Guía del usuario

AWS Modernización de mainframe

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

AWS Modernización de mainframe: Guía del usuario

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Las marcas comerciales y la imagen comercial de Amazon no se pueden utilizar en relación con ningún producto o servicio que no sea de Amazon, de ninguna manera que pueda causar confusión entre los clientes y que menosprecie o desacredite a Amazon. Todas las demás marcas registradas que no son propiedad de Amazon son propiedad de sus respectivos propietarios, que pueden o no estar afiliados, conectados o patrocinados por Amazon.

Table of Contents

¿Qué es la modernización del AWS mainframe 1
Características de la modernización del AWS mainframe 2
Patrones 3
¿Cómo empezar a modernizar el mainframe AWS 3
Servicios relacionados 4
Acceder a la modernización del AWS mainframe 5
¿Es la primera vez que utiliza Mainframe Modernization? AWS 5
Precios de la modernización de AWS mainframes 5
Configuración para la modernización del AWS mainframe 6
Inscríbase en una Cuenta de AWS 6
Creación de un usuario con acceso administrativo 6
Conceptos 9
Aplicación 9
Definición de aplicación 10
Trabajo por lotes 10
Configuración 11
Conjunto de datos 11
Entorno 11
Modernización de la unidad central 11
Viaje migratorio 11
Punto de montaje 12
Refactorización automatizada 12
Recreación de plataformas 12
Recurso 12
Motor de tiempo de ejecución 12
Enfoque de modernización 13
Fase de evaluación 13
Fase de movilización 14
Fase de migración y modernización 14
Fase de operación y optimización 15
Introducción 16
Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age 16
Requisitos previos 17
Paso 1: Cargar la aplicación de demostración 17

Paso 2: Crear la definición de aplicación ... 17
Paso 3: Crear un entorno de tiempo de ejecución ... 18
Paso 4: Crear una aplicación ... 23
Paso 5: Implementar una aplicación ... 25
Paso 6: Iniciar una aplicación ... 27
Paso 7: Acceder a la aplicación ... 27
Paso 8: Probar la aplicación ... 28
Eliminar recursos ... 30
Tutorial: Configurar el tiempo de ejecución gestionado para Rocket Software ... 30
Requisitos previos ... 31
Paso 1: creación y carga de un bucket de Amazon S3 ... 31
Paso 2: creación y configuración de un usuario de base de datos ... 33
Paso 3: creación y configuración de una AWS KMS key ... 35
Paso 4: creación y configuración de un secreto de base de datos de AWS Secrets Manager ... 36
Paso 5: Agrega el SSLMode al secreto ... 37
Paso 6: Crear un entorno de ejecución ... 38
Paso 7: Crea una aplicación ... 43
Paso 8: implementar una aplicación ... 49
Paso 9: Importar conjuntos de datos ... 51
Paso 10: Iniciar una aplicación ... 57
Paso 11: Conectarse a la CardDemo aplicación CICS ... 58
Eliminar recursos ... 65
Pasos a seguir a continuación ... 66
Ciclo de vida de componentes ... 67
Información general del ciclo de vida de los componentes ... 67
Actualización de versiones ... 69
AWS Descripción general de la versión de Mainframe Modernization Refactorization with AWS Blu Age ... 69
Aplicaciones administradas ... 71
Cree AWS recursos para una aplicación migrada ... 72
Permisos necesarios ... 72
Bucket de Amazon S3 ... 73
Database ... 73
AWS Key Management Service clave ... 74
AWS Secrets Manager secreto ... 74

Creación de una aplicación de ... 75
Creación de una aplicación de ... 75
Implementación de una aplicación de ... 76
Implementación de una aplicación de ... 76
Actualización de una aplicación de ... 77
Actualización de una aplicación de ... 78
Eliminación de una aplicación de ... 78
Eliminación de una aplicación de ... 79
Envío de trabajos por lotes para las aplicaciones de ... 79
Envío de un trabajo por lotes. ... 80
Reinicio de un trabajo por lotes ... 80
Cancelación de trabajos por lotes para las aplicaciones ... 82
Cancelación de un trabajo por lotes ... 82
Importación de conjuntos de datos para aplicaciones de ... 83
Importar un conjunto de datos ... 83
Exporte conjuntos de datos para aplicaciones ... 84
Exporte un conjunto de datos ... 84
Administración de las transacciones para las aplicaciones de ... 85
Gestión de las transacciones de las aplicaciones ... 85
Configure la aplicación gestionada por Rocket Software ... 86
Compatible con integraciones de terceros para Rocket Software ... 87
Configurar la aplicación gestionada AWS Blu Age ... 89
Estructura de las aplicaciones gestionadas de AWS Blu Age ... 89
Configuración del acceso a las utilidades para las aplicaciones administradas ... 92
Configuración de propiedades adicionales para aplicaciones administradas ... 103
Referencia de definición de aplicaciones ... 124
Sección de cabecera general ... 125
Descripción general de la sección de definiciones ... 127
AWS Ejemplo de definición de aplicación de Blu Age ... 127
AWS Detalles de la definición de Blu Age ... 128
Definición de la aplicación Rocket Software ... 134
Detalles de la definición de Rocket Software ... 136
Referencia de definición de conjuntos de datos ... 147
Propiedades comunes ... 148
Ejemplo de formato de solicitud de conjunto de datos para VSAM ... 150
Ejemplo de formato de solicitud de conjunto de datos para base GDG ... 152

Ejemplo de formato de solicitud de conjunto de datos para las generaciones PS o GDG 153
Ejemplo de formato de solicitud de conjunto de datos para PO 154
Entornos de tiempo de ejecución administrados 156
Creación de un entorno de tiempo de ejecución 157
Creación de un entorno de tiempo de ejecución 157
Actualizar un entorno de tiempo de ejecución 160
Actualizar un entorno de tiempo de ejecución 161
Periodo de mantenimiento 161
Detener un entorno de tiempo de ejecución 162
Detener un entorno de tiempo de ejecución 162
Reiniciar un entorno de tiempo de ejecución 163
Reiniciar un entorno de tiempo de ejecución 163
Eliminar un entorno de tiempo de ejecución 164
Eliminar un entorno de tiempo de ejecución 164
Pruebas de aplicaciones 166
Qué es Pruebas de aplicaciones 166
¿Es la primera vez que usa Application Testing? 167
Ventajas de Application Testing 168
Integración con AWS CloudFormation 168
Cómo funciona Pruebas de aplicaciones 169
Servicios relacionados 4
Acceso a Pruebas de aplicaciones 171
Precios de Pruebas de aplicaciones 171
Conceptos de Pruebas de aplicaciones 171
Caso de prueba 172
Conjunto de pruebas 172
Configuración del entorno de prueba 173
Cargar 173
Reproducción 173
Comparación 173
Comparación de bases de datos 173
Comparación de conjuntos de datos 174
Estado de comparación 175
Reglas de equivalencia 175
Comparación de conjuntos de datos de estado final 175
Comparaciones de bases de datos estado-progreso 176

Equivalencia funcional (FE) 176
Comparaciones de pantallas 3270 en línea 176
Datos de reproducción 176
Datos de referencia 177
Carga, reproducción y comparación 177
Diferencias 178
Equivalencias 178
Aplicación de origen 178
Aplicación de destino 178
Requisitos previos de Pruebas de aplicaciones 179
Flujos de trabajo de la consola en Pruebas de aplicaciones 179
Creación de casos de prueba en Pruebas de aplicaciones 180
Creación de conjuntos de pruebas en Pruebas de aplicaciones 182
Creación de configuraciones de entornos de prueba en Pruebas de aplicaciones 185
Tutorial: Configurar la CardDemo aplicación en las pruebas de aplicaciones 187
Requisitos previos 187
Paso 1: Prepárese para la configuración CardDemo 188
Paso 2: Crear todos los recursos necesarios 188
Paso 3: Implementar e iniciar la aplicación 189
Paso 4: Importar los datos iniciales 190
Paso 5: Conectarse a la CardDemo aplicación 191
Tutorial: Reproduzca y compare en AWS Blu Age usando CardDemo 192
Paso 1: Obtenga la imagen de EC2 Amazon Machine (AMI) de AWS Blu Age 192
Paso 2: Iniciar una EC2 instancia de Amazon con la AMI de AWS Blu Age 192
Paso 3: Cargue CardDemo los archivos dependientes a S3 194
Paso 4: Cargue las bases de datos e inicialice la aplicación CardDemo 194
Paso 5: Inicie el tiempo de ejecución de AWS Blu Age CloudFormation 197
Paso 6: Probar la EC2 instancia de Amazon de AWS Blu Age 199
Paso 7: Validar que los pasos anteriores se hayan completado correctamente 200
Paso 8: creación del caso de prueba 201
Paso 9: creación de una conjunto de pruebas 201
Paso 10: creación de una configuración de entorno de prueba 202
Paso 11: carga de los datos de entrada en el conjunto de pruebas 202
Paso 12: Reproducir y comparar 203
Páginas de código de conjuntos de datos admitidos en Pruebas de aplicaciones 203
Protección de datos en Pruebas de aplicaciones 214

Datos recopilados mediante las pruebas de la aplicación de modernización AWS del mainframe 215
Cifrado de datos en reposo para Pruebas de aplicaciones de AWS Mainframe Modernization 216
Creación de una clave administrada por el cliente 217
Especificación de una clave administrada por el cliente para Pruebas de aplicaciones de AWS Mainframe Modernization 219
AWS Contexto de cifrado de pruebas de aplicaciones de modernización de mainframe 219
Supervisión de sus claves de cifrado 220
Cifrado en tránsito 220
Cómo funciona Pruebas de aplicaciones con IAM 221
Políticas basadas en identidad 222
Políticas basadas en recursos 222
Acciones de políticas 223
Recursos de políticas 224
Claves de condición de políticas 226
ACLs 227
ABAC 227
Credenciales temporales 228
Sesiones de acceso directo 228
Roles de servicio 229
Roles vinculados a servicios 229
AWS Refactorización de Blu Age 230
AWS Lanzamientos de Blu Age 231
AWS Control de versiones de Blu Age 232
AWS Opciones de Blu Age Runtime 233
AWS Notas de lanzamiento de Blu Age 236
AWS Vulnerabilidades de seguridad de la era 348
Actualización de AWS Blu Age 350
AWS Ciclo vital de la era azul 352
AWS Conceptos de Blu Age Runtime 353
Arquitectura de alto nivel 353
Estructura de una aplicación modernizada 358
Descripción de los simplificadores de datos 395
AWS Blusam Blu Age 403
Programas disponibles en la aplicación web de utilidad 427

Consola de administración de Blusam 429
AWS Configuración de Blu Age Runtime 470
Principios básicos de la configuración de aplicaciones 471
Prioridad de la aplicación 473
JNDI para bases de datos 473
AWS Secretos de Blu Age Runtime 473
Otros archivos (groovy, sql, etc.) 486
Aplicación web adicional 486
Habilitación de propiedades 487
Propiedades de caché de Redis disponibles 579
Configuración de la seguridad para las aplicaciones de Gapwalk 595
AWS Tiempo de ejecución Blu Age APIs 612
Puntos finales para la construcción URLs 612
Puntos de conexión para la aplicación Gapwalk 613
Puntos de conexión REST de la consola de aplicaciones de Blusam 634
Administración de la consola de aplicaciones JICS 657
Estructuras de datos 678
Configurar AWS Blu Age Runtime (no gestionado) 688
AWS Requisitos previos de Blu Age Runtime 688
Incorporación de AWS Blu Age Runtime 689
Requisitos de configuración de infraestructura 694
AWS artefactos de Blu Age Runtime 701
Implemente AWS Blu Age Runtime en Amazon EC2 705
Implemente AWS Blu Age Runtime en Amazon ECS y Amazon EKS 716
Prueba la PlanetsDemo aplicación 724
Modificación del código fuente con el IDE para desarrolladores de Blu Age 728
Tutorial: Configurar AppStream 2.0 para el IDE de desarrolladores de AWS Blu Age 728
Tutorial: Utilice AWS Blu Age Developer en AppStream 2.0 734
AWS Preguntas frecuentes sobre Blu Age 751
General 751
AWS Blu Age Runtime 754
Datos 763
Transformación 765
Implementación 765
Replataforma de Rocket Software 769
Configurar Rocket Software (en Amazon EC2) 769

Requisitos previos de Rocket Software (en Amazon EC2) 770
Creación de puntos de conexión de VPC de Amazon para Amazon S3 770
Solicitud de actualización de la lista de permitidos de la cuenta 773
Crea el AWS Identity and Access Management rol 774
Concesión de permisos necesarios con License Manager 781
Suscríbase a Amazon Machine Images 782
Lanza una instancia de Rocket Software 786
Subred o VPC sin acceso a Internet 792
Configure la automatización AppStream 2.0 799
Configuración de la automatización al inicio de la sesión 800
Configuración de la automatización al final de la sesión 800
Ver conjuntos de datos como tablas en Enterprise Developer 800
Requisitos previos 801
Paso 1: Configurar la conexión ODBC al almacén de datos de Rocket Software (base de datos Amazon RDS) 802
Paso 2: Crear el archivo MFDBFH.cfg 804
Paso 3: Crear un archivo de estructura (STR) para el diseño del cuaderno 805
Paso 4: Crear una vista de base de datos con el archivo de estructura (STR) 807
Paso 5: Vea los conjuntos de datos de Rocket Software (anteriormente Micro Focus) en forma de tablas y columnas 808
Edite conjuntos de datos con las herramientas de archivos de datos de Enterprise Developer . 809
Requisitos previos 809
Inicie las herramientas de archivos de datos de Rocket Software (anteriormente Micro Focus) 810
Edite los conjuntos de datos de VSAM almacenados en la base de datos MFDBFH 810
Edite conjuntos de datos que no sean de VSAM almacenados en la base de datos MFDBFH 814
Edite conjuntos de datos VSAM y no VSAM almacenados en el sistema de archivos (EFS/FSx) 816
Tutoriales de Rocket Software 817
Tutorial: configurar la compilación de la aplicación BankDemo de muestra 817
Tutorial: Configuración CI/CD en proceso con Rocket Enterprise Developer 828
Tutorial: Cómo configurar la AppStream versión 2.0 para Enterprise Analyzer y Enterprise Developer 853
Tutorial: Usa plantillas con Rocket Enterprise Developer 862
Tutorial: configuración de Enterprise Analyzer 873

Tutorial: configuración de Enterprise Developer 884
Utilidades por lotes 890
Ubicación binaria 891
Utilidad por lotes M2SFTP 891
Utilidad por lotes M2WAIT 898
TXT2Utilidad de lotes de PDF 900
Utilidad por lotes M2DFUTIL 906
Utilidad por lotes M2RUNCMD 913
File Transfer 918
¿Qué es File Transfer? 918
Ventajas de AWS Mainframe Modernization File Transfer 919
Cómo funciona AWS Mainframe Modernization File Transfer 919
Instalación de un agente de File Transfer 920
Paso 1: creación de un conjunto de datos de zFS para M2-agent 921
Paso 2: formateo del conjunto de datos como zFS 921
Paso 3: montaje del sistema de archivos 921
Paso 4: verificación del montaje 922
Paso 5: introducción de OMV 922
Paso 6: establecimiento de la variable de entorno del directorio de instalación del agente ... 922
Paso 7: establecimiento de la variable de entorno del directorio de trabajo 922
Paso 8: creación del directorio de trabajo 922
Paso 9: copia del archivo tar del agente y del directorio de trabajo 923
Paso 10: Finalizar la instalación del agente 923
Configuración del agente de File Transfer 924
Paso 1: configuración de los permisos y control de tarea iniciada (STC) 924
Paso 2: creación de buckets de Amazon S3 925
Paso 3: Cree una clave de cifrado gestionada por el AWS KMS cliente 926
Paso 4: Cree un AWS Secrets Manager secreto para las credenciales del mainframe 927
Paso 5: creación de una política de IAM 928
Paso 6: creación de un usuario de IAM con credenciales de acceso a largo plazo 930
Paso 7: creación de un rol de IAM para que lo asuma el agente 931
Paso 8: configuración del agente 932
Creación de puntos de conexión de transferencia de datos 934
Creación de puntos de conexión de transferencia de datos 935
Creación de tareas de transferencia 937
Creación de tareas de transferencia 937

Visualización de las tareas de transferencia 941
Tutorial: Introducción a File Transfer 941
Descripción general 941
Paso 1: Transfiera el paquete tar de los archivos binarios del agente AWS a la partición lógica del mainframe 942
Paso 2: Configurar el agente de File Transfer en el mainframe de origen 942
Paso 3: Creación de un punto de conexión de transferencia de datos 942
Paso 4: Crear una tarea de transferencia 943
Paso 5: Ver el progreso de la tarea de transferencia 943
Páginas de códigos fuente y de destino admitidas 943
Tipos de conjuntos de datos de mainframe 943
Páginas de código admitidas 943
AWS Transform para mainframe 945
Ventajas principales 945
Tutorial sobre la transformación de la consola de aplicaciones de mainframe 946
Replicación de datos con Precisely 947
Requisitos previos 947
Suscripción a imágenes de máquina de Amazon 947
Lance la replicación de datos AWS de Mainframe Modernization con Precisely 948
Creación de una política de IAM 949
Creación de un rol de IAM 950
Adjunta el rol de IAM a la instancia de Amazon EC2 950
Conversión de ensambladores con mLogica 952
Qué es la conversión de ensambladores con mLogica 952
Compiladores de conversión de código 953
Arquitectura de conversión de código 953
Enfoque de automatización 954
Seguridad 954
Recursos adicionales 955
Descripción de la facturación por conversión de código 955
Alcance y facturación de conversión de código 955
Conceptos de conversión de código 958
Gestión de macros 958
Páginas de código (EBCDIC frente a ASCII) 958
CodeBuild 958
Descripción de los componentes y el proceso 959

AWS Mainframe Modernization contenedor ... 959
Bucket de proyectos de S3 ... 960
Ubicaciones de archivo de registro ... 960
Información general del proceso ... 961
Tutorial: conversión de código del ensamblador a COBOL ... 961
Requisitos previos ... 962
Paso 1: Comparta los activos de construcción con Cuenta de AWS ... 962
Paso 2: creación de buckets de Amazon S3 ... 963
Paso 3: creación de la política de IAM ... 963
Paso 4: creación de un rol de IAM ... 965
Paso 5: asociación de las políticas de IAM al rol de IAM. ... 966
Paso 6: Crea el CodeBuild proyecto ... 966
Paso 7: definición del proyecto y carga del código fuente ... 973
Paso 8: ejecución del análisis y comprensión de los informes ... 974
Paso 9: ejecución de la conversión del código ... 976
Paso 10: verificación de la conversión de código ... 980
Paso 11: descarga del código convertido ... 981
Eliminar recursos ... 981
Integración de Charon ... 983
Introducción a Charon-SSP ... 983
Sistemas operativos huéspedes compatibles ... 985
Requisitos previos de la instancia en la nube de Charon-SSP ... 986
Requisitos previos para las instancias ... 987
Creación y configuración de una instancia AWS en la nube para Charon (nueva GUI) ... 989
Requisitos previos generales ... 989
AWS Management Console Utilizándolo para lanzar una nueva instancia ... 990
Redefinición de plataforma con NTT DATA ... 996
Requisitos previos ... 996
Suscripción a imágenes de máquina de Amazon ... 996
AWS Lance la replataforma de modernización de mainframe con una instancia de NTT DATA ... 997
Introducción al uso de NTT Data ... 997
Tutorial: Implemente CardDemo la aplicación en NTT DATA ... 999
Diagrama de flujo de implementación ... 999
Requisitos previos ... 1000
Paso 1: preparación del entorno ... 1001

Paso 2: creación de una región de TPE 1001
Paso 3: creación del subsistema y del nodo de BPE 1002
Paso 4: Compilar e implementar la aplicación CardDemo 1011
Paso 5: importación del catálogo de BPE y TPE 1013
Paso 6: inicio y conexión de TPE con BPE 1013
Paso 7: ejecuta la aplicación CardDemo 1014
Solución de problemas 1020
Seguridad 1022
Protección de los datos 1023
Datos que recopila AWS Mainframe Modernization 1024
Cifrado de datos en reposo para el servicio de modernización AWS de mainframe 1025
Cómo utiliza AWS Mainframe Modernization las subvenciones en AWS KMS 1028
Creación de una clave administrada por el cliente 1030
Especificar una clave administrada por el cliente para AWS Mainframe Modernization 1031
AWS Contexto de cifrado de modernización del mainframe 1032
Supervisión de las claves de cifrado 1034
Más información 1049
Cifrado en tránsito 1049
Identity and Access Management 1050
Público 1050
Autenticación con identidades 1051
Administración de acceso mediante políticas 1055
Cómo funciona la modernización AWS del mainframe con la IAM 1058
Ejemplos de políticas basadas en identidades 1072
Solución de problemas 1075
Cómo utilizar roles vinculados a servicios 1077
Validación de conformidad 1081
Resiliencia 1082
Seguridad de la infraestructura 1082
AWS PrivateLink 1083
Consideraciones 1083
Creación de un punto de conexión de interfaz 1083
Creación de una política de punto de conexión 1084
Monitorización 1086
Monitorización con CloudWatch 1086
Métricas del entorno de tiempo de ejecución 1087

Métricas de aplicación 1088
Dimensiones 1094
Registro de llamadas a la API de CloudTrail con 1094
AWS Información sobre la modernización de mainframes en CloudTrail 1095
Descripción de las entradas de los archivos de registro de modernización de AWS mainframe 1096
Solución de problemas en M2 1098
Troubleshooting error: Time out while waiting for data set name to be unlocked 1098
Causa habitual 1099
Resolución 1099
Forzar la liberación del bloqueo 1099
Configuración del mecanismo de reparación automática de Blusam 1100
Administrador de bloqueos de Blusam 1101
Troubleshooting error: Cannot access an application URL 1102
Causa habitual 1102
Resolución 1102
Solución de problemas: AWS Blu Insights no se abre desde la consola 1103
Causa habitual 1103
Resolución 1104
Troubleshooting error: Environment unhealthy 1104
Causa habitual 1105
Resolución 1105
Solución de problemas de licencia de Rocket Software 1105
Verifica que la EC2 instancia de Amazon tenga la función de licencia de IAM 1106
Uso del Analizador de accesibilidad 1107
Ejecute el daemon de licencias 1107
Problemas de licencia con Enterprise Server o las herramientas de compilación empresarial en Linux después de aplicar parches al sistema operativo 1108
Historial de documentos 1109
..... mcxvii

¿Qué es la modernización del AWS mainframe?

AWS La modernización de mainframe le ayuda a modernizar sus aplicaciones de mainframe para convertirlas en entornos de tiempo de ejecución gestionados. AWS Ofrece herramientas y recursos para ayudarle a planificar e implementar la migración y modernización. Puede analizar sus aplicaciones de mainframe existentes, desarrollarlas o actualizarlas mediante COBOL o PL/I implementar un proceso automatizado para la integración y la entrega continuas de las aplicaciones. (CI/CD) Puede elegir entre patrones automatizados de refactorización y de redefinición de plataforma, en función de las necesidades de sus clientes. Si es un consultor que ayuda a un cliente a migrar sus cargas de trabajo de mainframe, puede utilizar las herramientas de modernización del AWS mainframe en todas las fases del proceso de migración y modernización, desde la planificación inicial hasta las operaciones en la nube posteriores a la migración.

Puede usar AWS Mainframe Modernization para crear y administrar de manera eficiente el entorno de ejecución de sus aplicaciones de mainframe, así como AWS para administrar y monitorear sus aplicaciones modernizadas.

Temas

- Características de la modernización del AWS mainframe
- Patrones
- ¿Cómo empezar a modernizar el mainframe AWS
- Servicios relacionados
- Acceder a la modernización del AWS mainframe
- ¿Es la primera vez que utiliza Mainframe Modernization? AWS
- Precios de la modernización de AWS mainframes

> **Note**
>
> ¿Ha contactado con socios competentes en migración de AWS mainframe o con servicios AWS profesionales para su proyecto de modernización de mainframe? De no ser así, le recomendamos que contrate a expertos para su proyecto.
>
> - AWS Socios competentes en materia de modernización de mainframes
> - Servicios profesionales de AWS

Las características y los casos de uso de la modernización de AWS mainframes respaldan un enfoque de modernización evolutivo, que proporciona beneficios a corto plazo al mejorar la agilidad y muchas oportunidades para optimizar e innovar más adelante. Para obtener más información, consulte Enfoque de modernización.

Características de la modernización del AWS mainframe

AWS Las funciones de modernización del mainframe admiten los siguientes casos de uso:

- Evaluación: la capacidad de evaluación de AWS Mainframe Modernization puede ayudarle a evaluar, definir y planificar un proyecto de migración y modernización.
- Refactorización: con la tecnología de AWS Blu Age, puede utilizar la refactorización para convertir lenguajes de programación de aplicaciones antiguos, crear macroservicios o microservicios y modernizar las interfaces de usuario () y los paquetes de software de aplicaciones. UIs

  AWS Blu Insights ya está disponible mediante el inicio de sesión único. AWS Management Console Ya no tiene que gestionar credenciales de AWS Blu Insights independientes. Puede acceder a las funciones de AWS AWS Blu Age Codebase y Transformation Center directamente desde. AWS Management Console
- Redefinición de la plataforma: con la tecnología de la solución Micro Focus Enterprise, puede llevar a la aplicación a un lugar donde gran parte del código fuente de la aplicación se vuelva a compilar sin cambios.
- IDE para desarrolladores: AWS Mainframe Modernization ofrece un entorno de desarrollo integrado (IDE) bajo demanda para que los desarrolladores puedan escribir código más rápido mediante la edición y la depuración inteligentes, la compilación instantánea de código y las pruebas unitarias.
- Tiempo de ejecución gestionado: el entorno de tiempo de ejecución gestionado de AWS Mainframe Modernization monitorea continuamente los clústeres para mantener las cargas de trabajo empresariales en funcionamiento mediante un procesamiento autorreparable y un escalado automatizado.
- Integración y entrega continuas(CI/CD): la CI/CD función AWS Mainframe Modernization ayuda a los equipos de desarrollo de aplicaciones a realizar cambios de código con mayor frecuencia y fiabilidad, lo que acelera la velocidad de migración, aumenta la calidad y ayuda a reducir el lanzamiento de nuevas funciones time-to-market empresariales.

- Integraciones con otros AWS servicios: la modernización del AWS mainframe admite AWS CloudFormation y permite una implementación repetible y AWS Key Management Service una mayor seguridad y conformidad. AWS PrivateLink
- Disponibilidad ampliada: la modernización de AWS mainframe ya está disponible en EE. UU. Este (Ohio), EE. UU. Oeste (Norte de California), Asia Pacífico (Bombay), Asia Pacífico (Seúl), Asia Pacífico (Singapur), Asia Pacífico (Tokio), Europa (Londres) y Europa (París).

Para obtener más información, consulte Características de modernización de AWS mainframe.

Patrones

El patrón de refactorización automatizada, impulsado por AWS Blu Age, se centra en acelerar la modernización al convertir toda la pila de aplicaciones antiguas y su capa de datos en una aplicación moderna basada en Java y, al mismo tiempo, preservar la equivalencia funcional. Durante esta transformación automatizada, crea una aplicación de varios niveles con una interfaz basada en Angular, un backend de Java con API y una capa de datos que accede a los almacenes de datos modernos. El proceso de refactorización proporciona una funcionalidad equivalente a la versión anterior para aumentar la automatización de los proyectos, lo que se traduce en velocidad, calidad y menor coste, lo que permite obtener beneficios empresariales más rápidamente. Para obtener más información, consulte AWS Mainframe Modernization Automated Refactor.

El patrón de cambio de plataforma, impulsado por la suite Enterprise de Rocket Software (anteriormente Micro Focus), se centra en preservar el lenguaje, el código y los artefactos de la aplicación para minimizar el impacto en los activos y los equipos de las aplicaciones. Ayuda a los clientes a mantener los conocimientos y las habilidades de la aplicación. Si bien los cambios en las aplicaciones son limitados, este patrón también facilita la modernización de la infraestructura y los procesos. La infraestructura se transforma en un servicio administrado moderno basado en la nube, mientras que los procesos se modifican para seguir las mejores prácticas de desarrollo de aplicaciones y operaciones de TI. Para obtener más información, consulte AWS Mainframe Modernization Replatform

¿Cómo empezar a modernizar el mainframe AWS

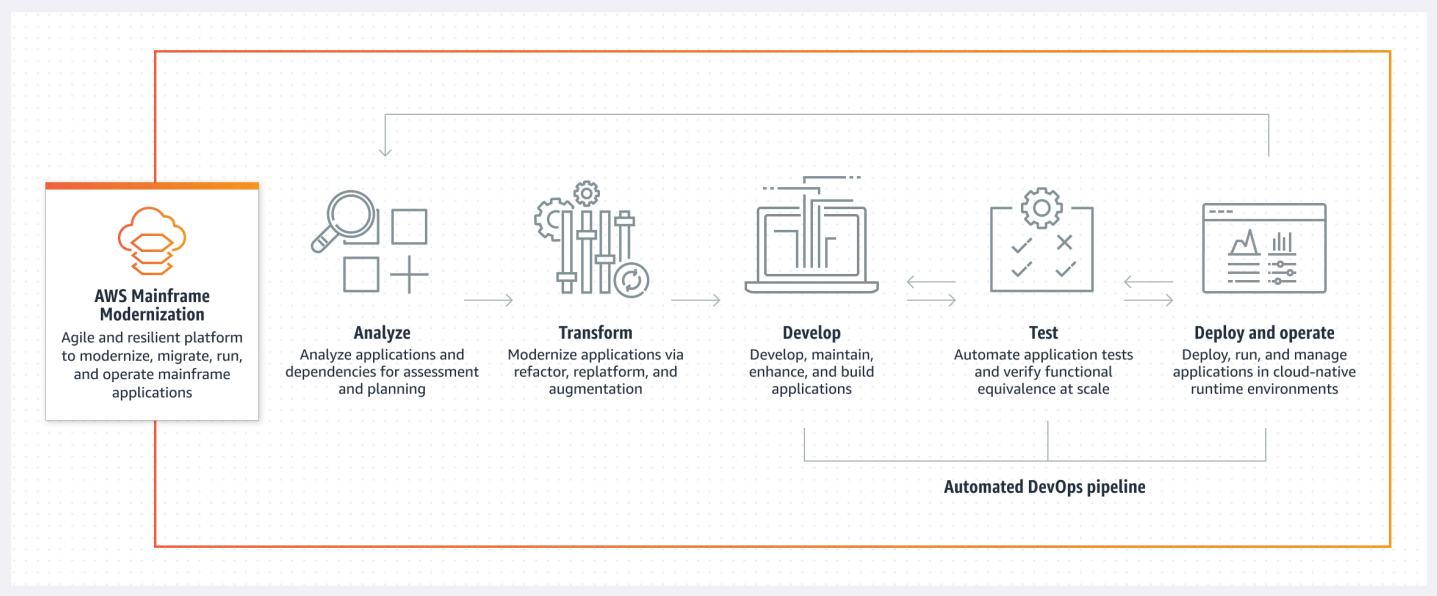
¡Pruébelo! Ofrecemos tutoriales y ejemplos de aplicaciones para ayudarle a hacerse una idea de lo que ofrece la modernización de AWS mainframes. Elija entre el tutorial Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age o el Tutorial: Configurar el tiempo de ejecución

gestionado para Rocket Software (anteriormente Micro Focus) para ver un step-by-step tutorial completo.

Si está interesado en la refactorización automatizada, consulte las herramientas de AWS Blu Age en. BluInsights También puede configurar la AppStream versión 2.0 para acceder al IDE para desarrolladores de AWS Blu Age o a las herramientas Rocket Enterprise Analyzer (anteriormente Micro Focus Enterprise Analyzer) y Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer).

Los tutoriales y las aplicaciones de muestra solo le dan una idea de lo que ofrece AWS Mainframe Modernization. Cuando esté listo para iniciar un proyecto de modernización, consulte los detalles sobre las etapas y las tareas de un proyecto de modernización en Enfoque de modernización.

El siguiente diagrama muestra el flujo de trabajo del servicio de modernización de AWS mainframe para analizar, transformar, desarrollar, probar, implementar y operar aplicaciones de mainframe.



Servicios relacionados

Además de Blu Insights para la refactorización automatizada, puede utilizar los siguientes AWS servicios con AWS Mainframe Modernization.

- Amazon RDS para alojar sus bases de datos migradas
- Amazon S3 para almacenar archivos binarios de aplicaciones y de definición
- Amazon FSx o Amazon EFS para almacenar datos de aplicaciones

- Amazon AppStream para acceder a las herramientas Rocket Enterprise Analyzer y Rocket Enterprise Developer
- AWS CloudFormation para la DevOps canalización automatizada que puede utilizar CI/CD para configurar sus aplicaciones migradas
- AWS Migration Hub
- AWS DMS para migrar sus bases de datos

Acceder a la modernización del AWS mainframe

Actualmente, puede acceder a AWS Mainframe Modernization a través de la consola que se encuentra en. https://console.aws.amazon.com/m2/ Para obtener una lista de las regiones en las que está disponible la modernización de AWS mainframes, consulte los puntos finales y las cuotas de modernización de AWS mainframes en la. Referencia general de Amazon Web Services

¿Es la primera vez que utiliza Mainframe Modernization? AWS

Si es la primera vez que utiliza AWS Mainframe Modernization, le recomendamos que comience leyendo las siguientes secciones:

- Comience con la modernización de AWS mainframe
- Configuración para la modernización del AWS mainframe

Precios de la modernización de AWS mainframes

AWS La modernización del mainframe cobra por el uso de instancias compatibles con los entornos de tiempo de ejecución gestionados. Además, AWS Mainframe Modernization ofrece algunas herramientas sin cargos adicionales. Usted es responsable de los cargos incurridos por otros AWS servicios que utilice en relación con la modernización del AWS mainframe. AWS avisará con 30 días de antelación antes de que entre en vigor cualquier cambio en los precios por el uso de AWS Mainframe Modernization. Para obtener más información, consulte Modernización de mainframe con. AWS

Con AWS Blu Insights, usted paga por el uso del Centro de Transformación. Para obtener más información, consulte AWS Mainframe Modernization pricing

Configuración para la modernización del AWS mainframe

Antes de empezar a utilizar AWS Mainframe Modernization, usted o su administrador deben registrarse Cuenta de AWS, crear un usuario con una configuración administrativa y proteger a los usuarios de IAM.

Temas

- Inscríbase en una Cuenta de AWS
- Creación de un usuario con acceso administrativo

Inscríbase en una Cuenta de AWS

Si no tiene uno Cuenta de AWS, complete los siguientes pasos para crearlo.

Para suscribirse a una Cuenta de AWS

1. Abrir https://portal.aws.amazon.com/billing/registro.
2. Siga las instrucciones que se le indiquen.

   Parte del procedimiento de registro consiste en recibir una llamada telefónica o mensaje de texto e indicar un código de verificación en el teclado del teléfono.

   Cuando te registras en un Cuenta de AWS, Usuario raíz de la cuenta de AWSse crea un. El usuario raíz tendrá acceso a todos los Servicios de AWS y recursos de esa cuenta. Como práctica recomendada de seguridad, asigne acceso administrativo a un usuario y utilice únicamente el usuario raíz para realizar tareas que requieren acceso de usuario raíz.

AWS te envía un correo electrónico de confirmación una vez finalizado el proceso de registro. En cualquier momento, puede ver la actividad de su cuenta actual y administrarla accediendo a https://aws.amazon.com/y seleccionando Mi cuenta.

Creación de un usuario con acceso administrativo

Después de crear un usuario administrativo Cuenta de AWS, asegúrelo Usuario raíz de la cuenta de AWS AWS IAM Identity Center, habilite y cree un usuario administrativo para no usar el usuario root en las tareas diarias.

Proteja su Usuario raíz de la cuenta de AWS

1. Inicie sesión AWS Management Consolecomo propietario de la cuenta seleccionando el usuario root e introduciendo su dirección de Cuenta de AWS correo electrónico. En la siguiente página, escriba su contraseña.

   Para obtener ayuda para iniciar sesión con el usuario raíz, consulte Iniciar sesión como usuario raíz en la Guía del usuario de AWS Sign-In .
2. Active la autenticación multifactor (MFA) para el usuario raíz.

   Para obtener instrucciones, consulte Habilitar un dispositivo MFA virtual para el usuario Cuenta de AWS raíz (consola) en la Guía del usuario de IAM.

Creación de un usuario con acceso administrativo

1. Activar IAM Identity Center.

   Consulte las instrucciones en Activar AWS IAM Identity Center en la Guía del usuario de AWS IAM Identity Center .
2. En IAM Identity Center, conceda acceso administrativo a un usuario.

   Para ver un tutorial sobre su uso Directorio de IAM Identity Center como fuente de identidad, consulte Configurar el acceso de los usuarios con la configuración predeterminada Directorio de IAM Identity Center en la Guía del AWS IAM Identity Center usuario.

Inicio de sesión como usuario con acceso de administrador

- Para iniciar sesión con el usuario de IAM Identity Center, use la URL de inicio de sesión que se envió a la dirección de correo electrónico cuando creó el usuario de IAM Identity Center.

  Para obtener ayuda para iniciar sesión con un usuario del Centro de identidades de IAM, consulte Iniciar sesión en el portal de AWS acceso en la Guía del AWS Sign-In usuario.

Concesión de acceso a usuarios adicionales

1. En IAM Identity Center, cree un conjunto de permisos que siga la práctica recomendada de aplicar permisos de privilegios mínimos.

Para conocer las instrucciones, consulte Create a permission set en la Guía del usuario de AWS IAM Identity Center .

2. Asigne usuarios a un grupo y, a continuación, asigne el acceso de inicio de sesión único al grupo.

   Para conocer las instrucciones, consulte Add groups en la Guía del usuario de AWS IAM Identity Center .

AWS Conceptos de modernización de mainframe

AWS La modernización del mainframe proporciona herramientas y recursos que le ayudan a migrar, modernizar y ejecutar las cargas de trabajo del mainframe. AWS Puede utilizar esta página para conocer varios conceptos relacionados con la modernización de AWS mainframes, como las aplicaciones, la modernización, los entornos, la replataforma, la refactorización y los motores de tiempo de ejecución.

Temas

- Aplicación
- Definición de aplicación
- Trabajo por lotes
- Configuración
- Conjunto de datos
- Entorno
- Modernización de la unidad central
- Viaje migratorio
- Punto de montaje
- Refactorización automatizada
- Recreación de plataformas
- Recurso
- Motor de tiempo de ejecución

Aplicación

Una carga de trabajo de mainframe en ejecución en Mainframe Modernization. AWS Una aplicación está formada por un conjunto de trabajos por lotes, transacciones interactivas (CICS o IMS) u otros componentes. Usted define el alcance. Debe definir y especificar los componentes o recursos que necesite la carga de trabajo, como las transacciones del CICS o los trabajos por lotes.

Definición de aplicación

La definición o especificación de los componentes y recursos que necesita una aplicación (carga de trabajo de mainframe) que se ejecuta en AWS Mainframe Modernization. Es importante separar la definición de la aplicación en sí, ya que es posible reutilizar la misma definición para varias etapas (preproducción, producción), representadas por diferentes entornos de tiempo de ejecución.

Trabajo por lotes

Un programa planificado que se ha configurado para ejecutarse sin requerir la interacción del usuario. En AWS Mainframe Modernization, necesitará almacenar los archivos de JCL de los trabajos por lotes y los archivos binarios de los trabajos por lotes se preparan en un bucket de Amazon Simple Storage Service (Amazon S3), y proporcionar la ubicación de ambos en el archivo de definición de la aplicación. Al ejecutar un trabajo por lotes, AWS Mainframe Modernization informa de los siguientes valores de estado:

Submitting

El trabajo por lotes está en proceso de enviarse.

Holding

El trabajo por lotes está en espera.

Dispatching

El trabajo por lotes está en proceso de enviarse.

Running

El trabajo se está ejecutando.

Cancelling

El trabajo por lotes está en proceso de ser cancelado.

Cancelled

El trabajo por lotes está cancelado.

Succeeded

El trabajo por lotes ha terminado de ejecutarse correctamente.

Failed

Se ha producido un error en el trabajo por lotes.

Succeeded With Warning

El trabajo por lotes ha terminado de ejecutarse correctamente y se informó de un error menor. El código de condición del trabajo devuelto como parte de la GetBatchJobExecution respuesta indica la causa del error.

Configuración

Las características de un entorno o una aplicación. Las configuraciones del entorno consisten en el tipo de motor, la versión del motor, los patrones de disponibilidad, las configuraciones opcionales del sistema de archivos y más.

Las configuraciones de las aplicaciones pueden ser estáticas o dinámicas. Las configuraciones estáticas solo cambian cuando se actualiza una aplicación mediante la implementación de una nueva versión. Las configuraciones dinámicas, que suelen ser una actividad operativa, como activar o desactivar el rastreo, cambian en cuanto se actualizan.

Conjunto de datos

Archivo que contiene datos para su uso por parte de las aplicaciones.

Entorno

Una combinación con nombre de recursos de AWS cómputo, un motor de tiempo de ejecución y detalles de configuración creada para alojar una o más aplicaciones.

Modernización de la unidad central

El proceso de migración de aplicaciones de un entorno de mainframe heredado a. AWS

Viaje migratorio

El end-to-end proceso de migración y modernización de las aplicaciones heredadas, normalmente consta de las siguientes fases: evaluar, movilizar, migrar y modernizar, y operar y optimizar.

Punto de montaje

Directorio de un sistema de archivos que proporciona acceso a los archivos almacenados en ese sistema.

Refactorización automatizada

El proceso de modernizar los artefactos de las aplicaciones heredadas para que se ejecuten en un entorno de nube moderno. Puede incluir la conversión de código y datos. Para obtener más información, consulte AWS Mainframe Modernization Automated Refactor.

Recreación de plataformas

Proceso de trasladar una aplicación y sus artefactos de una plataforma informática a otra diferente. Para obtener más información, consulte AWS Mainframe Modernization Replatform

Recurso

Componente físico o virtual de un sistema informático.

Motor de tiempo de ejecución

Software que facilita la ejecución de una aplicación.

Enfoque de modernización

La migración es compleja y tiene muchas variables. AWS La modernización de los mainframes ofrece un enfoque evolutivo que proporciona algunos beneficios a corto plazo, ya que mejora la agilidad y ofrece numerosas oportunidades para optimizar e innovar más adelante. Además, la modernización del AWS mainframe ayuda a simplificar el proceso y, al mismo tiempo, respeta los detalles de la empresa y el negocio de su cliente. Los dos enfoques principales que admite la modernización de los AWS mainframes son la refactorización automática o la replataforma. La opción que elija depende de la situación de su cliente.

La refactorización automatizada utiliza las herramientas de AWS Blu Age para convertir automáticamente el código, los datos y las dependencias a lenguajes, almacenes de datos y marcos modernos, al tiempo que garantiza la equivalencia funcional con las mismas funciones empresariales.

Replatforming utiliza las herramientas de Rocket Software (anteriormente Micro Focus) para transformar las cargas de trabajo del mainframe en servicios ágiles. AWS

Puede pensar en el proceso de modernización por etapas. La primera etapa incluye tres fases: evaluar, movilizar y migrar y modernizar. La siguiente etapa incluye la fase de operación y optimización, en la que puede identificar más oportunidades de innovación.

Temas

- Fase de evaluación
- Fase de movilización
- Fase de migración y modernización
- Fase de operación y optimización

Fase de evaluación

En el nivel más alto, la fase de evaluación analiza si está preparado para migrar. Usted define un modelo de negocio y, a continuación, capacita a su equipo con talleres y un día de inmersión (demostraciones y laboratorios) ofrecido por AWS. Los talleres y los días de inmersión abordan diferentes temas. Estas tareas se llevan a cabo fuera de la modernización del mainframe. AWS

Fase de movilización

En la fase de movilización, el proyecto comienza desde el principio y, a continuación, se ejecuta un proceso de descubrimiento que extrae los datos de las aplicaciones de su unidad central y los incorpora a una herramienta de migración. Identifica las aplicaciones que desea migrar y selecciona algunas de ellas para ponerlas a prueba. Usted perfecciona su modelo de negocio, redacta su plan de migración y decide cómo quiere gestionar la seguridad y el cumplimiento, la gobernanza de las cuentas y su modelo operativo. Configura un centro de excelencia en la nube con las personas adecuadas de su equipo. Ejecuta los programas piloto y documenta lo que ha aprendido. Es usted quien refine su plan de migración y su modelo de negocio. Muchas de estas tareas se llevan a cabo fuera de la modernización del AWS mainframe.

Fase de migración y modernización

La fase de migración y modernización se aplica a cada aplicación y consta de varias tareas, entre las que se incluyen la asignación de personal, la realización de un descubrimiento exhaustivo, la definición de la arquitectura de la aplicación adecuada, la configuración de los entornos de ejecución de las aplicaciones AWS, la reconfiguración de la plataforma o la refactorización del código, la integración con otros sistemas y, por supuesto, las pruebas. Al final de la fase, debe implementar en producción las aplicaciones se han reconfigurado o cuya plataforma se ha redefinido y pasar al nuevo sistema de AWS. La mayoría o la totalidad de estas tareas se llevan a cabo en AWS Mainframe Modernization, en otro AWS servicio o en una herramienta a la que Mainframe Modernization dé acceso. AWS

Si desea utilizar la refactorización automatizada, consulte Blu Insights. AWS Blu Insights ya está disponible AWS Management Console mediante el inicio de sesión único. Ya no tiene que gestionar credenciales de AWS Blu Insights independientes. Puede acceder a las funciones de AWS AWS Blu Age Codebase y Transformation Center directamente desde. AWS Management Console

Para migrar datos del mainframe a AWS, recomendamos el y el. AWS SCT AWS Database Migration Service Para obtener más información, consulte What is the AWS Schema Conversion Tool? el la guía del usuario de AWS Schema Conversion Tool y ¿Qué es AWS Database Migration Service? en la Guía el usuario de AWS Database Migration Service .

Fase de operación y optimización

En la fase de operación y optimización, usted se centra en supervisar las aplicaciones implementadas, administrar los recursos y garantizar que la seguridad y el cumplimiento estén actualizados. También evalúa las oportunidades para optimizar las cargas de trabajo migradas.

Comience con la modernización de AWS mainframe

Puede empezar a modernizar el AWS mainframe siguiendo los tutoriales que le presentan el servicio y cada motor de ejecución.

Temas

- Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age
- Tutorial: Configurar el tiempo de ejecución gestionado para Rocket Software (anteriormente Micro Focus)

Para seguir aprendiendo, consulte los siguientes tutoriales.

- Tutorial: Configuración de la versión de Rocket Software (anteriormente Micro Focus) para la aplicación BankDemo de muestra
- Tutorial: Configuración de un CI/CD canalización para su uso con Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)

Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age

Puede implementar una aplicación modernizada de AWS Blu Age en un entorno de ejecución de modernización de AWS mainframe con una aplicación de demostración especificada en este tutorial.

Temas

- Requisitos previos
- Paso 1: Cargar la aplicación de demostración
- Paso 2: Crear la definición de aplicación
- Paso 3: Crear un entorno de tiempo de ejecución
- Paso 4: Crear una aplicación
- Paso 5: Implementar una aplicación
- Paso 6: Iniciar una aplicación
- Paso 7: Acceder a la aplicación
- Paso 8: Probar la aplicación

- Eliminar recursos

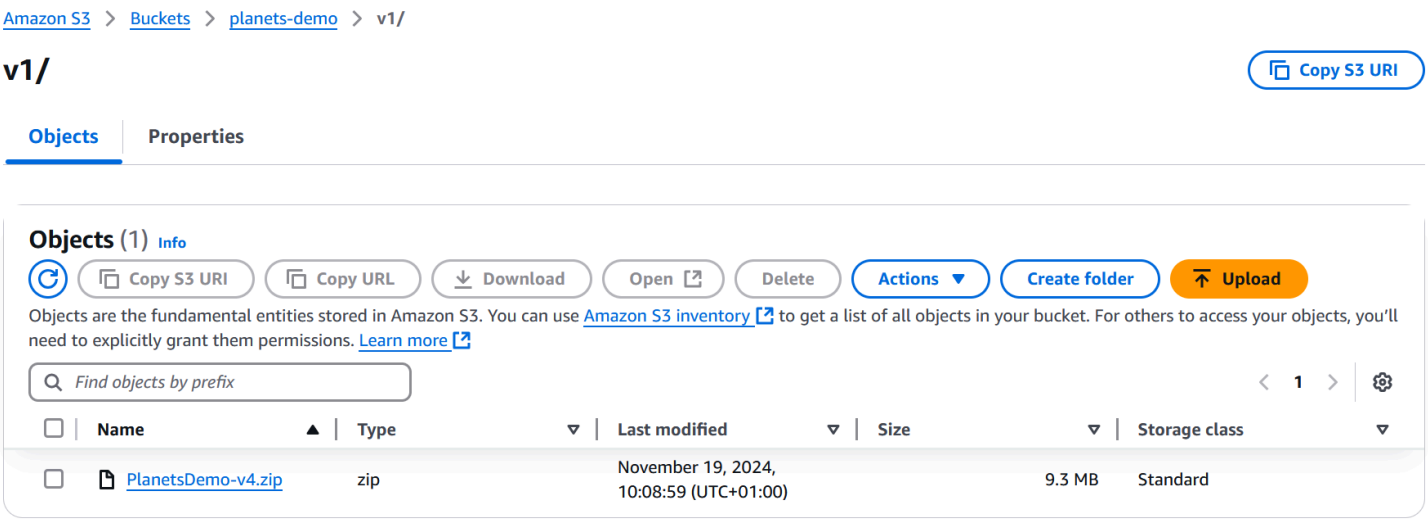
Requisitos previos

Para completar este tutorial, descargue el archivo PlanetsDemo de la aplicación de demostración - v4.zip.

La aplicación de demostración en ejecución requiere un navegador moderno para acceder a ella. El hecho de que ejecute este navegador desde su escritorio o desde una instancia de Amazon Elastic Compute Cloud, por ejemplo, dentro de la VPC, determinará su configuración de seguridad.

Paso 1: Cargar la aplicación de demostración

Cargue la aplicación de demostración en un bucket de Amazon S3. Asegúrese de que este bucket esté en la misma Región de AWS en la que va a implementar la aplicación. El siguiente ejemplo muestra un depósito denominado planets-demo, con un key prefix o carpeta denominado v1 y un archivo denominado. `planetsdemo-v4.zip`



> **Note**
>
> La carpeta del bucket es obligatoria.

Paso 2: Crear la definición de aplicación

Para implementar una aplicación en el entorno de ejecución administrado, necesita una definición de aplicación de modernización AWS del mainframe. Esta definición es un archivo JSON que describe

la ubicación y la configuración de la aplicación. El siguiente ejemplo es una definición de aplicación de este tipo para la aplicación de demostración:

```
{
    "template-version": "2.0",
    "source-locations": [{
        "source-id": "s3-source",
        "source-type": "s3",
        "properties": {
            "s3-bucket": "planets-demo",
            "s3-key-prefix": "v1"
        }
    }],
    "definition": {
        "listeners": [{
            "port": 8196,
            "type": "http"
        }],
        "ba-application": {
            "app-location": "${s3-source}/PlanetsDemo-v4.zip"
        }
    }
}
```

Cambie la `s3-bucket` entrada por el nombre del archivo zip de la aplicación de ejemplo (por ejemplo,`planets-demo`) y la `app-location` entrada por la ruta S3 en la que guardó el archivo zip de la aplicación de ejemplo (por ejemplo,`${s3-source}/PlanetsDemo-v4.zip`).

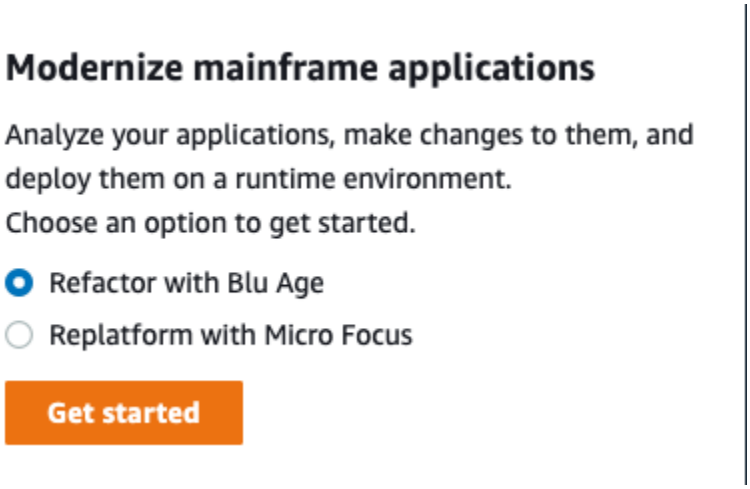
> **Note**
>
> Asegúrese de crear el archivo de definición de la aplicación en su local como un archivo de texto.

Para obtener más información sobre la definición de aplicaciones, consulte AWS Ejemplo de definición de aplicación de Blu Age.

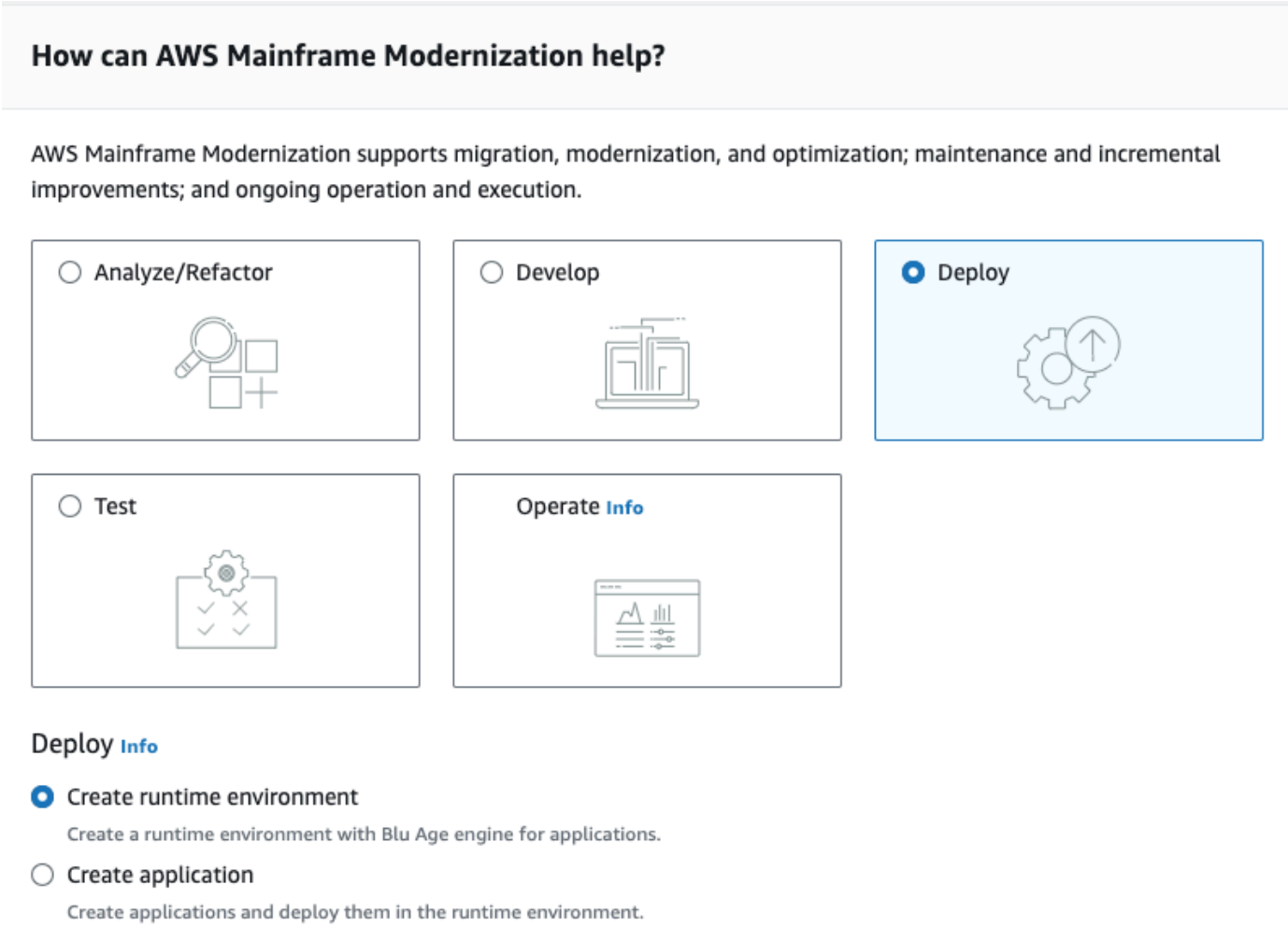
Paso 3: Crear un entorno de tiempo de ejecución

Para crear el entorno de ejecución de AWS Mainframe Modernization, lleve a cabo los siguientes pasos:

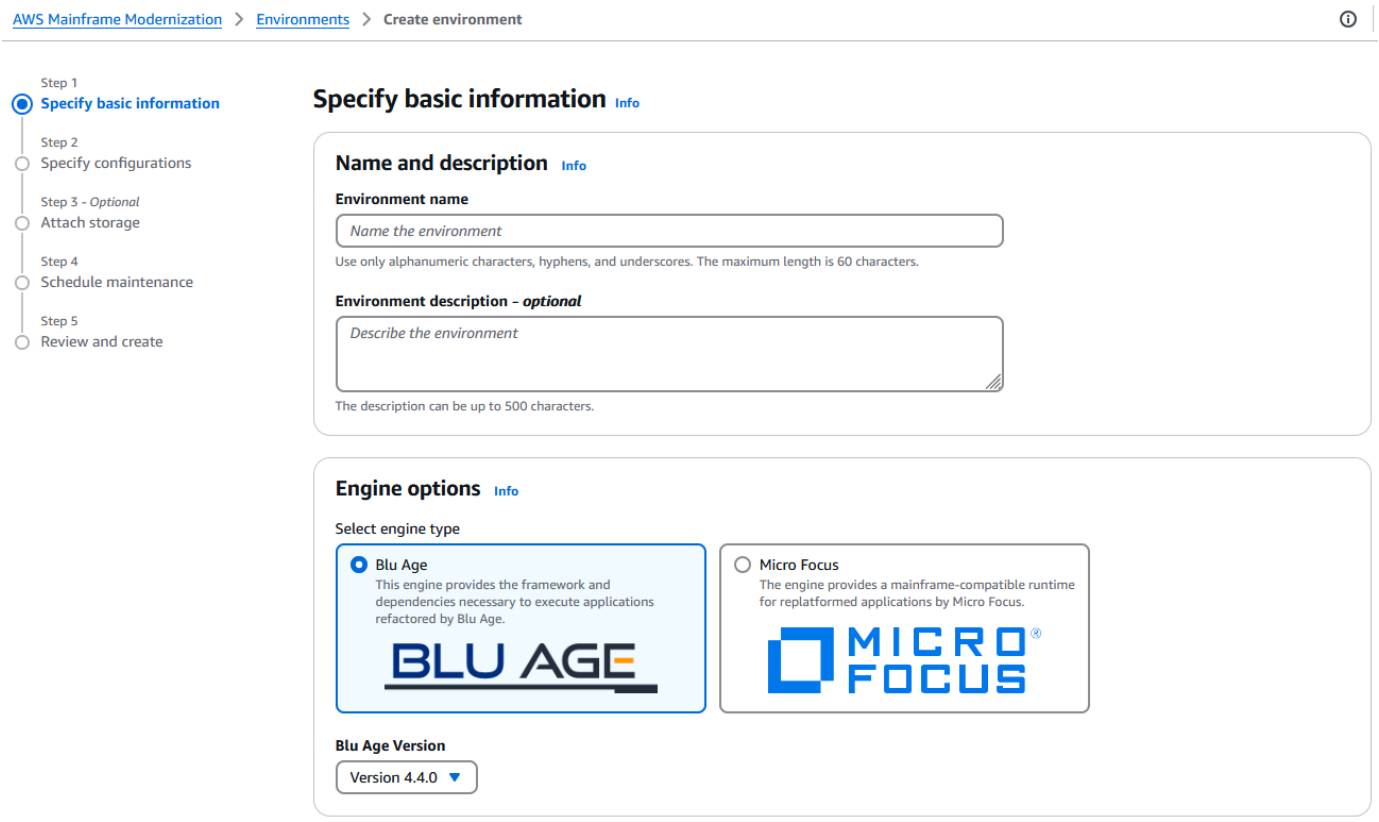
1. Abra la consola de AWS Mainframe Modernization.
2. En el Región de AWS selector, elija la región en la que desee crear el entorno. Región de AWS Debe coincidir con la región en la que creó el bucket de S3Paso 1: Cargar la aplicación de demostración.
3. En Modernice las aplicaciones de mainframe, elija Refactorizar con Blu Age y, a continuación, elija Comenzar.



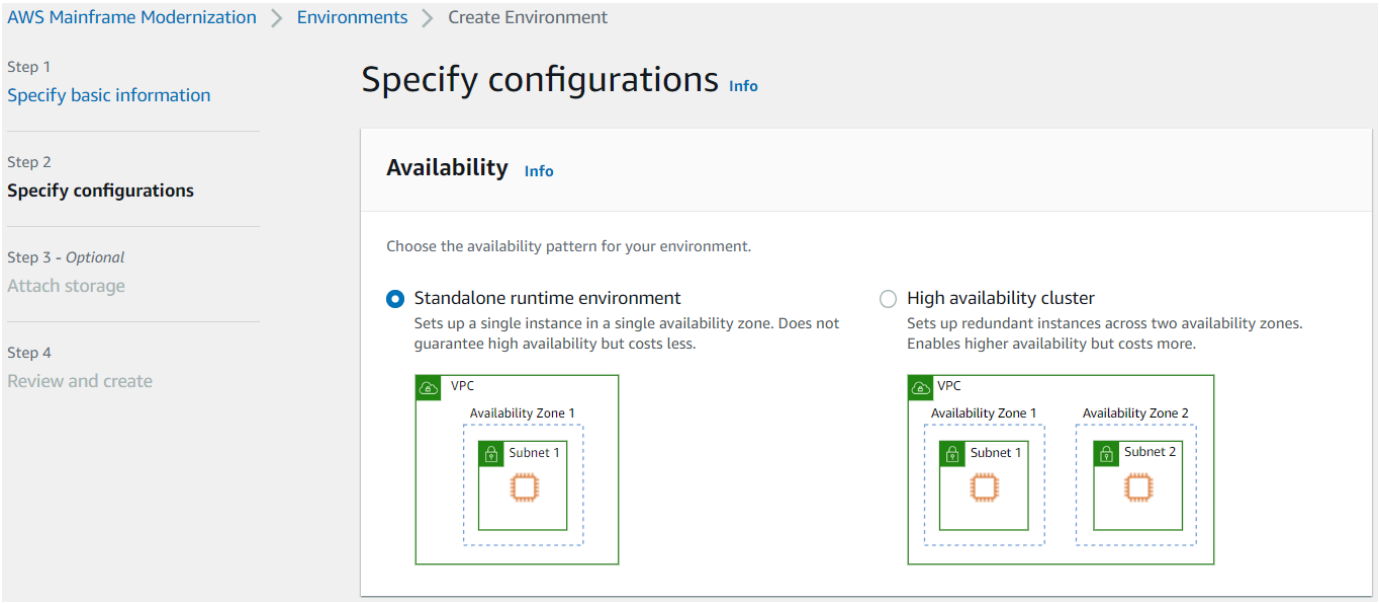
4. En Cómo puede ayudarle AWS Mainframe Modernization, elija Implementar y Crear entorno en tiempo de ejecución.



5. En el panel de navegación de la izquierda, seleccione Entornos y, a continuación, Crear entorno. En la página Especificar información básica, introduzca un nombre y una descripción para su entorno y, a continuación, asegúrese de que está seleccionado el motor AWS Blu Age. Si lo desea, puede agregar etiquetas al recurso creado. A continuación, elija Siguiente.

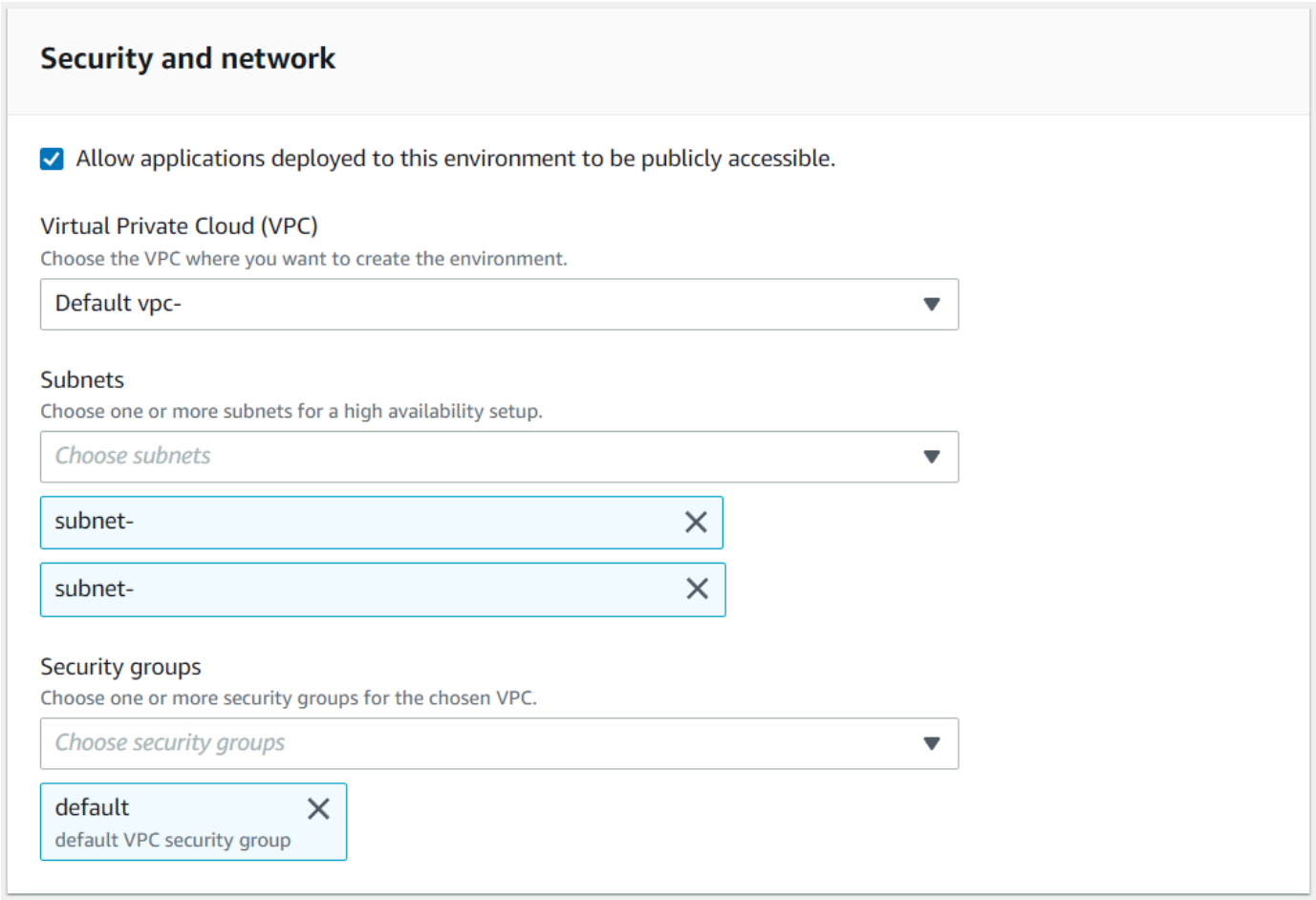


6. En la página Especificar las configuraciones, elija Entorno en tiempo de ejecución independiente.



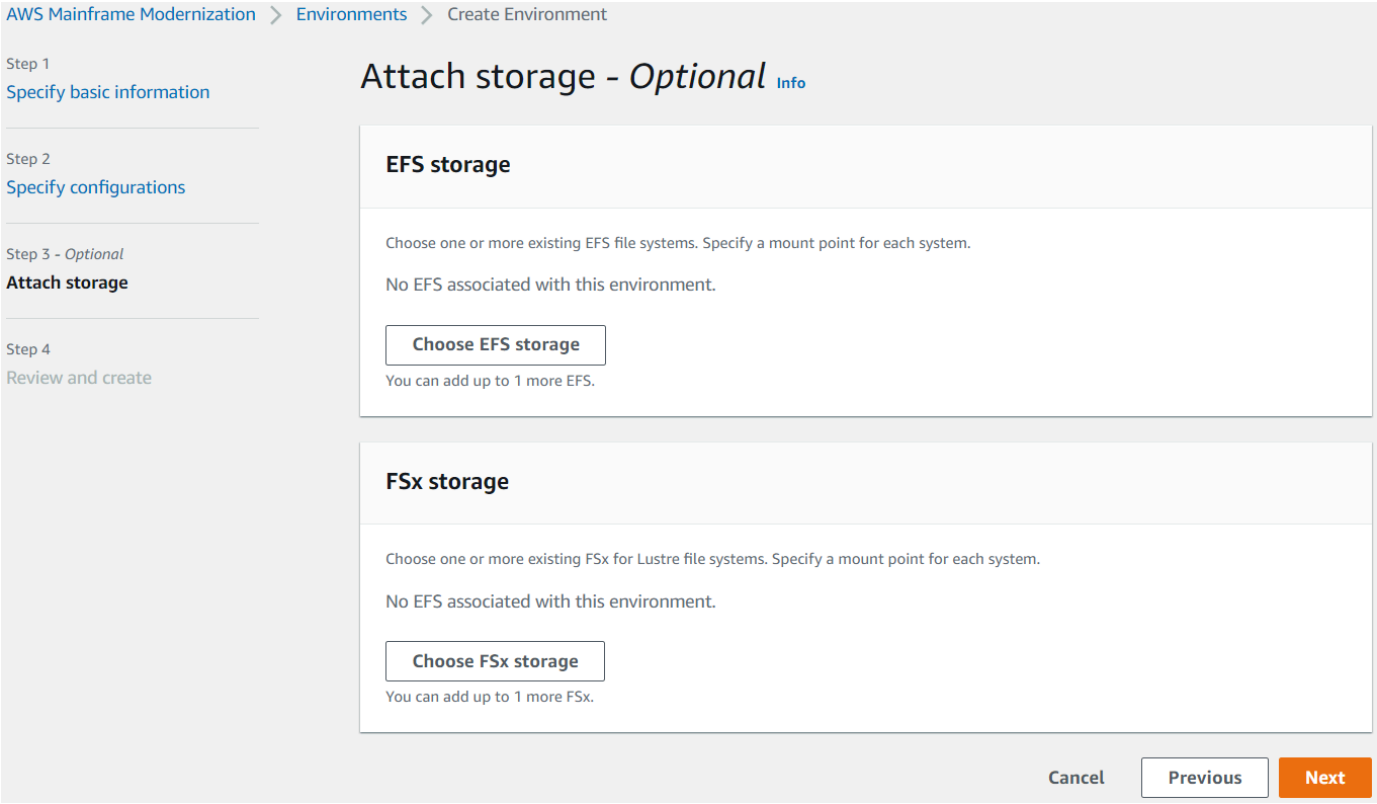
7. En Seguridad y red, realice los siguientes cambios:

- Seleccione Permitir que las aplicaciones implementadas en este entorno sean de acceso público. Esta opción asigna una dirección IP pública a la aplicación para que pueda acceder a ella desde el escritorio.
- Elija una VPC. Puede utilizar el valor Predeterminado.
- Seleccione dos subredes. Asegúrese de que las subredes permitan la asignación de direcciones IP públicas.
- Elija un grupo de seguridad. Puede utilizar el valor Predeterminado. Asegúrese de que el grupo de seguridad que elija permita el acceso desde una dirección IP al puerto que haya especificado en la propiedad `listener` de la definición de la aplicación. Para obtener más información, consulte Paso 2: Crear la definición de aplicación.



Si desea acceder a la aplicación desde fuera de la VPC que haya elegido, asegúrese de que las reglas de entrada de esa VPC estén configuradas correctamente. Para obtener más información, consulte Troubleshooting error: Cannot access an application URL.

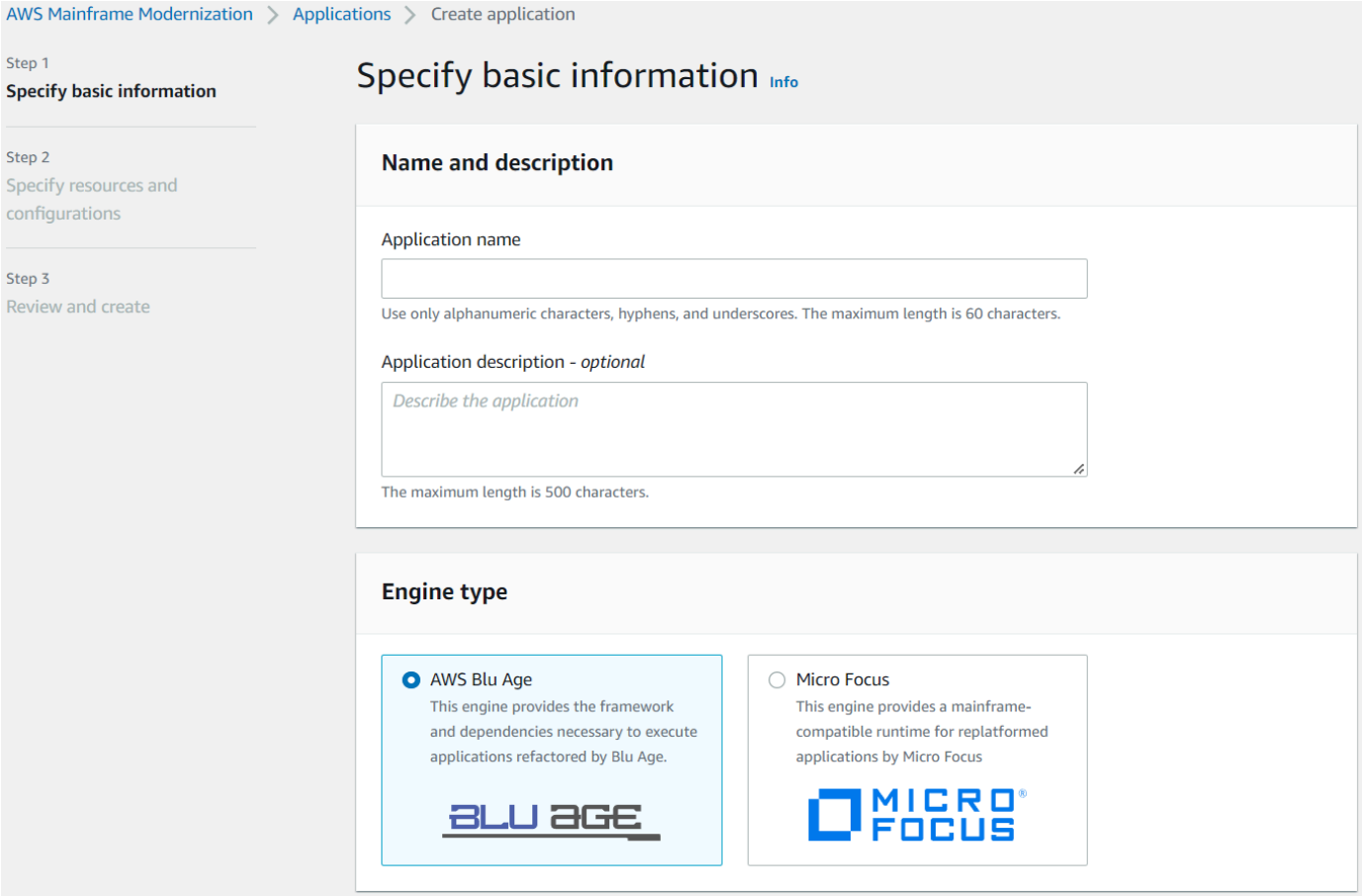
8. Elija Siguiente.
9. En Adjuntar almacenamiento: opcional, deje las selecciones predeterminadas y elija Siguiente.



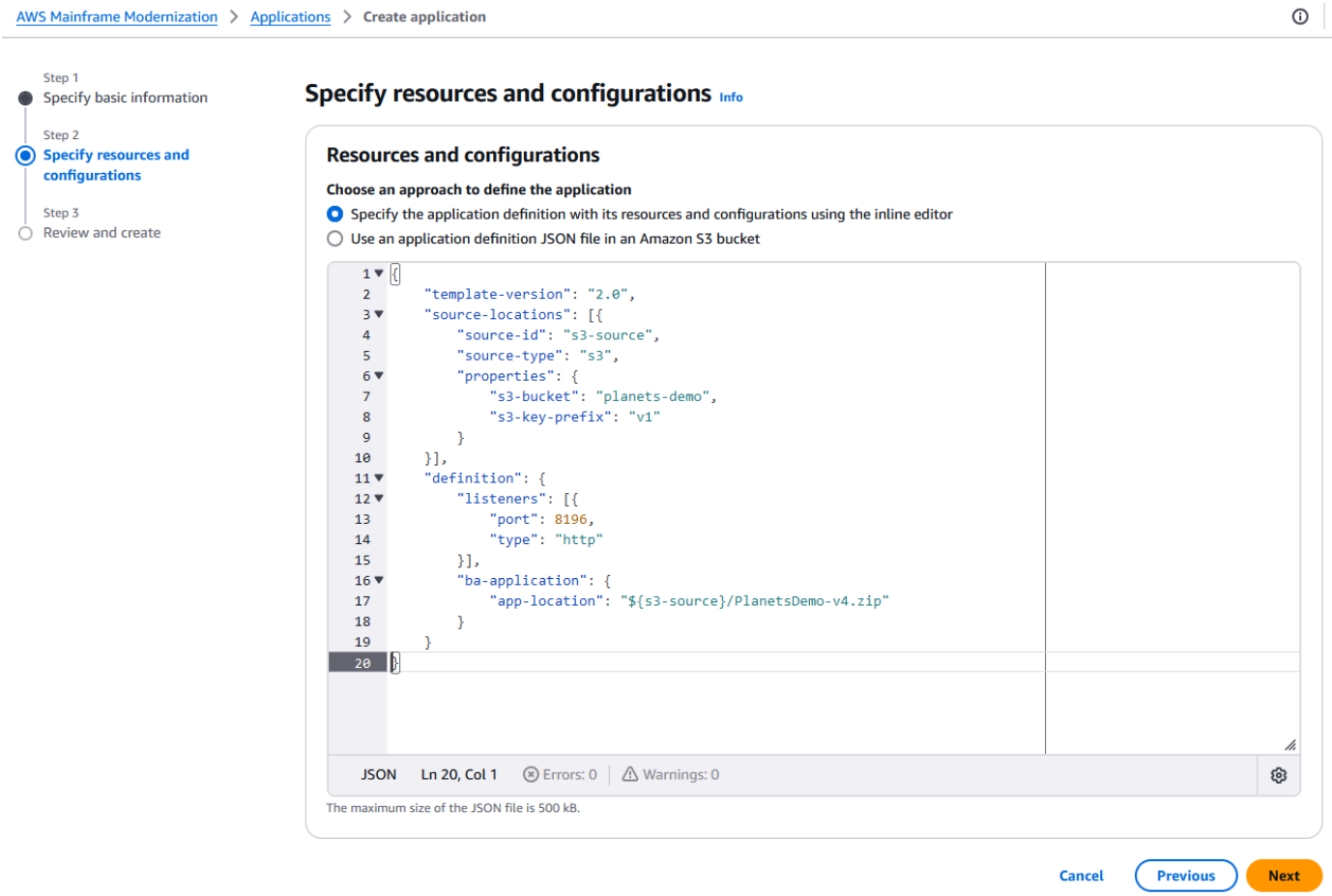
10. En Mantenimiento programado, elija Sin preferencia y, a continuación, elija Siguiente.
11. En Revisar y crear, revise la información y, a continuación, elija Crear entorno.

Paso 4: Crear una aplicación

1. Navegue hasta AWS Mainframe Modernization en AWS Management Console.
2. En el panel de navegación, elija Aplicaciones, y, a continuación, seleccione Crear aplicación. En la página Especificar información básica, introduzca un nombre y una descripción para la aplicación y, a continuación, asegúrese de que está seleccionado el motor AWS Blu Age. A continuación, elija Siguiente.



3. En la página Especificar recursos y configuraciones, copie y pegue la definición de aplicación actualizada en formato JSON que creó en the section called “Paso 2: Crear la definición de aplicación”.



4. En Revisar y crear, revise sus opciones y, a continuación, elija Crear aplicación.

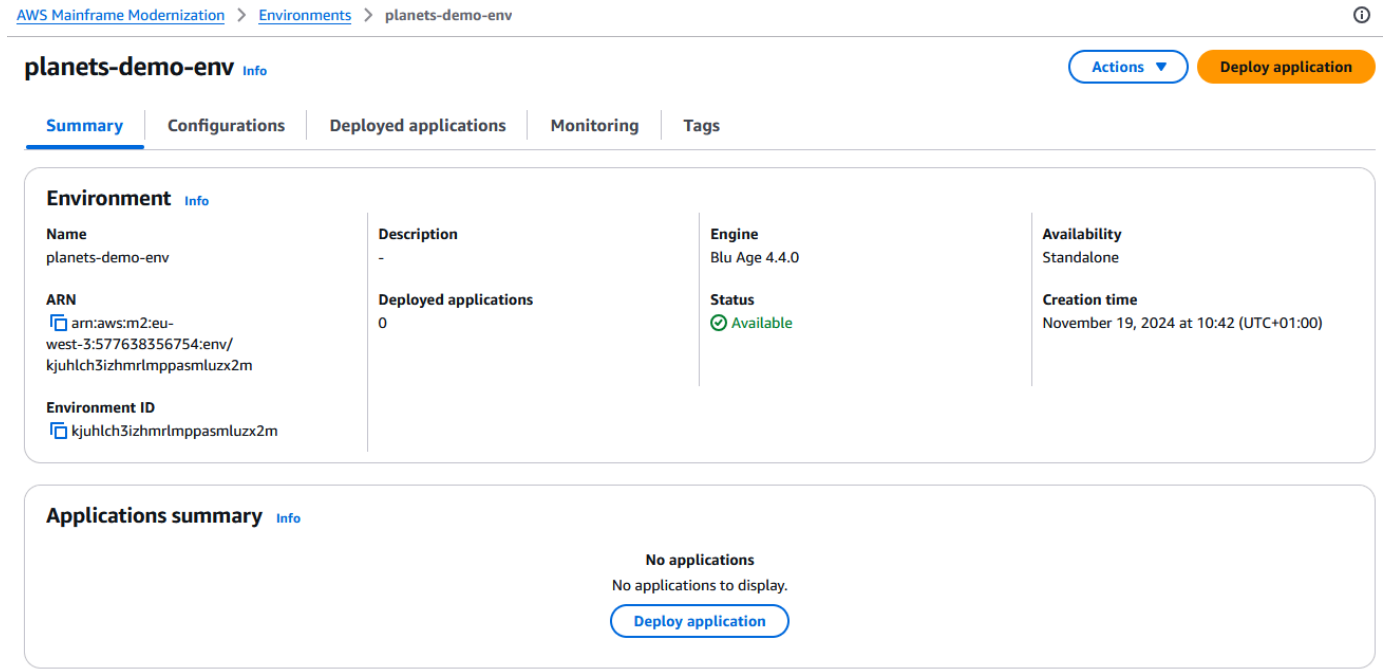
> **Note**
>
> Si se produce un error al crear la aplicación, compruebe la ruta S3 que ha introducido, ya que distingue entre mayúsculas y minúsculas.

Paso 5: Implementar una aplicación

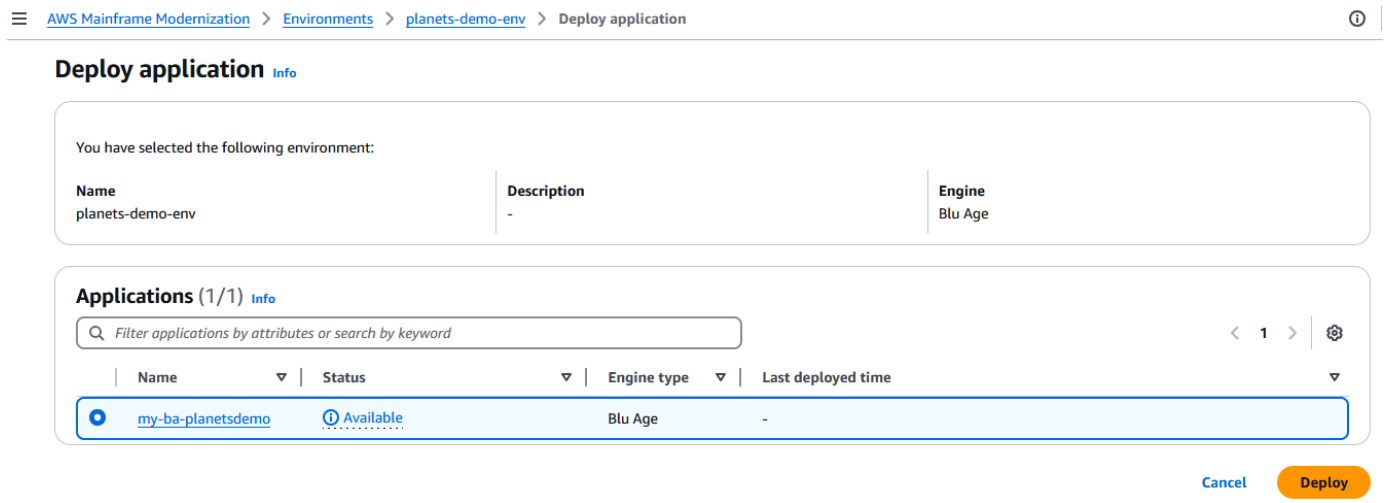
Una vez que haya creado correctamente el entorno de ejecución y la aplicación de AWS Mainframe Modernization y ambos estén en el estado Disponible, podrá implementar la aplicación en el entorno de ejecución. Para ello, siga los pasos que se describen a continuación:

1. Navegue hasta AWS Mainframe Modernization en la consola AWS de administración. En el panel de navegación, elija Entornos. Aparece la página de lista de entornos.

2. Elija el entorno de tiempo de ejecución creado anteriormente. Página de detalles del entorno.
3. Elija Implementar aplicación.



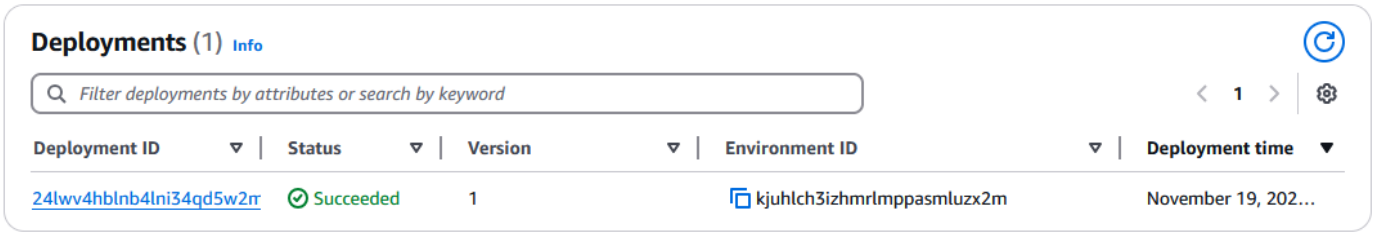
4. Elija la aplicación creada anteriormente y seleccione la versión en la que desee implementar la aplicación. A continuación, elija Implementar.



5. Espere hasta que la aplicación finalice su implementación. Verá un banner con el mensaje Application was deployed successfully.

Paso 6: Iniciar una aplicación

1. Navegue hasta AWS Mainframe Modernization en AWS Management Console y seleccione Aplicaciones.
2. Seleccione la aplicación y elija Implementaciones. El estado de la aplicación debe ser Finalizado con éxito.



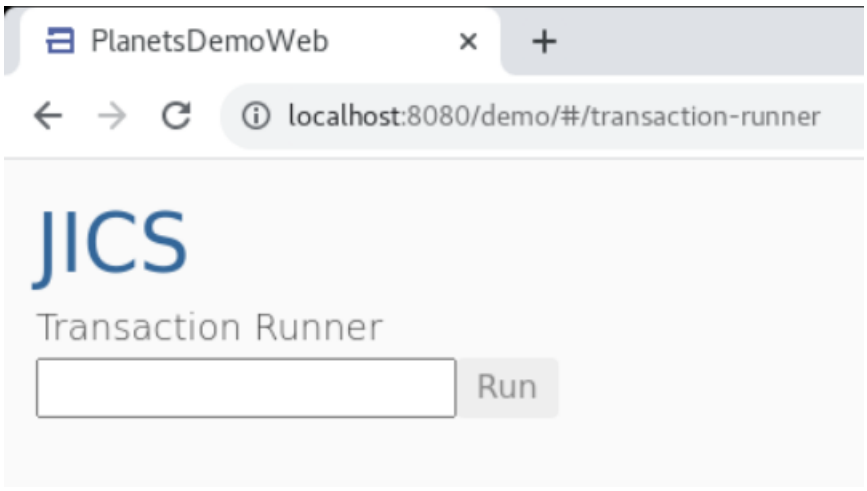
3. Elija Acciones y, a continuación, Iniciar aplicación.

Paso 7: Acceder a la aplicación

1. Espere hasta que la aplicación esté en estado En ejecución. Ver un banner con el mensaje Application was started successfully.
2. Copie el nombre de host DNS de la aplicación. Puede encontrarlo en la sección de Información de la aplicación de la aplicación.
3. En un navegador, navegue hasta `http://{hostname}:{portname}/PlanetsDemo-web-1.0.0/`, donde:

   - `hostname` es el nombre de host DNS copiado anteriormente.
   - `portname` es el puerto de Tomcat especificado en la definición de la aplicación que ha creado en el Paso 2: Crear la definición de aplicación.

   Aparece la pantalla de JICS.



Puede acceder a la aplicación, consulte Troubleshooting error: Cannot access an application URL.

> **Note**
>
> Si no se puede acceder a la aplicación y la regla de entrada del grupo de seguridad tiene seleccionada la opción “Mi IP” en el puerto 8196, especifique la regla que permita el tráfico desde LB i/p en el puerto 8196.

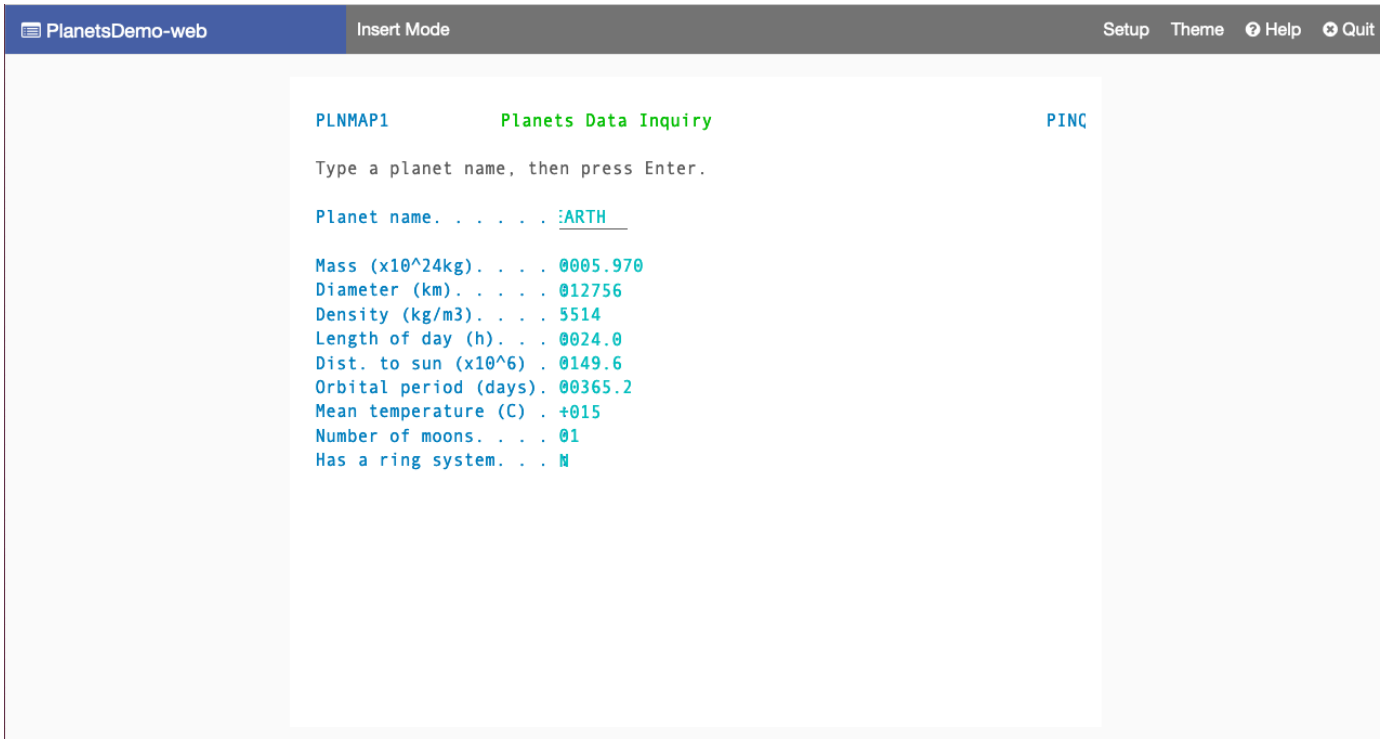
Paso 8: Probar la aplicación

En este paso, ejecutará una transacción en la aplicación migrada.

1. En la pantalla de JICS, introduzca `PINQ` en el campo de entrada y elija Ejecutar (o pulse Intro) para iniciar la transacción de la aplicación.

   Debería aparecer la pantalla de la aplicación de demostración.

2. Escriba el nombre de un planeta en el campo correspondiente y pulse Intro.



Deberían aparecer los detalles del planeta.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para evitar cargos adicionales. Para ello, complete los siguientes pasos.

- Si la aplicación de modernización del AWS mainframe sigue ejecutándose, deténgala.
- Elimine la aplicación. Para obtener más información, consulte Eliminar una AWS Mainframe Modernization aplicación.
- Eliminar un entorno de tiempo de ejecución Para obtener más información, consulte Elimine un entorno de ejecución AWS de modernización de mainframe.

Tutorial: Configurar el tiempo de ejecución gestionado para Rocket Software (anteriormente Micro Focus)

Puede implementar y ejecutar una aplicación en un entorno de tiempo de ejecución gestionado por AWS Mainframe Modernization con el motor de tiempo de ejecución de Rocket Software. Este tutorial muestra cómo implementar y ejecutar la aplicación de CardDemo ejemplo en un entorno de tiempo de ejecución gestionado por AWS Mainframe Modernization con el motor de tiempo de ejecución de Rocket Software. La aplicación CardDemo de ejemplo es una aplicación de tarjetas de crédito simplificada desarrollada para probar y mostrar la tecnología AWS y asociarla a los casos de uso de la modernización de mainframes.

En el tutorial, creará recursos en otros Servicios de AWS. Estos incluyen Amazon Simple Storage Service, Amazon Relational Database Service AWS Key Management Service y. AWS Secrets Manager

Temas

- Requisitos previos
- Paso 1: creación y carga de un bucket de Amazon S3
- Paso 2: creación y configuración de un usuario de base de datos
- Paso 3: creación y configuración de una AWS KMS key
- Paso 4: creación y configuración de un secreto de base de datos de AWS Secrets Manager
- Paso 5: Agrega el SSLMode al secreto
- Paso 6: Crear un entorno de ejecución

- Paso 7: Crea una aplicación
- Paso 8: implementar una aplicación
- Paso 9: Importar conjuntos de datos
- Paso 10: Iniciar una aplicación
- Paso 11: Conectarse a la CardDemo aplicación CICS
- Eliminar recursos
- Pasos a seguir a continuación

Requisitos previos

- Asegúrese de tener acceso a un emulador 3270 para usar la conexión CICS. Los emuladores 3270 gratuitos y de prueba están disponibles en sitios web de terceros. Como alternativa, puede iniciar una instancia de software Rocket de AWS Mainframe Modernization AppStream 2.0 y utilizar el emulador Rumba 3270 (no disponible de forma gratuita).

  Para obtener información acerca de AppStream 2.0, consulte. the section called “Tutorial: Cómo configurar la AppStream versión 2.0 para Enterprise Analyzer y Enterprise Developer”

  > **Note**
  >
  > Al crear la pila, elija la opción Enterprise Developer (ED) y no Enterprise Analyzer (EA).

- Descargue la aplicación CardDemo de ejemplo y descomprima el archivo descargado en cualquier directorio local. Este directorio contendrá un subdirectorio llamado `CardDemo_runtime`.
- Identifique una VPC en su cuenta en la que pueda definir los recursos creados en este tutorial. La VPC necesitará subredes al menos en dos zonas de disponibilidad. Para obtener más información acerca de Amazon VPC, consulte la Guía del usuario de Amazon VPC.

Paso 1: creación y carga de un bucket de Amazon S3

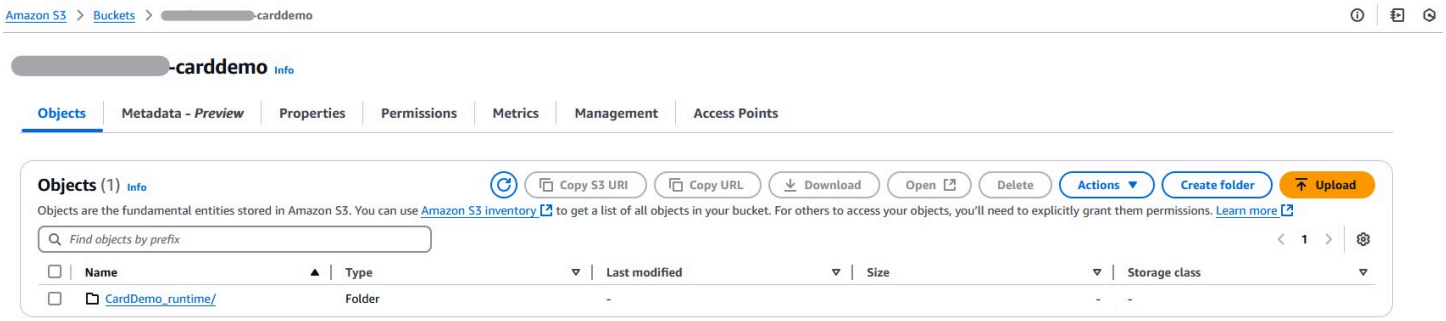
En este paso, debe crear un depósito de Amazon S3 y cargar CardDemo archivos en este depósito. Más adelante en este tutorial, utilizará estos archivos para implementar y ejecutar la aplicación de CardDemo muestra en un entorno de ejecución gestionado por Rocket Software AWS Mainframe Modernization.

> **Note**
>
> No tiene que crear un bucket de S3 nuevo, pero el bucket que elija debe estar en la misma región que los demás recursos que se utilizan en este tutorial.

Creación de un bucket de Amazon S3

1. Abra la consola de Amazon S3 y elija Crear bucket.
2. En Configuración general, elija la región de AWS en la que desee crear el entorno de ejecución gestionado por software Rocket Modernization de AWS Mainframe.
3. Introduzca un Nombre del bucket, por ejemplo, `yourname-aws-region-carddemo`. Conserve la configuración predeterminada y elija Crear bucket. Como alternativa, también puede copiar la configuración de un bucket de Amazon S3 existente y, a continuación, elegir Crear bucket.
4. Elija el bucket que acaba de crear y, a continuación, elija Cargar.
5. En la sección Cargar, elija Agregar carpeta y, a continuación, busque el directorio `CardDemo_runtime` desde su equipo local.
6. Elija Cargar para iniciar el proceso de carga. Los tiempos de carga varían en función de las velocidades de conexión.
7. Cuando se complete la carga, confirme que todos los archivos se han cargado correctamente y, a continuación, seleccione Cerrar.

Su bucket de Amazon S3 ahora contiene la carpeta `CardDemo_runtime`.



Para obtener más información sobre buckets de S3, consulte Creación, configuración y uso de buckets de Amazon S3.

Paso 2: creación y configuración de un usuario de base de datos

En este paso, va a crear una base de datos PostgreSQL en Amazon Relational Database Service (Amazon RDS). Para el tutorial, esta base de datos contiene los conjuntos de datos que la aplicación de CardDemo ejemplo utiliza para las tareas de los clientes relacionadas con las transacciones con tarjetas de crédito.

Para crear una nueva base de datos en Amazon RDS.

1. Abra la consola de Amazon RDS.
2. Elija la región de AWS donde desee crear la instancia de base de datos.
3. Seleccione Databases (Bases de datos) en el panel de navegación.
4. Elija Crear base de datos y, a continuación, Creación estándar.
5. En Tipo de motor, elija PostgreSQL.
6. Elija una Versión del motor de 15 o superior.

> **Note**
>
> Guarde la versión del motor, ya que la necesitará más adelante en este tutorial.

7. En Plantillas, elija Nivel gratuito.
8. Cambie el Identificador de instancias de bases de datos por algo significativo, por ejemplo, `MicroFocus-Tutorial`.
9. Absténgase de administrar las credenciales maestras en AWS Secrets Manager. En su lugar, introduzca una nueva contraseña maestra y confírmela.

> **Note**
>
> Guarde el nombre de usuario y la contraseña que utiliza para la base de datos. Los almacenará de manera segura en los siguientes pasos de este tutorial.

10. En Conectividad, elija la VPC en la que desee crear el entorno de tiempo de ejecución gestionado por AWS Mainframe Modernization.
11. Elija Creación de base de datos.

Para crear un grupo de parámetros personalizado en Amazon RDS

1. En el panel de navegación de la consola de Amazon RDS, elija Grupos de parámetros y, a continuación, Crear grupo de parámetros.
2. En la ventana Crear grupo de parámetros, en Familia de grupos de parámetros, seleccione la opción Postgres que coincida con la versión de su base de datos.

   > **Note**
   >
   > Algunas versiones de Postgres requieren un Tipo. Seleccione Grupo de parámetros de base de datos si es necesario. Introduzca un Nombre del grupo y Descripción para el grupo de parámetros.

3. Seleccione Crear.

Configuración del grupo de parámetros personalizado

1. Elija el grupo de parámetros recién creado.
2. Seleccione Acciones y, a continuación, Editar.
3. Filtre por `max_prepared_transactions` cambie el valor del parámetro a 100.
4. Elija Save changes (Guardar cambios).

Asociación del grupo de parámetros personalizado con la base de datos

1. En el panel de navegación de la consola de Amazon RDS, elija Bases de datos y, a continuación, la instancia de base de datos que desee modificar.
2. Elija Modify (Modificar). Aparece la página Modify DB instance (Modificar instancia de base de datos).

   > **Note**
   >
   > La opción Modificar no estará disponible hasta que la base de datos haya terminado de crearse y realizar copias de seguridad, lo que puede tardar varios minutos.

3. En la página Modificar la instancia de base de datos, vaya a Configuración adicional y cambie el Grupo de parámetros de base de datos al suyo. Si su grupo de parámetros no está disponible en la lista, compruebe si se creó con la versión de base de datos correcta.

4. Elija Continuar y consulte el resumen de las modificaciones.
5. Para aplicar los cambios inmediatamente, elija Aplicar inmediatamente.
6. Seleccione Modificar la instancia de base de datos para guardar los cambios.

Para obtener más información, consulte Trabajo con los grupos de parámetros.

> **Note**
>
> También puede utilizar una base de datos PostgreSQL de Amazon Aurora AWS con la modernización del mainframe, pero no hay ninguna opción de capa gratuita. Para obtener más información, consulte Uso de Amazon Aurora PostgreSQL.

Paso 3: creación y configuración de una AWS KMS key

Para almacenar de forma segura las credenciales de la instancia de Amazon RDS, cree primero una AWS KMS key.

Para crear un AWS KMS key

1. Abra la Consola de Key Management Service.
2. Elija Create Key (Crear clave).
3. Deje los valores predeterminados de Simétrica para el tipo de clave y Cifrado y descifrado para el uso de clave.
4. Elija Siguiente.
5. Proporcione a la clave un Alias, como `MicroFocus-Tutorial-RDS-Key`, y una descripción opcional.
6. Elija Siguiente.
7. Asigne un administrador clave marcando la casilla situada junto a su usuario o función.
8. Elija Siguiente.
9. Asigna el permiso de uso de claves marcando la casilla situada junto a tu usuario o rol.
10. Elija Siguiente.
11. En la pantalla de revisión, edite la Política de claves y, a continuación, introduzca lo siguiente dentro de la matriz de Instrucción existente:

```
{
     "Sid" : "Allow access for Mainframe Modernization Service",
     "Effect" : "Allow",
        "Principal" : {
           "Service" : "m2.amazonaws.com"
                 },
      "Action" : "kms:Decrypt",
      "Resource" : "*"
},
```

Esta política otorga a AWS Mainframe Modernization permisos de descifrado mediante esta política clave específica.

12. Elija Finalizar para crear la clave.

Para obtener más información, consulte Creación de claves en la Guía para AWS Key Management Service desarrolladores.

Paso 4: creación y configuración de un secreto de base de datos de AWS Secrets Manager

Ahora guarde las credenciales de la base de datos de forma segura con AWS Secrets Manager y AWS KMS key.

Para crear y configurar un secreto de AWS Secrets Manager base de datos

1. Abra la consola de Secrets Manager.
2. En el panel de navegación, elija Secretos.
3. En Secretos, elija Almacenar un secreto nuevo.
4. Establezca Tipo de secreto en Credenciales para la base de datos de Amazon RDS.
5. Introduzca las Credenciales que ha especificado al crear la base de datos.
6. En Clave de cifrado, seleccione la clave que ha creado en el paso 3.
7. En la sección Base de datos, seleccione la base de datos que ha creado para este tutorial y, a continuación, elija Siguiente.
8. En Nombre del secreto, introduzca un nombre, como, por ejemplo, `MicroFocus-Tutorial-RDS-Secret` y una descripción opcional.

9. En la sección Permisos de recursos, elija Editar permisos y, reemplace el contenido por la siguiente política:

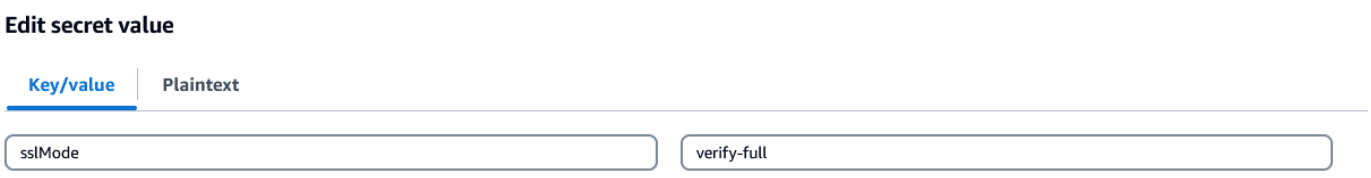
```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect" : "Allow",
            "Principal" : {
                "Service" : "m2.amazonaws.com"
            },
            "Action" : "secretsmanager:GetSecretValue",
            "Resource" : "*"
        }
    ]
}
```

10. Seleccione Save.
11. Elija Siguiente para las pantallas siguientes y, a continuación, Almacenar.

Paso 5: Agrega el SSLMode al secreto

Para añadir el SSLMode al secreto

1. Actualice la lista de secretos para ver el nuevo secreto.
2. Elige el secreto recién creado en el paso 4 y anótalo `Secret ARN` porque lo necesitarás más adelante en el tutorial.
3. En la pestaña Información general del secreto, elija Recuperar valor del secreto.
4. Elija Editar y, a continuación, Agregar fila.
5. Agregue una Clave para `sslMode` con un Valor de `verify-full`:

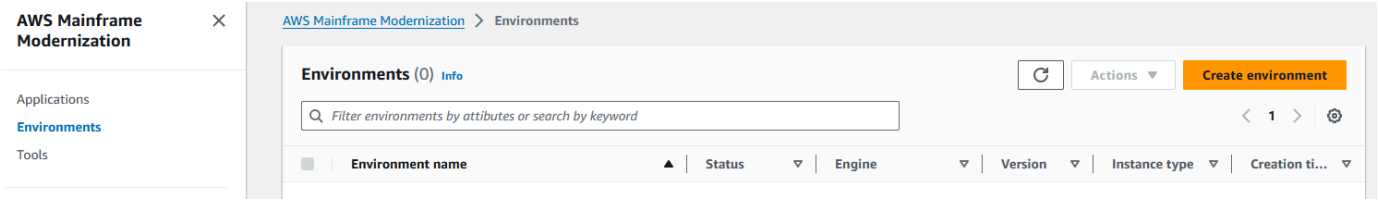


6. Seleccione Save.

Paso 6: Crear un entorno de ejecución

Para crear un entorno de tiempo de ejecución

1. Abra la consola de AWS Mainframe Modernization.
2. En el panel de navegación, elija Entornos. A continuación, elija Crear entorno.



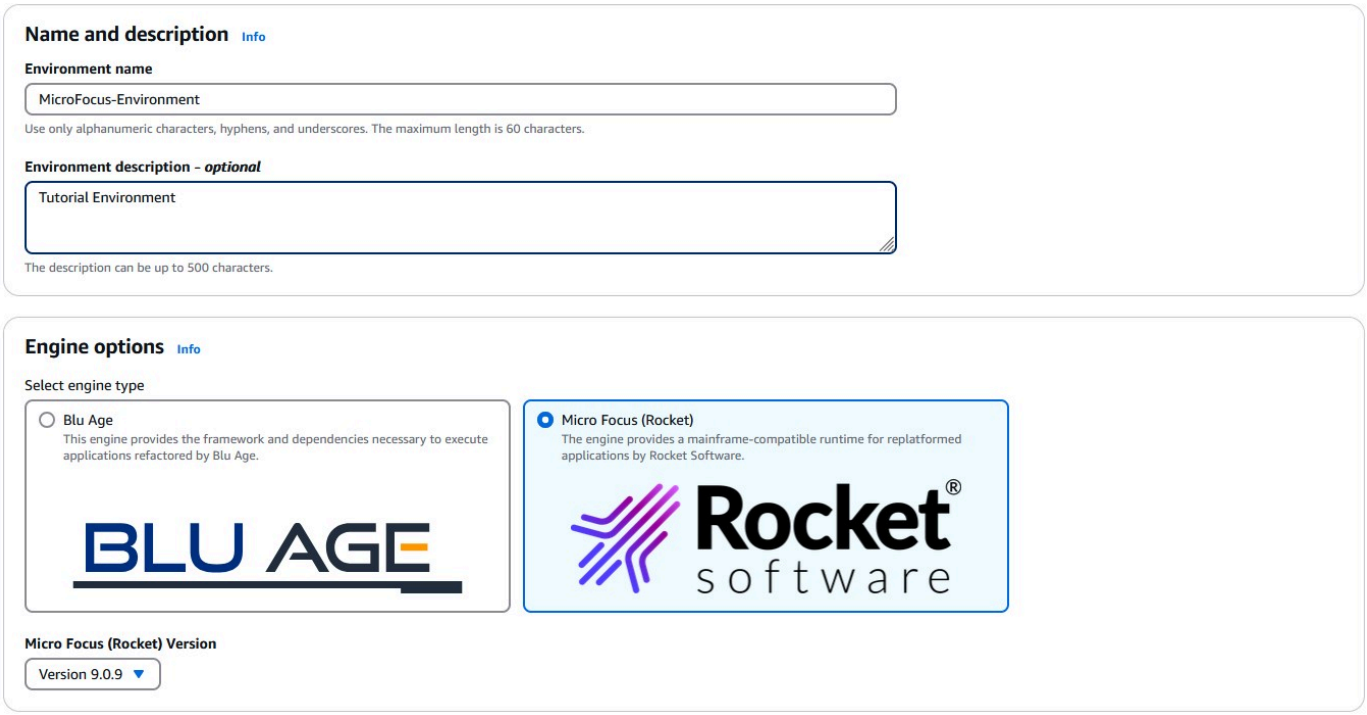
3. En Especificar información básica,

   a. Introduzca `MicroFocus-Environment` para el nombre del entorno.

   b. En las opciones del motor, asegúrese de seleccionar Micro Focus (Rocket).

   c. Elija la versión más reciente de Micro Focus (Rocket).

   d. Elija Siguiente.



4. Configure el entorno

   a. En Disponibilidad, elija Clúster de alta disponibilidad.

b. En Recursos, elija M2.c5.large o M2.m5.large para el tipo de instancia, así como el número de instancias que desee. Especifique hasta dos instancias.

c. En Seguridad y red, elija Permitir que las aplicaciones implementadas en este entorno sean de acceso público y elija al menos dos subredes públicas.

d. Elija Siguiente.

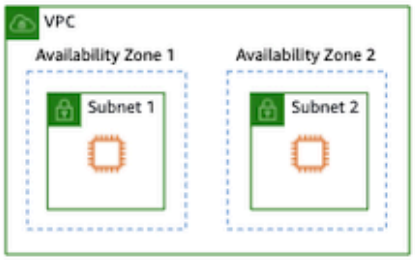
Specify configurations Info

Availability Info

Choose the availability pattern for your environment.

- Standalone runtime environment
  Sets up a single instance in a single availability zone. Does not guarantee high availability but costs less.

- High availability cluster
  Sets up redundant instances across two availability zones. Enables higher availability but costs more.



Resources

Instance type
Choose the instance type for your high availability cluster.

M2.m5.large

Desired capacity
Specify the desired number of instances.

2

Security and network

Allow applications deployed to this environment to be publicly accessible.

Virtual Private Cloud (VPC)
Choose the VPC where you want to create the environment.

Default vpc-15

Subnets
Choose one or more subnets for a high availability setup.

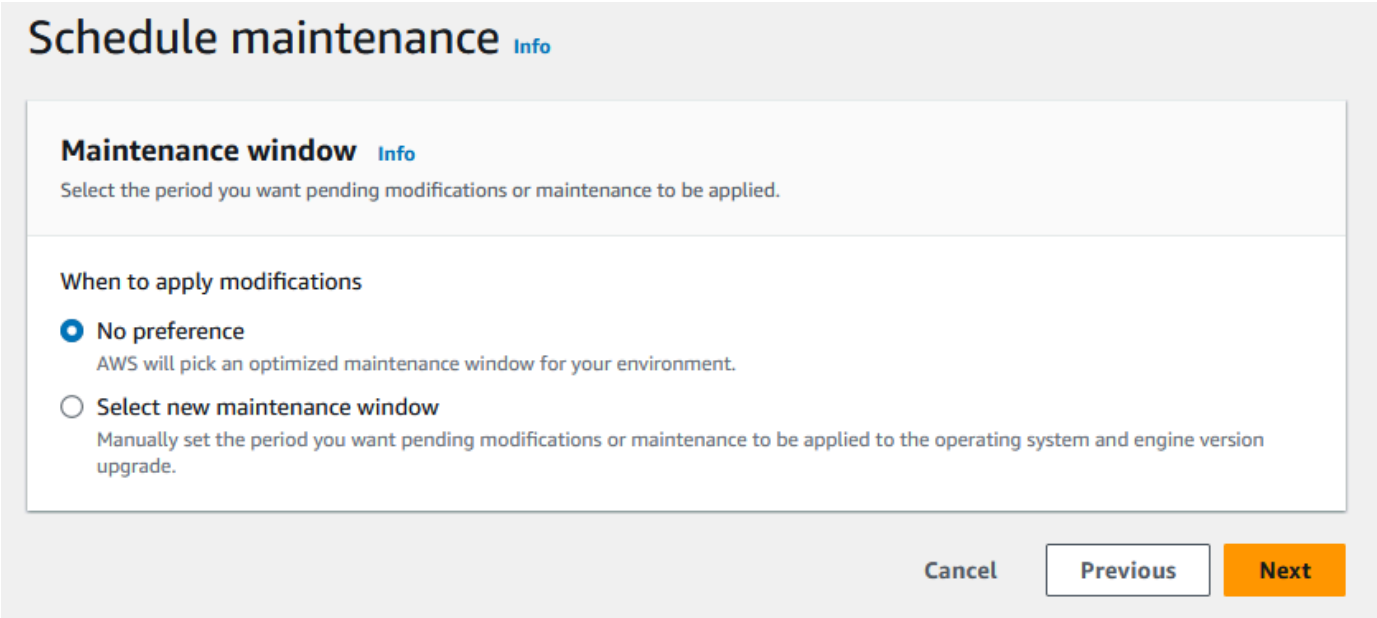
Choose subnets

subnet-56f1a | us-west-2a

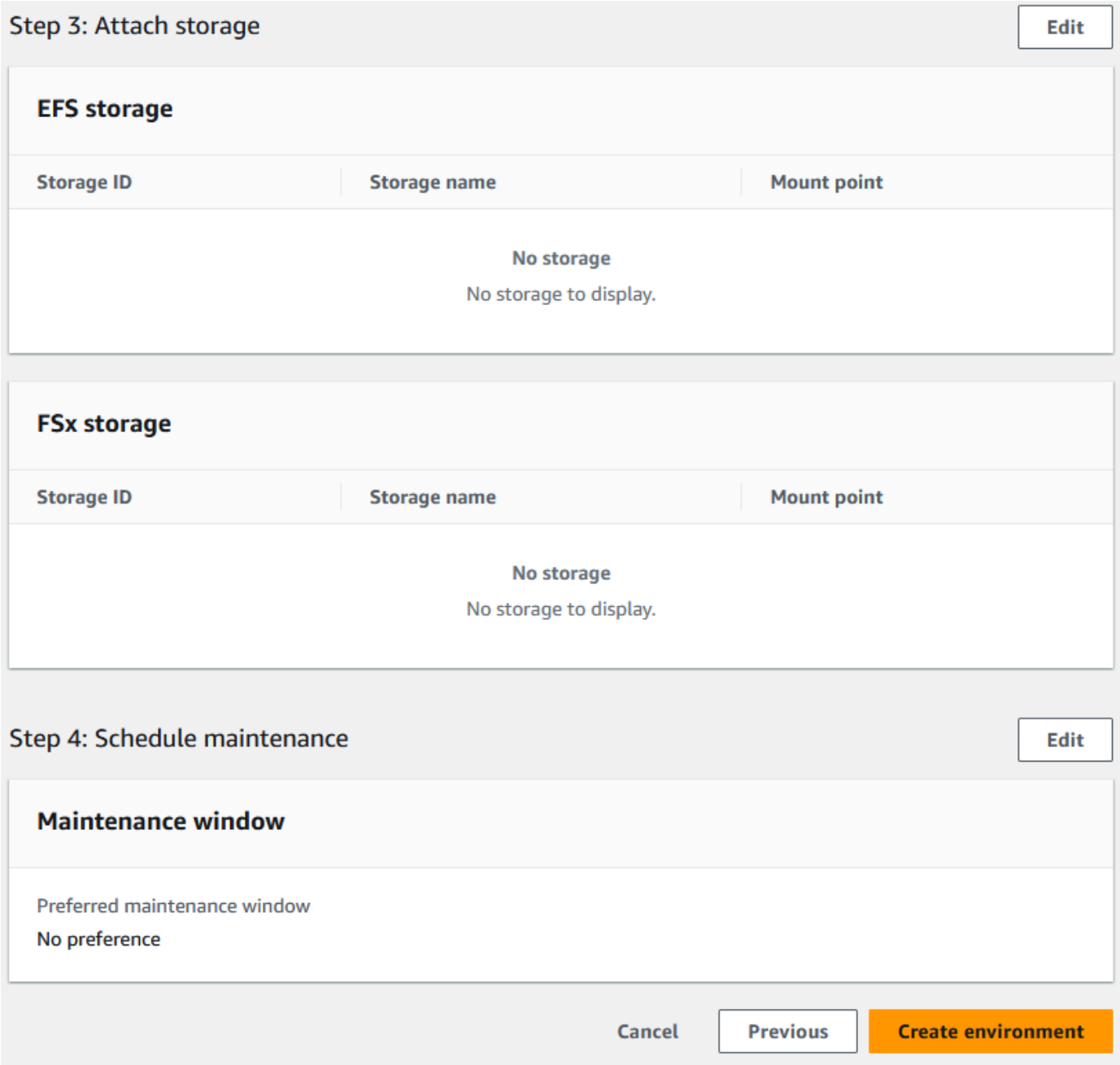
subnet-6685 | us-west-2b

Security groups
Choose one or more security groups for the chosen VPC.

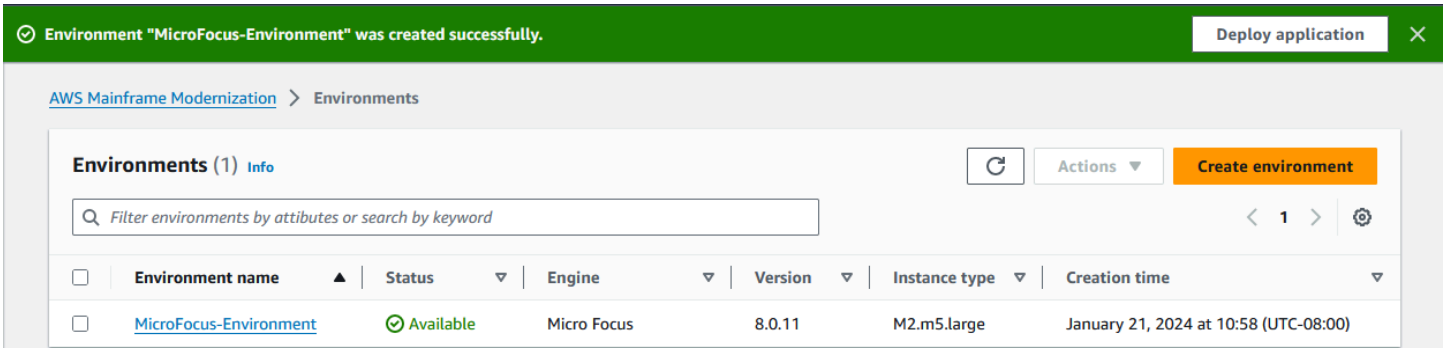
5. En la página Adjuntar almacenamiento, elija Paso siguiente.
6. En Mantenimiento programado, elija Sin preferencia y, a continuación, Siguiente.



7. En la página Revisar y crear, revise todas las configuraciones que ha proporcionado para el entorno de tiempo de ejecución y, a continuación, seleccione Crear entorno.



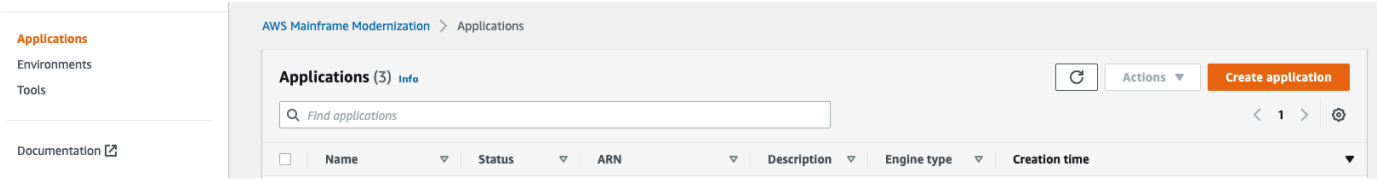
Cuando haya creado su entorno, aparecerá un banner que dice `Environment` *`name`* `was created successfully`y el campo Estado cambiará a Disponible. El proceso de creación del entorno tarda varios minutos, pero puede continuar con los siguientes pasos mientras se ejecuta.



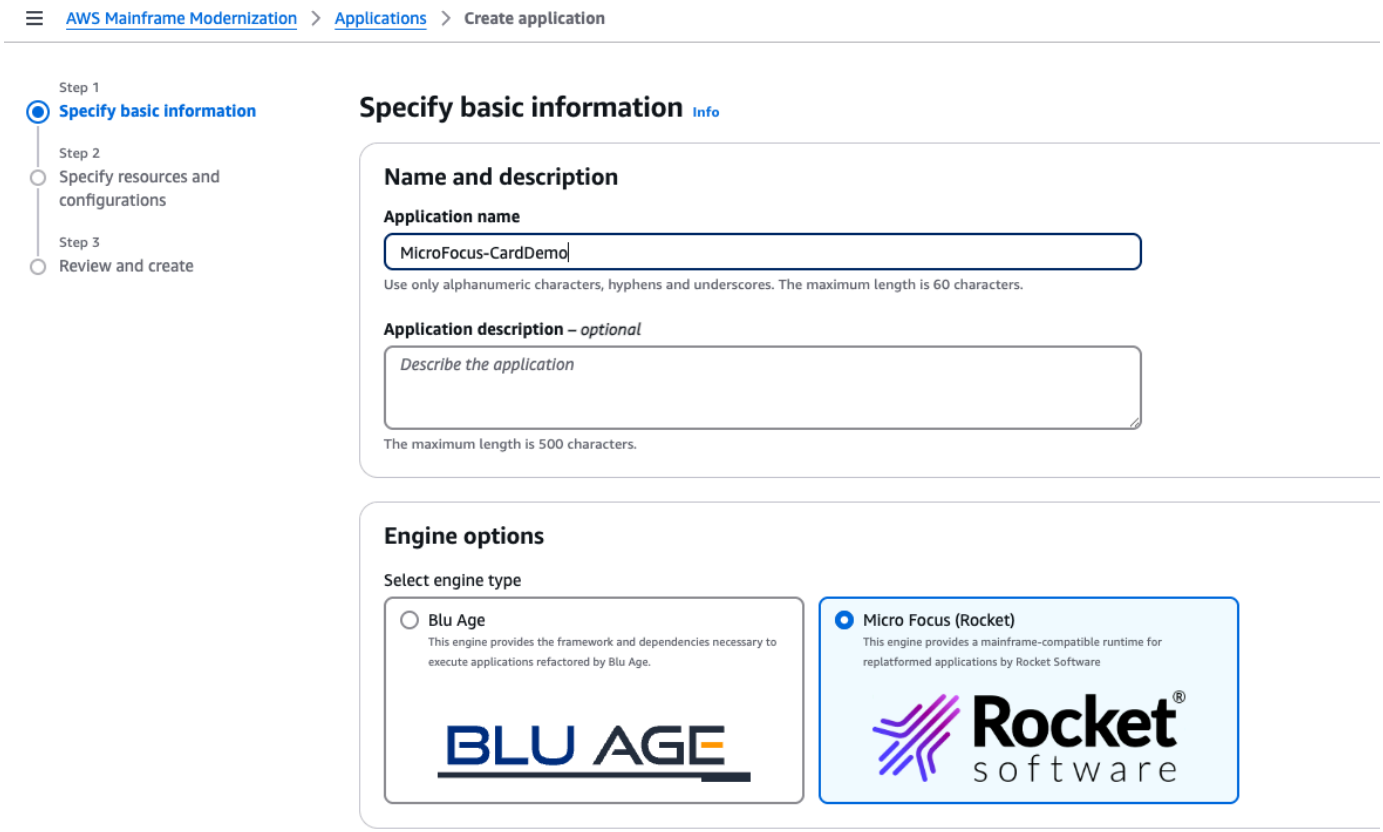
Paso 7: Crea una aplicación

Para crear una aplicación

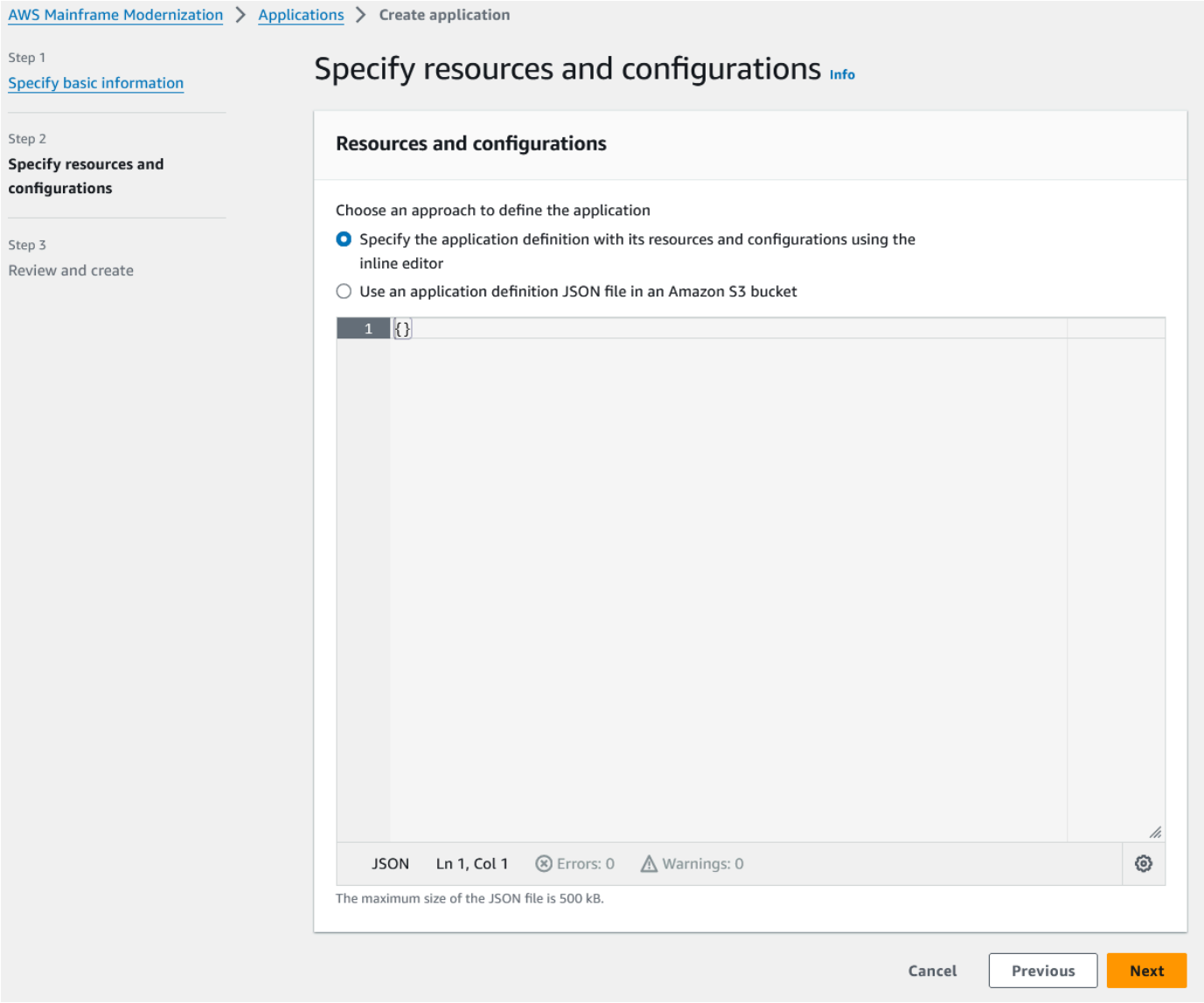
1. En el panel de navegación, elija Aplicaciones. Elija Crear aplicación.



2. En la página Crear aplicación, en Especificar información básica, introduzca el nombre `MicroFocus-CardDemo` de la aplicación y, en Tipo de motor, asegúrese de seleccionar Micro Focus (Rocket). A continuación, elija Siguiente.



3. En Especificar recursos y configuraciones, elija la opción para especificar la definición de la aplicación con sus recursos y configuraciones mediante el editor integrado.



Introduzca la siguiente definición de aplicación en el editor:

```
{
  "template-version": "2.0",
  "source-locations": [
    {
      "source-id": "s3-source",
      "source-type": "s3",
      "properties": {
        "s3-bucket": "yourname-aws-region-carddemo",
        "s3-key-prefix": "CardDemo_runtime"
      }
    }
```

```
  ],
  "definition": {
    "listeners": [
      {
        "port": 6000,
        "type": "tn3270"
      }
    ],
    "dataset-location": {
      "db-locations": [
        {
          "name": "Database1",
          "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx"
        }
      ]
    },
    "batch-settings": {
      "initiators": [
        {
          "classes": [
               "A",
               "B"
            ],
          "description": "initiator_AB...."
        },
        {
          "classes": [
                "C",
                "D"
             ],
          "description": "initiator_CD...."
        }
      ],
      "jcl-file-location": "${s3-source}/catalog/jcl"
    },
    "cics-settings": {
      "binary-file-location": "${s3-source}/loadlib",
      "csd-file-location": "${s3-source}/rdef",
      "system-initialization-table": "CARDSIT"
    },
    "xa-resources": [
      {
```

```
        "name": "XASQL",
        "secret-manager-arn":
 "arn:aws:secretsmanager:Region:123456789012:secret:MicroFocus-Tutorial-RDS-Secret-
xxxxxx",
        "module": "${s3-source}/xa/ESPGSQLXA64.so"
      }
    ]
  }
}
```

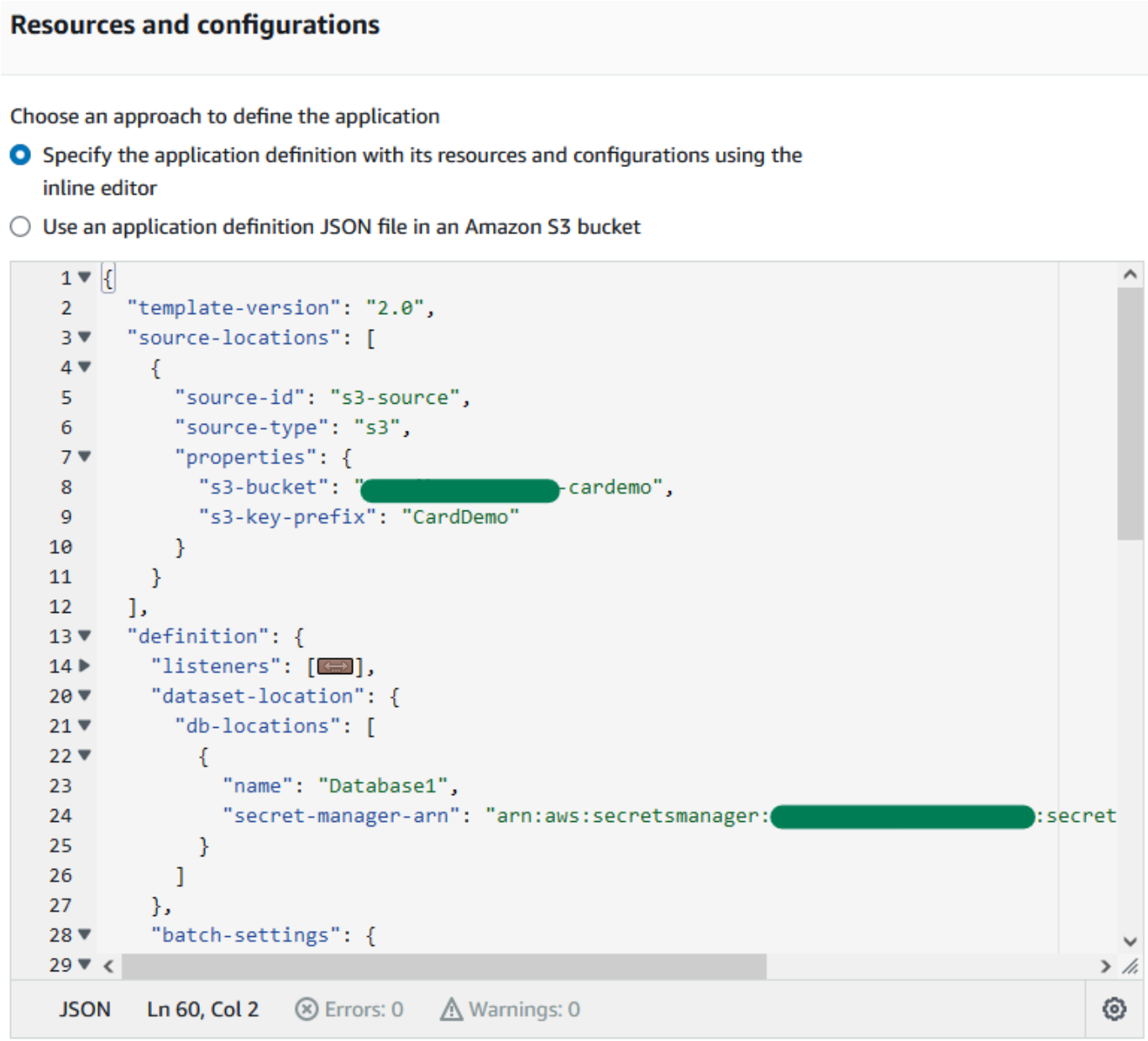
> **Note**
>
> Este archivo está sujeto a cambios.

4. Edite el JSON de la aplicación en el objeto properties de source-locations de la siguiente manera:

   a. Reemplace el valor de `s3_bucket` por el nombre del bucket de Amazon S3 que ha creado en el Paso 1.

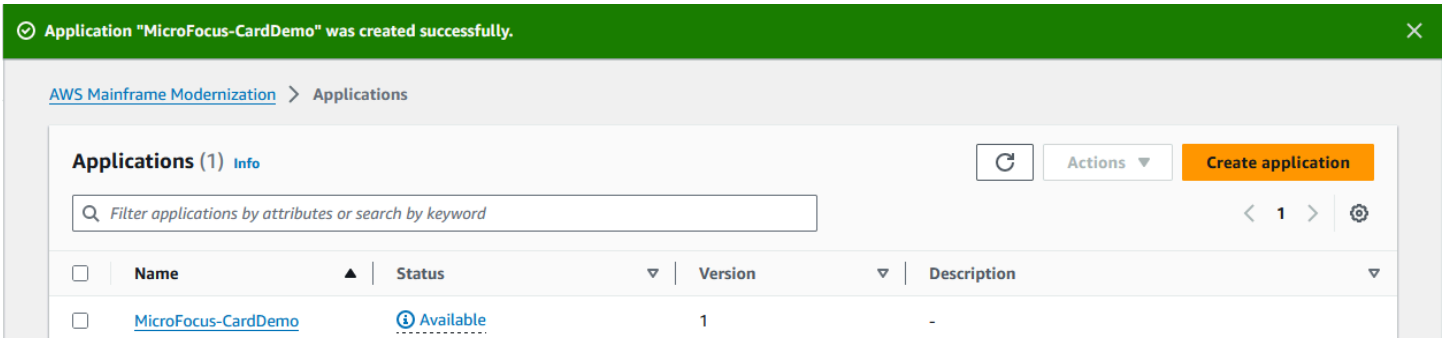
   b. Sustituya el valor for por `s3-key-prefix` la carpeta (key prefix) en la que cargó los archivos de CardDemo ejemplo. Si ha cargado el directorio `CardDemo` en un bucket de Amazon S3, no es necesario cambiar `s3-key-prefix`.

   c. Reemplace ambos valores de `secret-manager-arn` por el ARN del secreto de base de datos que ha creado en el Paso 4.



Para obtener más información sobre la definición de aplicaciones, consulte Definición de la aplicación Rocket Software (anteriormente Micro Focus).

5. Elija Siguiente para continuar.
6. En la página Revisar y crear, revise la información que ha proporcionado y, a continuación, elija Crear aplicación.

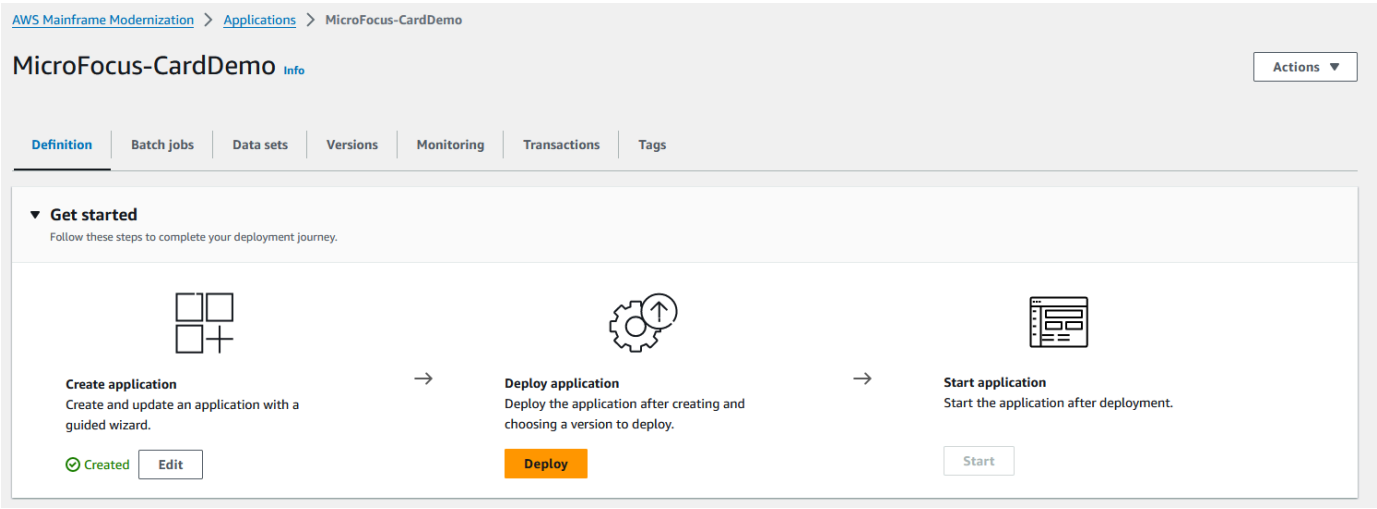


Cuando haya creado su entorno, aparecerá un banner que indica `Application` *`name`* `was created successfully`. El campo Estado cambia a Disponible.

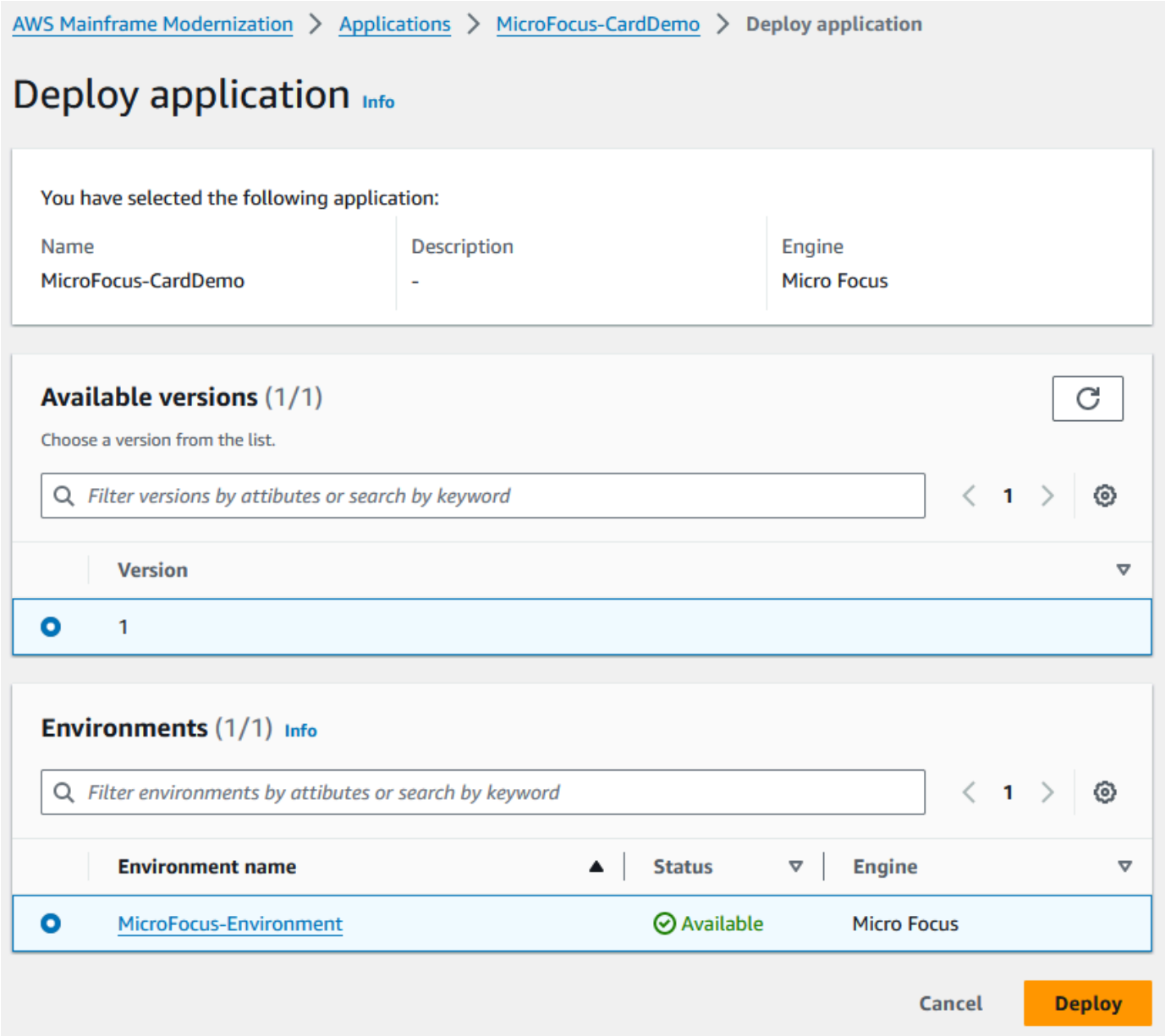
Paso 8: implementar una aplicación

Para implementar una aplicación

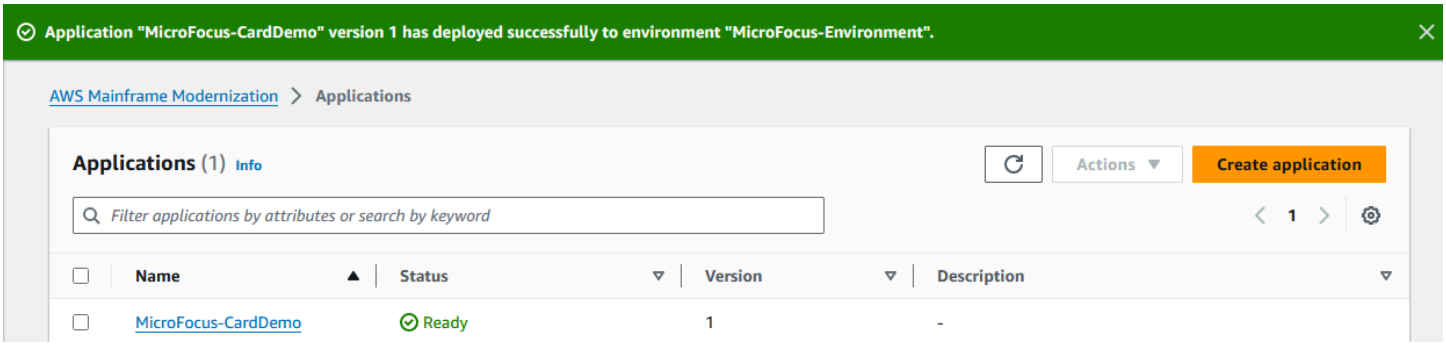
1. En el panel de navegación, elija Aplicaciones y, a continuación, `MicroFocus-CardDemo`.
2. En Implementar aplicación, elija Implementar.



3. Elija la versión más reciente de la aplicación y el entorno que ha creado anteriormente y, a continuación, Implementar.



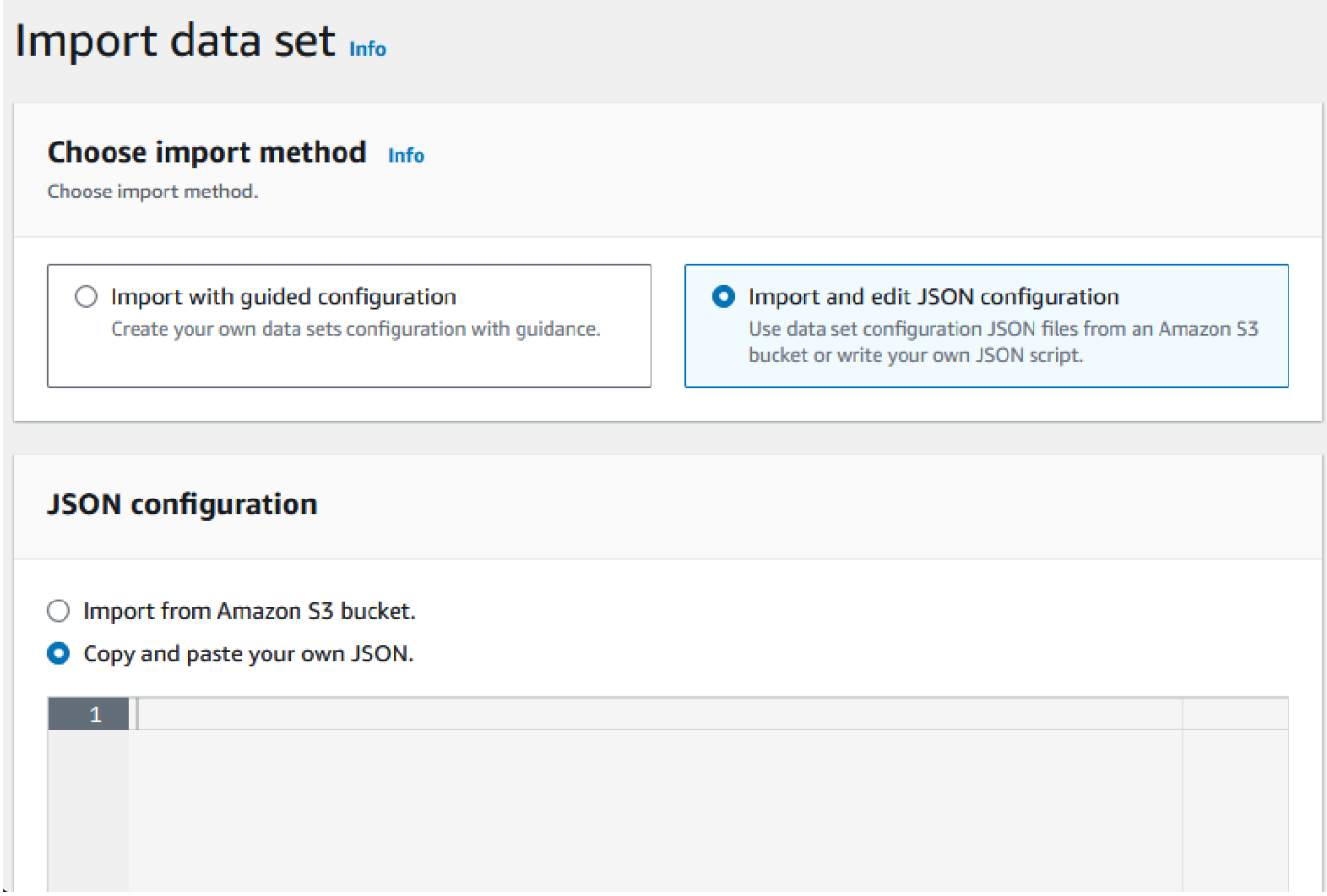
Cuando la CardDemo aplicación se implementa correctamente, el estado cambia a Listo.



Paso 9: Importar conjuntos de datos

Importación de conjuntos de datos

1. En el panel de navegación, elija Aplicaciones y, a continuación, elija la aplicación.
2. Elija la pestaña Conjuntos de datos. A continuación, elija Import.
3. Elija Importación y edición de la configuración de JSON y, a continuación, Copiar y pegar su propio JSON.



4. Copie y pegue el siguiente JSON, pero no elija Enviar todavía. Este JSON contiene todos los conjuntos de datos necesarios para la aplicación de demostración, pero necesita los detalles del bucket de Amazon S3.

```
{
    "dataSets": [
        {
            "dataSet": {
                "storageType": "Database",
```

```
                "datasetName": "AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 300,
                    "max": 300
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 11,
                            "offset": 16
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
```

```
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 150,
                    "max": 150
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
```

```
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 9,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 500,
                    "max": 500
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CUSTDATA.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.CARDXREF.VSAM.AIX.PATH",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
```

```
                        "primaryKey": {
                            "length": 11,
                            "offset": 25
                        }
                    }
                },
                "recordLength": {
                    "min": 50,
                    "max": 50
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.CARDXREF.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
                "datasetName": "AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 16,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 350,
                    "max": 350
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT"
            }
        },
        {
            "dataSet": {
                "storageType": "Database",
```

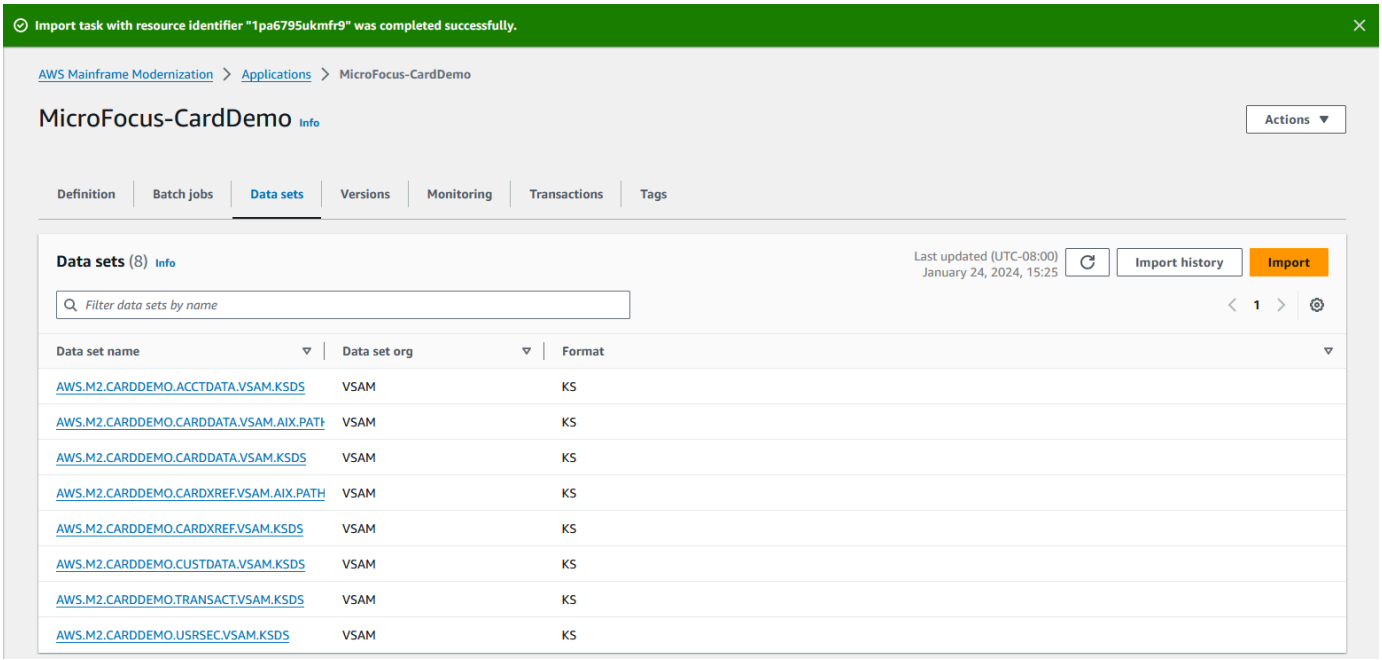
```
                "datasetName": "AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS",
                "relativePath": "DATA",
                "datasetOrg": {
                    "vsam": {
                        "format": "KS",
                        "encoding": "A",
                        "primaryKey": {
                            "length": 8,
                            "offset": 0
                        }
                    }
                },
                "recordLength": {
                    "min": 80,
                    "max": 80
                }
            },
            "externalLocation": {
                "s3Location": "s3://<s3-bucket-name>/CardDemo_runtime/catalog/data/
AWS.M2.CARDDEMO.USRSEC.VSAM.KSDS.DAT"
            }
        }
    ]
}
```

5. Sustituya cada aparición de `<s3-bucket-name>` (hay ocho) por el nombre del bucket de Amazon S3 que contiene la CardDemo carpeta, por ejemplo,`your-name-aws-region-carddemo`.

> **Note**
>
> Para copiar el URI de Amazon S3 de la carpeta de Amazon S3, seleccione la carpeta y, a continuación, elija Copiar el URI de Amazon S3.

6. Seleccione Enviar.

   Cuando finalice la importación, aparecerá un banner con el siguiente mensaje: `Import task with resource identifier` *`name`* `was completed successfully.` Se muestra una lista de los conjuntos de datos importados.

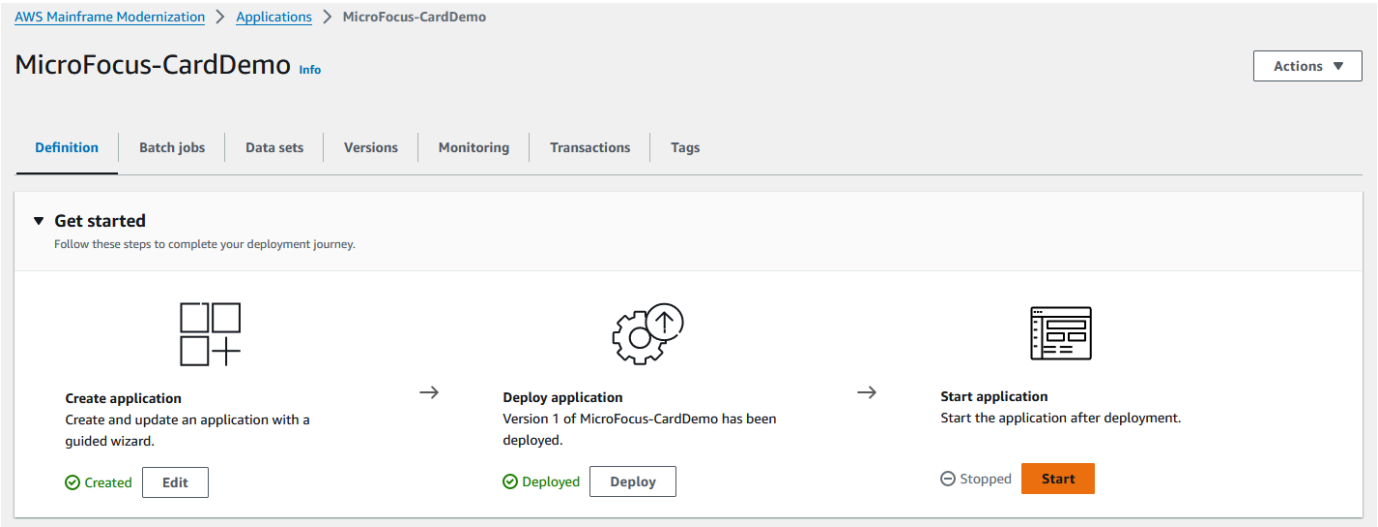


Para ver el estado de todas las importaciones de conjuntos de datos, también puede seleccionar Historial de importación en la pestaña Conjuntos de datos.

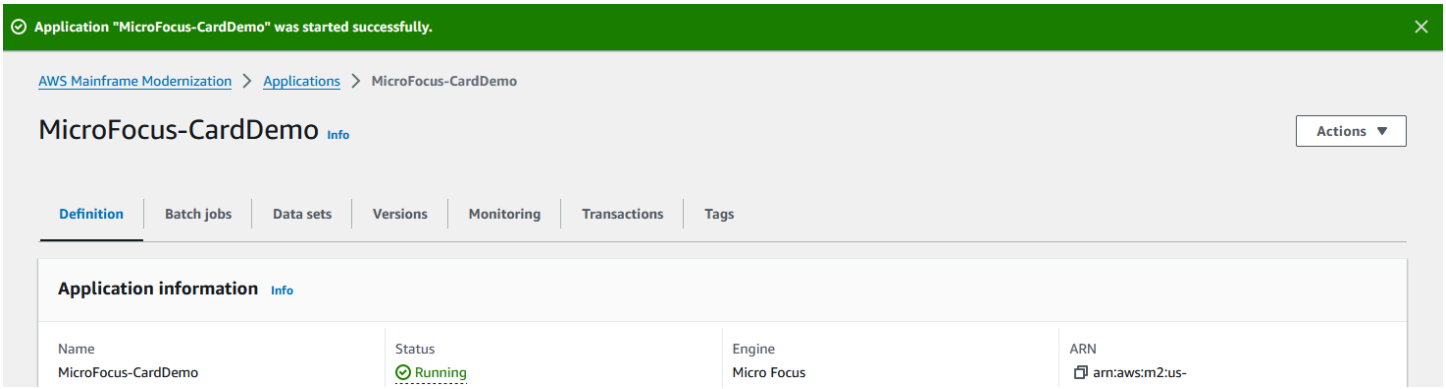
Paso 10: Iniciar una aplicación

Inicio de una aplicación

1. En el panel de navegación, elija Aplicaciones y, a continuación, elija la aplicación.
2. Elija Iniciar aplicación.



Cuando la CardDemo aplicación comienza a ejecutarse correctamente, aparece un banner con el siguiente mensaje:`Application` *`name`* `was started successfully`. El campo Estado cambia a En ejecución.



Paso 11: Conectarse a la CardDemo aplicación CICS

Antes de conectarse, asegúrese de que la VPC y el grupo de seguridad que ha especificado para la aplicación son los mismos que los que ha aplicado a la interfaz de red desde la que se conectará.

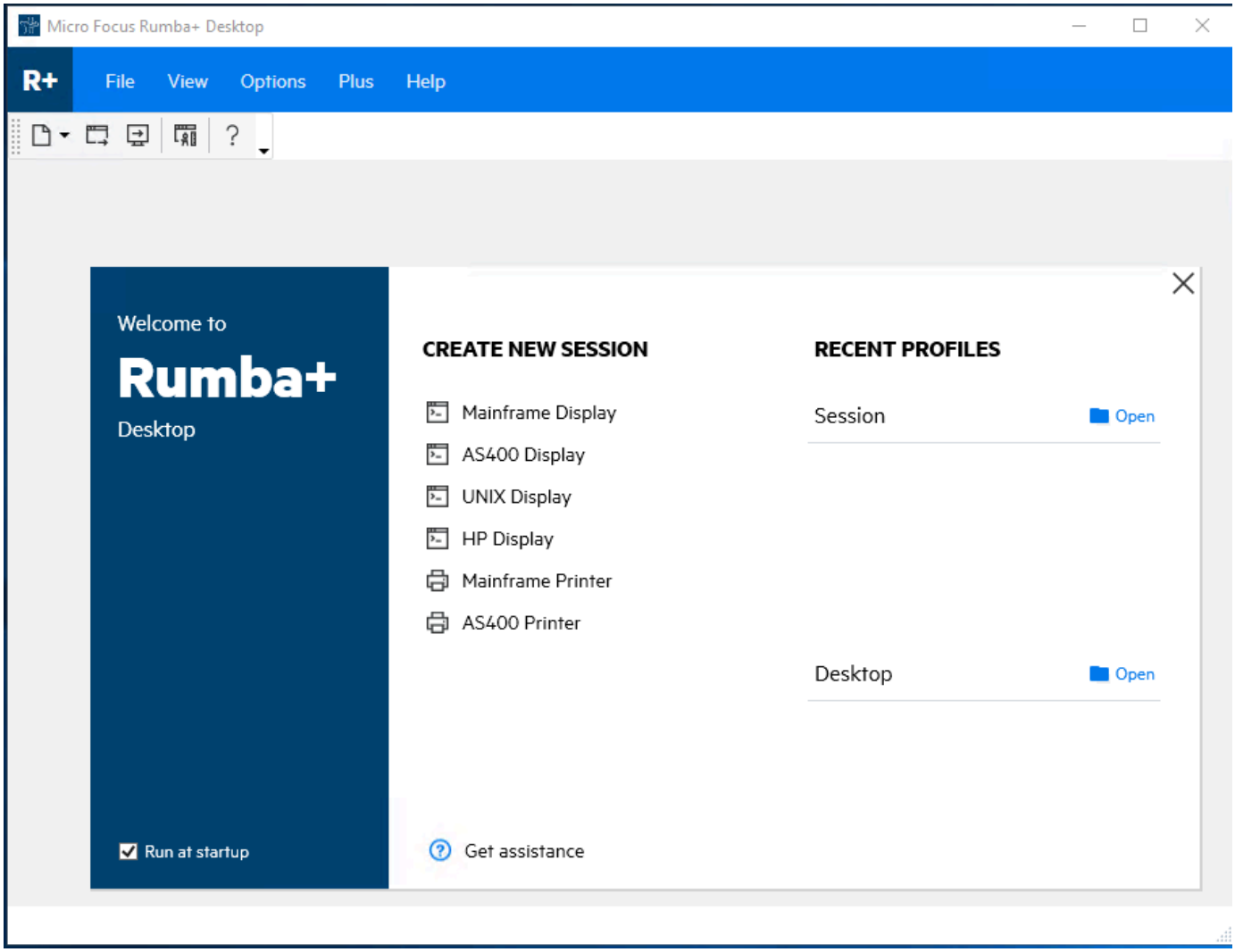
Para configurar la conexión TN327 0, también necesita el nombre de host DNS y el puerto de la aplicación.

Configuración y conexión de una aplicación al mainframe mediante un emulador de terminal

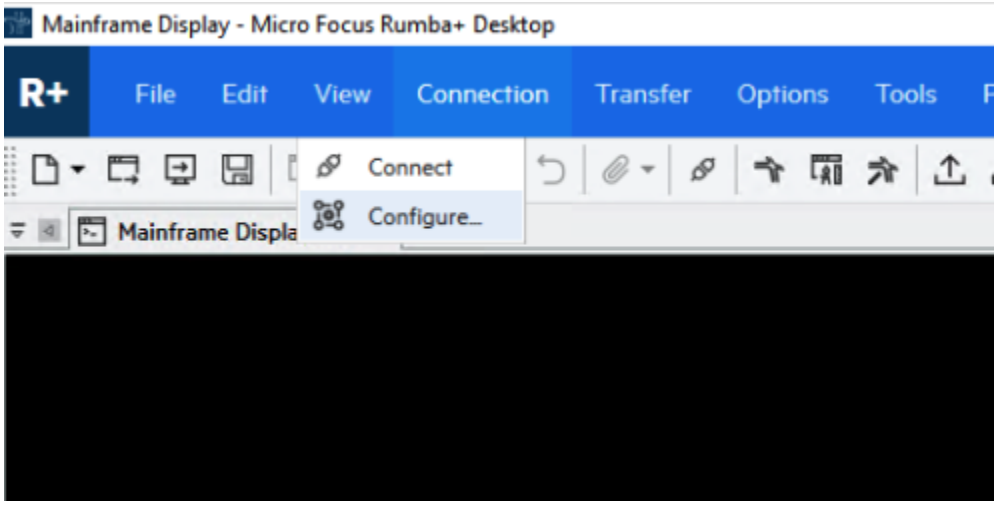
1. Abra la consola de modernización del AWS mainframe, seleccione Aplicaciones y, a continuación, elija. `MicroFocus-CardDemo`
2. Elija el icono de copiar para copiar el Nombre de host DNS. Asegúrese también de anotar el número de Puertos.
3. Inicie un emulador de terminal. En este tutorial se utiliza Micro Focus Rumba+.

   > **Note**
   >
   > Los pasos de configuración varían según el emulador.

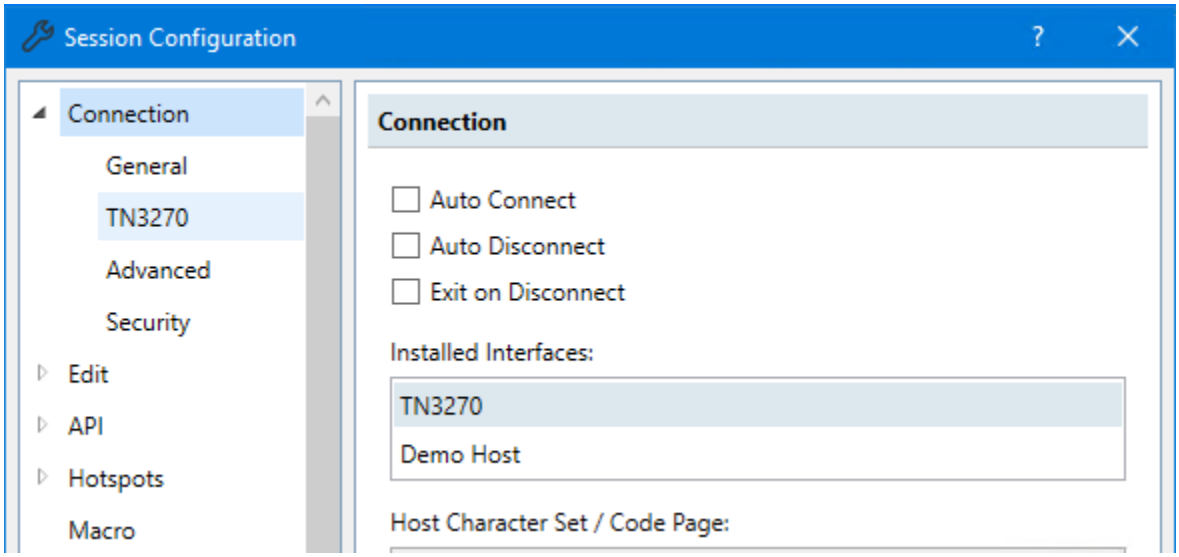
4. Elija Visualización de mainframe.



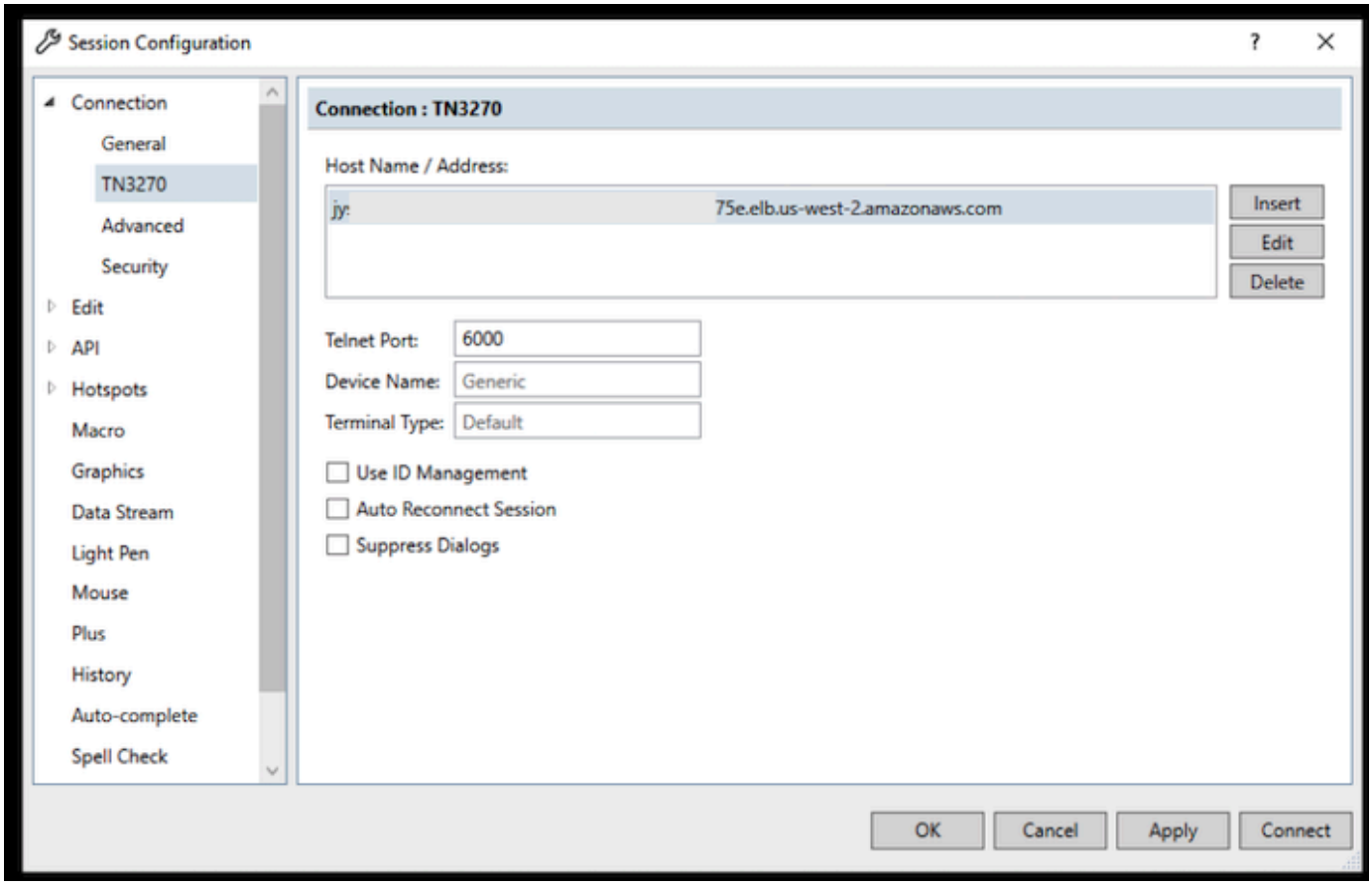
5. Elija Conexión y, a continuación, Configurar.



6. En Interfaces instaladas, elija `TN3270` y, a continuación, seleccione `TN3270` de nuevo en el menú Conexión.



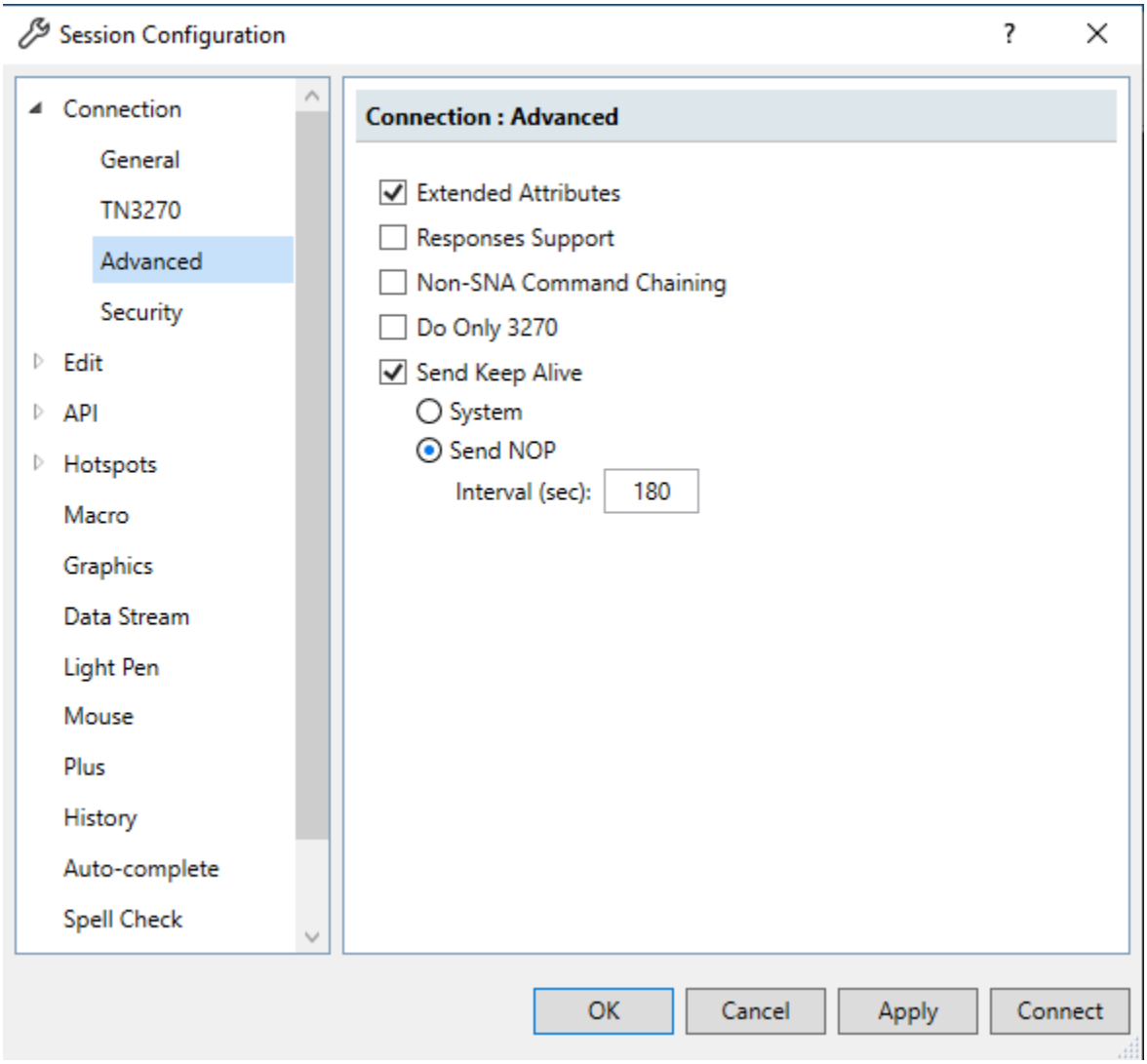
7. Elija Insertar y pegue el `DNS Hostname` para la aplicación. Especifique `6000` para el Puerto Telnet.



> **Note**
>
> Si utiliza AWS AppStream 2.0 en un navegador y tiene dificultades para pegar valores, consulte Solución de problemas de usuario de la AppStream versión 2.0.

8. En Conexión, elija Avanzado y, a continuación, elija Enviar mantenimiento activo y Enviar NOP, e introduzca 180 para el Intervalo.

> **Note**
>
> Si configuras el ajuste keep alive en tu terminal TN327 0 en al menos 180 segundos, te asegurarás de que el Network Load Balancer no interrumpa la conexión.



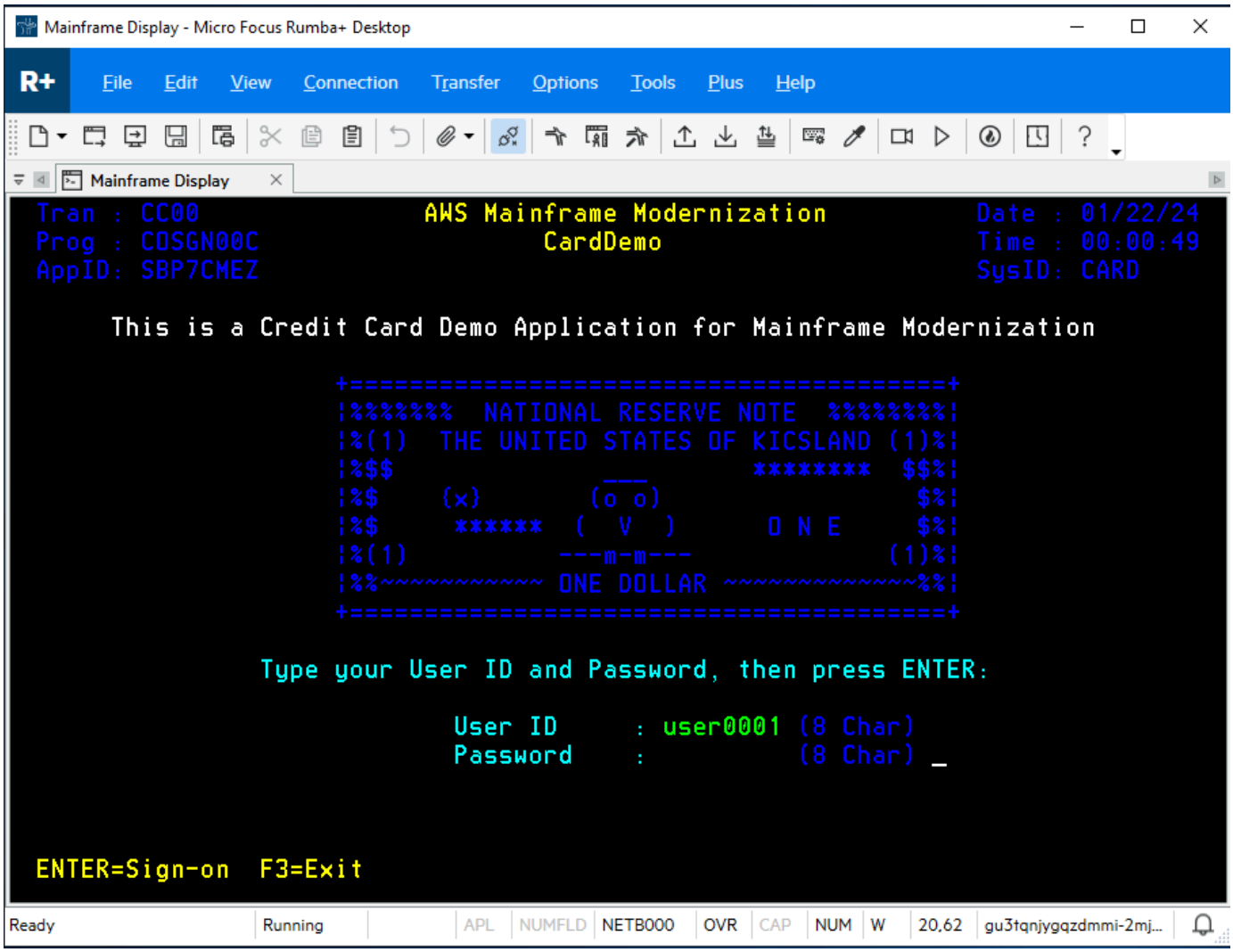
9. Elija Conectar.

> **Note**
>
> Si se produce un error en la conexión:
>
> - Si usa AppStream 2.0, confirme que la VPC y el grupo de seguridad especificados para el entorno de la aplicación son los mismos que los de la flota 2.0. AppStream
> - Utilice el Analizador de accesibilidad de la VPC para analizar la conexión. Puede acceder al Analizador de accesibilidad a través de la consola.
> - Como paso de diagnóstico, intente agregar o cambiar las reglas de entrada del grupo de seguridad para que la aplicación permita el tráfico hacia el puerto 6000 desde

cualquier lugar (es decir, el bloque CIDR 0.0.0.0/0). Si se conecta correctamente, sabrá que el grupo de seguridad estaba bloqueando el tráfico. Cambie el origen del grupo de seguridad por uno más específico. Para obtener más información sobre los grupos de seguridad, consulte Conceptos básicos de los grupos de seguridad.

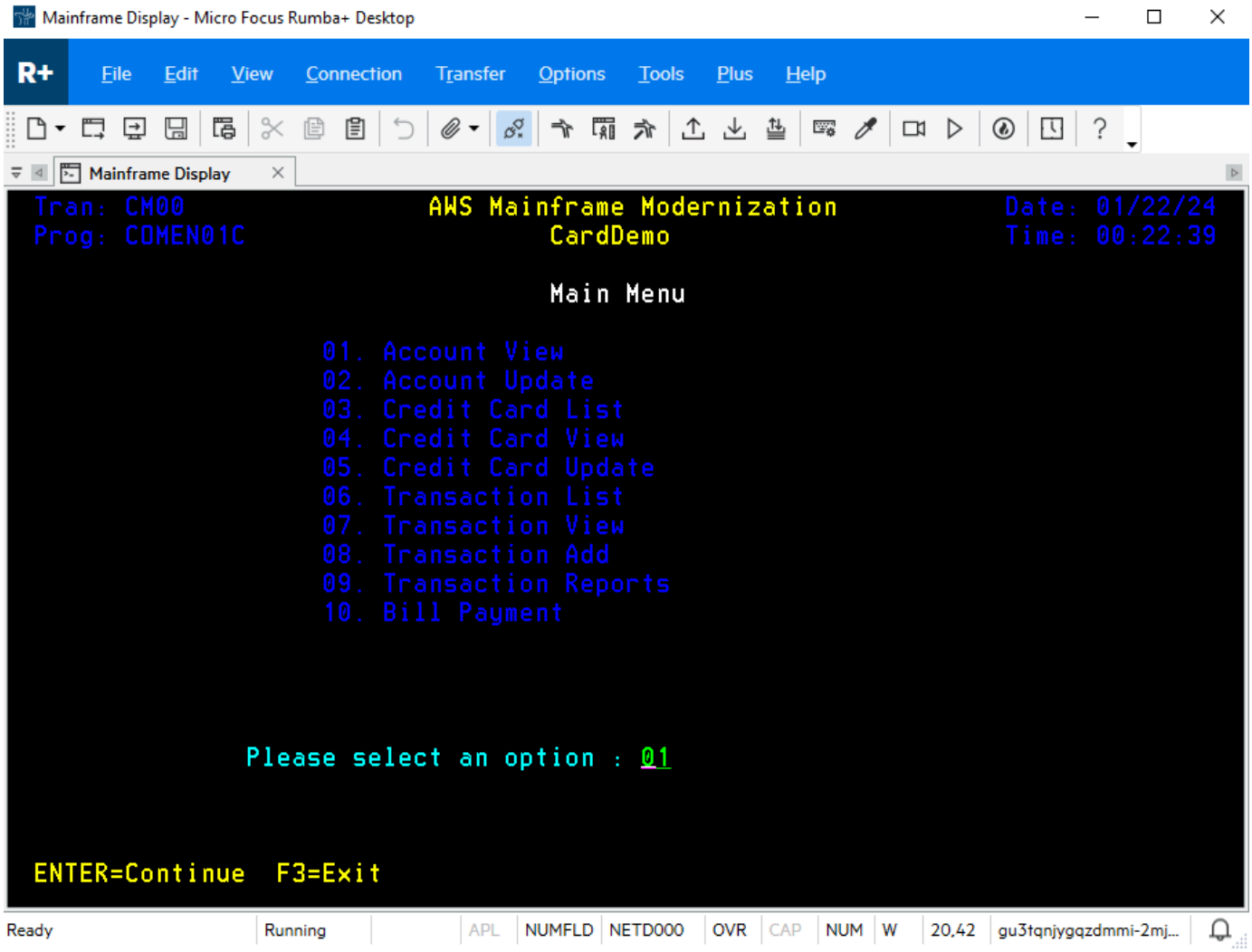
10. Introduzca `USER0001` para el usuario y `password` para la contraseña.

> **Note**
>
> En Rumba, el valor predeterminado para Borrar es ctrl-r y el predeterminado para Reset es ctrl-shift-z ctrl-r.



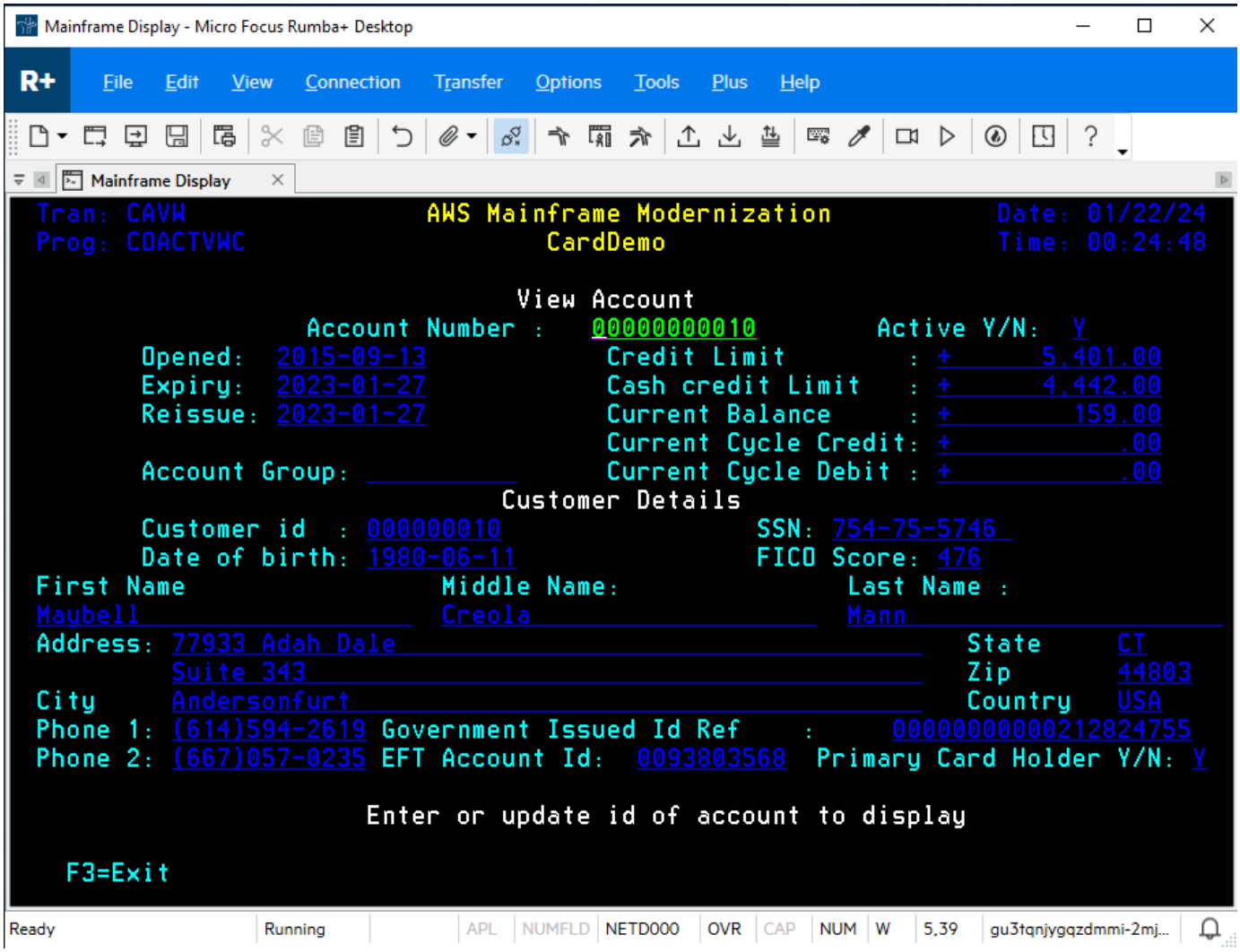
11. Después de iniciar sesión correctamente, podrá navegar por la CardDemo aplicación.

12. Introduzca `01` para la vista de la cuenta.



13. Introduzca `00000000010` para el número de cuenta y pulse Intro en el teclado.

> **Note**
>
> Otras cuentas válidas son `0000000011` y `00000000020`.



14. Pulse F3 para salir del menú y F3, para salir de la transacción.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para evitar cargos adicionales. Para ello, complete los siguientes pasos.

- Si es necesario, detenga la aplicación.
- Elimine la aplicación. Para obtener más información, consulte Eliminar una AWS Mainframe Modernization aplicación.
- Eliminar un entorno de tiempo de ejecución Para obtener más información, consulte Elimine un entorno de ejecución AWS de modernización de mainframe.

- Elimine los buckets de Amazon S3 que ha creado para este tutorial. Para obtener más información, consulte Eliminar un bucket en la Guía del usuario de Amazon S3.
- Elimine el AWS Secrets Manager secreto que creó para este tutorial. Para obtener más información, consulte Eliminación de un secreto.
- Elija la clave de KMS que ha creado para este tutorial. Para obtener más información, consulte Eliminación de claves de AWS KMS.
- Elija la base de datos de Amazon RDS que ha creado para este tutorial. Para obtener más información, consulte Eliminar la EC2 instancia y la instancia de base de datos en la Guía del usuario de Amazon RDS.
- Si ha agregado una regla de grupo de seguridad para el puerto 6000, elimínela.

Pasos a seguir a continuación

Para obtener información sobre cómo configurar un entorno de desarrollo para sus aplicaciones modernizadas, consulte el tutorial: Configurar la AppStream versión 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer.

AWS Ciclo de vida de componentes de modernización de mainframe

Cada componente de la modernización del AWS mainframe pasa por actualizaciones de versión y por un ciclo de vida de desarrollo. Puede utilizar esta página como información general para comprender estos componentes, sus planes de actualización de versiones y la forma en que AWS Mainframe Modernization comunica el lanzamiento o la obsolescencia de estos componentes o sus versiones.

Información general del ciclo de vida de los componentes

AWS El ciclo de vida de la modernización del mainframe describe el enfoque y los plazos para lanzar y dar soporte a los componentes del servicio de modernización del AWS mainframe durante todo su ciclo de vida. Ofrecer un ciclo de vida predecible y coherente le ayuda a planificar, probar e implementar las versiones más recientes.

Todos los componentes AWS de modernización de AWS mainframe proporcionados se benefician del soporte de producto que se proporciona Soporte desde el momento de su lanzamiento hasta su retirada, según el calendario de lanzamiento de cada componente. Puede obtener más información sobre el Soporte alcance y las actividades en Compare Soporte Plans. Durante los proyectos de modernización activos, normalmente recomendamos que el servicio de atención al cliente sea prestado primero por equipos de prestación de servicios profesionales, de acuerdo con la declaración de trabajo.

AWS Mainframe Modernization lanza algunos componentes con versiones originadas por proveedores, que pueden ser AWS ella misma, AWS socios selectos o comunidades. Para cada componente de modernización AWS del mainframe, una versión tiene un número de versión principal y un número de versión secundario. Cada componente tiene su propia numeración de versiones principal y secundaria.

Para los componentes versionados, tenemos las siguientes intenciones:

- Publicar versiones más recientes de los componentes de modernización de AWS mainframe de forma regular o según la demanda de los clientes. Si desea una versión más reciente de un componente y aún no está disponible en el servicio de modernización del AWS mainframe, puede realizar una solicitud explícita a través de la solicitud de características Soporte del producto (PFR).

- Hacer que las fechas de finalización y retirada del soporte de las versiones específicas de los componentes de AWS Mainframe Modernization coincidan con las fechas de finalización del soporte del proveedor de componentes.
- Notificación a los clientes aproximadamente un año antes de la retirada de la versión principal de un componente

Si bien nos esforzamos por cumplir con estas pautas, en algunos casos, es posible que retiremos versiones específicas antes y con plazos de notificación más breves. Por ejemplo, podemos retirar rápidamente una versión con problemas de seguridad y con un plazo de notificación más corto. También podemos retirar anticipadamente versiones secundarias cuando una versión secundaria tenga errores importantes o problemas de seguridad que se hayan resuelto en una versión secundaria posterior. En el improbable caso de que se produzcan estas circunstancias, notificaremos a los clientes y les informaremos sobre el plan y el calendario de retirada. Hay circunstancias específicas que pueden dictar plazos diferentes según la situación.

> **Note**
>
> Las actualizaciones críticas de los componentes pueden estar disponibles en cualquier momento. Por ejemplo, es posible que las nuevas versiones estén disponibles rápidamente por motivos de seguridad o para proporcionar correcciones a los entornos de producción. En el caso de las solicitudes presentadas Soporte, el plan de soporte determina los procesos, la gravedad y los tiempos de respuesta.

Cuando se retira una versión de un componente, AWS Mainframe Modernization no distribuye estas versiones a los clientes para nuevas implementaciones. Por consiguiente, estas versiones tampoco las admite Soporte. Los clientes que ejecuten despliegues de componentes existentes después de la fecha de retirada de su versión deben ser conscientes de los riesgos que conlleva hacerlo. AWS no es responsable de proporcionar actualizaciones de seguridad, soporte técnico ni correcciones urgentes para las versiones de componentes retiradas. Además, no eliminamos el acceso ni los recursos de su entorno de manera automática. Le recomendamos encarecidamente que compruebe si hay nuevas versiones cada 3 meses y que actualice todos los componentes de modernización AWS del mainframe a versiones compatibles recientes.

Actualización de versiones

AWS La modernización del mainframe ofrece versiones más recientes de cada componente compatible para que pueda disponer de up-to-date las funciones y actualizaciones de mantenimiento más recientes. Las versiones más recientes pueden incluir correcciones de errores, así como mejoras de la seguridad y de otro tipo para los componentes. Le recomendamos que actualice periódicamente para beneficiarse de las correcciones de seguridad, las correcciones de errores y las mejoras de características. Cuando AWS Mainframe Modernization publique una nueva versión, podrá elegir cómo y cuándo actualizar las implementaciones existentes. Hay dos tipos de actualizaciones: actualizaciones de versiones principales y actualizaciones de versiones secundarias. En general, una actualización de la versión principal del motor puede introducir cambios que no son compatibles con las aplicaciones existentes. En ese caso, es posible que se requieran cambios sustanciales en la aplicación para la actualización de una versión principal. Por contraste, una actualización de una versión secundaria incluye cambios compatibles con las versiones anteriores de las aplicaciones existentes. Es posible que se requieran pocos cambios o ninguno para actualizar una versión secundaria.

Debe realizar pruebas de no regresión antes de actualizar las versiones de los componentes. Se recomienda utilizar canalizaciones de DevOps prueba e implementación. DevOps Los canales de prueba se pueden crear durante los proyectos de modernización y deben mantenerse para automatizar las pruebas de las aplicaciones al realizar actualizaciones de componentes y cambios en el código de las aplicaciones. También puede utilizar implementaciones azules/verdes, o implementaciones canario durante las actualizaciones. Puede obtener más información sobre estas implementaciones y la administración de cambios en AWS Well-Architected Reliability Pillar.

AWS Descripción general de la versión de Mainframe Modernization Refactorization with AWS Blu Age

Con el tiempo de ejecución de AWS Blu Age, la versión sigue un `Major.Minor.Patch` patrón. Por ejemplo, para la versión `4.1.0` de tiempo de ejecución de AWS Blu Age, la versión principal es 4, la versión secundaria es 1 y la versión del parche es 0.

Tenemos la intención de lanzar nuevas versiones principales del entorno de ejecución de AWS Blu Age cuando se produzcan cambios importantes en el tiempo de ejecución o en sus dependencias. AWS Las versiones principales del entorno de ejecución de Blu Age están disponibles durante al menos 12 meses, a menos que aparezcan algunas vulnerabilidades y exposiciones comunes () CVEs. El soporte abarca los errores en las características de tiempo de ejecución, tal como

se menciona en nuestra documentación. En el caso de que las versiones Critical y High CVEs dependan del entorno de ejecución (Spring, Java, Tomcat y otras), la duración del soporte de la versión principal se reduce a 6 meses para High CVEs y a 3 meses para Critical a CVEs partir de la fecha de lanzamiento de la nueva versión en tiempo de ejecución que corrija el CVE, a menos que se indique explícitamente lo contrario.

Pretendemos lanzar nuevas versiones secundarias de AWS Blu Age todos los meses. Se espera que los clientes actualicen las versiones periódicamente para obtener las últimas correcciones de seguridad y de errores, así como las mejoras de características. Los proyectos activos que aún no estén en producción deben adoptar la versión del tiempo de ejecución más reciente tan pronto como esté disponible.

En la última versión secundaria se proporcionan nuevas correcciones para la versión principal concreta en la que se genera un problema. Si necesita nuevas correcciones, debe actualizar a una nueva versión secundaria para aplicarlas.

Las versiones parcheadas de las versiones compatibles se proporcionan solo para corregir defectos críticos de tiempo de ejecución que no estaban presentes en las versiones secundarias compatibles anteriores.

Las versiones preliminares alfa son versiones de corta duración que están disponibles para su rápida iteración durante los proyectos de entrega. Las correcciones de los problemas detectados en las versiones preliminares de la versión alfa se incluyen en las versiones secundarias posteriores, ya que no se incluyen parches en las versiones preliminares de la versión alfa.

Puede encontrar las fechas de lanzamiento y los detalles sobre cada versión de tiempo de ejecución en las the section called “AWS Notas de lanzamiento de Blu Age”.

Los análisis de seguridad los realiza Amazon Inspector.

Comprenda las aplicaciones administradas en AWS Mainframe Modernization

Si eres nuevo, AWS Mainframe Modernization consulta los siguientes temas para empezar:

- ¿Qué es la modernización del AWS mainframe?
- Configuración para la modernización del AWS mainframe
- Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age
- Tutorial: Configurar el tiempo de ejecución gestionado para Rocket Software (anteriormente Micro Focus)

Una aplicación AWS Mainframe Modernization contiene una carga de trabajo de mainframe migrada. La aplicación es análoga a una carga de trabajo en la unidad central y está asociada a un entorno de tiempo de ejecución. Puede añadir archivos por lotes y conjuntos de datos a las aplicaciones y supervisarlas a medida que se ejecutan. Puede crear aplicaciones de AWS Mainframe Modernization para cada carga de trabajo que migre. Al crear una AWS Mainframe Modernization aplicación, se especifica el motor en el que se ejecuta la aplicación al crearla. Elija AWS Blu Age si utiliza el patrón de refactorización automática y Rocket Software (anteriormente Micro Focus) si utiliza el patrón de replataforma.

Temas

- Cree AWS recursos para una aplicación migrada
- Cree una AWS Mainframe Modernization aplicación
- Implemente una AWS Mainframe Modernization aplicación
- Actualice una AWS Mainframe Modernization aplicación
- Eliminar una AWS Mainframe Modernization aplicación
- Envíe trabajos por lotes para AWS Mainframe Modernization las solicitudes
- Cancela los trabajos por lotes para las aplicaciones AWS Mainframe Modernization
- Importe conjuntos de datos para AWS Mainframe Modernization aplicaciones
- Exporte conjuntos de datos para AWS Mainframe Modernization aplicaciones
- Administración de las transacciones para las aplicaciones de AWS Mainframe Modernization
- Configurar la aplicación gestionada por Rocket Software (anteriormente Micro Focus)

- Configurar la aplicación gestionada AWS Blu Age
- AWS Mainframe Modernization referencia de definición de aplicación
- AWS Referencia de definición de conjuntos de datos de modernización de mainframe

Cree AWS recursos para una aplicación migrada

Para ejecutar la aplicación migrada AWS, debe crear algunos AWS recursos con otros Servicios de AWS. Los recursos que debe crear incluyen lo siguiente:

- Un depósito de S3 para almacenar el código de la aplicación, la configuración, los archivos de datos y otros elementos necesarios.
- Una base de datos de Amazon RDS o Amazon Aurora para almacenar los datos que requiere la aplicación.
- Y AWS KMS key, que es necesaria AWS Secrets Manager para crear y almacenar secretos.
- Un secreto de Secrets Manager para guardar las credenciales de la base de datos

> **Note**
>
> Cada aplicación migrada requiere su propio conjunto de estos recursos. Se trata de un conjunto mínimo. Es posible que la aplicación también requiera recursos adicionales, como secretos de Amazon Cognito o colas de MQ.

Permisos necesarios

Asegúrese de que tiene los siguientes permisos:

- `s3:CreateBucket`, `s3:PutObject`
- `rds:CreateDBInstance`
- `kms:CreateKey`
- `secretsmanager:CreateSecret`

Bucket de Amazon S3

Tanto las aplicaciones refactorizadas como las que han redefinido su plataforma requieren un bucket de Amazon S3 que se configura de la siguiente manera:

```
bucket-name/root-folder-name/application-name
```

bucket-name

Cualquier nombre dentro de las restricciones de nomenclatura de Amazon S3. Le recomendamos que incluya el nombre de la Región de AWS como parte del nombre de su bucket. Asegúrese de crear el bucket en la misma región en la que tiene previsto implementar la aplicación migrada.

root-folder-name

Nombre necesario para cumplir con las restricciones de la definición de la aplicación, que se crea como parte de la AWS Mainframe Modernization aplicación. Puede utilizar el `root-folder-name` para distinguir entre las distintas versiones de una aplicación, por ejemplo, la V1 y la V2.

application-name

El nombre de la aplicación migrada, por ejemplo, PlanetsDemo o BankDemo.

Database

Tanto las aplicaciones refactorizadas como las reconfiguradas pueden requerir una base de datos. Debe crear, configurar y administrar la base de datos de acuerdo con los requisitos específicos de cada motor de tiempo de ejecución. AWS Mainframe Modernization admite el cifrado en tránsito en esta base de datos. Si habilita el SSL en su base de datos, asegúrese de especificar `sslMode` en el secreto de la base de datos junto con los detalles de conexión de la base de datos. Para obtener más información, consulte AWS Secrets Manager secreto.

Si utiliza el patrón de refactorización de AWS Blu Age y necesita una BluSam base de datos, el motor de ejecución de AWS Blu Age espera una base de datos PostgreSQL de Amazon Aurora, que debe crear, configurar y administrar. La BluSam base de datos es opcional. Cree esta base de datos solo si su aplicación lo requiere. Para crear la base de datos, siga los pasos que se indican en Creación de un clúster de base de datos de Amazon Aurora en la Guía del usuario de Amazon Aurora.

Si utiliza el patrón de cambio de plataforma de Rocket Software, puede crear una base de datos PostgreSQL de Amazon RDS o Amazon Aurora. Para crear la base de datos, siga los pasos que

se indican en Creación de una instancia de base de datos de Amazon RDS en la Guía del usuario de Amazon RDS o en Creación de un clúster de base de datos de Amazon Aurora en la Guía del usuario de Amazon Aurora.

Para ambos motores de ejecución, debe almacenar las credenciales de la base de datos para AWS Secrets Manager cifrarlas. AWS KMS key

AWS Key Management Service clave

Debe almacenar las credenciales de la base de datos de la aplicación de forma segura en AWS Secrets Manager. Para crear un secreto en Secrets Manager, debe crear una AWS KMS key. Para crear una nueva clave de KMS, siga los pasos de Creación de claves en la Guía para desarrolladores de AWS Key Management Service .

Tras crear la clave, debe actualizar la política de claves para conceder permisos de AWS Mainframe Modernization descifrado. Añada las siguiente instrucciones de política:

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "kms:Decrypt",
   "Resource" : "*"
   }
```

AWS Secrets Manager secreto

Debe almacenar las credenciales de la base de datos de la aplicación de forma segura en AWS Secrets Manager. Para crear un secreto, siga los pasos de Creación de un secreto en la Guía del usuario de AWS Secrets Manager .

AWS Mainframe Modernization admite el cifrado en tránsito en esta base de datos. Si habilita el SSL en su base de datos, asegúrese de especificar `sslMode` en el secreto de la base de datos junto con los detalles de conexión de la base de datos. Puede especificar uno de los siguientes valores para `sslMode`: `verify-full`, `verify-ca` o `disable`.

Durante el proceso de creación de la clave, elija Permisos de recursos. Opcional y, a continuación, elija Editar permisos. En el editor de políticas, añada una política basada en recursos, como la siguiente, para recuperar el contenido de los campos cifrados.

```
{
   "Effect" : "Allow",
   "Principal" : {
   "Service" : "m2.amazonaws.com"
   },
   "Action" : "secretsmanager:GetSecretValue",
   "Resource" : "*"
   }
```

Cree una AWS Mainframe Modernization aplicación

Utilice la AWS Mainframe Modernization consola para crear una AWS Mainframe Modernization aplicación. La creación de una aplicación le permite realizar tareas con la carga de trabajo del mainframe migrado.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Creación de una aplicación de

Para crear una aplicación

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que desee crear la aplicación.
3. En la página Aplicaciones, seleccione Crear aplicación.
4. En la página Especificar información básica, en la sección Nombre y descripción, introduzca un nombre para la aplicación.
5. En el campo Descripción de la aplicación, escriba una descripción de la aplicación. Esta descrpción ayudarle a usted y a otros usuarios a identificar el propósito de la aplicación.
6. En la sección Tipo de motor, selecciona Blu Age para la refactorización automática o Micro Focus (Rocket) para la replataforma.
7. En la sección de claves KMS, elija Personalizar la configuración de cifrado si desea utilizar una clave gestionada por el cliente. AWS KMS Para obtener más información, consulte Cifrado de datos en reposo para el servicio de modernización AWS de mainframe.

> **Note**
>
> De forma predeterminada, AWS Mainframe Modernization cifra los datos con una AWS KMS clave que le AWS Mainframe Modernization pertenece y administra por usted. Sin embargo, puede optar por utilizar una AWS KMS clave gestionada por el cliente.

8. (Opcional) Elija una AWS KMS clave por nombre o por nombre de recurso de Amazon (ARN), o bien elija Crear una AWS KMS clave para ir a la AWS KMS consola y crear una clave nueva AWS KMS .
9. (Opcional) En la sección Etiquetas, seleccione Agregar nueva etiqueta para añadir una o varias etiquetas a la aplicación. Una etiqueta de aplicación es una etiqueta de atributo personalizada que le ayuda a organizar y administrar sus AWS recursos.
10. Elija Siguiente.
11. En la sección Recursos y configuraciones, utilice el editor en línea para introducir la definición de la aplicación. También puede elegir Use an application definition JSON file in an Amazon S3 bucket e indicar la ubicación de la definición de aplicación que desea usar. Para obtener más información, consulte AWS Ejemplo de definición de aplicación de Blu Age o Definición de la aplicación Rocket Software (anteriormente Micro Focus).
12. Elija Siguiente.
13. En la página Revisar y crear, revise la información que ha proporcionado y, a continuación, elija Crear origen de datos.

Implemente una AWS Mainframe Modernization aplicación

Use la AWS Mainframe Modernization consola para implementar una AWS Mainframe Modernization aplicación. Tiene que implementar sus aplicaciones en un entorno de tiempo de ejecución antes de realizar las tareas.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Implementación de una aplicación de

Para ejecutar una AWS Mainframe Modernization aplicación, primero debe implementarla en un entorno de ejecución. Una aplicación puede tener más de una versión. Cada versión de una

aplicación tiene su propia definición de aplicación. Para implementar una aplicación, debe especificar la versión que desea implementar.

Solo puede implementar una versión de una aplicación determinada a la vez. Si implementa una versión de una aplicación y decide implementar una versión diferente, primero debe detener la aplicación si está en ejecución.

Para implementar una aplicación

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que desee crear la aplicación.
3. En la página Aplicaciones, seleccione la aplicación que desea implementar.
4. Elija Implementar aplicación.
5. En la sección Versiones disponibles, elija la versión que desea implementar.
6. En la sección Entornos, elija un entorno de tiempo de ejecución en el que desea que se ejecute la aplicación.
7. Elija Implementar.

Para implementar una versión diferente de una aplicación implementada

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que desee crear la aplicación.
3. En la página Aplicaciones, seleccione la aplicación que desea implementar.
4. En el menú Acciones, elija Detener aplicación.
5. Cuando la aplicación se detenga, elija Implementar aplicación.
6. En la sección Versiones disponibles, elija la versión que desea implementar. En la sección Entornos, el entorno en el que ya está desplegada la aplicación aparece preseleccionado.
7. Elija Implementar.

Actualice una AWS Mainframe Modernization aplicación

Use la AWS Mainframe Modernization consola para actualizar una AWS Mainframe Modernization aplicación. Al actualizar una aplicación se creará una versión nueva de la aplicación.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Actualización de una aplicación de

Una AWS Mainframe Modernization aplicación puede tener varias versiones, cada una con su propia definición de aplicación. Para actualizar una aplicación, proporcione una nueva definición de aplicación. Así se creará una versión nueva de la aplicación.

Para actualizar una aplicación

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación que desea actualizar.
3. En la página Aplicaciones, seleccione la aplicación que desea actualizar.
4. En la página de detalles de la aplicación, en la sección Definición actual, elija Editar para actualizar la definición de la aplicación actual.
5. En la página Actualizar aplicación, utilice el editor en línea para actualizar la definición de la aplicación actual.

   También puede elegir Use an application definition JSON file in an Amazon S3 bucket e indicar la ubicación de la definición de aplicación que desea usar. Para obtener más información, consulte AWS Ejemplo de definición de aplicación de Blu Age o Definición de la aplicación Rocket Software (anteriormente Micro Focus).
6. Cuando haya terminado de actualizar la definición de la aplicación, seleccione Actualizar.

> **Note**
>
> Tras actualizar la aplicación, debe volver a implementarla. Para obtener más información, consulte Implemente una AWS Mainframe Modernization aplicación.

Eliminar una AWS Mainframe Modernization aplicación

Puede eliminar una AWS Mainframe Modernization aplicación de un entorno mediante la AWS Mainframe Modernization consola.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Eliminación de una aplicación de

Si necesita eliminar una AWS Mainframe Modernization aplicación y está en ejecución, asegúrese de detenerla primero. Puede ver el estado de la aplicación en la página Aplicaciones.

Para eliminar una aplicación de

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación que desea eliminar del entorno.
3. En la página Aplicaciones, elija la aplicación que desea eliminar del entorno y, a continuación, elija Acciones.
4. (Opcional) Si el estado de la aplicación es`Running`, seleccione Detener aplicación.
5. Seleccione Eliminar del entorno.

El proceso de eliminación se inicia de forma inmediata.

Envíe trabajos por lotes para AWS Mainframe Modernization las solicitudes

En AWS Mainframe Modernization ella puede enviar trabajos por lotes para sus aplicaciones. Puede enviar o cancelar trabajos por lotes y revisar los detalles sobre la ejecución de los trabajos por lotes. Cada vez que envía un trabajo por lotes, AWS Mainframe Modernization crea una ejecución de trabajo por lotes independiente. Puede supervisar la ejecución de este trabajo. Puede buscar los trabajos por lotes por su nombre y suministrar archivos JCL o de script a los trabajos por lotes.

> **Important**
>
> Si cancela un trabajo por lotes, no se elimina el trabajo, Cancela una ejecución concreta del trabajo por lotes. Los registros del trabajo por lotes permanecen disponibles para que pueda consultarlos en los detalles de la ejecución del trabajo por lotes.

Si su trabajo por lotes requiere acceso a uno o más conjuntos de datos, utilice la AWS Mainframe Modernization consola para importar los conjuntos de datos. Para obtener más información, consulte Importe conjuntos de datos para AWS Mainframe Modernization aplicaciones.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe y en Cree una AWS Mainframe Modernization aplicación.

Temas

- Envío de un trabajo por lotes.
- Reinicio de un trabajo por lotes

Envío de un trabajo por lotes.

Para enviar un trabajo por lotes.

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación para la que desea enviar un trabajo por lotes.
3. En la página Aplicaciones, seleccione la aplicación para la que desea enviar un trabajo por lotes.

   > **Note**
   >
   > Para poder enviar un trabajo por lotes a una aplicación, debe implementar la aplicación de forma correcta.

4. En la página de detalles de la aplicación, seleccione Trabajos por lotes.
5. Seleccione Enviar el trabajo.
6. En la sección Seleccionar un script, elija un script. Puede buscar el script que desee por nombre.
7. Seleccione Enviar el trabajo.

Reinicio de un trabajo por lotes

Reinicio de un trabajo por lotes

> **Important**
>
> El reinicio de un trabajo por lotes está disponible en las siguientes versiones de motor:

- Versión 8.0.6 o superior del motor de entorno Micro Focus (Rocket). También debe tener un EFS o un sistema de FSx archivos conectado a su entorno.
- AWS Versión 4.3.0 o superior del motor de entorno Blu Age. También debe tener un EFS o un sistema de FSx archivos adjunto si se trata de un entorno de alta disponibilidad.

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se crearon la aplicación y el trabajo por lotes.
3. En la página Aplicaciones, seleccione la aplicación en la que desea reiniciar un trabajo por lotes.
4. En la página de detalles de la aplicación, seleccione Trabajos por lotes.
5. Seleccione el trabajo por lotes que desee reiniciar en la lista generada. Vaya al menú Acciones y elija Reiniciar trabajo.
6. Especifique cómo desea que se reinicie el trabajo por lotes. Puede hacer lo siguiente con el motor de entorno Micro Focus (Rocket) y el motor de entorno AWS Blu Age:

   - En el caso del motor de entorno Micro Focus (Rocket), puede elegir entre reiniciar desde el principio o reiniciar siguiendo pasos o pasos sucesivos.
     - La opción Reiniciar desde el principio le permite reiniciar todos los pasos de un trabajo por lotes desde el principio.
     - La opción Reiniciar por pasos o procsteps le permite elegir un paso o paso de procedimiento específico que desee reiniciar y, opcionalmente, un paso o paso de procedimiento que desee finalizar.

   > **Note**
   >
   > El paso final o paso de procedimiento debe ser mayor o igual que el número de paso de inicio o de paso de procedimiento.

   - En el caso del motor de entorno AWS Blu Age, puede reiniciar la ejecución más reciente de un trabajo por lotes a partir de un paso de JCL/PROC anteriormente fallido o realizar un reinicio retrasado omitiendo los pasos que anteriormente se habían realizado correctamente.
     - Puede elegir un nombre de paso específico que desee reiniciar.

- Si lo desea, puede usar Omitir paso para omitir el paso seleccionado y reiniciar desde el siguiente paso del flujo de trabajo.

7. Seleccione Enviar el trabajo.

Cancela los trabajos por lotes para las aplicaciones AWS Mainframe Modernization

En AWS Mainframe Modernization ella puede cancelar los trabajos por lotes para sus aplicaciones. Puede revisar los detalles sobre la ejecución de trabajos por lotes. Cada vez que envía un trabajo por lotes, AWS Mainframe Modernization crea una ejecución de trabajo por lotes independiente. Puede supervisar la ejecución de este trabajo. Puede buscar los trabajos por lotes por su nombre y suministrar archivos JCL o de script a los trabajos por lotes.

> **Important**
>
> Si cancela un trabajo por lotes, no se elimina el trabajo, Cancela una ejecución concreta del trabajo por lotes. Los registros del trabajo por lotes permanecen disponibles para que pueda consultarlos en los detalles de la ejecución del trabajo por lotes.

Cancelación de un trabajo por lotes

Al cancelar un trabajo por lotes, no se elimina un trabajo por lotes, sino la ejecución de las tareas de ese trabajo por lotes. Puede seguir viendo los detalles del trabajo por lotes.

Cancelación de un trabajo por lotes

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región con la aplicación para sus trabajos por lotes.
3. En la lista de trabajos por lotes, busque y seleccione el trabajo por lotes que desea cancelar.
4. Seleccione Acciones y, a continuación, Cancelar trabajo.
5. Seleccione Cancelar trabajo por lotes.

Esto cancelará las tareas de trabajo por lotes que tuviera programadas para su ejecución.

Importe conjuntos de datos para AWS Mainframe Modernization aplicaciones

Con AWS Mainframe Modernization ella puede importar conjuntos de datos para usarlos con sus aplicaciones. Puede especificar los conjuntos de datos que se van a importar en un archivo JSON almacenado en un bucket de Amazon S3 o puede especificar los valores de configuración de los conjuntos de datos por separado. Tras importar los conjuntos de datos, puede revisar los detalles de la tarea de importación para confirmar que se importaron los conjuntos de datos que quería. Todos los conjuntos de datos catalogados de una aplicación se muestran juntos en la consola.

Utilice la AWS Mainframe Modernization consola para importar conjuntos de datos para una AWS Mainframe Modernization aplicación.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe y en Cree una AWS Mainframe Modernization aplicación.

Importar un conjunto de datos

Para importar un conjunto de datos

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación para la que desea importar los conjuntos de datos.
3. En la página Aplicaciones, seleccione la aplicación para la que desea importar conjuntos de datos.
4. En la página de detalles de la aplicación, elija Conjuntos de datos.
5. Seleccione Importar.
6. Realice una de las siguientes acciones:
   - Elija Use data set configuration JSON file in an Amazon S3 bucket e indique la ubicación de la configuración del conjunto de datos.
   - Elija Specify the data set configuration values separately con la configuración indicada. Consulte the section called “Referencia de definición de conjuntos de datos” para conocer los detalles de la definición específica.

     Introduzca el nombre, la organización del conjunto de datos (VSAM, GDG, PO, PS), la ubicación y la ubicación externa de Amazon S3, así como los ajustes de los parámetros

de cada valor de configuración del conjunto de datos. En la configuración guiada, también puede elegir Generar JSON para revisar la configuración de JSON a partir de sus datos.

7. Elija Enviar.

Exporte conjuntos de datos para AWS Mainframe Modernization aplicaciones

Con AWS Mainframe Modernization ella puede exportar conjuntos de datos para usarlos con sus aplicaciones. Puede especificar los conjuntos de datos que se van a exportar en un archivo JSON almacenado en un bucket de Amazon S3 o puede especificar los valores de configuración de los conjuntos de datos por separado. Tras exportar los conjuntos de datos, puede revisar los detalles de la tarea de exportación para confirmar que se exportaron los conjuntos de datos que quería.

Utilice la AWS Mainframe Modernization consola para exportar conjuntos de datos para una AWS Mainframe Modernization aplicación.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe y en Cree una AWS Mainframe Modernization aplicación.

Exporte un conjunto de datos

Para exportar un conjunto de datos

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación para la que desea importar los conjuntos de datos.
3. En la página Aplicaciones, elija la aplicación a la que desee exportar los conjuntos de datos.
4. En la página de detalles de la aplicación, elija Conjuntos de datos.
5. Seleccione Exportar.
6. Realice una de las siguientes acciones:
   - Elija Use data set configuration JSON file in an Amazon S3 bucket e indique la ubicación de la configuración del conjunto de datos.
   - Elija Specify the data set configuration values separately con la configuración indicada. Para obtener más información, consulte the section called “Referencia de definición de conjuntos de datos”.

Introduzca el nombre del conjunto de datos, la ubicación externa de Amazon S3 y la configuración de los parámetros para cada valor de configuración del conjunto de datos. En la configuración guiada, también puede elegir Generar JSON para revisar la configuración de JSON a partir de sus datos.

7. Elija Enviar.

Administración de las transacciones para las aplicaciones de AWS Mainframe Modernization

Con AWS Mainframe Modernization ella, puede ejecutar una aplicación, previa solicitud, al mismo tiempo que muchos otros usuarios que envían solicitudes para ejecutar la misma aplicación con los mismos archivos y programas. Una transacción consta de uno o más programas de aplicación que llevan a cabo el procesamiento necesario.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe y en Cree una AWS Mainframe Modernization aplicación.

Gestión de las transacciones de las aplicaciones

Para administrar las transacciones de las aplicaciones

1. Abra la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el Región de AWS selector, elija la región en la que se creó la aplicación que desea ejecutar.
3. En la página Aplicaciones, seleccione la aplicación en la que desea administrar transacciones.
4. En la pestaña Transacciones, en Transaction resources, elija cómo quiere que se muestren sus recursos en la lista desplegable. Puede mostrar los recursos según los recursos de transacciones, grupos, listas o SITs.

   - Los recursos de transacciones le permiten elegir el tipo de recurso según las definiciones de archivos, las definiciones de transacciones, las definiciones de programas o las definiciones de colas de datos transitorios.

> **Note**
>
> El AWS Mainframe Modernization servicio admite tipos de recursos adicionales para gestionar las transacciones de las aplicaciones y se puede acceder a él desde la consola.

- Los grupos son una colección de recursos de transacciones. Puede elegir los grupos que desea asociar a su recurso de transacción.
- Las listas son una colección ordenada de grupos. Puede ver todos sus recursos y grupos de transacciones en una vista de lista. La lista de startup determina qué recursos se cargan cuando se inicializa el servidor.
  - Con el motor de refactorización AWS Blu Age, usted especifica las listas que se incluirán al inicio. No existe ningún límite en el número de listas.
  - Con el motor de replataforma de Rocket Software, puede especificar hasta cuatro listas en un SIT.
- La SIT (tabla de inicialización del sistema) muestra todas las configuraciones de transacciones disponibles. Puede SITs buscarlas por propiedades (nombre, descripción y listas de inicio). También puede elegir listas para asociarlas a la SIT que haya elegido.

> **Note**
>
> SITs solo se aplican al motor de replataforma de Rocket Software.

5. Elija un recurso de transacción para mostrar toda la información del recurso. También puede ver todos los atributos asociados a su recurso de transacción.

Configurar la aplicación gestionada por Rocket Software (anteriormente Micro Focus)

Puede configurar sus aplicaciones con el motor de ejecución de Rocket Software para personalizar propiedades adicionales, incluidas las integraciones.

Temas

- Compatible con integraciones de terceros para Rocket Software
  - Impresoras

Compatible con integraciones de terceros para Rocket Software

Para utilizar integraciones de terceros, su entorno gestionado de modernización de AWS mainframe debe utilizar una versión del motor de Rocket Software que sea compatible con este tipo de configuración. Se admiten las versiones de motor con el sufijo R (por ejemplo, la versión 9.0.9.R). Eso significa que la versión 9.0.9.R del motor incluye soporte de instalación de clientes para las integraciones de terceros, pero la versión 9.0.9, no.

Impresoras

Los recursos de la impresora se configuran mediante la definición de la aplicación Rocket Software, tal como se describe en la the section called “Impresoras: opcionales” sección.

Una definición de impresora puede definir un módulo de salida personalizado o proporcionado por un servicio para la impresora. Algunos ejemplos de posibles configuraciones de módulos de salida son:

1. Ejemplo de archivo binario proporcionado de servicio de carga.

```
...
{
    "name": "p1",
    "classes": [
        "AB"
    ],
    "description": "Using service managed LRS Queue exit module",
    "exit-module": {
        "name": "lrsprte6"
    }
},
...
```

2. Ejemplo de suministro de archivo binario desde S3.

```
"exit-module": {
        "name": "s3Exit",
        "module": "${s3-source}/3pa/s3Exit.so"
}
```

3. Ejemplo de suministro de binario desde EFS.

> **Note**
>
> Para utilizar el montaje de EFS, debe estar asociado durante la creación del entorno, junto con algunos valores adicionales que se deben establecer, como `program-path`.

```
...
"batch-settings": {
    "jes-printers": [
        {
            "name": "p3",
            "classes": [
                "EF"
            ],
            "description": "Using binary from customer provided exit module on EFS
 Mount",
            "exit-module": {
                "name": "efsExit"
            }
        }
    ],
    "program-path": "$EFS_MOUNT/path/to/directory/containing/binaries/"
},
"runtime-settings": {
    "environment-variables": {
        "EFS_MOUNT": "/m2/mount/efs"
    }
}
...
```

Cola de LRS: opcional

Para utilizar la cola LRS, debes utilizar un motor de Rocket Software que admita artefactos de terceros (es decir, motores que terminen con `.R`). Además de configurar una impresora con un módulo de salida que apunte al `lrsprte6` nombre de la entrada del módulo de salida, la cola LRS requiere una variable de entorno adicional, tal y como se define en el bloque preexistente de «ajustes del tiempo de ejecución» de la definición de la aplicación Rocket Software.

LRSQ_ADDRESS

(Obligatorio) Especifica la dirección del servidor de LRS al que debe enviarse el módulo de salida de impresión del LRSQ.

Impresoras LRS: la configuración de una impresora LRS requiere la definición de una impresora JES, tal como se especifica en la sección the section called “Impresoras: opcionales”.

Además, LRSQ_ADDRESS debe especificarse como parte del campo `runtime-settings` de la definición de la aplicación.

```
"runtime-settings": {
    "environment-variables": {
        "LRSQ_ADDRESS": "<lrsq-address>"
    }
}
```

Configurar la aplicación gestionada AWS Blu Age

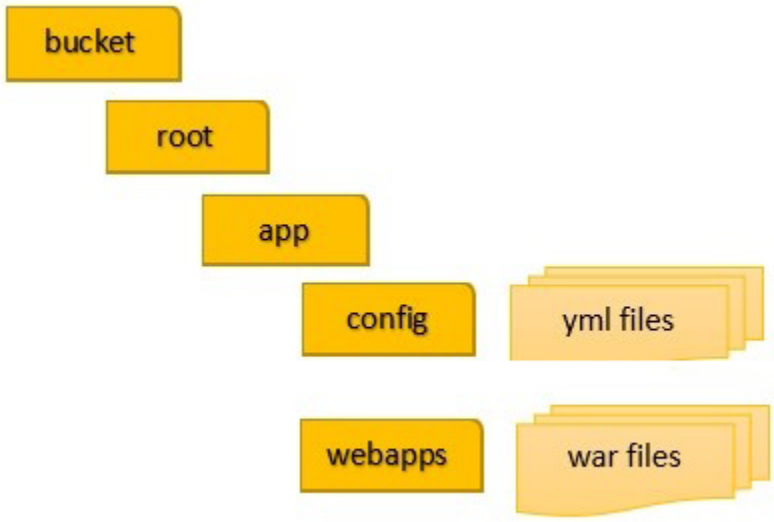
Puede configurar la aplicación para que incluya acceso a utilidades heredadas. También puede personalizar propiedades adicionales. Para comprender qué puede configurar y dónde, consulte la the section called “Estructura de las aplicaciones gestionadas de AWS Blu Age” sección para comprender la estructura general de una aplicación modernizada de AWS Blu Age.

Temas

- Estructura de las aplicaciones gestionadas de AWS Blu Age
- Configuración del acceso a las utilidades para las aplicaciones administradas
- Agregue propiedades de configuración para la aplicación gestionada con el motor AWS Blu Age

Estructura de las aplicaciones gestionadas de AWS Blu Age

Si utiliza el patrón de refactorización de AWS Blu Age, el motor de ejecución de AWS Blu Age espera la siguiente estructura dentro de la `application-name` carpeta de su depósito de S3:



config

Contiene los archivos YAML del proyecto. Se trata de los archivos YAML específicos de su aplicación, que suelen tener un nombre similar `application-planetsdemo.yaml` y no el `application-main.yaml` archivo que AWS Mainframe Modernization proporciona y configura automáticamente para usted.

webapps

Contiene los archivos `war` de la aplicación. Estos archivos son el resultado del proceso de modernización.

Una aplicación también puede tener las siguientes carpetas opcionales:

jics/sql

Contiene el script `initJics.sql` que inicializa la base de datos JICS para su aplicación.

scripts

Contiene scripts de la aplicación, que también puede proporcionar directamente dentro de los archivos `war`.

sql

Contiene archivos de SQL de la aplicación, que también puede proporcionar directamente dentro de los archivos `war`.

lnk

Contiene archivos de LNK de la aplicación, que también puede proporcionar directamente dentro de los archivos `war`.

extra

Contiene archivos jar que pueden proporcionar capacidades adicionales para la aplicación modernizada.

Administración de las opciones Java de una aplicación

Para administrar algunas opciones Java para la aplicación, agregue un archivo de propiedades llamado `tomcat.properties` a la carpeta `application-name`. Este archivo puede tener tres propiedades: `xms`, que especifica el consumo mínimo de memoria de Java, `xmx`, que especifica el consumo máximo de memoria de Java y `dnscachettl`, que administra la duración de la caché para resoluciones de DNS. El siguiente ejemplo muestra el contenido de un archivo `tomcat.properties` válido:

```
xms=512M
xmx=1G
dnscachettl=5
```

Los valores que especifique para las dos primeras propiedades pueden estar en cualquiera de las siguientes unidades:

- Bytes: no especifique una unidad.
- Kilobytes: añada una K al valor.
- Megabytes: añada una M al valor.
- Gigabytes: añada una G al valor.

El valor de la tercera propiedad representa la duración de la caché en segundos y puede tener un valor de -1 (almacenar en caché para siempre) o puede oscilar entre 0 (no almacenar nunca en caché) y 999. En el contexto de las implementaciones de aplicaciones administradas, el valor predeterminado es -1.

Configuración del acceso a las utilidades para las aplicaciones administradas

Al refactorizar una aplicación de mainframe con AWS Blu Age, es posible que necesites proporcionar soporte para varios programas utilitarios de plataformas antiguas, como IDCAMS, INFUTILB, SORT, etc., si tu aplicación depende de ellos. AWS La refactorización de Blu Age proporciona este acceso con una aplicación web dedicada que se implementa junto con las aplicaciones modernizadas. Esta aplicación web requiere un archivo de configuración, `application-utility-pgm.yml`, que usted debe proporcionar. Si no proporciona este archivo de configuración, la aplicación web no se podrá implementar junto con la suya y no estará disponible.

Temas

- Propiedades de configuración

En este tema se describen todas las propiedades posibles que puede especificar en el archivo de configuración `application-utility-pgm.yml`, junto con sus valores predeterminados. En este tema se describen las propiedades obligatorias y opcionales. A continuación se muestra un archivo de configuración completo de ejemplo. Incluye las propiedades en el orden que recomendamos. Puede utilizar este ejemplo como punto de partida para su propio archivo de configuración.

```
# If the datasource support mode is not static-xa, spring JTA transactions
 autoconfiguration must be disabled
 spring.jta.enabled: false
 logging.config: 'classpath:logback-utility.xml'

 # Encoding
 encoding: cp1047

 # Encoding to be used by INFUTILB and DSNUTILB to generate and read SYSPUNCH files
 sysPunchEncoding: cp1047

 # Utility database access
 spring.aws.client.datasources.primary.secret: `arn:aws:secretsmanager:us-
west-2:111122223333:secret:business-FfmXLG`

 treatLargeNumberAsInteger: false

 # Zoned mode : valid values = EBCDIC_STRICT, EBCDIC_MODIFIED, AS400
 zonedMode: EBCDIC_STRICT
```

```
jcl.type: mvs

# Unload properties
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntaxe to specify the byte value
unload:
 sqlCodePointShift: 384
 nbi:
   whenNull: "6F"
   whenNotNull: "00"
 useDatabaseConfiguration: false
 format:
   date: MM/dd/yyyy
   time: HH.mm.ss
   timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
 chunkSize:500
 fetchSize: 500
 varCharIsNull: false
 columnFiller: space

# Load properties
# Batch size for DSNUTILB Load Task
load:
 sqlCodePointShift: 384
 batchSize: 500
 format:
   localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
   dbDate: yyyy-MM-dd
   localTime: 'HH:mm:ss|HH.mm.ss'
   dbTime: 'HH:mm:ss'

table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

Propiedades de configuración

Puede especificar las siguientes propiedades en su archivo de configuración.

spring.jta.enabled

(Opcional) Controla si está habilitada la compatibilidad con JTA. Para las utilidades, recomendamos que establezca este valor en `false`.

```
spring.jta.enabled : false
```

logging.config

(Obligatorio) Especifica la ruta al archivo de configuración del registrador dedicado. Le recomendamos utilizar el nombre `logback-utility.xml` y proporcionar este archivo como parte de la aplicación modernizada. La forma habitual de organizar estos archivos es colocar todos los archivos de configuración del registrador en el mismo lugar, normalmente en la subcarpeta `/config/logback`, donde `/config` es la carpeta que contiene los archivos de configuración YAML. Para obtener más información, consulte Chapter 3: Logback configuration en la documentación de Logback.

```
logging.config : classpath:logback-utility.xml
```

encoding

(Obligatorio) Especifica el conjunto de caracteres que utiliza el programa de utilidades. En la mayoría de los casos, al migrar desde plataformas z/OS, este conjunto de caracteres es una variante del EBCDIC y debe coincidir con el conjunto de caracteres configurado para las aplicaciones modernizadas. Si no se establece, el valor predeterminado es `ASCII`.

```
encoding : cp1047
```

sysPunchEncoding

(Opcional) Especifica el conjunto de caracteres que utilizan INFUTILB y DSNUTILB para generar y leer los archivos SYSPUNCH. Si utiliza los archivos SYSPUNCH de la plataforma anterior tal como están, este valor debe ser una variante EBCDIC. Si no se establece, el valor predeterminado es `ASCII`.

```
sysPunchEncoding : cp1047
```

Configuración del origen de datos

Algunas utilidades relacionadas con las bases de datos, como LOAD y UNLOAD, requieren acceder a una base de datos de destino a través de un origen de datos. Al igual que otras definiciones de fuentes de datos incluidas en AWS Mainframe Modernization, este acceso requiere su uso. AWS Secrets Manager Las propiedades que señalan a los secretos adecuados de Secrets Manager son las siguientes:

Origen de datos principal

Esta es la base de datos de aplicaciones empresariales principal.

spring.aws.client.datasources.primary.secret

(Opcional) Especifica el secreto de Secrets Manager que contiene las propiedades del origen de datos.

```
spring.aws.client.datasources.primary.secret: datasource-secret-ARN
```

spring.aws.client.datasources.primary.dbname

(Opcional) Especifica el nombre de la base de datos de destino si este no se proporciona directamente en el secreto de la base de datos, junto con la propiedad `dbname`.

```
spring.aws.client.datasources.primary.dbname: target-database-name
```

spring.aws.client.datasources.primary.type

(Opcional) Especifica el nombre completo de la implementación del grupo de conexión que se usará. El valor predeterminado es `com.zaxxer.hikari.HikariDataSource`.

```
spring.aws.client.datasources.primary.type: target-datasource-type
```

Si el tipo de origen de datos principal es `com.zaxxer.hikari.HikariDataSource`, puede especificar propiedades adicionales de la siguiente manera:

spring.datasource.primary.[property_name]

(Opcional) Puede usar este formato para especificar propiedades adicionales para configurar la implementación de un grupo de conexiones de origen de datos principal.

A continuación, se muestra un ejemplo de un origen de datos principal de tipo `com.zaxxer.hikari.HikariDataSource`.

```
spring:
   datasource:
       primary:
         autoCommit: XXXX
         maximumPoolSize: XXXX
         keepaliveTime: XXXX
         minimumIdle: XXXX
         idleTimeout: XXXX
         connectionTimeout: XXXX
         maxLifetime: XXXX
```

Otros orígenes de datos de utilidad

Además del origen de datos principal, puede proporcionar otros orígenes de datos de utilidad.

spring.aws.client.utility.pgm.datasources.names

(Opcional) Especifica la lista de nombres de orígenes de datos de utilidad.

```
spring.aws.client.utility.pgm.datasources.names: dsname1, dsname2, dsname3
```

spring.aws.client.utility.pgm.datasources.[dsname].secret

(Opcional) Especifica el ARN del secreto en SSM que aloja las propiedades del origen de datos. Proporcione [dsname] en la lista de nombres especificada en `spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.secret: datasource-secret-ARN
```

spring.aws.client.utility.pgm.datasources.[dsname].dbname

(Opcional) Especifica el nombre de la base de datos de destino si el nombre de la base de datos no se proporciona directamente en el secreto de la base de datos mediante la propiedad `dbname`. Proporcione [dsname] en la lista de nombres especificada en `spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.dbname: target-database-name
```

spring.aws.client.utility.pgm.datasources.[dsname].type

(Opcional) Especifica el nombre completo de la implementación del grupo de conexión que se usará. El valor predeterminado es `com.zaxxer.hikari.HikariDataSource`. Proporcione [dsname] en la lista de nombres especificada en `spring.aws.client.utility.pgm.datasources.names`.

```
spring.aws.client.utility.pgm.datasources.dsname1.type: target-datasource-type
```

Si el tipo de origen de datos de utilidad es `com.zaxxer.hikari.HikariDataSource`, puede proporcionar propiedades adicionales de la siguiente manera:

spring.datasource.[dsname].[property_name]

(Opcional) Especifica una colección de propiedades adicionales para configurar la implementación de un grupo de conexiones de orígenes de datos de utilidad. Proporcione [dsname] en la lista de nombres especificada en `spring.aws.client.utility.pgm.datasources.names`. Especifique las propiedades en el siguiente formato: `property_name : value`

A continuación se muestra un ejemplo de orígenes de datos de utilidad adicionales del tipo `com.zaxxer.hikari.HikariDataSource`:

```
spring:
   datasource:
       dsname1:
         connectionTimeout: XXXX
         maxLifetime: XXXX
       dsname2:
         connectionTimeout: XXXX
         maxLifetime: XXXX
       dsname3:
         connectionTimeout: XXXX
         maxLifetime: XXXX
```

treatLargeNumberAsInteger

(Opcional) Relacionado con las especificaciones del motor de base de datos Oracle y el uso de las DSNTEP2 DSNTEP4 utilidades. Si establece este indicador en true, los números grandes que provienen de la base de datos Oracle (NUMBER (38,0)) se tratan como números enteros. Predeterminado: `false`

```
treatLargeNumberAsInteger : false
```

zonedMode

(Opcional) Establece el modo zonificado para codificar o decodificar los tipos de datos zonificados. Esta configuración influye en la forma en que se representan los dígitos de los signos. Los siguientes valores son válidos:

- EBCDIC_STRICT: predeterminado Utilice una definición estricta para la gestión de signos. Dependiendo de si el conjunto de caracteres es EBCDIC o ASCII, la representación de dígitos con signo utiliza los siguientes caracteres:
  - Caracteres EBCDIC que corresponden a bytes (`Cn+Dn`) para representar rangos de dígitos positivos y negativos (`+0` a `+9`, `-0` a `-9`). Los caracteres se muestran como `{`,`A` a `I`, `}`, `J` a `R`
  - Caracteres ASCII que corresponden a bytes (`3n+7n`) para representar rangos de dígitos positivos y negativos (`+0` a `+9`, `-0` a `-9`). Los caracteres se muestran como `0` a `9`, `p` a `y`
- EBCDIC_MODIFIED: utilice una definición modificada para la gestión de los signos. Tanto para EBDIC como para ASCII, la misma lista de caracteres representa los dígitos del signo, es decir, `+0` a `+9` se asignan a `{` + `A` a `I` y `-0` a `-9` se asignan a `}` + `J` a `R`. \
- AS400: Úselo para activos heredados modernizados que provienen de las plataformas iSeries (AS400).

```
zonedMode:EBCDIC_STRICT
```

jcl.type

(Opcional) Indica el tipo heredado de los scripts JCL modernizados. La utilidad IDCAMS utiliza esta configuración para personalizar el código de retorno si el JCL que invoca es de tipo `vse`. Los valores válidos son los siguientes:

- `mvs` (predeterminado)
- `vse`

```
jcl.type : mvs
```

Propiedades relacionadas con las utilidades de descarga de bases de datos

Utilice estas propiedades para configurar las utilidades que descargan tablas de bases de datos en conjuntos de datos. Todas las demás propiedades son opcionales.

En este ejemplo se muestran todas las propiedades de descarga posibles.

```
# Unload properties
 # For date/time: if use database configuration is enabled, formats are ignored
 # For nbi; use hexadecimal syntaxe to specify the byte value
 unload:
 sqlCodePointShift: 0
 nbi:
 whenNull: "6F"
 whenNotNull: "00"
 useDatabaseConfiguration: false
 format:
 date: MM/dd/yyyy
 time: HH.mm.ss
 timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
 chunkSize: 0
 fetchSize: 0
 varCharIsNull: false
 columnFiller: space
```

sqlCodePointShift

(Opcional) Especifica un valor entero que representa el desplazamiento de puntos del código SQL utilizado en los datos. El valor predeterminado es 0. Esto significa que no se produce ningún cambio de punto de código. Alinee esta configuración con el parámetro de cambio de punto del código SQL que se utiliza en las aplicaciones modernizadas. Cuando se utiliza el desplazamiento de puntos de código, el valor más común para este parámetro es 384.

```
unload.sqlCodePointShift: 0
```

nbi

(Opcional) Especifica un byte indicador nulo. Se trata de un valor hexadecimal (en forma de cadena) que se añade a la derecha del valor de los datos. Los dos valores posibles son los siguientes:

- whenNull: agrega el valor hexadecimal cuando el valor de los datos es nulo. El valor predeterminado es 6`. A veces se utiliza el valor alto FF en su lugar.

  ```
  unload.nbi.whenNull: "6F"
  ```

- whenNotNull: Agregue el valor hexadecimal cuando el valor de los datos no sea nulo, pero la columna sí lo sea. Valor predeterminado: `00` (valor bajo).

```
unload.nbi.whenNotNull: "00"
```

useDatabaseConfiguration

(Opcional) Especifica las propiedades de formato de fecha y hora. Se utiliza para tratar los objetos de fecha y hora en las consultas UNLOAD. El valor predeterminado es `false`.

- Si se establece en `true`, utiliza las propiedades `pgmDateFormat`, `pgmTimeFormat` y `pgmTimestampFormat` del archivo de configuración principal (`application-main.yml`).
- Si se establece en `false`, utiliza las siguientes propiedades de formato de fecha y hora:
  - `unload.format.date`: especifica un patrón de formato de fecha. El valor predeterminado es `MM/dd/yyyy`.
  - `unload.format.time`: especifica un patrón de formato de hora. El valor predeterminado es `HH.mm.ss`.
  - `unload.format.timestamp`: especifica un patrón de formato de marca de tiempo. El valor predeterminado es `yyyy-MM-dd-HH.mm.ss.SSSSSS`.

chunkSize

(Opcional) Especifica el tamaño de los fragmentos de datos que se utilizan para crear conjuntos de datos SYSREC. Estos conjuntos de datos son el objetivo de la operación de descarga de conjuntos de datos, con operaciones en paralelo. El valor predeterminado es `0` (sin fragmentos).

```
unload.chunkSize:0
```

fetchSize

(Opcional) Especifica el tamaño de la recuperación de datos. El valor es el número de registros que se van a recuperar al mismo tiempo cuando se utiliza una estrategia de fragmentos de datos. Predeterminado: `0`.

```
unload.fetchSize:0
```

varCharIsNulo

(Opcional) Especifica cómo gestionar una columna varchar que no admite valores null con contenido en blanco. El valor predeterminado es `false`.

Si establece este valor en `true`, el contenido de la columna se trata como una cadena vacía a efectos de descarga, en lugar de como una cadena de espacios individuales. Establezca este indicador como `true` solo para el caso del motor de base de datos Oracle.

```
unload.varCharIsNull: false
```

columnFiller

(Opcional) Especifica el valor que se utilizará para rellenar las columnas descargadas en las columnas de varchar. Los valores posibles son espacios o valores bajos. Valor predeterminado: space

```
unload.columnFiller: space
```

Propiedades relacionadas con la carga de base de datos

Utilice estas propiedades para configurar las utilidades que cargan registros de conjuntos de datos en una base de datos de destino, por ejemplo, DSNUTILB. Todas las demás propiedades son opcionales.

En este ejemplo se muestran todas las propiedades de carga posibles.

```
# Load properties
 # Batch size for DSNUTILB Load Task
 load:
 sqlCodePointShift: 384
 batchSize: 500
 format:
 localDate: dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd
 dbDate: yyyy-MM-dd
 localTime: HH:mm:ss|HH.mm.ss
 dbTime: HH:mm:ss

 table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
```

sqlCodePointTurno

(Opcional) Especifica un valor entero que representa el desplazamiento de puntos del código SQL utilizado en los datos. El valor predeterminado es 0, lo que significa que las aplicaciones no desplazan los puntos de código. Alinee esta configuración con el parámetro de cambio de punto del código SQL que se utiliza en las aplicaciones modernizadas. Cuando se utiliza el desplazamiento de puntos de código, el valor más común para este parámetro es 384.

```
load.sqlCodePointShift : 384
```

batchSize

(Opcional) Especifica un valor entero que representa el número de registros que se tratarán antes de enviar una instrucción de lote real a la base de datos. El valor predeterminado es 0.

```
load.batchSize: 500
```

format

(Opcional) Especifica los patrones de formato de fecha y hora que se utilizarán para las conversiones de fecha y hora durante las operaciones de carga de la base de datos.

- `load.format.localDate`: patrón de formato de fecha local. El valor predeterminado es `dd.MM.yyyy|dd/MM/yyyy|yyyy-MM-dd`.
- `load.format.dbDate`: patrón de formato de fecha de la base de datos. El valor predeterminado es `yyyy-MM-dd`.
- `load.format.localTime`: patrón de formato de hora local. El valor predeterminado es `HH:mm:ss|HH.mm.ss`.
- `load.format.dbTime`: patrón de formato de hora de la base de datos. El valor predeterminado es `HH:mm:ss`.

table-mappings

(Opcional) Especifica una colección de asignaciones proporcionadas por el cliente entre nombres de tablas antiguos y modernos. El programa de utilidades DSNUTILB usa estas asignaciones.

Especifique los valores en el siguiente formato: MODERN_TABLE_NAME : LEGACY_TABLE_NAME

A continuación se muestra un ejemplo:

```
table-mappings:
 TABLE_1_NAME : LEGACY_TABLE_1_NAME
 TABLE_2_NAME : LEGACY_TABLE_2_NAME
 ...
 TABLE_*N*_NAME : LEGACY_TABLE_*N*_NAME
```

> **Note**
>
> Cuando se inicia la aplicación de la utilidad, registra de forma explícita todas las asignaciones proporcionadas.

Agregue propiedades de configuración para la aplicación gestionada con el motor AWS Blu Age

Puedes añadir un archivo a la `config` carpeta de tu aplicación refactorizada que te dará acceso a las nuevas funciones del motor de ejecución Blu Age. AWS Debe asignar un nombre a este archivo `user-properties.yml`. Este archivo no reemplaza la definición de la aplicación, sino que la amplía. En este tema se describen las propiedades que puede incluir en el archivo `user-properties.yml`.

> **Note**
>
> No puede cambiar algunos parámetros porque están controlados por la modernización del AWS mainframe o por la definición de la aplicación. Todos los parámetros especificados en la definición de la aplicación para su aplicación tienen prioridad sobre los parámetros que especifique en `user-properties.yml`.

Para obtener más información sobre las aplicaciones refactorizadas, consulte Estructura de las aplicaciones gestionadas de AWS Blu Age.

El siguiente diagrama muestra dónde ubicar el `user-properties.yml` archivo dentro de la estructura de la aplicación de ejemplo AWS Blu Age, PlanetsDemo.

```
PlanetsDemo-v1/
   ## config/
   #  ## application-PlanetsDemo.yml
```

```
#  ## user-properties.yml
## jics/
## webapps/
```

Referencia de propiedades de configuración

Esta es la lista de propiedades disponibles. Todos los parámetros son opcionales.

Temas

- Propiedades de la aplicación Gapwalk
- Propiedades del script de procesamiento por lotes de Gapwalk
- Propiedades de Gapwalk Blugen
- Propiedades del comando Gapwalk CL
- Propiedades del ejecutor Gapwalk CL
- Propiedades JHDB de Gapwalk
- Propiedades JICS de Gapwalk
- Propiedades de tiempo de ejecución de Gapwalk
- Propiedades del programa de utilidades Gapwalk
- Otras propiedades

Propiedades de la aplicación Gapwalk

bluesam.fileLoading.commitInterval

Opcional. El intervalo de confirmación de Blusam.

Tipo: número

Predeterminado: 100000

card.encoding

Opcional. Codificación de tarjetas: para usar con `useControlMVariable`.

Tipo: cadena

Predeterminado: CP1145

checkinputfilesize

Opcional. Especifica si se debe activar una comprobación si el tamaño del archivo es múltiplo del tamaño del registro.

Tipo: booleano

Predeterminado: false

database.cursor.overflow.allowed

Opcional. Especifica si se debe permitir que el cursor se desborde. Configúrelo en `true` para realizar una siguiente llamada en el cursor, sea cual sea su posición. Configúrelo en `false` para comprobar si el cursor está en la última posición antes de realizar una siguiente llamada con el cursor. Actívela solo si el cursor es SCROLLABLE (SENSITIVE o INSENSITIVE)

Tipo: booleano

Predeterminado: true

DataSimplifier. onInvalidNumericDatos

Opcional. Cómo reaccionar al decodificar datos numéricos no válidos. Los valores permitidos son: `reject`, `toleratespaces`, `toleratespaceslowvalues`, `toleratemost`.

Tipo: cadena

Predeterminado: rechazar

defaultKeepExistingArchivos

Opcional. Especifica si se debe establecer el valor anterior predeterminado del conjunto de datos.

Tipo: booleano

Predeterminado: false

disposition.checkexistence

Opcional. Especifica si se debe publicar una verificación de la existencia del archivo para el conjunto de datos con DISP SHR u OLD.

Tipo: booleano

Predeterminado: false

externalSort.threshold

Opcional. El umbral de clasificación: cuándo cambiar a una ordenación externa (fusión).

Tipo: cadena

Predeterminado: null

```
externalSort.threshold: 12MB
```

blockSizeDefault

Opcional. El tamaño de bloque predeterminado que se utilizará para los bytes de BDW.

Tipo: número

Predeterminado: 32760

```
blockSizeDefault: 32760
```

forceHR

Opcional. Especifica si se debe utilizar SYSPRINT inteligible, ya sea en la consola o en la salida de un archivo.

Tipo: booleano

Predeterminado: false

forcedDate

Opcional. Fuerza una fecha y hora específicas en la base de datos. Úselo solo durante el desarrollo y las pruebas.

Predeterminado: null

```
forcedDate: 2022-08-26T12:59:58.123456+01:57
```

frozenDate

Opcional. Congela la fecha y la hora de la base de datos. Úselo solo durante el desarrollo y las pruebas.

Predeterminado: false

```
frozenDate: false
```

ims.messages.extendedSize

Opcional. Especifica si se debe establecer el tamaño extendido en los mensajes ims.

Tipo: booleano

Predeterminado: false

lockTimeout

Opcional. El tiempo de espera en milisegundos de una transacción cuando no se puede adquirir un bloqueo dentro de un período de tiempo específico.

Tipo: número

Predeterminado: 500

mapTransfo.prefixes

Opcional. Lista de prefijos que se utilizarán al transformar las variables de ControlM. Cada uno separado por comas.

Tipo: cadena

Predeterminado: &,@,%%

consulta. useConcatCondition

Opcional. Especifica si la condición clave se crea mediante concatenación de claves o no.

Tipo: booleano

Predeterminado: false

rollbackOnRTE

Opcional. Especifica si se debe anular la transacción implícita de la unidad de ejecución en las excepciones de tiempo de ejecución.

Tipo: booleano

Predeterminado: false

sctThreadLimit

Opcional. El límite de subprocesos para activar scripts.

Tipo: número

Valor predeterminado: 5

sqlCodePointTurno

Opcional. El cambio de punto del código sql. Cambia el punto de código de los caracteres de control que podemos encontrar al migrar datos de rdbms antiguos a un rdbms moderno. Por ejemplo, puede especificar `384` para que coincida con un carácter `\u0180`.

Tipo: número

Predeterminado: 0

sqlIntegerOverflowPermitido

Opcional. Especifica si se permite el desbordamiento de enteros de SQL, es decir, si se permite colocar valores más grandes en la variable host.

Tipo: booleano

Predeterminado: false

stepFailWhenUna curva

Opcional. Especifica si se debe generar un abend si un paso falla o completa la ejecución.

Tipo: booleano

Predeterminado: true

stopExecutionWhenProgNotFound

Opcional. Especifica si se debe detener la ejecución si no se encuentra un programa. Si se establece en `true`, interrumpe la ejecución si no se encuentra ningún programa.

Tipo: booleano

Predeterminado: true

uppercaseUserInput

Opcional. Especifica si la entrada del usuario debe estar en mayúsculas.

Tipo: booleano

Predeterminado: true

Utilice Control MVariable

Opcional. Especifica si se debe utilizar la especificación control-M para el reemplazo de variables.

Tipo: booleano

Predeterminado: false

jcl.checkpoint.expireTimeout

Opcional. Especifica el tiempo que se deben retener los puntos de comprobación de JCL en el proveedor de persistencia o en el registro en memoria.

Tipo: número

Predeterminado: -1

jcl.checkpoint. expireTimeoutUnit

Opcional. Especifica la unidad de duración de tiempo de la propiedad `jcl.checkpoint.expireTimeout`. Valores constantes de enumeración admitidos: `java.util.concurrent.TimeUnit`.

Tipo: cadena

Predeterminado: SECONDS

Propiedades del script de procesamiento por lotes de Gapwalk

encoding

Opcional. La codificación utilizada en los proyectos de scripts de procesamiento por lotes (no con groovy). Espera una codificación válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: cadena

Predeterminado: ASCII

Propiedades de Gapwalk Blugen

managers.trancode

Opcional. El mapeo de transcodificación del administrador de diálogos. Le permite asignar un código de transacción del JICS a un administrador de diálogos. El formato esperado es `trancode1:dialogManager1;trancode2:dialogManager2;`.

Tipo: cadena

Predeterminado: null

`managers.trancode: OR12:MYDIALOG1`

Propiedades del comando Gapwalk CL

commands-off

Opcional. Lista de comandos a desactivar, separados por comas. Los valores permitidos son `PGM_BASIC`, `RCVMSG`, `SNDRCVF`, `CHGVAR`, `QCLRDTAQ`, `RTVJOBA`, `ADDLFM`, `ADDPFM`, `RCVF`, `OVRDBF`, `DLTOVR`, `CPYF`, `SNDDTAQ`. Útil cuando se desea deshabilitar o sobrescribir un programa existente. `PGM_BASIC`es un programa específico de AWS Blu Age Runtime diseñado con fines de depuración.

Tipo: cadena

Predeterminado: null

spring.datasource.primary.jndi-name

Opcional. El origen de datos principal de la Interfaz de Nombrado y Directorio Java (jndi).

Tipo: cadena

Predeterminado: jdbc/primary

zonedMode

Opcional. El modo para codificar o decodificar tipos de datos zonificados. Los valores permitidos son `EBCDIC_STRICT` / `EBCDIC_MODIFIED` / `AS400`.

Tipo: cadena

Predeterminado: EBCDIC_STRICT

Propiedades del ejecutor Gapwalk CL

cl.configuration.context.encoding

Opcional. La codificación de los archivos CL. Espera una codificación válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: cadena

Predeterminado: CP297

cl.zonedMode

Opcional. El modo para codificar o decodificar los comandos del lenguaje de control (CL). Los valores permitidos son `EBCDIC_STRICT` / `EBCDIC_MODIFIED` / `AS400`.

Tipo: cadena

Predeterminado: EBCDIC_STRICT

Propiedades JHDB de Gapwalk

ims.programas

Opcional. Lista de programas de IMS que se van a utilizar. Separe cada parámetro con punto y coma (`;`) y cada transacción con una coma (`,`). Por ejemplo: `ims.programs: PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`

Tipo: cadena

Predeterminado: null

jhdb.checkpointPath

Opcional. Si `jhdb.checkpointPersistence` no es `none`, este parámetro le permite configurar la ruta de persistencia del punto de comprobación (ubicación de almacenamiento del archivo checkpoint.dat), todos los datos de los puntos de comprobación contenidos en el registro se serializan y se hace una copia de seguridad en un archivo (checkpoint.dat) ubicado en la carpeta proporcionada. Tenga en cuenta que esta copia de seguridad solo afecta a los datos de los puntos de control (scriptId, stepId, posición de la base de datos y área del punto de control).

Tipo: cadena

Predeterminado: file:./setup/

jhdb.checkpointPersistence

Opcional. El modo de persistencia del punto de comprobación. Los valores permitidos son `none` / `add` / `end`. Utilice `add` para conservar los puntos de comprobación cuando se crea y se

agrega uno nuevo al registro. Utilice `end` para conservar el punto de comprobación al cerrar el servidor. Cualquier otro valor deshabilita la persistencia. Tenga en cuenta que cada vez que se añada un nuevo punto de control al registro, todos los puntos de comprobación existentes se serializarán y el archivo se borrará. No es un anexo a los datos existentes en el archivo. Por lo tanto, dependiendo del número de puntos de comprobación, puede tener algún efecto en el rendimiento.

Tipo: cadena

Predeterminado: none

jhdb.configuration.context.encoding

Opcional. La codificación JHDB (base de datos jerárquica de Java). Espera una cadena de codificación válida `CP1047`, `IBM930`, `ASCII`, `UTF-8`...

Tipo: cadena

Predeterminado: CP297

jhdb. identificationCardData

Opcional. Se utiliza para codificar algunos “datos de la tarjeta de identificación del operador” en el campo MID designado por el parámetro CARD.

Tipo: cadena

Predeterminado: “”

jhdb.lterm

Opcional. Le permiten forzar un ID de terminal lógico común en el caso de una emulación de IMS. Si no se establece, se utiliza sessionId.

Tipo: cadena

Predeterminado: null

jhdb.metadata.extrapath

Parámetro de configuración que especifica una carpeta raíz adicional específica para el tiempo de ejecución para las carpetas psbs y dbds.

Tipo: cadena

Predeterminado: file:./setup/

> **Note**
>
> Actualmente, debido a las restricciones de implementación, debe copiar sus directorios dbds y psbs en el directorio config de su aplicación o en un subdirectorio del directorio config: por ejemplo, config/setup
>
> ```
> config
> |- setup
>    |- dbds
>    |- psbs
> ```
>
> y configurarlo en application-jhdb.yml
> `jhdb.metadata.extrapath: file: ./config/setup/`

jhdb.navigation.cachenexts

Opcional. La duración de la caché (en milisegundos) utilizada en la navegación jerárquica de un RDBMS.

Tipo: número

Predeterminado: 5000

jhdb.consulta. limitJoinUsage

Opcional. Especifica si se debe utilizar el parámetro de límite de uso de uniones en los gráficos del RDBMS.

Tipo: booleano

Predeterminado: true

jhdb. use-db-prefix

Opcional. Especifica si se debe habilitar un prefijo de base de datos en la navegación jerárquica de un RDBMS.

Tipo: booleano

Predeterminado: true

Propiedades JICS de Gapwalk

jics.data. dataJsonInitUbicación

Opcional. Ubicación del archivo json preparado por el analizador a partir del análisis del CSD y utilizado para inicializar la base de datos jics,

Tipo: cadena

Predeterminado: “”

jics.db. dataScriptLocation

Opcional. Ubicación del script initJics.sql, preparado por el analizador a partir del análisis de las exportaciones de CSD desde el ordenador central.

Tipo: cadena

Predeterminado: “”

jics.db. dataTestQueryUbicación

Opcional. Ubicación de un script sql que contiene una única consulta sql que se espera que devuelva un recuento de objetos (por ejemplo: contar el número de registros de la tabla del programa jics). Si el recuento es 0, la base de datos se cargará mediante el script `jics.db.dataScriptLocation`; de lo contrario, se omitirá la carga de la base de datos.

Tipo: cadena

Predeterminado: “”

jics.db. ddlScriptLocation

Opcional. La ubicación del script ddl de Jics. Permite iniciar el esquema de la base de datos jics mediante un script .sql.

Tipo: cadena

Predeterminado: “”

```
jics.db.ddlScriptLocation: ./jics/sql/jics.sql
```

jics.db. schemaTestQueryUbicación

Opcional. Ubicación del archivo sql que debe contener una consulta única que devuelva el número de objetos del esquema jics (si los hay).

Tipo: cadena

Predeterminado: “”

jics. runUnitLauncherPool. Habilitar

Opcional. Especifica si se debe activar el grupo de lanzadores de unidades de ejecución en JICS.

Tipo: booleano

Predeterminado: false

jics. runUnitLauncherTamaño de la piscina

Opcional. El tamaño del conjunto de lanzadores de unidades de ejecución en JICS.

Tipo: número

Predeterminado: 20

jics. runUnitLauncherPool. Intervalo de validación

Opcional: el intervalo de validación del conjunto de lanzadores de unidades de ejecución en JICS, expresado en milisegundos.

Tipo: número

Predeterminado: 1000

jics.queues.sqs.region

Opcional. El Región de AWS para Amazon SQS, utilizado en JICS. Se recomienda configurar la misma región de la aplicación desplegada para garantizar el rendimiento, pero no es obligatorio.

Tipo: cadena

Predeterminado: eu-west-1

jics.xa.agent.timeout

Opcional. Define la duración máxima para que el agente xa responsable de gestionar las transacciones distribuidas complete sus operaciones.

Tipo: número

Predeterminado: null

mq.queues.sqs.region

Opcional. El Región de AWS para el servicio Amazon SQS MQ.

Tipo: cadena

Predeterminado: eu-west-3

Ejecutor de tareas. allowCoreThreadTimeOut

Opcional. Especifica si se debe permitir que se agote el tiempo de espera de los subprocesos principales en JCIS. Esto permite un crecimiento y una reducción dinámicos incluso en combinación con una cola distinta de cero (ya que el tamaño máximo del grupo solo aumentará una vez que la cola esté llena).

Tipo: booleano

Predeterminado: false

Ejecutor de tareas. corePoolSize

Opcional. Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño del grupo de núcleos.

Tipo: número

Valor predeterminado: 5

Ejecutor de tareas. maxPoolSize

Opcional. Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño máximo del grupo (número máximo de subprocesos paralelos).

Tipo: número

Predeterminado: 10

taskExecutor.queueCapacity

Opcional. Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño de la cola. (= número máximo de transacciones pendientes cuando se alcanza `taskExecutor.maxPoolSize`)

Tipo: número

Predeterminado: 50

Propiedades de tiempo de ejecución de Gapwalk

cacheMetadata

Opcional. Especifica si se deben almacenar en caché los metadatos de la base de datos.

Tipo: booleano

Predeterminado: true

check-groovy-file

Opcional. Especifica si se debe comprobar el contenido de los archivos groovy antes de registrarlos.

Tipo: booleano

Predeterminado: true

databaseStatistics

Opcional. Especifica si se permite a los compiladores de SQL recopilar y mostrar información estadística.

Tipo: booleano

Predeterminado: false

dateTimeFormat

Opcional. dateTimeFormat Describe cómo incluir el tipo de fecha y hora de la base de datos en las entidades simplificadoras de datos. Los valores permitidos son `ISO` / `EUR` / `USA` / `LOCAL`

Tipo: cadena

Predeterminado: ISO

dbDateFormat

Opcional. El formato de fecha de destino de la base de datos.

Tipo: cadena

Predeterminado: yyyy-MM-dd

dbTimeFormat

Opcional. El formato de hora de destino de la base de datos.

Tipo: cadena

Predeterminado: HH:mm:ss

dbTimestampFormat

Opcional. El formato de hora de destino de la base de datos.

Tipo: cadena

Predeterminado: yyyy-MM-dd hh:mm:ss.ssssss

fetchSize

Opcional. El valor fetchSize de los cursores. Se usa cuando se obtienen datos con fragmentos mediante utilidades de carga o descarga.

Tipo: número

Predeterminado: 10

Forzar la desactivación SQLTrim StringType

Opcional. Especifica si se debe deshabilitar el recorte de todos los parámetros de cadena SQL.

Tipo: booleano

Predeterminado: false

localDateFormat

Opcional. Lista de formatos de fecha locales. Separe cada formato con |.

Tipo: cadena

localTimeFormat

Opcional. Lista de formatos de hora locales. Separe cada formato con |.

Tipo: cadena

localTimestampFormat

Opcional. Lista de formatos de marca temporal locales. Separe cada formato con |.

Tipo: cadena

Predeterminado:

pgmDateFormat

Opcional. El formato de fecha y hora utilizado en los programas.

Tipo: cadena

Predeterminado: yyyy-MM-dd

pgmTimeFormat

Opcional. Formato de hora utilizado para la ejecución de los pgm (programas).

Tipo: cadena

Predeterminado: HH.mm.ss

pgmTimestampFormat

Opcional. Formato de marca temporal

Tipo: cadena

Predeterminado: yyyy-MM-dd-HH .mm.ss.ssssss

Propiedades del programa de utilidades Gapwalk

jcl.type

Opcional. tipo de archivo `.jcl`. Los valores permitidos son `jcl` / `vse`. Los comandos PRINT/ REPRO de la utilidad IDCAMS devuelven 4 si el archivo está vacío en el caso de jcl que no es vse.

Tipo: cadena

Predeterminado: mvs

listcat.variablelengthpreprocessor.enabled

Opcional. Especifica si se debe habilitar el preprocesador de longitud variable para el comando LISTCAT.

Tipo: booleano

Predeterminado: false

listcat.variablelengthpreprocessor.type

Opcional. El tipo de objetos que contiene el archivo listcat, si está activado `listcat.variablelengthpreprocessor.enabled`. Los valores permitidos son `rdw` / `bdw`.

Tipo: cadena

Predeterminado: rdw

load.batchSize

Opcional. El tamaño del lote de la utilidad de carga.

Tipo: número

Predeterminado: 0

load.format.dbDate

Opcional. El formato de base de datos de la utilidad de carga que se va a utilizar.

Tipo: cadena

Predeterminado: yyyy-MM-dd

load.format.dbTime

Opcional. La hora de base de datos de la utilidad de carga que se va a utilizar.

Tipo: cadena

Predeterminado: HH:mm:ss

load.format.localDate

Opcional. El formato de fecha local de la utilidad de carga que se va a utilizar.

Tipo: cadena

Predeterminado: dd/MM/yyyy dd.mm.yyyy| | aaaa-mm-dd

load.format.localTime

Opcional. El formato de hora local de la utilidad de carga que se va a utilizar.

Tipo: cadena

Predeterminado: hh:mm:ss|hh.mm.ss

cargar. sqlCodePointTurno

Opcional. El punto de cambio en el código SQL para la utilidad de carga. Ejecuta el proceso de cambio de caracteres. Necesario cuando la base de datos de destino DB2 es Postgresql.

Tipo: número

Predeterminado: 0

sysPunchEncoding

Opcional. El conjunto de caracteres de codificación de syspunch. Los valores admitidos son `Cp1047` / `ASCII`.

Tipo: cadena

Predeterminado: ASCII

treatLargeNumberAsInteger

Opcional. Especifica si se deben tratar los números grandes como `Integer`. Se tratan como `BigDecimal` por defecto.

Tipo: booleano

Predeterminado: false

unload.chunkSize

Opcional. Tamaño de fragmento utilizado para la utilidad de descarga.

Tipo: número

Predeterminado: 0

unload.columnFiller

Opcional. El rellenador de columnas de la utilidad de descarga.

Tipo: cadena

Predeterminado: space

unload.fetchSize

Opcional. Permite ajustar el tamaño de la búsqueda al manipular los cursores en la utilidad de descarga.

Tipo: número

Predeterminado: 0

unload.format.date

Opcional. Si `unload.useDatabaseConfiguration` está activado, el formato de fecha que se utilizará en la utilidad de descarga.

Tipo: cadena

Valor predeterminado: MM/dd/yyyy

unload.format.time

Opcional. Si `unload.useDatabaseConfiguration` está activado, el formato de hora que se utilizará en la utilidad de descarga.

Tipo: cadena

Predeterminado: HH.mm.ss

unload.format.timestamp

Opcional. Si `unload.useDatabaseConfiguration` está activado, el formato de marca de tiempo que se utilizará en la utilidad de descarga.

Tipo: cadena

Predeterminado: .mm.ss.ssssss yyyy-MM-dd-HH

descargar.nbi. whenNotNull

Opcional. El valor del indicador de bytes nulos (nbi) que se añade cuando el valor de la base de datos no es nulo.

Tipo: hexadecimal

Predeterminado: 00

unload.nbi.whenNull

Opcional. El valor del indicador de bytes nulos (nbi) que se añade cuando el valor de la base de datos es nulo.

Tipo: hexadecimal

Predeterminado: 6F

descargar.nbi. writeNullIndicator

Opcional. Especifica si se debe escribir el indicador nulo en el archivo de salida de descarga.

Tipo: booleano

Predeterminado: false

descargar. sqlCodePointTurno

Opcional. El punto de cambio en el código SQL para la utilidad de descarga. Ejecuta el proceso de cambio de caracteres. Necesario cuando la base de datos de destino DB2 es Postgresql.

Tipo: número

Predeterminado: 0

descargar. useDatabaseConfiguration

Opcional. Especifica si se debe usar la configuración de fecha u hora de application-main.yml en la utilidad de descarga.

Tipo: booleano

Predeterminado: false

descargar. varCharIsNulo

Opcional. Utilice este parámetro en el programa INFTILB, si se establece en `true`, y todos los campos que no admiten valores NULL con valores en blanco (espacios) devolverán una cadena vacía.

Tipo: booleano

Predeterminado: false

Otras propiedades

qtemp.cleanup.threshold.hours

Opcional. Para especificar cuándo está activado `qtemp.dblog`. La vida útil de la partición de base de datos (en horas).

Tipo: número

Predeterminado: 0

qtemp.dblog

Opcional. Si se debe habilitar el registro de la base de datos QTEMP.

Tipo: booleano

Predeterminado: false

qtemp.uuid.length

Opcional. La longitud de identificación única de QTEMP.

Tipo: número

Predeterminado: 9

quartz.scheduler. stand-by-if-error

Opcional. Especifica si se debe activar la ejecución del trabajo si el programador de trabajos está en modo de espera. Si es verdadero, cuando está habilitada, no se activa la ejecución de tareas.

Tipo: booleano

Predeterminado: false

warmUpCache

Opcional. Especifica si se deben cargar todos los datos de la tabla de comunicación de datos en una memoria caché de calentamiento al iniciar el servidor.

Tipo: booleano

Predeterminado: false

AWS Mainframe Modernization referencia de definición de aplicación

En AWS Mainframe Modernization, puede configurar las aplicaciones de mainframe migradas en un archivo JSON de definición de aplicaciones, que es específico del motor de ejecución que elija. La definición de una aplicación contiene tanto información general como información específica del motor. En este tema se describen las definiciones de aplicaciones de AWS Blu Age y Rocket Software (anteriormente Micro Focus) y se identifican todos los elementos obligatorios y opcionales.

Contenido

- Sección de cabecera general
- Descripción general de la sección de definiciones
- AWS Ejemplo de definición de aplicación de Blu Age
- AWS Detalles de la definición de Blu Age
  - Oyente(s): obligatorio
  - AWS Aplicación Blu Age: necesaria
  - BluSAM: opcional
  - AWS Colas de mensajes de Blue Age: opcionales
  - AWS Configuración de EFS de almacenamiento de aplicaciones Blu Age: opcional
- Definición de la aplicación Rocket Software (anteriormente Micro Focus)
- Detalles de la definición de Rocket Software
  - Oyente (es): obligatorio
  - Ubicaciones de los conjuntos de datos: obligatorias
  - Controlador de autenticación y autorización de Amazon Cognito: opcional
  - Controlador de LDAP y Active Directory: opcional
  - Configuración de lotes: obligatoria
  - Configuración de CICS: necesaria
  - Impresoras: opcionales
  - Recursos de XA: opcional
  - Configuración del tiempo de ejecución: opcional

Sección de cabecera general

Cada definición de aplicación comienza con información general sobre la versión de la plantilla y las ubicaciones de origen. La versión actual de la definición de la aplicación es la 2.0.

Utilice la siguiente estructura para especificar la versión de la plantilla y las ubicaciones de origen.

```
"template-version": "2.0",
    "source-locations": [
        {
            "source-id": "s3-source",
```

```
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket",
                "s3-key-prefix": "v1"
            }
        }
    ]
```

> **Note**
>
> Puede usar la siguiente sintaxis si desea introducir el ARN de S3 como s3-bucket:
>
> ```
> "template-version": "2.0",
>   "source-locations": [
>       {
>           "source-id": "s3-source",
>           "source-type": "s3",
>           "properties": {
>               "s3-bucket": "arn:aws:s3:::mainframe-deployment-bucket",
>               "s3-key-prefix": "v1"
>           }
>       }
>   ]
> ```

template-version

(Obligatorio) Especifica la versión del archivo de definición de aplicaciones. No cambie este valor. La única permitida actualmente es la 2.0. Especifique `template-version` con una cadena.

source-locations

Especifica las ubicaciones de los archivos y otros recursos que la aplicación necesita durante el tiempo de ejecución.

source-id

Especifica un nombre para la ubicación. Este nombre se utiliza para hacer referencia a la ubicación de origen según sea necesario en el JSON de definición de la aplicación.

source-type

Especifica el tipo del origen. El único valor permitido actualmente es s3.

properties

Proporciona los detalles de la ubicación de origen. Cada propiedad se especifica con una cadena.

- `s3-bucket`: obligatorio. Especifica el nombre del bucket de Amazon S3 donde se almacenan los archivos.
- `s3-key-prefix`: obligatorio. Especifica el nombre de la carpeta del bucket de Amazon S3 donde se almacenan los archivos.

Descripción general de la sección de definiciones

Especifica las definiciones de recursos de los servicios, la configuración, los datos y otros recursos típicos que la aplicación necesita para ejecutarse. Al actualizar la definición de una aplicación, AWS Mainframe Modernization detecta los cambios comparando las listas `source-locations` y `definition` de las versiones anterior y actual del archivo JSON de definición de la aplicación.

La sección de definición es específica del motor y está sujeta a cambios. En las siguientes secciones, se muestran ejemplos de definiciones de aplicaciones específicas de los motores para ambos motores.

AWS Ejemplo de definición de aplicación de Blu Age

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 8194,
            "type": "http"
        }],
        "ba-application": {
```

```
            "app-location": "${s3-source}/murachs-v6/"
        },
        "blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
        }
    }
}
```

AWS Detalles de la definición de Blu Age

Oyente(s): obligatorio

Especifique el puerto que utilizará para acceder a la aplicación a través del Elastic Load Balancing AWS Mainframe Modernization creado. Utilice la estructura siguiente:

```
"listeners": [{
            "port": 8194,
            "type": "http"
}],
```

puerto

(Obligatorio) Puede usar cualquier puerto disponible, excepto los puertos bien conocidos del 0 al 1023. Le recomendamos que utilice valores entre 8192 y 8199. Asegúrese de que no haya otros oyentes o aplicaciones funcionando en este puerto.

type

(Obligatorio) Actualmente, solo se admite `http`.

AWS Aplicación Blu Age: necesaria

Especifique la ubicación en la que el motor recoge el archivo de imagen de la aplicación mediante la siguiente estructura.

```
"ba-application": {
            "app-location": "${s3-source}/murachs-v6/",
            "files-directory": "/m2/mount/myfolder",
            "enable-jics": <true|false>,
            "enable-batch-restart": <true|false>,
            "shared-app-location": "${s3-source}/shared/"
},
```

app-location

La ubicación específica en Amazon S3 donde está almacenado el archivo de imagen de la aplicación.

files-directory

(Opcional) La ubicación de los archivos de entrada/salida de los lotes. Debe ser una subcarpeta de la configuración de Amazon EFS o Amazon FSx mount point a nivel de entorno. La subcarpeta debe ser propiedad de un usuario adecuado para que la utilice la aplicación Blu Age que se ejecuta en AWS Mainframe Modernization. Para ello, al adjuntar la unidad a una EC2 instancia de Amazon Linux, se `3001` debe crear un grupo con ID `101` y un usuario con ID, y la carpeta deseada debe ser propiedad de este usuario. Por ejemplo, de esta forma, Blu Age AWS Mainframe Modernization Managed puede utilizar la *`testclient`* carpeta.

```
groupadd -g 101 mygroup
useradd -M -g mygroup -p mypassword -u 3001 myuser
mkdir testclient
chown myuser:mygroup testclient
```

enable-jics

(Opcional) Especifica si se habilitará JICS. El valor predeterminado es true (verdadero). Si se establece en false, se evita que se genere la base de datos de JICS.

enable-batch-restart

(Opcional) Especifica si se debe habilitar la función de reinicio para los trabajos por lotes. El valor predeterminado es falso. Para obtener más información sobre las configuraciones de reinicio

por lotes, consulte las propiedades del motor AWS Blu Age con el prefijo `jcl.checkpoint` Propiedades de configuración para la aplicación gestionada con el motor AWS Blu Age.

shared-app-location

(Opcional) Otra ubicación en Amazon S3 donde se almacenan los elementos compartidos de aplicaciones. Puede contener el mismo tipo de estructura de aplicación que la ubicación de la aplicación.

BluSAM: opcional

Especifique la base de datos de BluSAM y la caché de Redis mediante la siguiente estructura.

```
"blusam": {
            "db": {
                "nb-threads": 8,
                "batch-size": 10000,
                "name": "blusam",
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:blusam-FfmXLG"
            },
            "redis": {
                "hostname": "blusam.c3geul.ng.0001.usw2.cache.amazonaws.com",
                "port": 6379,
                "useSsl": true,
                "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:111122223333:secret:bluesamredis-nioefm"
            }
}
```

db

Especifica las propiedades de la base de datos utilizada con la aplicación. La base de datos debe ser una base de datos de Aurora PostgreSQL. Puede especificar las propiedades siguientes:

- `nb-threads`: (opcional) Especifica cuántos subprocesos dedicados se utilizan para el mecanismo de escritura diferida en el que se basa el motor de Blusam. El valor predeterminado es 8.
- `batch-size`: (opcional) Especifica el umbral que utiliza el mecanismo de escritura diferida para iniciar las operaciones de almacenamiento por lotes. El umbral representa el número de registros modificados que iniciarán una operación de almacenamiento por lotes para

garantizar que los registros modificados se conserven. El desencadenante en sí se basa en una combinación del tamaño del lote y un tiempo transcurrido de un segundo, lo que se alcance primero. El valor predeterminado es 10000.

- `name`: (opcional) Especifica el nombre de la base de datos.
- `secret-manager-arn`: especifica el nombre de recurso de Amazon (ARN) del secreto que contiene las credenciales de la base de datos. Para obtener más información, consulte Paso 4: creación y configuración de un secreto de base de datos de AWS Secrets Manager.

Redis

Especifica las propiedades de la caché de Redis que la aplicación utiliza para almacenar los datos temporales que necesita en una ubicación central para mejorar el rendimiento. Le recomendamos que cifre y proteja con contraseña la caché de Redis.

- `hostname`: especifica la ubicación de la caché de Redis.
- `port`: especifica el puerto, normalmente el 6379, al que la caché de Redis envía y recibe las comunicaciones.
- `useSsl`: especifica si la caché de Redis está cifrada. Si lo está, establezca `useSsl` en false.
- `secret-manager-arn`: especifica el nombre de recurso de Amazon (ARN) del secreto que contiene la contraseña de la caché de Redis. Si la caché de Redis no está protegida con contraseña, no especifique `secret-manager-arn`. Para obtener más información, consulte Paso 4: creación y configuración de un secreto de base de datos de AWS Secrets Manager.

AWS Colas de mensajes de Blue Age: opcionales

Especifique los detalles de conexión JMS-MQ para AWS la aplicación Blu Age.

```
"message-queues": [
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr1",
      "channel": "mqChannel1",
      "hostname": "mqserver-host1",
      "port": 1414,
      "user-id": "app-user1",
      "ssl-cipher": "*TLS12ORHIGHER",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  },
```

```
  {
      "product-type": "JMS-MQ",
      "queue-manager": "QMgr2",
      "channel": "mqChannel2",
      "hostname": "mqserver-host2",
      "port": 1412,
      "user-id": "app-user2",
      "ssl-cipher": "*TLS12ORHIGHER",
      "secret-manager-arn": "arn:aws:secretsmanager:us-
west-2:123456789012:secret:sample/mq/test-279PTa"
  }
]
```

product-type

(Obligatorio) Especifica el tipo de producto. Actualmente, solo puede ser «JMS-MQ» para AWS aplicaciones de Blu Age.

queue-manager

(Obligatorio) Especifica el nombre del administrador de colas.

channel

(Obligatorio) Especifica el nombre del canal de conexión al servidor.

hostname

(Obligatorio) Especifica el nombre de host del servidor de cola de mensajes.

puerto

(Obligatorio) Especifica el número de puerto de agente de escucha en el que escucha el servidor.

user-id

(Obligatorio) Especifica el ID de cuenta de usuario autorizado para realizar operaciones de cola de mensajes en el canal especificado.

cifrado SSL

(Opcional) Especifica la especificación de cifrado SSL para la conexión.

secret-manager-arn

(Obligatorio) Especifica el nombre de recurso de Amazon (ARN) de Secrets Manager que proporciona la contraseña del usuario especificado.

AWS Configuración de EFS de almacenamiento de aplicaciones Blu Age: opcional

Especifique los detalles del punto de acceso EFS de almacenamiento de aplicaciones mediante la siguiente estructura.

```
"ba-application": {
            "file-permission-mask": "UMASK002"
},
"efs-configs": [
           {
            "file-system-id": "fs-01376dfsvfvrsvsr",
            "mount-point": "/m2/mount/efs-ap2",
            "access-point-id": fsap-0eaesefvrefrewgv8"
           }
]
```

file-system-id

(Obligatorio) El ID del sistema de archivos de EFS al que se aplica el punto de acceso. Patrón: "fs-([0-9a-f]{8,40}){1,128}$"

mount-point

(Obligatorio) El punto de montaje para el sistema de archivos de aplicación. Debe ser diferente al punto de montaje de almacenamiento de entorno.

access-point-id

(Obligatorio) El ID del punto de acceso, asignado por Amazon EFS. Patrón: "^fsap-([0-9a-f]{8,40}){1,128}$"

file-permission-mask

(Opcional) Define la máscara de creación de archivos para los archivos creados por el proceso de solicitud. Por ejemplo, si el valor se establece en `UMASK006`, todos los archivos tendrán el permiso 660. Esto significa que solo el propietario de los archivos y el grupo de archivos tendrán acceso de lectura y escritura, mientras que los demás usuarios no tendrán ningún permiso.

> **Note**
>
> El valor establecido para este campo solo se tiene en cuenta al utilizar el almacenamiento de EFS de aplicación.

> **Note**
>
> Cuando se proporciona la configuración de EFS, se debe especificar el directorio de archivos en la sección de definición de la aplicación. Debe ser una subcarpeta de la configuración del punto de montaje de Amazon EFS en la aplicación.

Definición de la aplicación Rocket Software (anteriormente Micro Focus)

La siguiente sección de ejemplos de definición es para el motor de ejecución de Rocket Software y contiene elementos obligatorios y opcionales.

```
{
 "template-version": "2.0",
 "source-locations": [
        {
            "source-id": "s3-source",
            "source-type": "s3",
            "properties": {
                "s3-bucket": "mainframe-deployment-bucket-aaa",
                "s3-key-prefix": "v1"
            }
        }
    ],
    "definition" : {
        "listeners": [{
            "port": 5101,
            "type": "tn3270"
        }],
        "dataset-location": {
            "db-locations": [{
                "name": "Database1",
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }]
        },
        "cognito-auth-handler": {
            "user-pool-id": "cognito-idp.us-west-2.amazonaws.com/us-west-2_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "us-west-2:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
        },
        "ldap-ad-auth-handler": {
            "ldap-ad-connection-secrets": [LIST OF AD-SECRETS]
```

```
        },
        "batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
            ]
        },
        "cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
        },
        "jes-printers": [
           {
             "name": "printerName",
              "classes": [
                 "A",
                 "B"
             ],
             "description": "printer desc....",
             "exit-module": {
                 "name": "lrsprte6"
                }
             }
         ],
        "xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "xa-connection-type": "postgres",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
        }],
        "runtime-settings": {
          "base-configuration-location": "${s3-source}/exported.json",
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs",
            "LRSQ_ADDRESS": "<lrsq-address>"
```

```
          }
        }
    }
}
```

Detalles de la definición de Rocket Software

El contenido de la sección de definición del archivo de definición de la aplicación de Rocket Software varía en función de los recursos que la aplicación de mainframe migrada requiera en tiempo de ejecución.

Oyente (es): obligatorio

Especifique un oyente mediante la siguiente estructura:

```
"listeners": [{
    "port": 5101,
    "type": "tn3270"
}],
```

puerto

En el caso de tn3270, el valor predeterminado es 5101. Para otros tipos de oyentes de servicios, el puerto varía. Puede usar cualquier puerto disponible, excepto los puertos bien conocidos del 0 al 1023. Cada oyente debe tener un puerto distintivo. Los oyentes no deben compartir puertos. Para obtener más información, consulte Listener Control en la documentación de Micro Focus Enterprise Server.

type

Especifica el tipo de oyente del servicio. Para obtener más información, consulte Listeners en la documentación de Micro Focus Enterprise Server.

Ubicaciones de los conjuntos de datos: obligatorias

Especifique la ubicación del conjunto de datos mediante la siguiente estructura.

```
"dataset-location": {
            "db-locations": [{
                "name": "Database1",
```

```
                "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456"
            }],
 }
```

db-locations

Especifica la ubicación de los conjuntos de datos que crea la aplicación migrada. Actualmente, solo AWS Mainframe Modernization admite conjuntos de datos de una única base de datos de VSAM.

- `name`: especifica el nombre de la instancia de base de datos que contiene los conjuntos de datos que crea la aplicación migrada.
- `secret-manager-arn`: especifica el nombre de recurso de Amazon (ARN) del secreto que contiene las credenciales de la base de datos.

Controlador de autenticación y autorización de Amazon Cognito: opcional

AWS Mainframe Modernization utiliza Amazon Cognito para la autenticación y autorización de las aplicaciones migradas. Especifique el controlador de autenticación de Amazon Cognito mediante la siguiente estructura.

```
"cognito-auth-handler": {
            "user-pool-id": "cognito-idp.Region.amazonaws.com/Region_rvYFnQIxL",
            "client-id": "58k05jb8grukjjsudm5hhn1v87",
            "identity-pool-id": "Region:64464b12-0bfb-4dea-ab35-5c22c6c245f6"
}
```

user-pool-id

Especifica el grupo de usuarios de Amazon Cognito que se AWS Mainframe Modernization utiliza para autenticar a los usuarios de la aplicación migrada. El del grupo Región de AWS de usuarios debe coincidir con el de Región de AWS la AWS Mainframe Modernization aplicación.

client-id

Especifica la aplicación migrada a la que puede acceder el usuario autenticado.

identity-pool-id

Especifica el grupo de identidades de Amazon Cognito en el que el usuario autenticado intercambia un token del grupo de usuarios por credenciales que permiten al usuario acceder a

AWS Mainframe Modernization. El Región de AWS del grupo de identidades debe coincidir con el Región de AWS de la AWS Mainframe Modernization aplicación.

Controlador de LDAP y Active Directory: opcional

Puede integrar la aplicación con Active Directory (AD) o cualquier tipo de servidor LDAP para que los usuarios de la aplicación puedan utilizar sus credenciales de LDAP/AD para la autorización y la autenticación.

Para integrar su aplicación con AD

1. Siga los pasos descritos en Configuring Active Directory for Enterprise Server Security, en la documentación de Micro Focus Enterprise Server.
2. Cree un AWS Secrets Manager secreto con el AD/LDAP details for each AD/LDAP servidor que desee usar con la aplicación. Para obtener información sobre cómo crear un secreto, consulte Crear un secreto de AWS Secrets Manager en la Guía del AWS Secrets Manager usuario. Para el tipo de secreto, elija Otro tipo de secreto e incluya los siguientes pares clave-valor.

```
{
    "connectionPath"       : "<HOST-ADDRESS>:<PORT>",
    "authorizedId"         : "<USER-FULL-DN>",
    "password"             : "<PASSWORD>",
    "baseDn"               : "<BASE-FULL-DN>",
    "userClassDn"          : "<USER-TYPE>",
    "userContainerDn"      : "<USER-CONTAINER-DN>",
    "groupContainerDn"     : "<GROUP-CONTAINER-DN>",
    "resourceContainerDn"  : "<RESOURCE-CONTAINER-DN>"
}
```

⚠ Recomendaciones de seguridad

- Para `connectionPath`, AWS Mainframe Modernization es compatible con los protocolos LDAP y LDAP over SSL (LDAPS). Recomendamos utilizar el LDAPS porque es más seguro y evita que las credenciales aparezcan en las transmisiones de red.
- Para `authorizedId` y `password`, le recomendamos que especifique las credenciales de un usuario sin más permisos que los permisos de verificación y de solo lectura más restrictivos que se requieren para que la aplicación se ejecute.

- Le recomendamos que rote las credenciales AD/LDAP periódicamente.
- No cree usuarios de AD con el nombre de usuario `awsuser` o `mfuser`. Estos dos nombres de usuario están reservados para el uso de AWS .

A continuación se muestra un ejemplo.

```
{
    "connectionPath" : "ldaps://msad4.m2.example.people.aws.dev:636",
    "authorizedId" :
 "CN=LDAPUser,OU=Users,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "password" : "ADPassword",
    "userContainerDn" : "CN=Enterprise Server Users,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "groupContainerDn" : "CN=Enterprise Server Groups,CN=Micro Focus,CN=Program
 Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev",
    "resourceContainerDn" : "CN=Enterprise Server Resources,CN=Micro
 Focus,CN=Program Data,OU=msad4,DC=msad4,DC=m2,DC=example,DC=people,DC=aws,DC=dev"
}
```

Cree el secreto con una clave de KMS administrada por el cliente. Debe conceder a AWS Mainframe Modernization los permisos `GetSecretValue` y `DescribeSecret` sobre el secreto y los permisos `Decrypt` y `DescribeKey` sobre la clave KMS. Para obtener más información, consulte los permisos de la clave KMS en la Guía del usuario. AWS Secrets Manager

3. Agregue el código siguiente a la definición de la aplicación.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": [LIST OF AD/LDAP SECRETS]
}
```

A continuación se muestra un ejemplo.

```
"ldap-ad-auth-handler": {
    "ldap-ad-connection-secrets": ["arn:aws:secrets:1234:us-east-1:secret:123456"]
}
```

Si la aplicación está integrada con LDAP y se ha iniciado, debe proporcionar las credenciales para ejecutar al menos una de las operaciones relacionadas con la aplicación mencionadas en la lista de autorizaciones admitidas.

El controlador de autenticación LDAP/AD está disponible para Micro Focus (Rocket) 8.0.11 y versiones posteriores.

> **Note**
>
> Actualmente, el administrador de LDAP debe proporcionar permisos de modificación sobre la utilidad `casstart` en los recursos del servidor empresarial OPERCMDS de su directorio LDAP. Esto debe hacerse para que todos los usuarios predeterminados necesarios (por ejemplo, CICSUSER, si la aplicación está relacionada con CICS) para iniciar la aplicación correctamente.

Especificación de las credenciales de usuario de LDAP para autenticación y autorización

1. Cree una AWS Secrets Manager con las siguientes claves y valores:

```
{
  "username" : "<USERNAME>",
  "password" : "<PASSWORD>"
}
```

> **Important**
>
> Debe tener los derechos para ejecutar `DescribeSecrets` y `GetSecretValue` en el Secrets Manager que se utilice. Además, asocie una clave de KMS y los permisos necesarios para el AWS Secrets Manager, como se menciona en Elegir un AWS KMS key.

2. Elija el parámetro de Secrets Manager.

   AWS console

   Al ejecutar las operaciones desde la AWS consola, habrá una opción para elegir el Secrets Manager que se debe pasar.

AWS CLI (or SDK)

Al ejecutar operaciones desde AWS CLI (o SDK), debe pasarse el parámetro de la API `auth-secrets-manager-arn` con el ARN de Secrets Manager.

Esta es la lista de operaciones de las aplicaciones que actualmente admiten autorización:

- `StartBatchJob`
- `CancelBatchJobExecution`
- `ListBatchJobRestartPoints`

Configuración de lotes: obligatoria

Especifique los detalles necesarios para los trabajos por lotes que se ejecutan como parte de la aplicación mediante la siguiente estructura.

```
"batch-settings": {
            "initiators": [{
                "classes": ["A", "B"],
                "description": "initiator...."
            }],
            "jcl-file-location": "${s3-source}/batch/jcl",
            "program-path": "/m2/mount/libs/loadlib:$EFS_MOUNT/emergency/loadlib",
            "system-procedure-libraries":"SYS1.PROCLIB;SYS2.PROCLIB",
            "aliases": [
               {"alias": "FDSSORT", "program": "SORT"},
               {"alias": "MFADRDSU", "program": "ADRDSSU"}
    ]
}
```

initiators

Especifica un iniciador de lotes que se inicia cuando la aplicación migrada se inicia correctamente y continúa ejecutándose hasta que la aplicación se detiene. Puede definir una o más clases por iniciador. También puede definir varios iniciadores. Por ejemplo:

```
"batch-settings": {
            "initiators": [
                {
```

```
                "classes": ["A", "B"],
                "description": "initiator...."
              },
              {
                "classes": ["C", "D"],
                "description": "initiator...."
              }

        ],
}
```

Para obtener más información, consulte To define a batch initiator or printer SEP en la documentación de Micro Focus Enterprise Server.

- `classes`: especifica las clases de trabajo que puede ejecutar el iniciador. Puede usar hasta 36 caracteres. Puede usar los siguientes caracteres: A-Z o 0-9.
- `description`: describe para qué sirve el iniciador.

jcl-file-location

Especifica la ubicación de los archivos JCL (Lenguaje de control de tareas) que requieren los trabajos por lotes que ejecuta la aplicación migrada.

program-path

Especifica la ruta necesaria para ejecutar trabajos por lotes cuando un programa de un JCL no se encuentra en la ubicación predeterminada. Los nombres de rutas diferentes están separados por punto y coma (:).

> **Note**
>
> La ruta del programa solo puede ser EFS.

system-procedure-libraries

Especifica los conjuntos de datos particionados predeterminados en los que se buscarán los procedimientos de JCL. Sin embargo, el procedimiento no se encuentra en el JCL ni mediante las instrucciones JCLLIB. Estos conjuntos de datos deben estar catalogados y debe usarse el nombre del catálogo. Y las entradas se separan con punto y coma (;).

alias

Define un mapeo entre los nombres de las utilidades y los programas utilizados en JCL y el nombre de implementación de la utilidad. AWS y las utilidades por lotes de terceros (por ejemplo, M2SFTP, M2WAIT, Syncsort, etc.) pueden tener alias de forma opcional para eliminar la necesidad de cambiar el JCL. Por ejemplo:

- FDSSORT Alias FDSSORT para SORT y Alias FDSICET para ICETOOL
- ADRDSSU Alias MFADRDSU para ADRDSSU
- Syncsort Alias DMXMFSRT para SORT

Configuración de CICS: necesaria

Especifique los detalles necesarios para las transacciones de CICS que se ejecutan como parte de la aplicación mediante la siguiente estructura.

```
"cics-settings": {
            "binary-file-location": "${s3-source}/cics/binaries",
            "csd-file-location": "${s3-source}/cics/def",
            "system-initialization-table": "BNKCICV"
}
```

binary-file-location

Especifica la ubicación de los archivos del programa de transacciones CICS.

csd-file-location

Especifica la ubicación del archivo de definición de recursos (CSD) del CICS para esta aplicación. Para obtener más información, consulte CICS Resource Definitions en la documentación de Micro Focus Enterprise Server.

system-initialization-table

Especifica la tabla de inicialización del sistema (SIT) que utiliza la aplicación migrada. El nombre de la tabla SIT puede tener hasta 8 caracteres. Puede usar A-Z, 0-9, $, @ y #. Para obtener más información, consulte CICS Resource Definitions en la documentación de Micro Focus Enterprise Server.

Impresoras: opcionales

Especifique una impresora jes mediante la siguiente estructura.

```
"jes-printers": [
    {
        "name": "printerName",
        "classes": [
            "A",
            "B"
        ],
        "description": "printer desc....",
        "exit-module": {
            "name": "lrsprte6",
            "module" : "program"
        }
    }
],
```

> **Note**
>
> Puede haber un máximo de 25 impresoras configuradas para una aplicación determinada.

nombre

(Obligatorio) Especifica el nombre que se asociará a este recurso de impresora. Los nombres deben ser únicos para cada impresora y se puede utilizar un límite de 128 caracteres alfanuméricos.

classes

(Obligatorio) Especifica las clases de salida aplicables a este recurso de impresora. Se puede utilizar un límite de 36 caracteres alfanuméricos.

description

(Opcional) Texto descriptivo adicional para la impresora.

exit-module

(Opcional) Especifica un módulo personalizado para la salida de impresión. No hay valores predeterminados; si no se especifica, no se utilizará ningún módulo de salida. Puede utilizar un

módulo de salida de impresión administrado o proporcionar el suyo. Los módulos de salida de impresión administrados se definen mediante un nombre reservado `lrsprte6` para la cola de LRS o bien, puede proporcionar el suyo propio mediante el parámetro del módulo para especificar la ubicación y el nombre.

La estructura `exit-module` tiene dos componentes:

- `name`: (obligatorio), si se usa `exit-module`. El nombre de la entrada del módulo de salida. El nombre de entrada del módulo de salida está limitado a un máximo de 8 caracteres.
- `module`: (opcional) La ubicación de S3 del binario del módulo de salida de impresión.

Puede ver más ejemplos de cómo definir el módulo de salida en la sección the section called “Impresoras”.

Recursos de XA: opcional

Especifique los detalles necesarios para los recursos de XA que requiere la aplicación mediante la siguiente estructura.

```
"xa-resources" : [{
            "name": "XASQL",
            "secret-manager-arn": "arn:aws:secrets:1234:us-east-1:secret:123456",
            "xa-connection-type": "postgres",
            "module": "${s3-source}/xa/ESPGSQLXA64.so"
}]
```

> **Note**
>
> Se ha actualizado la definición del recurso de XA para incluir un campo `xa-connection-type` opcional. Si no se proporciona, se supone que el tipo de conexión es postgres.

nombre

(Obligatorio) Especifica el nombre del recurso de XA.

secret-manager-arn

(Obligatorio) Especifica el nombre de recurso de Amazon (ARN) del secreto que contiene las credenciales para conectarse a la base de datos.

xa-connection-type

(Opcional) Especifica el tipo de conexión de recurso de XA.

module

(Obligatorio) Especifica la ubicación del archivo ejecutable del módulo conmutador RM. Para obtener más información, consulte Planificación y diseño XARs en la documentación de Micro Focus Enterprise Server.

Configuración del tiempo de ejecución: opcional

Especifique los detalles necesarios para la configuración del tiempo de ejecución para administrar las variables de entorno permitidas mediante la siguiente estructura.

```
"runtime-settings": {
          "base-configuration-location": "${s3-source}/exported.json",
          "environment-variables": {
            "ES_JES_RESTART": "N",
            "EFS_MOUNT": "/m2/mount/efs"
    }
}
```

base-configuration-location

(Opcional) Especifica la ubicación para la importación masiva de una configuración de servidor Micro Focus. Este archivo debe ser un JSON válido y estar presente en la misma ubicación de S3 que la ubicación de los artefactos de la aplicación definida anteriormente. Para exportar la configuración de una aplicación existente, consulte la sección Para exportar una región de la CESPAO en la documentación del software Rocket.

environment-variables

Especifica las variables de entorno compatibles con Micro Focus que se aplican al tiempo de ejecución de esta aplicación.

- `ES_JES_RESTART`es una variable de entorno de Rocket Software que permite reiniciar el procesamiento de JCL. Opcionalmente, también se puede utilizar `ES_ALLOC_OVERRIDE` como variable de entorno de Rocket Software.

- `EFS_MOUNT` es una variable de entorno personalizada que la aplicación puede utilizar para identificar dónde se encuentra el montaje de EFS del entorno.

Puede acceder a todas las variables de entorno de Rocket Software en la guía de Rocket Enterprise Server para UNIX.

AWS Referencia de definición de conjuntos de datos de modernización de mainframe

Si su aplicación requiere más de unos pocos conjuntos de datos para su procesamiento, es ineficiente ingresarlos uno por uno en la consola de modernización del AWS mainframe. En lugar de eso, se recomienda crear un archivo JSON para especificar cada conjunto de datos. Los distintos tipos de conjuntos de datos se especifican de forma diferente en el JSON, aunque muchos parámetros son comunes. En este documento se describen los detalles del JSON necesarios para importar diferentes tipos de conjuntos de datos.

> **Note**
>
> Antes de importar un conjunto de datos, debe transferirlo de la unidad central a AWS. Los conjuntos de datos deben encontrarse en un formato que se pueda cargar en el motor de tiempo de ejecución seleccionado. En muchos casos, puede ser un archivo secuencial, pero en el caso de VSAM de Rocket Software (anteriormente Micro Focus), tendrá que estar en su formato propietario. La utilidad `DFCONV` es el método sugerido para convertir el archivo. Especifique el nombre del bucket y la carpeta en el archivo JSON de definición del conjunto de datos.
> Para obtener más información sobre el motor de ejecución de Rocket Software, consulte Conversión de archivos por lotes DFCONV en la documentación de Rocket Software.
> Para obtener más información sobre AWS Blu Age, consulte. the section called “AWS Configuración de Blu Age Runtime”

Temas

- Propiedades comunes
- Ejemplo de formato de solicitud de conjunto de datos para VSAM
- Ejemplo de formato de solicitud de conjunto de datos para base GDG

- Ejemplo de formato de solicitud de conjunto de datos para las generaciones PS o GDG
- Ejemplo de formato de solicitud de conjunto de datos para PO

Propiedades comunes

Varios parámetros son comunes en todos los conjuntos de datos. Estos parámetros cubren las siguientes áreas:

- Información sobre el conjunto de datos (`datasetName`, `datasetOrg`, `recordLength`, `encoding`).
- Información sobre la ubicación desde la que está importando, es decir, la ubicación de origen del conjunto de datos. Esta no es la ubicación de la unidad central. Es la ruta a la ubicación de Amazon S3 en la que cargó el conjunto de datos (`externalLocation`).
- Información sobre la ubicación a la que está importando, es decir, la ubicación de destino del conjunto de datos. Esta ubicación puede ser una base de datos o un sistema de archivos, según el motor de tiempo de ejecución. (`storageType` y `relativePath`).
- Información sobre el tipo de conjunto de datos (tipo de conjunto de datos específico, formato, codificación, etc.).

Cada definición de conjunto de datos tiene la misma estructura JSON. El siguiente ejemplo de JSON muestra todos estos parámetros comunes.

```
{
    "dataSet": {
        "storageType": "Database",
        "datasetName": "MFI01V.MFIDEMO.BNKACC",
        "relativePath": "DATA",
        "datasetOrg": {
            "type": {
                type-specific properties
                    ...
            },
        },
    },
}
```

Las siguientes propiedades son comunes a todos los conjuntos de datos.

storageType

Obligatorio. Se aplica a la ubicación de destino. Especifica si el conjunto de datos se almacena en una base de datos o en un sistema de archivos. Los valores posibles son `Database` o `FileSystem`.

- AWS Motor de ejecución Blu Age: los sistemas de archivos no son compatibles. Debe utilizar una base de datos.
- Motor de ejecución de Rocket Software: se admiten bases de datos y sistemas de archivos. Puede utilizar Amazon Relational Database Service o Amazon Aurora para las bases de datos, y Amazon Elastic File System o FSx Amazon for Lustre para los sistemas de archivos.

datasetName

(Obligatorio) Especifica el nombre completo del conjunto de datos tal como aparece en el mainframe.

relativePath

(Obligatorio) Se aplica a la ubicación de destino. Especifica la ubicación relativa del conjunto de datos en la base de datos o el sistema de archivos.

datasetOrg

(Obligatorio) Especifica el tipo de conjunto de datos. Los valores posibles son `vsam`, `gdg`, `ps`, `po` o `unknown`.

- AWS Motor de ejecución Blu Age: solo se admiten conjuntos de datos de tipo VSAM.
- Motor de ejecución de Rocket Software: se admiten conjuntos de datos de tipo VSAM, GDG, PS, PO o de tipo desconocido.

> **Note**
>
> Si la aplicación requiere archivos que no sean archivos de datos COBOL, sino archivos PDF u otros archivos binarios, puede especificarlos de la siguiente manera:

```
"datasetOrg": {
            "type": PS {
                "format": U
            },
```

Ejemplo de formato de solicitud de conjunto de datos para VSAM

- AWS Motor de tiempo de ejecución Blu Age: compatible.
- Motor de tiempo de ejecución de Rocket Software: compatible.

Si va a importar conjuntos de datos de VSAM, especifique `vsam` como `datasetOrg`. El JSON debe parecerse al siguiente ejemplo:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.VSAM.KSDS",
    "relativePath": "DATA",
    "datasetOrg": {
        "vsam": {

            "encoding": "A",
            "format": "KS",
            "primaryKey": {
                "length": 11,
                "offset": 0
            }
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.VSAM.KSDS.DAT"
}
```

Los conjuntos de datos de VSAM admiten las siguientes propiedades.

encoding

(Obligatorio) Especifica la codificación del conjunto de caracteres del conjunto de datos. Los valores posibles son ASCII (`A`), EBCDIC (`E`) y Unknown (`?`).

format

(Obligatorio) Especifica el tipo de conjunto de datos de VSAM y el formato de registro.

- AWS Motor de ejecución Blu Age: los valores posibles son ESDS (`ES`) y KSDS (`KS`). El formato de registro puede ser fijo o variable.
- Motor de ejecución de Rocket Software: los valores posibles son ESDS (`ES`), KSDS (`KS`) y RRDS (). `RR` La definición de VSAM incluye el formato de registro, por lo que no es necesario especificarlo por separado.

primaryKey

(Obligatorio) Se aplica únicamente a los conjuntos de datos KSDS de VSAM. Especifica la clave principal. Consta del nombre de la clave principal, el desplazamiento y la longitud de la clave. `name` es opcional; `offset` y `length` son obligatorias.

recordLength

(Obligatorio) Especifica la longitud de un registro. Para los formatos de registro de longitud fija, estos valores deben coincidir.

- AWS Motor de tiempo de ejecución Blu Age: para VSAM, ESDS y KSDS, `min` es opcional y obligatorio. `max`
- Motor de tiempo de ejecución de Rocket Software: `min` y `max` es obligatorio.

externalLocation

(Obligatorio) Especifica la ubicación de origen: es decir, el bucket de Amazon S3 en el que cargó el conjunto de datos.

Propiedades específicas del motor de Blue Age

El motor de ejecución AWS Blu Age admite la compresión de conjuntos de datos VSAM. En el siguiente ejemplo se muestra cómo se puede especificar esta propiedad en JSON.

```
{
    common properties
        ...
        "datasetOrg": {
            "vsam": {
                common properties
                ...
                "compressed": boolean,
                common properties
                ...
            }
        }
```

```
}
```

Especifique la propiedad de compresión de la siguiente manera:

compression

(Opcional) Especifica si los índices de este conjunto de datos se almacenan como valores comprimidos. Si tiene un conjunto de datos grande (normalmente > 100 Mb), considere configurar este indicador en `true`.

Ejemplo de formato de solicitud de conjunto de datos para base GDG

- AWS Motor de tiempo de ejecución Blu Age: no compatible.
- Motor de tiempo de ejecución de Rocket Software: compatible.

Si va a importar conjuntos de datos base GDG, especifique `gdg` como `datasetOrg`. El JSON debe parecerse al siguiente ejemplo:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.GDG",
    "relativePath": "DATA",
    "datasetOrg": {
        "gdg": {
            "limit": "3",
            "rollDisposition": "Scratch and No Empty"
        }
    }
}
```

Los conjuntos de datos base de GDG admiten las siguientes propiedades.

límite

(Obligatorio) Especifica el número de generaciones activas o de sesgos. Para un clúster base GDG, el máximo es 255.

rollDisposition

(Obligatorio) Especifica cómo gestionar los conjuntos de datos de generación cuando se alcanza o se supera el máximo. Los valores posibles son `No Scratch and No Empty`, `Scratch and`

`No Empty`, `Scratch and Empty` o `No Scratch and Empty`. El valor predeterminado es `Scratch and No Empty`.

Ejemplo de formato de solicitud de conjunto de datos para las generaciones PS o GDG

- AWS Motor de tiempo de ejecución Blu Age: no compatible.
- Motor de tiempo de ejecución de Rocket Software: compatible.

Si va a importar conjuntos de datos de generación PS o GDG, especifique `ps` como `datasetOrg`. El JSON debe parecerse al siguiente ejemplo:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PS.FB",
    "relativePath": "DATA",
    "datasetOrg": {
        "ps": {
            "format": "FB",
            "encoding": "A"
        }
    },
    "recordLength": {
        "min": 300,
        "max": 300
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/catalog/data/AWS.M2.PS.LSEQ"
}
}
```

Los conjuntos de datos de generaciones de PS o GDG admiten las siguientes propiedades.

format

(Obligatorio) Especifica el formato de los registros del conjunto de datos. Los posibles valores son `F`, `FA`, `FB`, `FBA`, `FBM`, `FBS`, `FM`, `FS`, `LSEQ`, `U`, `V`, `VA`, `VB`, `VBA`, `VBM`, `VBS`, `VM` y `VS`.

encoding

(Obligatorio) Especifica la codificación del conjunto de caracteres del conjunto de datos. Los valores posibles son ASCII (`A`), EBCDIC (`E`) y Unknown () `?`.

recordLength

(Obligatorio) Especifica la longitud de un registro. Debe especificar la longitud mínima (`min`) y máxima (`max`) del registro. Para los formatos de registro de longitud fija, estos valores deben coincidir.

externalLocation

(Obligatorio) Especifica la ubicación de origen, es decir, el bucket de Amazon S3 en el que cargó el conjunto de datos.

Ejemplo de formato de solicitud de conjunto de datos para PO

Si va a importar conjuntos de datos de PO, especifique `po` como `datasetOrg`. El JSON debe parecerse al siguiente ejemplo:

```
{
    "storageType": "Database",
    "datasetName": "AWS.M2.PO.PROC",
    "relativePath": "DATA",
    "datasetOrg": {
        "po": {
            "format": "LSEQ",
            "encoding": "A",
            "memberFileExtensions": ["PRC"]
        }
    },
    "recordLength": {
        "min": 80,
        "max": 80
    }
},
"externalLocation": {
    "s3Location": "s3://$M2_DATA_STORE/source/proc/"
}
}
```

Los conjuntos de datos de PO admiten las siguientes propiedades.

format

(Obligatorio) Especifica el formato de los registros del conjunto de datos. Los posibles valores son `F`, `FA`, `FB`, `FBA`, `FBM`, `FBS`, `FM`, `FS`, `LSEQ`, `U`, `V`, `VA`, `VB`, `VBA`, `VBM`, `VBS`, `VM` y `VS`.

encoding

(Obligatorio) Especifica la codificación del conjunto de caracteres del conjunto de datos. Los valores posibles son ASCII (`A`), EBCDIC (`E`) y Unknown (`?`).

memberFileExtensions

(Obligatorio) Especifica una matriz que contiene una o más extensiones de nombre de archivo, lo que le permite especificar qué archivos se incluirán como miembros del PDS.

recordLength

(Opcional) Especifica la longitud de un registro. Tanto la longitud mínima (`min`) como la máxima (`max`) del registro son opcionales. Para los formatos de registro de longitud fija, estos valores deben coincidir.

externalLocation

(Obligatorio) Especifica la ubicación de origen, es decir, el bucket de Amazon S3 en el que cargó el conjunto de datos.

> **Note**
>
> La implementación actual del motor de ejecución de Rocket Software añade entradas PDS como conjuntos de datos dinámicos.

Entornos de tiempo de ejecución gestionados en la AWS modernización de mainframe

Si es la primera vez que utiliza la modernización de AWS mainframes, consulte los siguientes temas para empezar:

- ¿Qué es la modernización del AWS mainframe?
- Configuración para la modernización del AWS mainframe
- Comience con la modernización de AWS mainframe
- Tutorial: Configurar el tiempo de ejecución gestionado para AWS Blu Age
- Tutorial: Configurar el tiempo de ejecución gestionado para Rocket Software (anteriormente Micro Focus)

Un entorno de tiempo de ejecución en la modernización de AWS mainframe es una combinación determinada de recursos de AWS procesamiento, un motor de tiempo de ejecución y los detalles de configuración que especifique. El entorno de tiempo de ejecución aloja una o más aplicaciones. Las aplicaciones de AWS Mainframe Modernization contienen cargas de trabajo de mainframe migradas. Puede elegir el motor de tiempo de ejecución para los entornos que cree. Elija AWS Blu Age si utiliza el patrón de refactorización automática y Rocket Software (anteriormente Micro Focus) si utiliza el patrón de replataforma. También puede elegir la cantidad de recursos informáticos adecuada para su aplicación y, de forma opcional, conectar el almacenamiento a los entornos de ejecución. AWS La modernización del mainframe le permite CloudWatch utilizar las métricas y el registro de Amazon para que pueda supervisar su entorno de ejecución.

Temas

- Cree un entorno de ejecución de modernización AWS de mainframe
- Actualice un entorno de ejecución AWS de modernización de mainframe
- Detenga un entorno de AWS ejecución de modernización de mainframe
- Reinicie un entorno de ejecución AWS de modernización de mainframe
- Elimine un entorno de ejecución AWS de modernización de mainframe

Cree un entorno de ejecución de modernización AWS de mainframe

Utilice la consola de modernización de AWS mainframe para crear un entorno de modernización de AWS mainframe.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Creación de un entorno de tiempo de ejecución

Para crear un entorno de tiempo de ejecución

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región en la que desee crear el entorno.
3. En la página Entornos, elija Crear entorno.
4. En la página Especificaciones de inferencia proporcione la siguiente información:
    a. En la sección Nombre y descripción, introduzca un nombre para el entorno.
    b. (Opcional). En el campo Descripción del entorno, introduzca una descripción para el entorno. Esta descripción ayudarle a usted y a otros usuarios a identificar el propósito del entorno de tiempo de ejecución.
    c. En la sección de opciones de motor, elige Blu Age para la refactorización automática o Micro Focus (Rocket) para la replataforma.
    d. Elija una versión para el motor que ha seleccionado.
    e. (Opcional). En la sección Etiquetas, elija Agregar nueva etiqueta para agregar una o varias etiquetas al entorno. Una etiqueta de entorno es un atributo personalizado que le ayuda a organizar y administrar los recursos de AWS .
    f. Elija Siguiente.
5. En la página Especificaciones de ajuste, proporcione la siguiente información:
    a. En la sección Disponibilidad, elija un Entorno en tiempo de ejecución independiente o un Clúster de alta disponibilidad.

El patrón de disponibilidad determina la disponibilidad de la aplicación cuando se ejecute. La versión independiente está bien para fines de desarrollo. La de alta disponibilidad es para las aplicaciones que deben estar disponibles en todo momento.

b. En Recursos, elija un tipo de instancia y la capacidad deseada.

Estos recursos son las EC2 instancias de Amazon gestionadas por AWS Mainframe Modernization que alojarán su entorno de ejecución. Los entornos de tiempo de ejecución independientes ofrecen dos opciones de tipo de instancia y solo permiten una instancia. Los entornos de tiempo de ejecución de alta disponibilidad ofrecen dos opciones para el tipo de instancia y permiten hasta dos instancias.

Para obtener más información, consulte Tipos de EC2 instancias de Amazon y póngase en contacto con un especialista en AWS mainframe para obtener orientación.

6. En la sección Seguridad y red, realice lo siguiente:

a. Si desea que las aplicaciones sean de acceso público, seleccione Allow applications deployed to this environment to be publicly accessible.

b. Elija el tipo de red. Si lo desea IPv4, las aplicaciones del entorno de modernización de AWS mainframe solo atienden IPv4 solicitudes. En el modo de doble pila, las aplicaciones atenderán tanto a las solicitudes como a IPv4 las IPv6 solicitudes. Si elige el modo de doble pila, asegúrese de que haya al menos 1 VPC IPv6 con subredes habilitadas.

c. Elija una nube virtual privada (VPC).

d. Si utiliza el patrón de alta disponibilidad, elija dos o más subredes. Si utilizas el patrón independiente con el motor AWS Blu Age, elige dos o más subredes. Si utilizas el patrón independiente con el motor de Rocket Software, puedes especificar una subred.

e. Elija un grupo de seguridad para la VPC que ha seleccionado.

> **Note**
>
> AWS La modernización del mainframe crea un Network Load Balancer para que pueda distribuir las conexiones a su entorno de ejecución. Asegúrese de que las reglas de entrada y salida de su grupo de seguridad permitan el acceso desde una dirección IP al puerto que ha especificado en la propiedad `the section called “Oyente (es): obligatorio”` de la definición de la aplicación. Para obtener

más información, consulte Actualización de los grupos de seguridad del equilibrador de carga de red en la Guía del usuario para Equilibradores de carga de red.

f. En el campo de clave KMS, seleccione Personalizar la configuración de cifrado si desea utilizar un sistema gestionado por el cliente. AWS KMS key Para obtener más información, consulte Cifrado de datos en reposo para el servicio de modernización AWS de mainframe.

> **Note**
>
> De forma predeterminada, AWS Mainframe Modernization cifra sus datos con un archivo AWS KMS key que AWS Mainframe Modernization posee y administra por usted. Sin embargo, puede optar por utilizar una AWS KMS key administrada por el cliente.

g. (Opcional) Elija un AWS KMS key por nombre o nombre de recurso de Amazon (ARN). También puede elegir Crear un AWS KMS key para ir a la AWS KMS consola y crear uno nuevo. AWS KMS key

h. Elija Siguiente.

7. (Opcional) En la página Adjuntar almacenamiento, elija uno o más sistemas de FSx archivos Amazon EFS o Amazon.

   El sistema de archivos montado en un entorno de modernización de AWS mainframe debe ser propiedad de un usuario adecuado para que lo utilicen las aplicaciones que se ejecutan en la consola de modernización de mainframe de AWS.

   Para configurar estos ajustes de usuario, puede conectar la unidad a una EC2 instancia de Amazon de Linux. A continuación, cree un grupo con ID `101` y un usuario con un ID`3001`. Además, asegúrese de que la carpeta de datos deseada que utilizarán sus aplicaciones sea propiedad de este usuario.

   Por ejemplo, las aplicaciones de modernización de mainframe que se ejecutan en AWS Mainframe Modernization AWS Managed pueden usar la `myFiles` carpeta.

```
groupadd -g 101 mygroup
useradd -M -g mygroup -p mypassword -u 3001 myuser
mkdir myFiles
chown myuser:mygroup myFiles
```

> **Note**
>
> Para habilitar el acceso al sistema de archivos, se deben configurar las siguientes reglas de grupos de seguridad para establecer la conectividad de red entre la instancia de entorno EFS y M2:
>
> - Grupo de seguridad del entorno de M2: incluya una regla de salida que permita el tráfico a través del puerto NFS 2049.
> - Grupo de seguridad de destinos de montaje del sistema de archivos: incluya una regla de entrada que permita el tráfico a través del puerto NFS 2049 desde el grupo de seguridad de instancias (mostrado anteriormente) y una regla de salida que permita el tráfico a través del puerto NFS 2049.

8. Elija Siguiente.
9. En la sección Periodo de mantenimiento, elija cuándo desea aplicar los cambios pendientes al entorno.

   - Si selecciona Ninguna preferencia, AWS Mainframe Modernization seleccionará un intervalo de mantenimiento optimizado para usted.
   - Para especificar una ventana de mantenimiento concreta, seleccione Seleccionar nueva ventana de mantenimiento. Puede elegir el día de la semana, una hora de inicio y la duración del periodo de mantenimiento.

   Para obtener más información sobre los períodos de mantenimiento, consulte AWS Ventana de mantenimiento y modernización del mainframe.

   Elija Siguiente.
10. En la página Revisar y crear, revise la información que ha proporcionado y, a continuación, elija Crear entorno.

Actualice un entorno de ejecución AWS de modernización de mainframe

Utilice la consola de modernización de AWS mainframe para actualizar un entorno de ejecución de modernización de AWS mainframe. Puede actualizar la versión secundaria del motor de tiempo de

ejecución o el tipo de instancia que aloja el entorno de tiempo de ejecución. Puede elegir si aplicar las actualizaciones inmediatamente o durante el periodo de mantenimiento que prefiera.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Actualizar un entorno de tiempo de ejecución

Para actualizar un entorno de tiempo de ejecución

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región en la que se creó el entorno que desea actualizar.
3. En la página Entornos, elija el nombre del entorno que desea actualizar.
4. En la página de detalles del entorno, elija Acciones y, a continuación, seleccione Editar entorno.
5. Realice cualquiera de los siguientes cambios:

   - En la sección Opciones de motor, elija la versión de motor que desee.
   - En la sección Recursos, elija el tipo de instancia que desee.
   - En la sección de la Periodo de mantenimiento, elija el día, la hora y la duración que desee.

   > **Note**
   >
   > Los únicos cambios que puede optar por aplicar durante el periodo de mantenimiento son los cambios en la versión del motor. Debe aplicar todos los demás cambios inmediatamente.

6. Elija Siguiente.
7. En When to apply these changes, elija Inmediatamente o Durante el siguiente periodo de mantenimiento. A continuación, seleccione Actualizar entorno.

Si elige Inmediatamente, verá un mensaje cuando el entorno haya terminado de actualizarse.

AWS Ventana de mantenimiento y modernización del mainframe

Cada entorno de tiempo de ejecución tiene un periodo de mantenimiento semanal de dos horas. Todos los cambios en el sistema se aplican durante este tiempo. El periodo de mantenimiento

es su oportunidad de controlar cuándo se producirán las modificaciones, así como los parches de seguridad y software. Si hay un evento de mantenimiento programado para una semana determinada, comienza dentro del periodo de mantenimiento de dos horas asignados. La mayoría de los eventos de mantenimiento también se completan durante el periodo de mantenimiento de dos horas, aunque hay eventos de mantenimiento más largos que pueden tardar más de ese tiempo en completarse.

El periodo de mantenimiento de dos horas se selecciona al azar dentro de un bloque de 8 horas por región. Si no especifica un periodo de mantenimiento al crear un entorno de tiempo de ejecución, AWS Mainframe Modernization asigna una ventana de mantenimiento de dos horas un día de la semana seleccionado al azar.

AWS La modernización del mainframe consume algunos de los recursos de la instancia de su entorno mientras se realiza el mantenimiento. Es posible que observe un efecto mínimo en el rendimiento o algunas interrupciones en las aplicaciones durante el mantenimiento.

Detenga un entorno de AWS ejecución de modernización de mainframe

Utilice la consola de modernización de AWS mainframe para detener un entorno de ejecución de modernización de AWS mainframe. Al detener un entorno, se conservan las implementaciones de aplicaciones actuales y no se le cobrará por el entorno hasta que se reinicie.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Detener un entorno de tiempo de ejecución

Si necesita detener un entorno de ejecución de modernización de AWS mainframe, siga los mismos pasos que en la sección de actualización del entorno.

Utilice la consola de modernización de AWS mainframe para detener un entorno de ejecución de modernización de AWS mainframe. Al detener un entorno, se conservan las implementaciones de aplicaciones actuales y no se le cobrará por el entorno hasta que se reinicie.

> **Note**
>
> Debe detener todas las aplicaciones antes de detener el entorno.

Para detener un entorno de tiempo de ejecución

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región en la que se creó el entorno que desea detener.
3. En la página Entornos, elija el nombre del entorno que desea detener.
4. En la página de detalles del entorno, elija Acciones y, a continuación, seleccione Editar entorno.
5. En la página Editar entorno, busque la sección Recursos y actualice la capacidad deseada a cero.

   > **Note**
   >
   > Para detener un entorno, solo puede optar por detenerlo inmediatamente.

6. Elija Siguiente.
7. En When to apply these changes, elija Inmediatamente. A continuación, seleccione Actualizar entorno.

Aparece un mensaje cuando se actualiza la capacidad del entorno.

Reinicie un entorno de ejecución AWS de modernización de mainframe

Utilice la consola de modernización de AWS mainframe para reiniciar un entorno de ejecución de modernización de AWS mainframe. Al reiniciar un entorno de ejecución, se reanudará la facturación del entorno.

Reiniciar un entorno de tiempo de ejecución

Para reiniciar un entorno de ejecución de modernización de AWS mainframes, siga pasos similares a los de la sección de parada del entorno.

Para reiniciar un entorno de tiempo de ejecución

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región en la que se creó el entorno que desea reiniciar.
3. En la página Entornos, elija el nombre del entorno que desea reiniciar.

4. En la página de detalles del entorno, elija Acciones y, a continuación, seleccione Editar entorno.

> **Note**
>
> La capacidad deseada para un entorno independiente solo se puede actualizar a 1. Para reiniciar un entorno de tiempo de ejecución, solo puede elegir reiniciarlo inmediatamente.

5. En la página Editar entorno, busque la sección Recursos y actualice la capacidad deseada de cero a la que desee.
6. Elija Siguiente.
7. En When to apply these changes, elija Inmediatamente. A continuación, seleccione Actualizar entorno.

Aparece un mensaje cuando se actualiza la capacidad del entorno y se reinicia.

Elimine un entorno de ejecución AWS de modernización de mainframe

Utilice la consola de modernización de AWS mainframe para eliminar un entorno de ejecución de modernización de AWS mainframe.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe.

Eliminar un entorno de tiempo de ejecución

Si necesita eliminar un entorno de ejecución de AWS Mainframe Modernization, asegúrese de eliminar primero todas las aplicaciones implementadas del entorno. No puede eliminar un entorno de tiempo de ejecución en el que se implementan las aplicaciones.

Para eliminar un entorno

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región en la que se creó el entorno que desea eliminar.
3. En la página Entornos, elija el entorno que desee eliminar y, a continuación, elija Acciones y Eliminar entorno.

4. En la ventana Eliminar entorno, introduzca `delete` para confirmar que desea eliminar el entorno de tiempo de ejecución y, a continuación, seleccione Eliminar.

Pruebas de aplicaciones en la modernización de AWS mainframes

AWS Las pruebas de aplicaciones de modernización de mainframe proporcionan pruebas automatizadas de equivalencia funcional para sus proyectos de migración. AWS Las pruebas de aplicaciones de modernización de mainframes aceleran los proyectos de migración al aprovechar la elasticidad de la nube. Puede ejecutar conjuntos de pruebas independientes en tantos entornos paralelos como sea necesario, lo cual reduce los plazos de las pruebas. Entre las principales ventajas de las pruebas de aplicaciones se incluyen la rapidez y la agilidad de las pruebas, el alto grado de repetibilidad de las pruebas, la escalabilidad y la elasticidad integradas, la automatización a gran escala, la rentabilidad y la perfecta integración con AWS CloudFormation los entornos de prueba específicos.

Temas

- ¿Qué son las pruebas de aplicaciones de modernización de AWS mainframe?
- AWS Conceptos de pruebas de aplicaciones de modernización de mainframe
- AWS Requisitos previos para las pruebas de aplicaciones de modernización de mainframe
- Flujos de trabajo de la consola de Pruebas de aplicaciones
- Tutorial: Configurar la aplicación de CardDemo muestra en las pruebas de aplicaciones de modernización de AWS mainframe
- Tutorial: Reproduzca y compare en pruebas de aplicaciones de modernización de AWS mainframe con CardDemo AWS Blu Age implementado en Amazon EC2
- AWS Modernización de mainframe, pruebas de aplicaciones, conjuntos de datos compatibles y páginas de códigos
- Protección de datos en las pruebas de aplicaciones AWS de modernización de mainframe
- Cómo funcionan las pruebas de aplicaciones de modernización de AWS mainframe con IAM

¿Qué son las pruebas de aplicaciones de modernización de AWS mainframe?

Las pruebas tienen un impacto significativo en los proyectos de modernización. AWS Las pruebas de aplicaciones, una función de la modernización del AWS mainframe, proporcionan

pruebas automatizadas de equivalencia funcional para las aplicaciones migradas. Las pruebas de equivalencia funcional le ayudan a validar que las aplicaciones del mainframe Nube de AWS son equivalentes a las aplicaciones del mainframe. AWS Las pruebas de aplicaciones comparan automáticamente los cambios en los conjuntos de datos, los registros de bases de datos y las 3270 pantallas en línea entre su mainframe y. AWS Además, Pruebas de aplicaciones permite realizar pruebas repetibles, por lo que puede ejecutar sus escenarios de prueba muchas veces a medida que actualice la arquitectura de destino, resuelva problemas y avance hacia una aplicación completamente migrada. Tras la migración, puede seguir utilizando Application Testing para realizar pruebas de regresión, a fin de garantizar que las actualizaciones de los motores del tiempo de ejecución u otros componentes no provoquen regresiones. Las pruebas de aplicaciones son rentables: los entornos de prueba objetivo se crean utilizando las CloudFormation plantillas proporcionadas por el usuario y aprovechando Infrastructure-as-Code los conceptos (IaC). Pruebas de aplicaciones acelera los proyectos de migración gracias a la elasticidad de la nube. Puede ejecutar conjuntos de pruebas independientes en tantos entornos paralelos como sea necesario, lo cual reduce los plazos de las pruebas.

Temas

- ¿Es la primera vez que usa Application Testing?
- Ventajas de Application Testing
- Integración con AWS CloudFormation
- Cómo funciona Pruebas de aplicaciones
- Servicios relacionados
- Acceso a Pruebas de aplicaciones
- Precios de Pruebas de aplicaciones

¿Es la primera vez que usa Application Testing?

Si es la primera vez que usa Application Testing, le recomendamos que empiece leyendo las siguientes secciones:

- Conceptos de Pruebas de aplicaciones
- Tutorial: Configurar la CardDemo aplicación en las pruebas de aplicaciones
- the section called “Tutorial: Reproduzca y compare en AWS Blu Age usando CardDemo”

Ventajas de Application Testing

Pruebas de aplicaciones ofrece varias ventajas que le ayudarán en el proceso de migración:

- Aceleración de las pruebas, agilidad y flexibilidad
- Conceptos de pruebas: “Registrar una vez en el mainframe y reproducir varias veces en AWS”
- Creación de entornos objetivo mediante plantillas proporcionadas por el usuario mediante plantillas proporcionadas por los usuarios. CloudFormation
- Altos grados de repetibilidad de las pruebas
- Diseñado para la nube, pensando en la escalabilidad y la elasticidad
- Pruebas a gran escala con un alto grado de automatización
- Rentabilidad

Integración con AWS CloudFormation

Las pruebas de aplicaciones utilizan la infraestructura como código con AWS CloudFormation. Esta elección de diseño simplifica y mejora su experiencia de prueba. AWS CloudFormation le brinda autonomía e independencia para definir la mejor infraestructura para sus necesidades. Puede seleccionar o definir varios parámetros (tamaño de instancia, instancia de RDS, grupo de seguridad óptimo) de forma independiente. Puede agregar recursos, como una cola de Amazon SQS, que necesite para que su aplicación funcione correctamente en condiciones de prueba.

En las AWS CloudFormation plantillas que se proporcionan para su descarga, observará algunas características comunes:

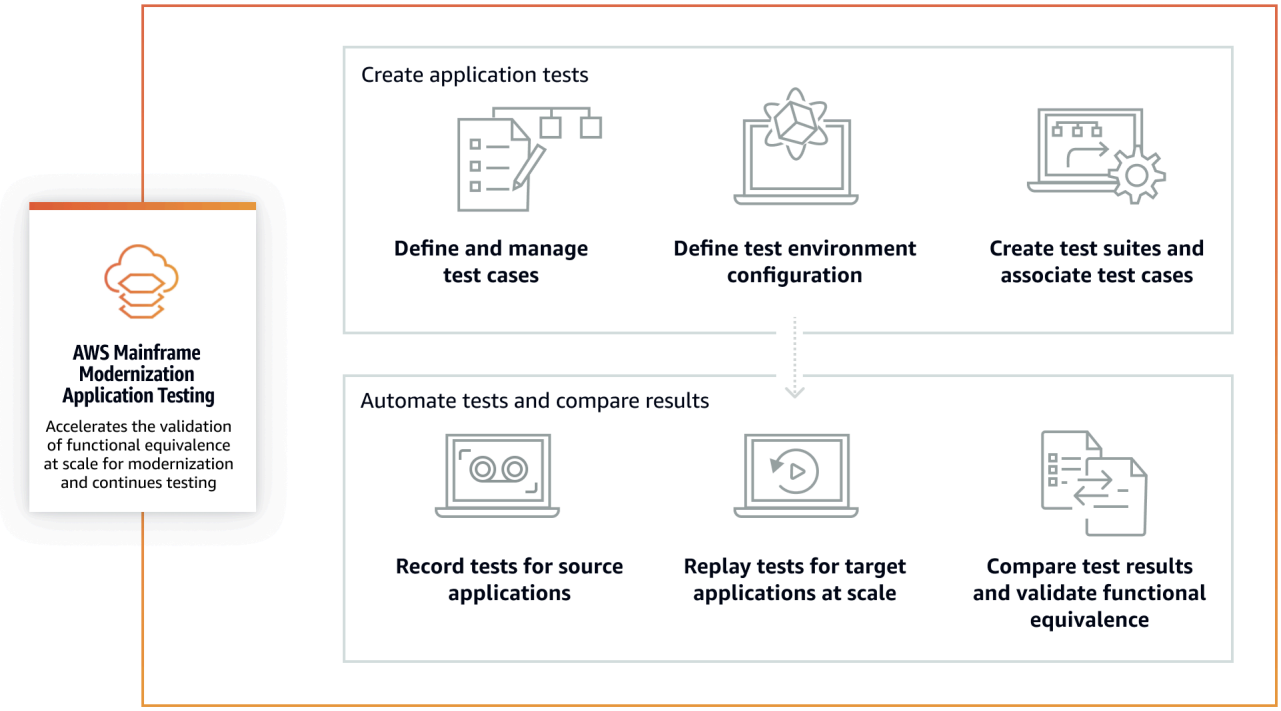
- Las pruebas de aplicaciones crean una pila completamente aislada, que incluye una aplicación y un entorno de ejecución de modernización del AWS mainframe, con sus propias definiciones de red y seguridad. Este conjunto aislado proporciona resiliencia, ya que otros actores del mismo sistema Cuenta de AWS no pueden interferir con la actividad de prueba. También evita situaciones en las que los operadores del sistema modifiquen la VPC o el grupo de seguridad predeterminados, lo que puede provocar errores en las actividades de prueba.
- El grupo de seguridad también permite controlar el acceso externo a los recursos utilizados en las pruebas. Por ejemplo, una base de datos puede contener datos confidenciales.
- El aislamiento total evita que otros actores que comparten la VPC espíen el tráfico.

- Mejora el rendimiento. Por ejemplo, la comunicación entre la aplicación de modernización del AWS mainframe que crea la plantilla y su base de datos de Amazon RDS se produce en una red independiente (una VPC privada), lo que evita que otros actores ralenticen el tráfico.

Le recomendamos que implemente también estas funciones en las AWS CloudFormation plantillas que cree.

Cómo funciona Pruebas de aplicaciones

La siguiente figura es una descripción general de cómo funciona Pruebas de aplicaciones.



- Puede transferir los datos de entrada de la fuente a AWS las herramientas que prefiera para la transferencia de datos de mainframe. File Transfer
- Aplica la misma lógica empresarial tanto en el origen como en el destino.
- Pruebas de aplicaciones compara automáticamente los datos de salida (conjuntos de datos, cambios en las bases de datos relacionales, pantallas 3270 en línea e interacciones de los usuarios) tanto del origen como del destino. Después de ejecutar el escenario de prueba en el ordenador central, se capturan los datos de salida y se transfieren al mismo y AWS, a continuación, se reproduce el escenario de prueba en el servidor de destino. Las pruebas de aplicaciones comparan automáticamente los datos de salida de la prueba AWS con los datos de salida de la fuente. Puede ver rápidamente qué registros son idénticos, equivalentes, diferentes

o faltantes. Además, puede definir reglas de equivalencia para que los registros que no sean idénticos pero tengan el mismo significado empresarial se entiendan como equivalentes.

El flujo de trabajo que se sigue en Pruebas de aplicaciones consta de los siguientes pasos:

1. Crear casos de prueba: los casos de prueba son la unidad más pequeña de las acciones de pruebas. Al crear un caso de prueba, también se identifican los tipos de datos que se van a comparar y que representan mejor la equivalencia funcional entre el origen y el destino.
2. Defina la configuración del entorno de prueba: especifique la configuración de su entorno especificando la AWS CloudFormation plantilla y los atributos adicionales.
3. Crear conjuntos de pruebas: los conjuntos de pruebas son una colección de casos de prueba.
4. Cargue conjuntos de datos en la fuente y reprodúzcalos en el destino: capture los conjuntos de datos de entrada y salida en la computadora central y cárguelos allí. AWS A continuación, se vuelve a reproducir el escenario de prueba en AWS.
5. Comparar conjuntos de datos de origen y destino: Pruebas de aplicaciones compara automáticamente los conjuntos de datos de salida, tanto de origen como de destino, para que pueda ver rápidamente qué es correcto y qué no.

Tanto la acción final de un escenario de prueba como el objetivo de todo el proceso es identificar las discrepancias entre las pruebas de origen y de destino. Application Testing compara la versión de origen y la versión de destino de los datos capturados en todos los canales de interacción durante la ejecución de la prueba. También compara los estados finales de los datos relevantes (tal como se definen en los casos de prueba).

Servicios relacionados

Las pruebas de aplicaciones son una característica de la modernización del AWS mainframe. También utiliza la infraestructura como código AWS CloudFormation para garantizar la repetibilidad, la automatización y la rentabilidad de las pruebas. Para obtener más información, consulte:

- AWS Modernización del mainframe
- AWS CloudFormation

Acceso a Pruebas de aplicaciones

Para acceder a la consola de pruebas de aplicaciones en https://console.aws.amazon.com/apptest/ o desde la consola de modernización del AWS mainframe, seleccione Application Testing en el panel de navegación izquierdo.

Precios de Pruebas de aplicaciones

Los precios de Application Testing se encuentran en Precios de AWS Mainframe Modernization.

AWS Conceptos de pruebas de aplicaciones de modernización de mainframe

AWS Las pruebas de aplicaciones utilizan términos que otros servicios de pruebas o paquetes de software podrían utilizar con un significado ligeramente diferente. En las siguientes secciones se explica cómo las pruebas de aplicaciones de modernización de AWS mainframes utilizan esta terminología.

Temas

- Caso de prueba
- Conjunto de pruebas
- Configuración del entorno de prueba
- Cargar
- Reproducción
- Comparación
- Comparación de bases de datos
- Comparación de conjuntos de datos
- Estado de comparación
- Reglas de equivalencia
- Comparación de conjuntos de datos de estado final
- Comparaciones de bases de datos estado-progreso
- Equivalencia funcional (FE)
- Comparaciones de pantallas 3270 en línea
- Datos de reproducción

- Datos de referencia
- Carga, reproducción y comparación
- Diferencias
- Equivalencias
- Aplicación de origen
- Aplicación de destino

Caso de prueba

Un caso de prueba es la unidad de acción individual más básica del flujo de trabajo de pruebas. Por lo general, un caso de prueba se utiliza para representar una unidad independiente de lógica empresarial que modifica los datos. Se realizarán comparaciones para cada caso de prueba. Los casos de prueba se agregan a un conjunto de pruebas. Los casos de prueba contienen metadatos sobre los artefactos de datos (conjuntos de datos, bases de datos) que modifica el caso de prueba y sobre las funciones empresariales que se activan durante la ejecución del caso de prueba: trabajos por lotes, cuadros de diálogo interactivos de tipo 3270 y otros. Por ejemplo, los nombres y las páginas de códigos de los conjuntos de datos.

Datos de entrada → Caso de prueba → Datos de salida

Los casos de prueba pueden ser en línea o por lotes:

- Los casos de prueba de la pantalla 3270 en línea son casos de prueba en los que el usuario ejecuta cuadros de diálogo interactivos en pantalla (3270) para leer, modificar o generar nuevos datos empresariales (registros de bases de datos o conjuntos de datos).
- Los casos de prueba por lotes son casos de prueba que requieren enviar un lote para leer, procesar y modificar o producir nuevos datos empresariales (conjuntos de datos o registros de bases de datos).

Conjunto de pruebas

Los conjuntos de pruebas tienen una colección de casos de prueba que se ejecutan en orden secuencial, uno por uno. La reproducción se realiza en un conjunto de pruebas. Todos los casos de prueba del conjunto de pruebas se ejecutan en el entorno de prueba de destino cuando se reproduce un conjunto de pruebas. Si hay diferencias después de comparar los artefactos de las pruebas de referencia y de reproducción, las diferencias se mostrarán en el nivel del caso de prueba.

Por ejemplo, el conjunto de pruebas A:

Caso de prueba 1, caso de prueba 2, caso de prueba 3, etc.

Configuración del entorno de prueba

La configuración del entorno de prueba le permite configurar el conjunto inicial de datos y parámetros de configuración (o recursos) con los CloudFormation que necesita para que la ejecución de la prueba sea repetible.

Cargar

La carga se realiza en un conjunto de pruebas. Durante la carga, debe proporcionar una ubicación de Amazon S3 que contenga los artefactos, conjuntos de datos y diarios CDC de las bases de datos relacionales del mainframe de origen con los que se compararán. Se considerarán datos de referencia del mainframe de origen. Durante la reproducción, los datos de reproducción generados se compararán con los datos de referencia cargados para garantizar la equivalencia de las aplicaciones.

Reproducción

Las reproducciones se realizan en un conjunto de pruebas. Durante la reproducción, las pruebas de aplicaciones de modernización del AWS mainframe utilizan el CloudFormation script para crear el entorno de prueba de destino y ejecutar la aplicación. Los conjuntos de datos y los registros de las bases de datos que se modifican durante la reproducción se capturan y se comparan con los datos de referencia del mainframe. Por lo general, cargará en el mainframe una vez y luego reproducirá varias veces, hasta que se alcance la equivalencia funcional.

Comparación

Las comparaciones se realizan automáticamente cuando la reproducción finaliza satisfactoriamente. Durante las comparaciones, los datos de referencia que cargó y capturó durante la fase de carga se comparan con los datos de reproducción generados durante la fase de reproducción. Las comparaciones se realizan por caso de prueba individual para conjuntos de datos, registros de bases de datos y pantallas en línea por separado.

Comparación de bases de datos

Application Testing emplea una función de comparación del progreso del estado al comparar los cambios en los registros de las bases de datos entre las aplicaciones de origen y destino.

La comparación estado-progreso compara las diferencias en cada ejecución individual de las instrucciones INSERT, UPDATE y DELETE, a diferencia de comparar las filas de la tabla al final del proceso. La comparación estado-progreso es más eficiente que las alternativas, ya que proporciona comparaciones más rápidas y precisas, pues solo se comparan los datos modificados y se detectan los errores que se corrigen automáticamente en el flujo de transacciones. Al utilizar la tecnología CDC (Changed Data Capture), Application Testing puede detectar cambios en las bases de datos de relaciones individuales y compararlos entre la de origen y la de destino.

Los cambios en la base de datos relacional se generan en el origen y el destino mediante el código de la aplicación probada con sentencias DML (lenguaje de modificación de datos) como SQL INSERT, UPDATE o DELETE, pero también indirectamente cuando la aplicación utiliza procedimientos almacenados, cuando los activadores de la base de datos se configuran en algunas tablas o cuando se utiliza CASCADE DELETE para garantizar la integridad referencial, lo que desencadena automáticamente eliminaciones adicionales.

Comparación de conjuntos de datos

Pruebas de aplicaciones compara automáticamente los conjuntos de datos de referencia y reproducción producidos en los sistemas de origen (registro) y de destino (reproducción).

Para comparar conjuntos de datos:

1. Comience con los mismos datos de entrada (conjuntos de datos, base de datos) tanto en el origen como en el destino.
2. Ejecute sus casos de prueba en el sistema de origen (mainframe).
3. Capture los conjuntos de datos producidos y cárguelos en un bucket de Amazon S3. Puede transferir los conjuntos de datos de entrada de la fuente a otros AWS mediante revistas, pantallas y conjuntos de datos de los CDC.
4. Especifique la ubicación del bucket de Amazon S3 en el que se cargaron los conjuntos de datos del mainframe al registrar el caso de prueba.

Una vez finalizada la reproducción, Pruebas de aplicaciones compara automáticamente los conjuntos de datos de referencia y de destino de salida y muestra si los registros son idénticos, equivalentes, diferentes o faltan. Por ejemplo, los campos de fecha relativos al momento de la ejecución de la carga de trabajo (día + 1, final del mes en curso, etc.) se consideran automáticamente equivalentes. Además, si lo desea, puede definir reglas de equivalencia para que los registros que no sean idénticos sigan teniendo el mismo significado empresarial y se marquen como equivalentes.

Estado de comparación

Application Testing utiliza los siguientes estados de comparación: IDENTICAL, EQUIVALENT y DIFFERENT.

IDENTICAL

Los datos de origen y destino son exactamente los mismos.

EQUIVALENT

Los datos de origen y destino contienen diferencias falsas consideradas equivalencias, como fechas o marcas horarias, que no afectan a la equivalencia funcional cuando son relativas al momento de la ejecución de la carga de trabajo. Puede definir reglas de equivalencia para identificar cuáles son estas diferencias. Cuando todos los conjuntos de pruebas repetidos en comparación con sus conjuntos de pruebas de referencia muestran el estado IDENTICAL o EQUIVALENT, el conjunto de pruebas no muestra diferencias.

DIFFERENT

Los datos de origen y de destino contienen diferencias, como un número diferente de registros en un conjunto de datos o valores diferentes en el mismo registro.

Reglas de equivalencia

Conjunto de reglas para identificar diferencias falsas que pueden considerarse resultados equivalentes. Las pruebas de equivalencia funcional offline (OFET) provocan inevitablemente diferencias en algunos resultados entre los sistemas de origen y de destino. Por ejemplo, las marcas de tiempo de actualización son diferentes según el diseño. Las reglas de equivalencia explican cómo ajustar esas diferencias y evitar los falsos positivos en el momento de la comparación. Por ejemplo, si una fecha es el tiempo de ejecución más 2 días en una columna de datos concreta, la regla de equivalencia lo describe y acepta un tiempo en el sistema de destino que sea un tiempo de ejecución en el objetivo más 2 días en lugar de un valor que sea estrictamente igual a la misma columna de la carga de referencia.

Comparación de conjuntos de datos de estado final

El estado final de los conjuntos de datos que se han creado o modificado, incluidos todos los cambios o actualizaciones realizados en los conjuntos de datos desde su estado inicial. En el caso

de los conjuntos de datos, Pruebas de aplicaciones examina los registros de esos conjuntos de datos al final de la ejecución de un caso de prueba y compara los resultados.

Comparaciones de bases de datos estado-progreso

Comparaciones de los cambios realizados en los registros de la base de datos como una secuencia de instrucciones DML (eliminar, actualizar, insertar) individuales. Application Testing compara los cambios individuales (insertar, actualizar o eliminar la fila de una tabla) de la base de datos de origen con la de destino e identifica las diferencias para cada cambio individual. Por ejemplo, se puede usar una instrucción INSERT individual para insertar en una tabla una fila con valores diferentes en la base de datos de origen en comparación con la base de datos de destino.

Equivalencia funcional (FE)

Dos sistemas se consideran funcionalmente equivalentes si producen los mismos resultados en todas las operaciones observables, con los mismos datos de entrada. Por ejemplo, dos aplicaciones se consideran funcionalmente equivalentes si los mismos datos de entrada producen datos de salida idénticos (a través de pantallas, cambios en el conjunto de datos o cambios en la base de datos).

Comparaciones de pantallas 3270 en línea

Compara la salida de las 3270 pantallas del ordenador central con la salida de las pantallas web de las aplicaciones modernizadas cuando el sistema de destino se ejecuta en un entorno de ejecución de AWS Blu Age. Nube de AWS Además, compara la salida de las 3270 pantallas del ordenador central con las 3270 pantallas de la aplicación realojada cuando el sistema de destino se ejecuta en un entorno de ejecución de Rocket Software (anteriormente Micro Focus). Nube de AWS

Datos de reproducción

Los datos de reproducción se utilizan para describir los datos generados al reproducir un conjunto de pruebas en el entorno de prueba de destino. Por ejemplo, los datos de reproducción se generan cuando se ejecuta un conjunto de pruebas en una aplicación del servicio de modernización de mainframe. AWS A continuación, los datos de reproducción se comparan con los datos de referencia capturados en el origen. Cada vez que reproduce la carga de trabajo en el entorno de destino, se genera una nueva generación de datos de reproducción.

Datos de referencia

Los datos de referencia se utilizan para describir los datos capturados en el mainframe de origen. Es la referencia con la que se compararán los datos generados por la reproducción (objetivo). Por lo general, por cada registro del mainframe que cree datos de referencia, habrá muchas repeticiones. Esto se debe a que los usuarios suelen capturar el estado correcto de la aplicación en el mainframe y reproducir los casos de prueba en la aplicación modernizada de destino para validar la equivalencia. Si se encuentran errores, se corrigen, y los casos de prueba se vuelven a reproducir. A menudo, se repiten varios ciclos, se corrigen errores y se vuelve a reproducir para validar la incidencia. Esto se conoce como el paradigma de prueba de capturar una vez y reproducir varias veces.

Carga, reproducción y comparación

Pruebas de aplicaciones opera en tres pasos:

- Carga: captura los datos de referencia creados en el mainframe para cada caso de prueba de un escenario de prueba. Pueden incluir pantallas 3270 en línea, conjuntos de datos y registros de bases de datos.
  - En el caso de pantallas 3270 en línea, debe utilizar el emulador de terminal Blu Insights para capturar la carga de trabajo de origen. Para obtener más información, consulte la documentación de Blu Insights.
  - En el caso de los conjuntos de datos, tendrá que capturar los conjuntos de datos generados por cada caso de prueba en el mainframe mediante herramientas habituales, como el FTP o el servicio de transferencia de conjuntos de datos que forma parte de AWS Mainframe Modernization.
  - Para los cambios en la base de datos, utilice la documentación de AWS Mainframe Modernization Data Replication with Precisely para recopilar y generar diarios de CDC que contengan los cambios.
- Reproducción: el escenario de prueba se reproduce en el entorno de destino. Todos los casos de prueba especificados en la ejecución del conjunto de pruebas. Los tipos de datos especificados creados por los casos de prueba individuales, como conjuntos de datos, cambios en las bases de datos relacionales o pantallas 3270, se capturarán de forma automática. Estos datos se conocen como datos de reproducción y se compararán con los datos de referencia capturados durante la fase de carga.

> **Note**
>
> Los cambios en la base de datos relacional requerirán opciones de configuración específicas del DMS en la plantilla de estado inicial. CloudFormation

- Comparación: se comparan los datos de referencia de la prueba de origen y los datos de reproducción de destino, y los resultados se muestran como datos idénticos, diferentes, equivalentes o faltantes.

Diferencias

Indica que se han detectado diferencias entre los conjuntos de datos de referencia y de reproducción mediante la comparación de los datos. Por ejemplo, un campo de una pantalla 3270 en línea que muestre valores diferentes desde el punto de vista de la lógica empresarial entre el mainframe de origen y la aplicación modernizada de destino se considerará una diferencia. Otro ejemplo es una carga en un conjunto de datos que no es idéntico entre las aplicaciones de origen y destino.

Equivalencias

Los registros equivalentes son registros que difieren entre los conjuntos de datos de referencia y los de reproducción, pero que no deben tratarse de forma diferente desde el punto de vista de la lógica empresarial. Por ejemplo, un registro que contenga la marca temporal del momento en que se produjo el conjunto de datos (tiempo de ejecución de la carga de trabajo). Mediante reglas de equivalencia personalizables, puede indicar a Application Testing que trate esa diferencia de falsos positivos como una equivalencia, incluso aunque muestre valores diferentes entre los datos de referencia y los de reproducción.

Aplicación de origen

La aplicación de mainframe de origen con la que se va a comparar.

Aplicación de destino

La aplicación nueva o modificada en la que se realizan las pruebas y que se comparará con la aplicación de origen para detectar cualquier defecto y lograr la equivalencia funcional entre las aplicaciones de origen y de destino. La aplicación de destino normalmente se ejecuta en la nube. AWS

AWS Requisitos previos para las pruebas de aplicaciones de modernización de mainframe

AWS La función de pruebas de aplicaciones de modernización de AWS mainframe de Mainframe Modernization le permite realizar pruebas de equivalencia funcional automatizadas para sus proyectos de migración. Para prepararse para usar las pruebas de aplicaciones en la consola de modernización del AWS mainframe, haga lo siguiente:

1. Definir los casos de prueba: defina las unidades básicas de pruebas que desea ejecutar y reproducir en un orden específico, para la aplicación de destino. Para obtener más información sobre cómo crear casos de pruebas, consulte the section called “Creación de casos de prueba en Pruebas de aplicaciones”.
2. Prepare la CloudFormation plantilla y los datos de entrada: cree una CloudFormation plantilla que se utilizará para aprovisionar el entorno de prueba de destino. Las variables de esta plantilla se utilizarán para agregar datos de entrada y nombres de variables de salida en su aplicación de AWS Mainframe Modernization. Para obtener información adicional, consulte Trabajar con una AWS CloudFormation plantilla en la Guía AWS CloudFormation del usuario.
3. Garantizar el acceso al mainframe y a la captura de datos: compruebe que tiene acceso al mainframe de origen. Esto también garantizará que pueda capturar y cargar los datos de origen generados por las aplicaciones que se ejecutan en el mainframe.

Flujos de trabajo de la consola de Pruebas de aplicaciones

AWS La consola de pruebas de aplicaciones de modernización de mainframe le ayuda a crear casos de prueba, conjuntos de pruebas y configuraciones de entornos de prueba.

Temas

- Cree casos de prueba en las pruebas de aplicaciones de modernización de AWS mainframe
- Cree conjuntos de pruebas en las pruebas de aplicaciones de modernización de AWS mainframe
- Cree configuraciones de entornos de prueba en las pruebas de aplicaciones de modernización de AWS mainframe

Cree casos de prueba en las pruebas de aplicaciones de modernización de AWS mainframe

Un caso de prueba es una unidad atómica que representa una acción determinada en su flujo de trabajo. Para obtener información adicional sobre varios conceptos, consulte ???.

> **Important**
>
> Primero debe crear al menos configuraciones de entorno de prueba antes de ejecutar los casos de prueba. Para crear su primera configuración de entorno, consulte the section called "Creación de configuraciones de entornos de prueba en Pruebas de aplicaciones".

Temas

- Creación de un caso de prueba por lotes
- Creación de un caso de prueba de pantalla 3270 en línea

Creación de un caso de prueba por lotes

Los casos de prueba por lotes le permiten enviar un lote para leer, procesar y modificar o producir nuevos datos empresariales (registros de conjuntos de datos y/o bases de datos).

Creación de un caso de prueba por lotes

1. Abra la consola de pruebas de aplicaciones de modernización de AWS mainframes en. https://console.aws.amazon.com/apptest/
2. En el Región de AWS selector, elija la región en la que estén disponibles las pruebas de aplicaciones.

   > **Note**
   >
   > En la actualidad, Pruebas de aplicaciones solo se encuentra disponible en las regiones del Este de EE. UU. (Norte de Virginia), Asia-Pacífico (Sídney), Europa (Fráncfort) y América del Sur (São Paulo).

3. En el panel de navegación izquierdo, seleccione Casos de prueba.

4. En Definir caso de prueba, introduzca el nombre del caso de prueba y la descripción opcional. Elija Lote en Tipo de caso de prueba.
5. Elija Siguiente.
6. (Opcional) En la página Especifique los parámetros de JCL de lote, agregue el nombre del JCL (lenguaje de control de tareas) y los parámetros de trabajo (nombres y valores).
7. Elija Siguiente.
8. En la página Origen de datos para capturar, puede elegir entre Cambios en bases de datos relacionales, Conjuntos de datos o ambos.

   - Seleccione Cambios en bases de datos relacionales cuando desee que el caso de prueba modifique los registros de la base de datos.
   - Elija Conjuntos de datos cuando desee que el caso de prueba modifique los conjuntos de datos. En Conjuntos de datos de salida, agregue el nombre del conjunto de datos de salida.

   > **Note**
   >
   > Puede agregar varios conjuntos de datos.

9. Elija Siguiente.
10. En la página Revisar y crear, revise toda la información y elija Crear caso de prueba.

Creación de un caso de prueba de pantalla 3270 en línea

Los casos de prueba de pantalla 3270 en línea son casos de prueba que le permiten ejecutar cuadros de diálogo de pantalla interactivos (3270) para leer, modificar o generar nuevos datos empresariales (registros de bases de datos o conjuntos de datos).

Creación de un caso de prueba de pantalla 3270 en línea

1. Abra la consola de pruebas de aplicaciones de modernización de AWS mainframe enhttps://console.aws.amazon.com/apptest/.
2. En el Región de AWS selector, elija la región en la que estén disponibles las pruebas de aplicaciones.

> **Note**
>
> En la actualidad, Pruebas de aplicaciones solo se encuentra disponible en las regiones del Este de EE. UU. (Norte de Virginia), Asia-Pacífico (Sídney), Europa (Fráncfort) y América del Sur (São Paulo).

3. En el panel de navegación izquierdo, seleccione Casos de prueba.
4. En Definir caso de prueba, introduzca el nombre del caso de prueba y la descripción opcional. Seleccione Pantallas 3270 en línea en Tipo de caso de prueba.
5. Elija Siguiente.

> **Note**
>
> La pantalla 3270 en línea no necesita que especifique los parámetros de JCL.

6. Elija Siguiente.
7. En la página Origen de datos para capturar, la selección predeterminada es Pantallas 3270 en línea. Además, puede elegir Cambios en bases de datos relacionales en Conjuntos de datos.

   - Seleccione Cambios en bases de datos relacionales cuando desee que el caso de prueba modifique los registros de la base de datos.
   - Elija Conjuntos de datos cuando desee que el caso de prueba modifique los conjuntos de datos. En Conjuntos de datos de salida, agregue el nombre del conjunto de datos de salida.

   > **Note**
   >
   > Puede agregar varios conjuntos de datos.

8. Elija Siguiente.
9. En la página Revisar y crear, revise toda la información y elija Crear caso de prueba.

Cree conjuntos de pruebas en las pruebas de aplicaciones de modernización de AWS mainframe

Los conjuntos de pruebas son una serie de casos de prueba que se ejecutan en orden secuencial. Los conjuntos de pruebas son importantes para la reproducción de los casos de prueba.

> **Important**
>
> Para crear conjuntos de pruebas, debe tener al menos un caso de prueba. Puede crear su primer caso de prueba con the section called “Creación de casos de prueba en Pruebas de aplicaciones”.

Para obtener información adicional sobre varios conceptos, consulte the section called “Conceptos de Pruebas de aplicaciones”.

Temas

- Creación de un conjunto de pruebas
- Carga de datos de referencia
- Reproducción y comparación

Creación de un conjunto de pruebas

Los conjuntos de pruebas le permiten ejecutar distintos casos de prueba, reproducirlos y compararlos posteriormente.

Para crear un conjunto de pruebas:

1. Abra la consola de pruebas de aplicaciones de modernización de AWS mainframes en. https://console.aws.amazon.com/apptest/
2. En el Región de AWS selector, elija la región en la que estén disponibles las pruebas de aplicaciones.

   > **Note**
   >
   > En la actualidad, Pruebas de aplicaciones solo se encuentra disponible en las regiones del Este de EE. UU. (Norte de Virginia), Asia-Pacífico (Sídney), Europa (Fráncfort) y América del Sur (São Paulo).

3. En el panel de navegación izquierdo, seleccione Casos de prueba.
4. Elija Crear conjunto de pruebas.
5. En la sección Crear conjuntos de pruebas, busque casos de prueba en la biblioteca de casos de prueba y elija Agregar casos de prueba seleccionados.

> **Note**
>
> Puede agregar hasta 20 casos de prueba en un conjunto de pruebas.

6. En el panel Conjunto de pruebas, introduzca el nombre del conjunto de pruebas y una descripción opcional. Además, seleccione entre el tiempo de ejecución administrado o el tiempo de ejecución no administrado, lo que definirá cómo el conjunto de pruebas configura y desconfigura una AWS aplicación de modernización del mainframe. Si lo desea, añada el URI JSON S3 del conjunto de AWS datos de importación de Mainframe Modernization.
7. En la sección Casos de prueba agregados, apile los casos de prueba en el orden en que desee cargarlos y vuelva a reproducirlos.
8. Elija Crear conjunto de pruebas.

Carga de datos de referencia

Cargue los datos de referencia del mainframe a AWS Application Testing. Solo necesita guardar los datos de referencia cargados la primera vez. El servicio de pruebas puede reutilizar los resultados cargados desde el origen y compararlos consecutivamente con los resultados reproducidos en el destino.

Carga de los datos de referencia

1. En la sección Conjuntos de pruebas, elija el conjunto de pruebas para cargar los datos de referencia.
2. Seleccione Cargar.
3. En la página Cargar datos de referencia, seleccione los casos de prueba que desee reproducir. Complete los campos para la fecha de captura de datos, la ubicación del diario de cambios de la base de datos en S3, la ubicación en S3 de los conjuntos de datos y elija Cargar.

Reproducción y comparación

El proceso de reproducción y comparación asocia su caso de prueba al entorno de prueba de destino y ejecuta la aplicación. Debe cargar los datos antes de ejecutar el proceso de reproducción.

Reproducción y comparación

1. En la sección Conjuntos de pruebas, elija el conjunto de pruebas para reproducir.
2. Elija Reproducir y comparar.
3. En la página Información general de reproducción y comparación, seleccione la configuración del entorno de prueba y revise la información. La función Editar le permite editar cualquier campo de configuración del entorno de prueba. También puede encontrar parámetros de AWS CloudFormation .
4. En la sección Casos de prueba que se reproducirán, elija los casos de prueba y colóquelos en el orden en el que desee reproducirlos.
5. Elija Reproducir y comparar.

Cree configuraciones de entornos de prueba en las pruebas de aplicaciones de modernización de AWS mainframe

Las configuraciones del entorno de prueba le permiten configurar el conjunto inicial de datos y parámetros de configuración (o recursos) AWS CloudFormation que necesita para que la ejecución de la prueba sea repetible.

Para obtener información adicional sobre varios conceptos, consulte the section called “Conceptos de Pruebas de aplicaciones”.

Creación de una configuración de entornos

Configure su entorno de prueba para reproducir y comparar los casos de prueba en Pruebas de aplicaciones.

Establecimiento de las configuraciones del entorno de prueba

1. Abra la consola de pruebas de aplicaciones de modernización de AWS mainframe en. https://console.aws.amazon.com/apptest/
2. En el Región de AWS selector, elija la región en la que estén disponibles las pruebas de aplicaciones.

> **Note**
>
> En la actualidad, Pruebas de aplicaciones solo se encuentra disponible en las regiones del Este de EE. UU. (Norte de Virginia), Asia-Pacífico (Sídney), Europa (Fráncfort) y América del Sur (São Paulo).

3. En el panel de navegación izquierdo, seleccione Configuraciones del entorno de prueba.
4. Elija Crear configuración del entorno de prueba.
5. En el panel Crear configuración del entorno de prueba, introduzca el nombre y la descripción. Añada también su bucket de Amazon S3 que contiene la CloudFormation plantilla para las pruebas de aplicaciones. Además, puede añadir los parámetros CloudFormation de entrada que se utilizarán durante la creación de la CloudFormation pila.
6. Especifique la aplicación de AWS Mainframe Modernization que se verá afectada por esta configuración de prueba. Añada el nombre de la variable de salida para el identificador de la aplicación de modernización del AWS mainframe y el motor de ejecución (AWS Blu Age no gestionado o gestionado por Rocket Software (anteriormente Micro Focus)).

> **Note**
>
> El nombre de la variable de salida del ID de la aplicación AWS Mainframe Modernization debe coincidir con el nombre de la variable de salida de la CloudFormation plantilla para la creación de la pila.

> **Important**
>
> El tiempo de ejecución no administrado de AWS Blu Age también requiere que especifique el nombre de la variable de salida para el ID del servicio de punto final de la VPC, el nombre de la variable de salida para el puerto de escucha y el nombre de la variable de salida para el nombre. WebApp Estos nombres deben coincidir con los nombres de las variables de salida de la plantilla. CloudFormation

7. (Opcional) Se puede definir un atributo adicional, como el nombre de la variable de salida, para el Nombre de recurso de Amazon (ARN) de la tarea del Database Migration Service (DMS), que se utiliza para capturar los cambios en las bases de datos relacionales. Otro atributo es el URI de DDL S3 de la base de datos de origen.

> **Important**
>
> El nombre de la variable de salida debe coincidir con el nombre de la variable de la CloudFormation plantilla.

8. (Opcional) Personalice su clave de Key Management Service (KMS). Para obtener más información, consulte Administración del acceso a las claves en la Guía para desarrolladores de AWS Key Management Service .
9. Elija Crear configuración del entorno de prueba.

Tutorial: Configurar la aplicación de CardDemo muestra en las pruebas de aplicaciones de modernización de AWS mainframe

Para este tutorial, debe crear una AWS CloudFormation pila que le ayude a configurar la aplicación de CardDemo ejemplo para el cambio de plataforma con el servicio gestionado de Micro Focus on AWS Mainframe Modernization y funciones que incluyen las pruebas de aplicaciones de modernización de AWS mainframes. En el tutorial se describe un ejemplo de AWS CloudFormation plantilla que puede utilizar para crear la pila. También proporcionamos un archivo comprimido con los artefactos de aplicación necesarios. La plantilla de ejemplo proporciona una base de datos, un entorno de tiempo de ejecución, una aplicación y un entorno de red completamente aislado.

Esta plantilla crea varios AWS recursos. Estos recursos se le facturarán si crea una pila a partir de esta plantilla.

Requisitos previos

- Descargue y descomprima `IC3-card-demo-zip` y `datasets_Mainframe_ebcdic.zip`. Estos archivos contienen la CardDemo muestra y los conjuntos de datos de muestra para su uso en las pruebas de AWS aplicaciones.
- Cree un bucket de Amazon S3 para almacenar los CardDemo archivos y otros artefactos. Por ejemplo, `my-carddemo-bucket`.

Paso 1: Prepárese para la configuración CardDemo

Cargue los archivos de CardDemo muestra y edite la AWS CloudFormation plantilla que creará la CardDemo aplicación.

1. Cargue las carpetas `datasets_Mainframe_ebcdic` y `IC3-card-demo` que descomprimió anteriormente en su bucket.
2. Descarga la `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation plantilla de tu paquete. Está en la carpeta `IC3-card-demo`.
3. Edita la `aws-m2-math-mf-carddemo.yaml` AWS CloudFormation plantilla de la siguiente manera:
   - Cambie el parámetro `BucketName` por el nombre del bucket que definió anteriormente, por ejemplo `my-carddemo-bucket`.
   - Cambie la `ImportJsonPath` a la ubicación del archivo `mf-carddemo-datasets-import.json` en su bucket. Por ejemplo, `s3://my-carddemo-bucket/IC3-card-demo/mf-carddemo-datasets-import.json`. Al actualizar este valor se asegura que la salida `M2ImportJson` tenga el valor correcto.
   - (Opcional) Adapte los parámetros `EngineVersion` y `InstanceType` para que coincidan con sus estándares.

> **Note**
>
> No modifique los resultados `M2EnvironmentId` y `M2ApplicationId`. Application Testing utiliza esos valores para localizar los recursos con los que interactuará.

Paso 2: Crear todos los recursos necesarios

Ejecute su AWS CloudFormation plantilla personalizada para crear todos los recursos que necesita para completar este tutorial correctamente. Esta plantilla configura la CardDemo aplicación para que pueda utilizarla en las pruebas.

1. Inicie sesión en la AWS CloudFormation consola y elija Crear pila y, a continuación, elija Con nuevos recursos (estándar).
2. En Requisito previo: preparar la plantilla, elija La plantilla está lista.

3. En Especificar plantilla, seleccione Cargar un archivo de plantilla y, a continuación, elija Elegir archivo.
4. Navegue hasta el lugar donde descargó `aws-m2-math-mf-carddemo.yaml`, elija ese archivo y, a continuación, elija Siguiente.
5. En Especificar los detalles de la pila, proporcione un nombre para la pila para que pueda encontrarla fácilmente en una lista y, a continuación, seleccione Siguiente.
6. En Configurar opciones de pila, mantenga los valores predeterminados y seleccione Siguiente.
7. En Revisar, comprueba qué AWS CloudFormation es lo que te está creando y, a continuación, selecciona Enviar.

Se tarda entre 10 y 15 minutos en AWS CloudFormation crear la pila.

> **Note**
>
> La plantilla está configurada para agregar un sufijo único a los nombres de los recursos que crea. Esto significa que puede crear varias instancias de esta plantilla de pila en paralelo, una característica clave para Pruebas de aplicaciones que le permite ejecutar varios conjuntos de pruebas al mismo tiempo.

Paso 3: Implementar e iniciar la aplicación

Implemente la CardDemo aplicación que AWS CloudFormation creó para usted y asegúrese de que se esté ejecutando.

1. Abra la consola de modernización del AWS mainframe y seleccione Aplicaciones en el menú de navegación de la izquierda.
2. Elija la CardDemo aplicación, que tendrá un nombre parecido `aws-m2-math-mf-carddemo-abc1d2e3` a.
3. Elija Acciones y, a continuación, Implementar aplicación.
4. En Entornos, seleccione el entorno de tiempo de ejecución que corresponda a la aplicación. Tendrá el mismo identificador único adjunto al final del nombre. Por ejemplo, `aws-m2-math-mf-carddemo-abc1d2e3`.
5. Elija Implementar. Espere hasta que la aplicación se implemente correctamente y tenga el estado Listo.

6. Seleccione la aplicación, elija Acciones y, a continuación, Iniciar aplicación. Espere hasta que la aplicación esté en estado En ejecución.
7. En la página de detalles de la aplicación, copie el Puerto y el Nombre de host DNS que necesita para conectarse a la aplicación en ejecución.

Paso 4: Importar los datos iniciales

Para utilizar la aplicación de CardDemo ejemplo, debe importar un conjunto inicial de datos. Siga estos pasos:

1. Descargue el archivo `mf-carddemo-datasets-import.json`.
2. Abra el archivo con el editor de texto que prefiera.
3. Localice el parámetro `s3Location` y actualice el valor para que apunte al bucket de Amazon S3 que creó.
4. Realice este mismo cambio todas las veces que aparezca `s3Location` y, a continuación, guarde el archivo.
5. Inicie sesión en la consola de Amazon S3 y vaya al bucket que creó anteriormente.
6. Cargue el archivo `mf-carddemo-datasets-import.json` personalizado.
7. Abra la consola de modernización del AWS mainframe y seleccione Aplicaciones en el menú de navegación de la izquierda.
8. Elija la CardDemo aplicación.
9. Elija Conjuntos de datos y, a continuación, Importar.
10. Navegue hasta la ubicación en Amazon S3 en la que cargó el archivo JSON personalizado y seleccione Enviar.

Este trabajo importa 23 conjuntos de datos. Para supervisar el resultado del trabajo de importación, compruebe la consola. Cuando todos los conjuntos de datos se hayan importado correctamente, conéctese a la aplicación.

> **Note**
>
> Al utilizar esta plantilla en Application Testing, la salida `M2ImportJson` gestiona automáticamente el proceso de importación.

Paso 5: Conectarse a la CardDemo aplicación

Conéctese a la aplicación de CardDemo muestra con el emulador 3270 de su elección.

- Cuando la aplicación esté en ejecución, utilice el emulador 3270 para conectarse a la aplicación y especifique el nombre del servidor DNS y el nombre del puerto, si es necesario.

  Por ejemplo, si utiliza el emulador c3270 de código abierto, el comando tendrá el siguiente aspecto:

  ```
  c3270 -port port-number DNS-hostname
  ```

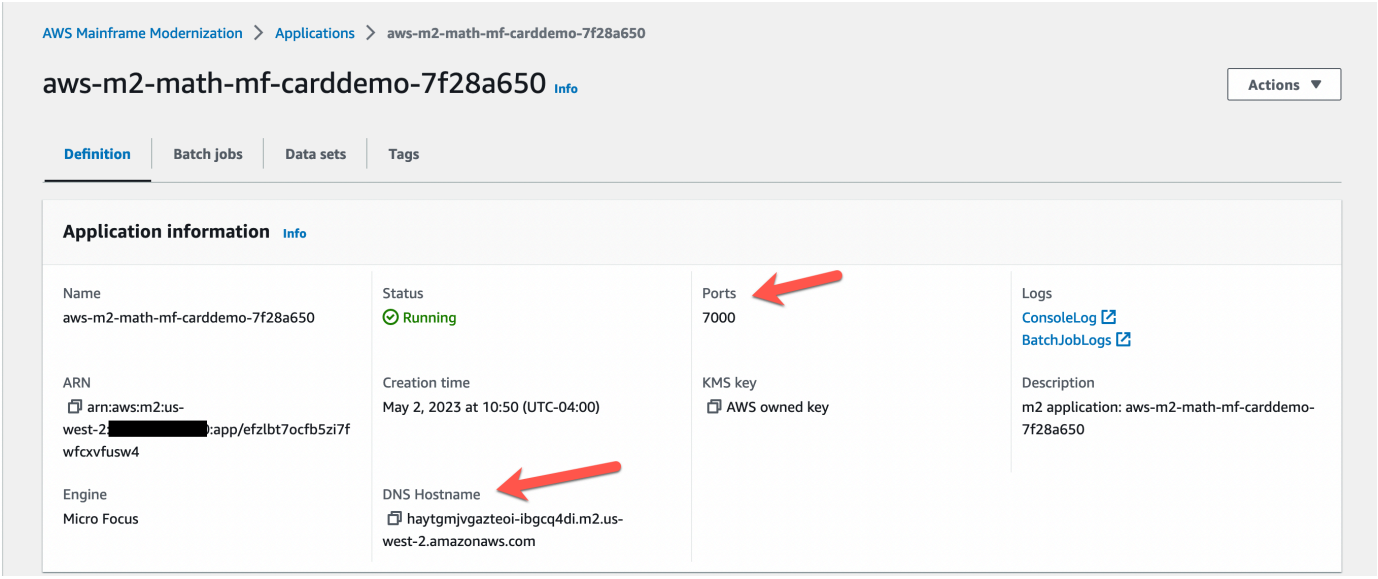
  puerto

  El puerto especificado en la página de detalles de la aplicación. Por ejemplo, 6000.

  Hostname

  El nombre de host de DNS especificado en la página de detalles de la aplicación.

  En la siguiente figura se muestra dónde encontrar el puerto y el nombre de host de DSN.



Tutorial: Reproduzca y compare en pruebas de aplicaciones de modernización de AWS mainframe con CardDemo AWS Blu Age implementado en Amazon EC2

En este tutorial, completarás los pasos necesarios para reproducir y comparar las cargas de trabajo de prueba con la CardDemo aplicación que se ejecuta en AWS Blu Age implementada en Amazon. EC2

Paso 1: Obtenga la imagen de EC2 Amazon Machine (AMI) de AWS Blu Age

Sigue las instrucciones del tutorial de configuración de AWS Blu Age Runtime (en Amazon EC2) para conocer los pasos de incorporación necesarios para acceder a AWS Blu Age en Amazon EC2 AMI.

Paso 2: Iniciar una EC2 instancia de Amazon con la AMI de AWS Blu Age

1. Configura tus AWS credenciales.
2. Identifique la ubicación del archivo binario Amazon EC2 AMI 3.5.0 (solo CLI o versión AWS Blu Age) del bucket de Amazon S3:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/
aws s3 ls s3://aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1/3.5.0/AMI/
```

> **Note**
>
> La característica de Pruebas de aplicaciones solo está disponible para su uso en 4 regiones de producción (us-east-1, sa-east-1, eu-central-1 y ap-southeast-2).

3. Restaure la AMI en su cuenta con el siguiente comando:

```
aws ec2 create-restore-image-task --object-key 3.5.0/AMI/ami-0182ffe3b9d63925b.bin
 --bucket aws-bluage-runtime-artifacts-xxxxxxx-eu-west-1 --region eu-west-1 --name
 "AWS BLUAGE RUNTIME AMI"
```

> **Note**
>
> Cambie el nombre del archivo bin de la AMI y la región en la que quiera crear la AMI.

4. Tras crear una EC2 instancia de Amazon, puede encontrar el ID de AMI correcto que se restauró (AMI) desde el bucket de Amazon S3 en el catálogo de EC2 imágenes de Amazon.

> **Note**
>
> En este tutorial, el ID de AMI es ami-0d0fafcc636fd1e6d y debe cambiarlo en los distintos archivos de configuración por el que se le proporcionó.

1. Si se produce un create-restore-image-task error en aws ec2, compruebe su versión de Python y CLI mediante el siguiente comando:

```
aws --version
```

> **Note**
>
> La versión de Python debe ser >= 3 y la versión de la CLI debe ser >= 2.

2. Si estas versiones están obsoletas, se debe actualizar la CLI. Para actualizar la CLI:
   a. Siga las instrucciones en Instalación o actualización de la versión más reciente de la CLI de AWS.
   b. Elimine la CLI v1 con el siguiente comando:

```
sudo yum remove awscli
```

   c. E instale la CLI v2 con el siguiente comando:

```
curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

   d. Por último, compruebe la versión de Python y la CLI con el siguiente comando:

```
aws --version
```

3. A continuación, puede rehacer el aws create-restore-image-task ec2.

Paso 3: Cargue CardDemo los archivos dependientes a S3

Copie el contenido de las carpetas databases, file-system y userdata. Descarga y descomprime las CardDemo aplicaciones. Estas tres carpetas deben copiarse en uno de sus buckets denominado amzn-s3-demo-bucket en esta documentación.

Paso 4: Cargue las bases de datos e inicialice la aplicación CardDemo

Cree una EC2 instancia temporal de Amazon que utilizará como recurso informático para generar las instantáneas de base de datos necesarias para la CardDemo aplicación. Esta EC2 instancia no ejecutará la CardDemo aplicación por sí misma, sino que generará las instantáneas de la base de datos que se utilizarán más adelante.

Comience por editar la CloudFormation plantilla proporcionada llamada 'load-and-create-ba-snapshots.yml.' Esta es la CloudFormation plantilla que se utiliza para crear la EC2 instancia de Amazon que se utiliza para generar las instantáneas de la base de datos.

1. Genera y proporciona el EC2 key pair que se utilizará en la EC2 instancia. Para obtener más información, consulte Crear pares de claves.

   Ejemplo:

   ```
   Ec2KeyPair:
       Description: 'ec2 key pair'
       Default: 'm2-tests-us-west-2'
       Type: String
   ```

2. Especifique la ruta de Amazon S3 de la carpeta en la que haya colocado la carpeta database del paso anterior:

   ```
   S3DBScriptsPath:
       Description: 'S3 DB scripts folder path'
       Type: String
       Default: 's3://amzn-s3-demo-bucket/databases'
   ```

3. Especifique la ruta de Amazon S3 de la carpeta en la que haya colocado la carpeta file-system del paso anterior:

```
S3ApplicationFilesPath:
    Description: 'S3 application files folder path'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/file-system'
```

4. Especifique la ruta de Amazon S3 de la carpeta en la que haya colocado la carpeta userdata del paso anterior:

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/userdata'
```

5. Especifique también una ruta de Amazon S3 en la que guardará los archivos de resultados para utilizarlos en el siguiente paso.

```
S3SaveProducedFilesPath:
    Description: 'S3 path folder to save produced files'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/post-produced-files'
```

6. Cambie el ID de la AMI por el correcto obtenido anteriormente en este tutorial mediante la siguiente plantilla:

```
  BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0bd41245734fd20d9'
    Type: String
```

- Si lo desea, puede cambiar el nombre de las tres instantáneas que se crearán al ejecutar las bases de datos de carga con CloudFormation ellas. Estarán visibles en la CloudFormation pila a medida que se vaya creando y se utilizarán más adelante en este tutorial. Recuerde anotar los nombres utilizados para las instantáneas de base de datos.

```
SnapshotPrimary:
    Description: 'Snapshot Name DB BA Primary'
    Type: String
```

```
    Default: 'snapshot-primary'

  SnapshotBluesam:
    Description: 'Snapshot Name DB BA Bluesam'
    Type: String
    Default: 'snapshot-bluesam'

  SnapshotJics:
    Description: 'Snapshot Name DB BA Jics'
    Type: String
    Default: 'snapshot-jics'
```

> **Note**
>
> En este documento, damos por sentado que el nombre de las instantáneas sigue siendo coherente.

7. Ejecute CloudFormation con CLI o AWS consola mediante el botón y el asistente Create Stack. Al final del proceso, debería ver tres instantáneas en la consola de RDS con el nombre que eligió, seguido de un ID único. Necesitará estos nombres en el siguiente paso.

> **Note**
>
> RDS agregará sufijos a los nombres de las instantáneas definidos en la plantilla. AWS CloudFormation Asegúrese de obtener el nombre completo de la instantánea desde RDS antes de ir al paso siguiente.

Ejemplo de comando de la CLI

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots --
template-url https://your-apptest-bucket.s3.us-west-2.amazonaws.com/load-and-
create-ba-snapshots.yml --capabilities CAPABILITY_NAMED_IAM
```

También puede comprobar en la ruta de Amazon S3 que proporcionó para S3 SaveProducedFilesPath que los conjuntos de datos se crearon correctamente.

Paso 5: Inicie el tiempo de ejecución de AWS Blu Age CloudFormation

Se utiliza CloudFormation para ejecutar la EC2 instancia de Amazon con la aplicación CardDemo AWS Blu Age. Debe reemplazar algunas variables del CloudFormation nombre `m2-with-ba-using-snapshots-https-authentication.yml` editando el archivo YAML o modificando los valores de la consola durante el lanzamiento del CFN.

1. Modifique `AllowedVpcEndpointPrincipals` para especificar qué cuenta llegará al punto final de la VPC para acceder al tiempo de ejecución de AWS Blu Age mediante los siguientes comandos:

```
AllowedVpcEndpointPrincipals:
    Description: 'comma-separated list of IAM users, IAM roles, or AWS accounts'
    Default: 'apptest.amazonaws.com'
    Type: String
```

2. Cambie el valor de SnapshotPrimaryDb las SnapshotBlusamDb variables y SnapshotJicsDb el nombre de las instantáneas. Obtenga también los nombres de las instantáneas de RDS después de crearlas en el paso anterior.

```
SnapshotPrimary:
    Description: 'Snapshot DB cluster for DB Primary'
    Type: String
    Default: 'snapshot-primary87d067b0'

  SnapshotBluesam:
    Description: 'Snapshot DB cluster for DB Bluesam'
    Type: String
    Default: 'snapshot-bluesam87d067b0'

  SnapshotJics:
    Description: 'Snapshot DB cluster for DB Jics'
    Type: String
    Default: 'snapshot-jics87d067b0'
```

> **Note**
>
> RDS agregará su propio sufijo a los nombres de las instantáneas.

3. Proporcione el par de EC2 claves de Amazon para la EC2 instancia mediante este comando:

```
Ec2KeyPair:
    Description: 'ec2 key pair'
    Default: 'm2-tests-us-west-2'
    Type: String
```

4. Proporcione el ID de AMI que obtuvo durante el proceso de registro de la AMI para la variable BaaAmiIdmediante:

```
BaaAmiId:
    Description: 'ami id (AL2) for ba anywhere'
    Default: 'ami-0d0fafcc636fd1e6d'
    Type: String
```

5. Proporcione la ruta de la carpeta Amazon S3 que utilizó en el paso anterior para guardar los archivos generados mediante el siguiente comando:

```
S3ApplicationFilesPath:
    Description: 'bucket name'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/post-produced-files'
```

6. Por último, indique la ruta de la carpeta s3- userdata-folder-path:

```
S3UserDataPath:
    Description: 'S3 userdata folder path'
    Type: String
    Default: 's3://amzn-s3-demo-bucket/userdata'
```

- (Opcional) Puede activar el modo HTTPS y la autenticación HTTP básica para tomcat. Aunque la configuración predeterminada también funcionaría.

> **Note**
>
> De forma predeterminada, el modo HTTPS está deshabilitado y configurado en modo HTTP en el parámetro BacHttpsMode:

Por ejemplo:

```
BacHttpsMode:
    Description: 'http or https for Blue Age Runtime connection mode '
    Default: 'http'
    Type: String
    AllowedValues: [http, https]
```

- (Opcional) Para habilitar el modo HTTPS, debe cambiar el valor a HTTPS y proporcionar el ARN de su certificado ACM cambiando el valor de la ACMCert variable Arn:

```
ACMCertArn:
    Type: String
    Description: 'ACM certificate ARN'
    Default: 'your arn certificate'
```

- (Opcional) La autenticación básica está deshabilitada de forma predeterminada con el parámetro WithBacBasicAuthenticationestablecido en false. Puede habilitarla, estableciendo el valor como true.

```
WithBacBasicAuthentication:
    Description: 'false or true for Blue Age Runtime Basic Authentication '
    Default: false
    Type: String
    AllowedValues: [true, false]
```

7. Cuando haya completado la configuración, podrá crear la pila mediante la CloudFormation plantilla editada.

Paso 6: Probar la EC2 instancia de Amazon de AWS Blu Age

Ejecuta manualmente la CloudFormation plantilla para crear la EC2 instancia de Amazon de AWS Blu Age para la CardDemo aplicación y asegurarte de que se inicie sin errores. Esto se hace para comprobar que la CloudFormation plantilla y todos los requisitos previos son válidos, antes de utilizar la CloudFormation plantilla con la función de prueba de aplicaciones. Luego, puedes usar Application Testing para crear automáticamente la EC2 instancia Amazon de AWS Blu Age de destino durante la reproducción y la comparación.

1. Ejecuta el comando CloudFormation create stack para crear la EC2 instancia de Amazon de AWS Blu Age y proporciona la CloudFormation plantilla m2- with-ba-using-snapshots -https-authentication.yml que editaste en el paso anterior:

```
aws cloudformation create-stack --stack-name load-and-create-ba-snapshots —-
template-url https://apptest-ba-demo.s3.us-west-2.amazonaws.com/m2-with-ba-using-
snapshots-https-authentication.yml --capabilities CAPABILITY_NAMED_IAM --region us-
west-2
```

> **Note**
>
> Recuerde especificar la región correcta en la que se restauró la AMI de AWS Blu Age.

2. Comprueba que todo funciona correctamente buscando en la consola la EC2 instancia de Amazon en ejecución. Conexión mediante el Administrador de sesiones
3. Después de conectarte a la EC2 instancia de Amazon, usa los siguientes comandos:

```
sudo su
cd /m2-anywhere/tomcat.gapwalk/velocity/logs
cat catalina.log
```

4. Asegúrese de que no haya excepciones ni errores en el registro.
5. A continuación, compruebe que la aplicación responde mediante este comando:

```
curl http://localhost:8080/gapwalk-application/
```

Ahí podrá ver el mensaje: “Jics application is running”.

Paso 7: Validar que los pasos anteriores se hayan completado correctamente

En los siguientes pasos, utilizaremos las pruebas de aplicaciones de modernización AWS del mainframe para reproducir y comparar los conjuntos de datos creados por la aplicación. CardDemo Estos pasos se basan en la finalización satisfactoria de todos los pasos anteriores de este tutorial. Valide lo siguiente antes de continuar:

1. Has creado correctamente la EC2 instancia de AWS Blu Age on Amazon a través de la AWS CloudFormation plantilla.
2. El servicio Tomcat de la era AWS azul de Amazon ya EC2 está en funcionamiento, sin excepciones.

Cuando ejecute la EC2 instancia con la CardDemo aplicación, complete los siguientes pasos en la consola de pruebas de aplicaciones para reproducir y comparar los conjuntos de datos por lotes.

Paso 8: creación del caso de prueba

En este paso, se crea el caso de prueba que se utilizará para comparar los conjuntos de datos creados en la aplicación de CardDemo.

1. Cree un nuevo caso de prueba. Póngale un nombre y una descripción.
2. Especifique `CREASTMT.JCL` como nombre de JCL.
3. Agregue los siguientes conjuntos de datos a la definición del caso de prueba:

Name	CCSID	RecordFormat	RecordLength
AWS.M2.CARDDEMO.STATEMNT.PS	“037”	FB	80
AWS.M2.CARDDEMO.STATEMNT.HTML	“037”	FB	100

> **Note**
>
> El nombre de JCL y los detalles del conjunto de datos deben coincidir.

Paso 9: creación de una conjunto de pruebas

1. Cree un nuevo conjunto de pruebas y póngale un nombre y una descripción.
2. Agregue el caso de prueba que ha creado en el paso anterior a su conjunto de pruebas.
3. Una vez creado el conjunto de pruebas, capture los casos de prueba en el mainframe y cargue los datos de referencia del mainframe en Application Testing. AWS
4. Elija Crear conjunto de pruebas.

Paso 10: creación de una configuración de entorno de prueba

1. Cree una nueva configuración de entorno de prueba y póngale un nombre y una descripción.
2. Añada su plantilla. CloudFormation También puede añadir el nombre y el valor del parámetro de entrada desde su CloudFormation plantilla.
3. Elija el servicio de modernización de AWS mainframe (AWS Blu Age, no gestionado) como entorno de ejecución.
4. Agregue el nombre de la variable de salida para el nombre del ID de la aplicación de modernización de AWS mainframe, el nombre de la variable de salida para el ID del servicio de punto final de la VPC, el nombre de la variable de salida para el puerto Listener y el nombre de la variable de salida para el nombre. WebApp

   > **Note**
   >
   > Los nombres de estos campos deben coincidir con los nombres de las variables de salida de la CloudFormation plantilla que AWS Mainframe Modernization devolverá durante la creación de la pila.

5. (Opcional) Elija el nombre de la variable de salida para el ARN de tarea del DMS (Database Migration Service) y la ubicación del URI de S3 de la base de datos de origen DDL (lenguaje de definición de bases de datos).
6. (Opcional) Personalice su clave de Key Management Service (KMS). Para obtener más información, consulte Administración del acceso a las claves en la Guía para desarrolladores de AWS Key Management Service .
7. Elija Crear configuración del entorno de prueba.

Paso 11: carga de los datos de entrada en el conjunto de pruebas

En este paso, ejecutará casos de prueba en el origen. Para ello:

1. Descargue y ejecute los conjuntos de datos que se originaron en la ejecución principal de la aplicación. CardDemo
2. Suba la carpeta descomprimida a su bucket de Amazon S3. Este bucket de Amazon S3 debe estar en la misma región de que los demás recursos de Application Testing.

> **Note**
>
> Debe haber dos archivos con nombres coincidentes con los nombres de los conjuntos de datos aprobados en el caso de prueba anterior.

3. En la página Información general del conjunto de pruebas, elija el botón Cargar.
4. En la página Cargar datos de referencia, especifique la ubicación de Amazon S3 en la que ha cargado los conjuntos de datos obtenidos del mainframe de origen.
5. Elija Cargar para iniciar el proceso de carga.

> **Note**
>
> Espere a que se complete el registro antes de reproducir y comparar.

Paso 12: Reproducir y comparar

Ejecute el conjunto de pruebas y los casos de prueba en el EC2 entorno objetivo de AWS AWS Blu Age on Amazon. Pruebas de aplicaciones capturará los conjuntos de datos producidos por la reproducción y los comparará con los conjuntos de datos de referencia que se registraron en el mainframe.

1. Elija Reproducir y comparar. Crear la CloudFormation pila y realizar la comparación debería tardar unos tres minutos.

Una vez que todo esté completo, debería disponer de los resultados de la comparación con algunas diferencias creadas intencionadamente para esta demostración.

AWS Modernización de mainframe, pruebas de aplicaciones, conjuntos de datos compatibles y páginas de códigos

Utilice la siguiente tabla para determinar si el identificador de conjunto de caracteres codificado (CCSID) de sus datos es compatible con las pruebas de AWS aplicaciones. Si sus datos utilizan un CCSID no compatible, le recomendamos que lo convierta en un CCSID compatible o que se ponga en contacto con nosotros para obtener ayuda.

CCSID	Juegos de caracteres	Descripción
37	IBM037, IBM-037, Cp037	Host: EE. UU., Canadá (ESA), Países Bajos, Portugal, Brasil, Australia, Nueva Zelanda
273	IBM273, IBM-273, Cp273	Host: Austria, Alemania
277	IBM277, IBM-277, Cp277	Host: Dinamarca, Noruega
278	IBM278, IBM-278, Cp278	Host: Finlandia, Suecia
280	IBM280, IBM-280, Cp280	Host: Italia
284	IBM284, IBM-284, Cp284	Host: España, Latinoamérica (español)
285	IBM285, IBM-285, Cp285	Host: Reino Unido
297	IBM297, IBM-297, Cp297	Host: Francia
300	IBM-300	JAPAN DB EBCDIC
301	IBM-301	Datos de PC: DB de Japón
437	IBM437, IBM-437, US-ASCII, ASCII, Cp437, US-ASCII	Datos de PC: base de PC EE. UU., muchos otros países
500	IBM500, IBM-500, Cp500	Host: Bélgica, Canadá (AS/400), Suiza, Latin-1 internacional
720	IBM-720	MSDOS ÁRABE
737	IBM-737, x- IBM737	MSDOS GRIEGO
75	IBM775, IBM-775	MSDOS BÁLTICO
808	IBM-808	Datos de PC: cirílico, ruso, con euro

CCSID	Juegos de caracteres	Descripción
813	ISO-8859-7, _7 ISO8859	ISO 8859-7: Grecia
819	ISO-8859-1, _1 ISO8859	ISO 8859-1: países Latin-1
833	IBM-833	EBCDIC COREANO
834	IBM-834, x- IBM834	EBCDIC DE DB COREANO
835	IBM-835	EBCD DE DB CHINO-T
836	IBM-836	EBCDIC CHINO-S
837	IBM-837	EBCDIC CHINO-S
850	IBM850, IBM-850, Cp850	Datos de PC: países Latin-1
855	IBM855, IBM-855, Cp855	Datos de PC: cirílico
856	IBM-856, x-, Cp856 IBM856	Datos de PC: hebreo
858	IBM00858, IBM-858, Cp858	Datos de PC: países Latin-1, con euro
859	IBM-859	Datos de PC: LATIN-9
860	IBM860, IBM-860	Datos de PC: portugués
861	IBM861, IBM-861	Datos de PC: Islandia
862	IBM862, IBM-862, Cp862	Datos de PC: hebreo (migración)
863	IBM863, IBM-863	Datos de PC: Canadá
865	IBM865, IBM-865, Cp865	Datos de PC: Din./Noruega
866	IBM866, IBM-866, Cp866	Datos de PC: cirílico, Rusia
867	IBM-867	Datos de PC: hebreo con euro

CCSID	Juegos de caracteres	Descripción
870	IBM870, IBM-870, Cp870	Host: Latin-2 multilingüe
871	IBM871, IBM-871, Cp871	Host: Islandia
874	x- IBM874	Datos de PC: tailandés
875	IBM-875, x-, Cp875 IBM875	Host: Grecia
897	IBM-897	Datos de PC: SB de Japón
912	ISO-8859-2, _2 ISO8859	ISO 8859-2: multilingüe Latin-2
915	ISO-8859-5_5 ISO8859	ISO 8859-5: cirílico
916	ISO-8859-8, _8 ISO8859	ISO 8859-8: hebreos
918	IBM918, IBM-918, Cp918	Host: urdu
920	ISO-8859-9 ISO8859, _9	ISO 8859-9: Latin-5 (ECMA-128, Turquía TS-5881)
921	IBM-921, x-D, CP921 IBM921	Datos de PC: Letonia, Lituania
922	IBM-922, x-D, CP922 IBM922	Datos de PC: Estonia
923	ISO-8859-15, Cp923, _15_FDIS ISO8859	ISO 8859-15: Latin-9
924	IBM-924	ISO 8859-15: Latin-9
927	IBM-927	Datos de PC: chino-T
930	IBM-930, x-0, CP930 IBM93	Host katakana: SBCS extendido. Host kanji: DBCS que incluye 4370 caracteres definidos por el usuario
932	IBM-932	Datos de PC: mix de Japón

CCSID	Juegos de caracteres	Descripción
933	IBM-933, x-D, Cp933 IBM933	Host: SBCS extendido. Host: DBCS, que incluye 1880 caracteres definidos por el usuario y 11 172 caracteres en hangul completo
935	IBM-935, x-D, Cp935 IBM935	Host: SBCS extendido. Host: DBCS que incluye 1880 caracteres definidos por el usuario.
937	IBM-937, x-D, Cp937 IBM937	Host: SBCS extendido. Host: DBCS que incluye 6204 caracteres definidos por el usuario
939	IBM-939, x-D, CP939 IBM939	Host latino: SBCS extendido. Host kanji: DBCS que incluye 4370 caracteres definidos por el usuario.
942	IBM-942, IBM-942C, x-C, CP942, IBM942 CP942C IBM942	Datos de PC: SBCS extendido . Datos de PC: DBCS que incluye 1880 caracteres definidos por el usuario.
943	IBM-943, IBM-943C, Shift_JIS , windows-31j, windows-932, x- C, Cp943, Cp943C, IBM943 IBM943 MS932	Datos de PC: SBCS. Datos de PC: DBCS para entornos abiertos que incluye 1880 caracteres IBM definidos por el usuario.
947	IBM-947	CHINO-T BIG-5

CCSID	Juegos de caracteres	Descripción
948	IBM-948, x-D, CP948 IBM948	Datos de PC: SBCS extendido . Datos de PC: DBCS que incluye 6204 caracteres definidos por el usuario
949	IBM-949, IBM-949C, x-C, CP949, CP949C IBM949 IBM949	Código IBM KS - datos de PC: SBCS. Código IBM KS - datos de PC: DBCS que incluye 1880 caracteres definidos por el usuario
950	Big5, IBM-950IBM95, x-0, CP950	Datos de PC: SBCS (IBM). BIG5 Datos de PC: DBCS, incluidos 13 493 caractere s CNS, 566 caracteres seleccionados por IBM y 6204 caracteres definidos por el usuario
951	IBM-951	Datos de PC: IBM KS
954	EUC-JP, IBM-954, IBM-954C	G0: JIS X201 latino. G1: JIS X208-1990. G1: JIS X201 katakana. G1: JIS X212
964	EUC-TW, IBM-964, x-, Cp964 IBM964	G0: ASCII. G1: CNS 11 643 plano 1. G1: CNS 11 643 plano 2.
970	EUC-KR, x-0, Cp970 IBM97	G0: ASCII. G1: KSC X5601-1989 que incluye 1880 caracteres definidos por el usuario
971	IBM-971	EUC COREANO

CCSID	Juegos de caracteres	Descripción
1006	IBM-1006, x-006, Cp1006 IBM1	ISO-8: urdu
1025	IBM-1025, x-025, CP1025 IBM1	Host: cirílico multilingüe
1026	IBM1026, IBM-1026, Cp1026	Host: Latin-5 (Turquía)
1027	IBM-1027	EBCD JAPONÉS LATINO
1041	IBM-1041	Datos de PC: Japón
1043	IBM-1043	Datos de PC: chino-T
1046	IBM-1046, IBM-1046S, x-046 IBM1	ÁRABE - PC
1047	IBM1047, IBM-1047	Host: Latin-1
1051	hp-roman8	EMULACIÓN HP
1088	IBM-1088	Datos de PC: Corea KS
1089	ISO-8859-6, _6 ISO8859	ISO 8859-6: árabe
1097	IBM-1097, x-097, CP1097 IBM1	Host: farsi
1098	IBM-1098, x-098, CP1098 IBM1	Datos de PC: farsi
112	IBM-1112, x-, CP1112 IBM1112	Host: Letonia, Lituania
114	IBM-1114	Datos de PC: T-CH SB
1115	IBM-1115	Datos de PC: S-CH SB

CCSID	Juegos de caracteres	Descripción
1122	IBM-1122, x-, Cp1122 IBM1122	Host: Estonia
123	IBM-1123, x-, Cp1123 IBM1123	Host: Ucrania cirílico
124	IBM-1124, x-, CP1124 IBM1124	8 bits: cirílico, Bielorrusia
1140	IBM01140, IBM-1140, Cp1140	Host: EE. UU., Canadá (ESA), Países Bajos, Portugal, Brasil, Australia, Nueva Zelanda, con euro
1141	IBM01141, IBM-1141, Cp1141	Host: Austria, Alemania, con euro
1142	IBM01142, IBM-1142, Cp1142	Host: Dinamarca, Noruega, con euro
1143	IBM01143, IBM-1143, Cp1143	Host: Finlandia, Suecia, con euro
1144	IBM01144, IBM-1144, Cp1144	Host: Italia, con euro
1145	IBM01145, IBM-1145, Cp1145	Host: España, Latinoamérica (español), con euro
1146	IBM01146, IBM-1146, Cp1146	Host: Reino Unido, con euro
1147	IBM01147, IBM-1147, Cp1147	Host: Francia, con euro
1148	IBM01148, IBM-1148, Cp1148	Host: Bélgica, Canadá (AS/400), Suiza, Latin-1, con euro
1149	IBM01149, IBM-1149, Cp1149	Host: Islandia, con euro

CCSID	Juegos de caracteres	Descripción
1200	UTF-16BE	Unicode con conjunto de caracteres 65535. En ausencia de una marca de orden de bytes (BOM), se supone que es UTF-16 BE (big-endian).
1202	UTF-16LE	UTF-16 LE con IBM PUA
1204	UTF-16	UTF-16 con IBM PUA
1208	UTF-8, UTF-8J, UTF8	Unicode con conjunto de caracteres 65535. UTF-8.
1232	UTF-32BE	UTF-32 BE con IBM PUA
1234	UTF-32LE	UTF-32 LE con IBM PUA
1236	UTF-32	UTF-32 con IBM PUA
1351	IBM-1351	JAPÓN ABIERTO
1362	IBM-1362	COREANO MS-WIN
1363	IBM-1363, IBM-1363C, windows-949, MS949	Datos de PC: MS Windows coreano SBCS. Datos de PC: MS Windows coreano DBCS, incluido el hangul completo 11 172
1364	IBM-1364	Host: SBCS extendido. Host: DBCS, que incluye 1880 caracteres definidos por el usuario y 11 172 caracteres en hangul completo

CCSID	Juegos de caracteres	Descripción
1370	IBM-1370	Datos de PC: SBCS extendido , con euro. Datos de PC: DBCS que incluye 6204 caracteres definidos por el usuario, con euro
1371	IBM-1371	Host: SBCS extendido, con euro. Host: DBCS que incluye 6204 caracteres definidos por el usuario, con euro
1375	Big5-HKSCS	Big-5 Ext mixto para HKSCS
1380	IBM-1380	Datos de PC: S-CH SB
1381	IBM-1381, x-, Cp1381 IBM1381	Datos de PC: SBCS extendido (IBM GB). Datos de PC: DBCS (IBM GB), incluidos 31 caracteres seleccionados por IBM y 1880 definidos por el usuario
1382	IBM-1382	EUC CHINO-S
1383	EUC-CN, IBM-1383 GB2312, x-, Cp1383 IBM1383	G0: ASCII. G1: conjunto GB 2312-80
1385	IBM-1385	Datos de PC: S-CH GBK
1386	GBK, IBM-1386, windows-936, MS936	Datos de PC: GBK chino-S y IBM BIG-5 chino-T. Datos de PC: S-CH GBK
138	IBM-1388	Host: SBCS extendido. Host: DBCS, que incluye 1880 caracteres definidos por el usuario

CCSID	Juegos de caracteres	Descripción
1390	IBM-1390	Host katakana: SBCS extendido, con euro. Host kanji: DBCS que incluye 6205 caracteres definidos por el usuario
1399	IBM-1399	Host latino: SBCS extendido , con euro. Host kanji: DBCS que incluye 4370 caracteres definidos por el usuario, con euro
5050	JIS0201, JIS0208, JIS0212, JIS0201, JIS0208, JIS0212	G0: JIS X201 latino. G1: JIS X208-1990. G1: JIS X201 katakana. G1: JIS X212
5054	ISO-2022-JP	TCP JAPONÉS
5346	windows-1250, Cp1250	MS Windows: Latin-2, versión 2 con euro
5347	windows-1251, Cp1251	MS Windows: cirílico, versión 2 con euro
5348	windows-1252, Cp1252	MS Windows: países Latin-1, versión 2 con euro
5349	windows-1253, Cp1253	MS Windows: Grecia, versión 2 con euro
5350	windows-1254, Cp1254	MS Windows: Turquía, versión 2 con euro
5351	windows-1255, Cp1255	MS Windows: hebreo, versión 2 con euro

CCSID	Juegos de caracteres	Descripción
5352	windows-1256, windows-1 256S, Cp1256	MS Windows: árabe, versión 2 con euro
5353	windows-1257, Cp1257	MS Windows: región báltica, versión 2 con euro
5354	windows-1258, Cp1258	MS Windows: vietnamita, versión 2 con euro
5488	GB18030	GB18030, datos de 1 byte 030, datos de 2 bytes GB18 030, datos de 4 bytes GB18
9030	IBM-838, Cp838	Host: SBCS extendido tailandés.
9066	IBM-874, Cp874	Datos de PC: SBCS extendido tailandés
9400	CESU-8	CESU-8 con IBM PUA
25546	ISO-2022-KR	TCP COREANO
33722	IBM-33722, IBM-33722C	IBMeucJP

Protección de datos en las pruebas de aplicaciones AWS de modernización de mainframe

Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulta las Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulta la publicación de blog sobre el Modelo de responsabilidad compartida de AWS y GDPR en el Blog de seguridad de AWS .

Le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). Como resultado, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Utilice SSL/TLS para comunicarse con los recursos. AWS Se recomienda el uso de TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con. AWS CloudTrail
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.
- Si necesita módulos criptográficos validados por FIPS 140-2 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulte Estándar de procesamiento de la información federal (FIPS) 140-2.

Se recomienda evitar usar información confidencial o sensible, como, por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre (por ejemplo, el campo Nombre). Esto incluye cuando trabaja con la modernización del AWS mainframe, las pruebas de aplicaciones o cualquier otro tipo de pruebas Servicios de AWS mediante la consola, la API o AWS CLI AWS SDKs Cualquier dato que introduzca en etiquetas o campos de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, evite usar información de credenciales en la URL para validar la solicitud para ese servidor.

Datos recopilados mediante las pruebas de la aplicación de modernización AWS del mainframe

AWS Las pruebas de aplicaciones de modernización del mainframe recopilan varios tipos de datos sobre usted:

- `Resource definition`: la definición de recurso indica los datos que se pasan a Pruebas de aplicaciones al crear o actualizar un recurso del tipo de caso de prueba, conjunto de pruebas o configuración de prueba.

- `Scripts for replay`: Se trata de scripts que se pasan a las pruebas de aplicación para su aplicación de modernización AWS del mainframe.
- `Data for comparison`: son conjuntos de datos o archivos de captura de datos de cambios (CDC) de base de datos que se pasan a Pruebas de aplicaciones para su comparación.

AWS Las pruebas de aplicaciones de modernización de mainframe almacenan estos datos de forma nativa en. AWS Los datos que recopilamos sobre usted se almacenan en un bucket Amazon S3 de Amazon S3 gestionado por las pruebas de aplicaciones de modernización del AWS mainframe. Al eliminar un recurso, los datos asociados se eliminan del bucket de Amazon S3.

Al iniciar una ejecución de prueba para reproducir para cargas de trabajo interactivas, Pruebas de aplicaciones de AWS Mainframe Modernization descarga el script en un contenedor Fargate administrado por Amazon ECS y respaldado por almacenamiento efímero para realizar la reproducción. El archivo de script se elimina una vez finalizada la reproducción y el archivo de salida generado por el script se almacena en el bucket de Amazon S3 administrado por Pruebas de aplicaciones en su cuenta. El archivo de salida de reproducción se elimina del bucket de Amazon S3 al eliminar la ejecución de prueba.

Del mismo modo, al iniciar una ejecución de prueba para comparar archivos (conjuntos de datos o cambios en la base de datos), Pruebas de aplicaciones de AWS Mainframe Modernization descarga los archivos en un contenedor Fargate administrado por Amazon ECS y con copia en almacenamiento efímero para realizar la comparación. Los archivos descargados se eliminan en cuanto finaliza la operación de comparación. Los datos de salida de la comparación se almacenan en el bucket de Amazon S3 administrado por Pruebas de aplicaciones en la cuenta. Los datos de salida se eliminan del bucket de S3 al eliminar la ejecución de la prueba.

Puede utilizar todas las opciones de cifrado de Amazon S3 disponibles para proteger sus datos al colocarlos en el depósito de Amazon S3 que las pruebas de aplicaciones de modernización de AWS mainframe utilizan para comparar archivos.

Cifrado de datos en reposo para Pruebas de aplicaciones de AWS Mainframe Modernization

AWS Las pruebas de aplicaciones de modernización del mainframe se integran con AWS Key Management Service (KMS) para proporcionar un cifrado transparente del lado del servidor (SSE) en todos los recursos dependientes que almacenan datos de forma permanente. Entre los ejemplos de recursos se incluyen Amazon Simple Storage Service, Amazon DynamoDB y Amazon Elastic Block

Store. AWS Las pruebas de aplicaciones de modernización de mainframe crean y administran AWS KMS claves de cifrado simétricas para usted. AWS KMS

El cifrado de los datos en reposo de forma predeterminada ayuda a reducir la sobrecarga operativa y la complejidad que implica la protección de los datos confidenciales. Al mismo tiempo, le permite probar aplicaciones seguras que requieren estrictos requisitos normativos y de conformidad del cifrado.

No puede deshabilitar esta capa de cifrado ni seleccionar un tipo de cifrado alternativo al crear casos de prueba, conjuntos de pruebas o configuraciones de prueba.

Puede utilizar su propia clave gestionada por el cliente para los archivos y AWS CloudFormation plantillas de comparación a fin de cifrar Amazon S3. Puede usar esta clave para cifrar todos los recursos creados para las ejecuciones de pruebas en Pruebas de aplicaciones.

> **Note**
>
> Los recursos de DynamoDB siempre se cifran mediante Clave administrada de AWS una cuenta de servicio en Application Testing. No se pueden cifrar los recursos de DynamoDB con una clave administrada por el cliente.

AWS Las pruebas de aplicaciones de modernización de mainframe utilizan la clave administrada por el cliente para las siguientes tareas:

- Exportación de conjuntos de datos de Pruebas de aplicaciones a Amazon S3.
- Carga de archivos de salida de comparación en Amazon S3.

Para más información, consulte las claves administradas por el cliente en la Guía para desarrolladores de AWS Key Management Service .

Creación de una clave administrada por el cliente

Puede crear una clave simétrica gestionada por el cliente utilizando la AWS Management Console o la. AWS KMS APIs

Para crear una clave simétrica administrada por el cliente

Siga los pasos para crear una clave simétrica gestionada por el cliente que se indican en la Guía para desarrolladores de AWS Key Management Service .

Política de claves

Las políticas de clave controlan el acceso a la clave administrada por el cliente. Cada clave administrada por el cliente debe tener exactamente una política de clave, que contiene instrucciones que determinan quién puede usar la clave y cómo puede utilizarla. Cuando crea la clave administrada por el cliente, puede especificar una política de clave.

El siguiente es un ejemplo de política clave de acceso limitado ViaService que permite a Application Testing escribir en su cuenta los datos generados por la reproducción y la comparación. Debe asociar esta política al rol de IAM al invocar la API de `StartTestRun`.

Example

```
{
    "Sid": "TestRunKmsPolicy",
    "Action": ["kms:Decrypt", "kms:GenerateDataKey"],
    "Effect": "Allow",
    "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/TestRunRole"
    },
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "kms:ViaService": ["s3.amazonaws.com"]
        },
        "ForAnyValue:StringEquals": {
            "kms:EncryptionContextKeys": "aws:apptest:testrun"
        }
    }
}
```

Para obtener más información, consulte Administración del acceso a las claves en la Guía para desarrolladores de AWS Key Management Service .

Para obtener información sobre la solución de problemas de acceso a las claves, consulte la Guía para desarrolladores de AWS Key Management Service .

Especificación de una clave administrada por el cliente para Pruebas de aplicaciones de AWS Mainframe Modernization

Al crear una configuración de prueba, puede especificar una clave administrada por el cliente introduciendo un KEY ID. Pruebas de aplicaciones se utiliza para cifrar los datos cargados en el bucket de Amazon S3 durante la ejecución de la prueba.

- KEY ID: identificador de clave para una clave administrada por el cliente. Introduzca el ID de la clave, el ARN de la clave, el nombre de alias o el ARN del alias.

Para añadir la clave gestionada por el cliente al crear una configuración de prueba con el AWS CLI, especifique el parámetro de la `kmsKeyId` siguiente manera:

```
create-test-configuration --name test \
--resources '[{
    "name": "TestApplication",
    "type": {
        "m2ManagedApplication": {
            "applicationId": "wqju4m2dcz3rhny5fpdozrsdd4",
            "runtime": "MicroFocus"
        }
    }
}]' \
--service-settings '{
    "kmsKeyId": "arn:aws:kms:us-west-2:111122223333:key/05d467z6-c42d-40ad-b4b7-274e68b14013"
}'
```

AWS Contexto de cifrado de pruebas de aplicaciones de modernización de mainframe

Un contexto de cifrado es un conjunto opcional de pares clave-valor que pueden contener información contextual adicional sobre los datos.

AWS KMS utiliza el contexto de cifrado como datos autenticados adicionales para respaldar el cifrado autenticado. Al incluir un contexto de cifrado en una solicitud de cifrado de datos, AWS KMS vincula el contexto de cifrado a los datos cifrados. Para descifrar los datos, debe incluir el mismo contexto de cifrado en la solicitud.

AWS Modernización del mainframe: aplicación, prueba, contexto de cifrado

AWS Las pruebas de aplicaciones de modernización de mainframes utilizan el mismo contexto de cifrado en todas las operaciones AWS KMS criptográficas relacionadas con una ejecución de prueba, donde la clave es `aws:apptest:testrun` y el valor es el identificador único de la ejecución de la prueba.

Example

```
"encryptionContext": {
        "aws:apptest:testrun": "u3qd7uhdandgdkhhi44qv77iwq"
}
```

Uso del contexto de cifrado para la supervisión

Si utiliza una clave simétrica administrada por el cliente para cifrar la ejecución de prueba, también puede utilizar el contexto de cifrado en los registros y registros de auditoría para identificar cómo se está utilizando la clave administrada por el cliente al cargar los datos en Amazon S3.

Supervisión de las claves de cifrado para Pruebas de aplicaciones de AWS Mainframe Modernization

Cuando utiliza una clave gestionada por el AWS KMS cliente con sus recursos de pruebas de aplicaciones de modernización de AWS mainframe, puede utilizarla AWS CloudTrailpara realizar un seguimiento de las solicitudes que las pruebas de aplicaciones de modernización de AWS mainframe envían a Amazon S3 al cargar objetos.

Cifrado en tránsito

En los casos de prueba en los que se definen los pasos para probar las cargas de trabajo transaccionales, los intercambios de datos entre el emulador de terminal gestionado por Application Testing que ejecuta sus scripts de selenio y los puntos finales de la aplicación AWS Mainframe Modernization no se cifran durante el tránsito. AWS Las pruebas de aplicaciones de modernización del mainframe se utilizan AWS PrivateLink para conectarse al punto final de la aplicación para intercambiar datos de forma privada sin exponer el tráfico a través de la Internet pública.

AWS Las pruebas de aplicaciones de modernización de mainframes utilizan HTTPS para cifrar el servicio. APIs Todas las demás comunicaciones dentro de las pruebas de aplicaciones de

modernización del AWS mainframe están protegidas por la VPC del servicio o el grupo de seguridad, así como por HTTPS.

El cifrado básico en tránsito está configurado de manera predeterminada, pero no se aplica a las pruebas de carga de trabajo interactivas basadas en el protocolo `TN3270`.

Cómo funcionan las pruebas de aplicaciones de modernización de AWS mainframe con IAM

Antes de usar la IAM para administrar el acceso a las pruebas de aplicaciones de modernización del AWS mainframe, infórmese sobre las funciones de IAM disponibles para su uso en las pruebas de aplicaciones de modernización del mainframe. AWS

Funciones de IAM que puede utilizar con las pruebas de aplicaciones de modernización de mainframes AWS

Característica de IAM	AWS Soporte para pruebas de aplicaciones de modernización de mainframes
Políticas basadas en identidades	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí
Recursos de políticas	Sí
Claves de condición de política	Sí
ACLs	No
ABAC (etiquetas en políticas)	Sí
Credenciales temporales	Sí
Sesiones de acceso directo (FAS)	Sí
Roles de servicio	No
Roles vinculados al servicio	No

Para obtener una visión general de cómo funcionan las pruebas de aplicaciones y otros AWS servicios de modernización del AWS mainframe con la mayoría de las funciones de IAM, consulte AWS los servicios que funcionan con IAM en la Guía del usuario de IAM.

Políticas basadas en la identidad para las pruebas de aplicaciones de modernización de mainframes AWS

Compatibilidad con las políticas basadas en identidad: sí

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. No es posible especificar la entidad principal en una política basada en identidad porque se aplica al usuario o rol al que está asociada. Para obtener más información sobre los elementos que puede utilizar en una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Ejemplos de políticas basadas en la identidad para las pruebas de aplicaciones de modernización de mainframes AWS

Para ver ejemplos de políticas basadas en la identidad de las pruebas de aplicaciones de modernización de AWS mainframes, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Políticas basadas en recursos en las pruebas de aplicaciones de modernización de mainframes AWS

Admite políticas basadas en recursos: no

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios puede utilizarlos para controlar el acceso a un recurso específico. Para

el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Añadir a una política en función de recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando el principal y el recurso son diferentes Cuentas de AWS, el administrador de IAM de la cuenta de confianza también debe conceder a la entidad principal (usuario o rol) permiso para acceder al recurso. Para conceder el permiso, adjunte la entidad a una política basada en identidad. Sin embargo, si la política basada en recursos concede acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para obtener más información, consulte Cross account resource access in IAM en la Guía del usuario de IAM.

Medidas normativas para las pruebas de AWS aplicaciones de modernización de mainframes

Compatibilidad con las acciones de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Las acciones políticas suelen tener el mismo nombre que la operación de AWS API asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de las acciones de prueba de las aplicaciones de modernización del AWS mainframe, consulte las acciones definidas por las pruebas de las aplicaciones de modernización del AWS mainframe en la referencia de autorización del servicio.

Las acciones políticas en las pruebas de aplicaciones de modernización de AWS mainframes utilizan el siguiente prefijo antes de la acción:

```
apptest
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
      "apptest:CreateTestCase",
      "apptest:StartTestRun"
]
```

Puede utilizar caracteres comodín (*) para especificar varias acciones . Por ejemplo, para especificar todas las acciones que comiencen con la palabra `List`, incluya la siguiente acción:

```
"Action": "apptest:List*"
```

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Recursos de políticas para las pruebas de aplicaciones de modernización de AWS mainframes

Compatibilidad con los recursos de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puedes realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puedes hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utiliza un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Puede restringir el acceso a recursos específicos de pruebas de aplicaciones de modernización de AWS mainframes utilizándolos ARNs para identificar el recurso al que se aplica la política de IAM.

Para obtener más información sobre el formato de ARNs, consulte Amazon Resource Names (ARNs) en Referencia general de AWS.

Por ejemplo, un caso de prueba de una aplicación de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:apptest:regionId:accountId:testcase/service-generated-unique-identifier"
```

Una configuración de prueba de aplicaciones de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:apptest:regionId:accountId:testconfiguration/service-generated-unique-identifier"
```

Un conjunto de pruebas de aplicaciones de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:apptest:regionId:accountId:testsuite/service-generated-unique-identifier"
```

La ejecución de una prueba de una aplicación de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:apptest:regionId:accountId:testrun/service-generated-unique-identifier"
```

No todas las acciones de prueba de aplicaciones de modernización AWS del mainframe admiten permisos a nivel de recursos. Para las acciones que no admiten permisos de recursos, debe utilizar el comodín (*).

Las siguientes acciones de prueba de aplicaciones de modernización de AWS mainframe no admiten permisos a nivel de recursos.

```
ListTestCases
ListTestConfigurations
ListTestRuns
ListTestSuites
ListTagsForResource
```

Para ver una lista de los tipos de recursos de pruebas de aplicaciones de modernización de AWS mainframe y sus tipos de recursos ARNs, consulte los recursos definidos por las pruebas de aplicaciones de modernización de mainframe de AWS en la referencia de autorización de servicios. Para obtener información acerca de las acciones con las que puede especificar el ARN de cada recurso, consulte Acciones definidas por Pruebas de aplicaciones de Mainframe Modernization.

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Claves de condición de política para las pruebas de aplicaciones de modernización de AWS mainframes

Compatibilidad con claves de condición de políticas específicas del servicio: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puedes realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puedes crear expresiones condicionales que utilizan operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación `AND` lógica. Si especifica varios valores para una única clave de condición, AWS evalúa la condición mediante una `OR` operación lógica. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puedes utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puedes conceder un permiso de usuario de IAM para acceder a un recurso solo si está etiquetado con su nombre de usuario de IAM. Para más información, consulta Elementos de la política de IAM: variables y etiquetas en la Guía del usuario de IAM.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Para ver una lista de las claves de condición de las pruebas de las aplicaciones de modernización de AWS mainframes, consulte las claves de condición para las pruebas de las aplicaciones de

modernización de AWS mainframes en la Referencia de autorización de servicio. Para obtener más información sobre las acciones y los recursos con los que puede utilizar una clave de condición, consulte Acciones definidas por Pruebas de aplicaciones de AWS Mainframe Modernization.

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Listas de control de acceso (ACLs) en las pruebas de aplicaciones de modernización de AWS mainframes

Soporta ACLs: No

Las listas de control de acceso (ACLs) controlan qué directores (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLs son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Control de acceso basado en atributos (ABAC) con Pruebas de aplicaciones de AWS Mainframe Modernization

Admite ABAC (etiquetas en las políticas): sí

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puede adjuntar etiquetas a las entidades de IAM (usuarios o roles) y a muchos AWS recursos. El etiquetado de entidades y recursos es el primer paso de ABAC. A continuación, designa las políticas de ABAC para permitir operaciones cuando la etiqueta de la entidad principal coincida con la etiqueta del recurso al que se intenta acceder.

ABAC es útil en entornos que crecen con rapidez y ayuda en situaciones en las que la administración de las políticas resulta engorrosa.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/key-name`, `aws:RequestTag/key-name` o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más información sobre ABAC, consulte Definición de permisos con la autorización de ABAC en la Guía del usuario de IAM. Para ver un tutorial con los pasos para configurar ABAC, consulta Uso del control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

Uso de credenciales temporales en las pruebas de aplicaciones de modernización AWS de mainframes

Compatibilidad con credenciales temporales: sí

Algunas Servicios de AWS no funcionan cuando se inicia sesión con credenciales temporales. Para obtener información adicional, incluida la información sobre cuáles Servicios de AWS funcionan con credenciales temporales, consulta Cómo Servicios de AWS funcionan con IAM en la Guía del usuario de IAM.

Utiliza credenciales temporales si inicia sesión en ellas AWS Management Console mediante cualquier método excepto un nombre de usuario y una contraseña. Por ejemplo, cuando accedes AWS mediante el enlace de inicio de sesión único (SSO) de tu empresa, ese proceso crea automáticamente credenciales temporales. También crea credenciales temporales de forma automática cuando inicia sesión en la consola como usuario y luego cambia de rol. Para obtener más información sobre el cambio de roles, consulte Cambio de un usuario a un rol de IAM (consola) en la Guía del usuario de IAM.

Puedes crear credenciales temporales manualmente mediante la AWS CLI API o. AWS A continuación, puede utilizar esas credenciales temporales para acceder AWS. AWS recomienda generar credenciales temporales de forma dinámica en lugar de utilizar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.

Reenvío de las sesiones de acceso de Pruebas de aplicaciones de AWS Mainframe Modernization

Admite sesiones de acceso directo (FAS): sí

Cuando utilizas un usuario o un rol de IAM para realizar acciones en AWSél, se te considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos del principal que llama y los que solicita Servicio de AWS para realizar solicitudes a los servicios descendentes. Servicio de AWS Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS recursos para completarse. En este caso, debe tener permisos para realizar

ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.

> **Important**
>
> Estos tokens permiten a AWS Mainframe Modernization Application Testing acceder a los datos de los clientes sin su consentimiento explícito; por ejemplo, AWS Mainframe Modernization Application Testing entrega los resultados de las pruebas a un bucket de Amazon S3 de un cliente sin obtener el permiso explícito del cliente. Es posible que necesite actualizar la documentación de cumplimiento en consecuencia.

Funciones de servicio para las pruebas de aplicaciones de AWS modernización de mainframes

Compatible con roles de servicio: No

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

Roles vinculados al servicio para Pruebas de aplicaciones de AWS Mainframe Modernization

Compatibilidad con roles vinculados al servicio: no

Un rol vinculado a un servicio es un tipo de rol de servicio que está vinculado a un. Servicio de AWS El servicio puedes asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Refactorización automática de aplicaciones con AWS Blu Age

La refactorización automatizada con AWS Blu Age proporciona una end-to-end solución para migrar y modernizar sus aplicaciones de mainframe. Los pasos del proceso de refactorización son los siguientes:

- Análisis del inventario
- Análisis de las dependencias
- Transformación automática del código
- Captura y administración de escenarios de prueba

Puede completar los pasos anteriores en la herramienta Blu Insights, disponible mediante el inicio de sesión único desde la consola de modernización del mainframe. AWS Para obtener más información sobre Blu Insights, consulte la documentación de Blu Insights.

Cuando esté satisfecho con el código fuente transformado, es el momento de continuar AWS y completar los siguientes pasos:

- Cree e implemente la aplicación refactorizada.
- Implemente y supervise su aplicación en AWS Mainframe Modernization.

AWS Blu Age ofrece varias opciones de tiempo de ejecución para adaptarse a diferentes escenarios de despliegue y preferencias operativas. Estos incluyen tiempos de ejecución gestionados y no gestionados, cada uno con su propio conjunto de características y objetivos de despliegue.

Para obtener una descripción completa de las opciones de tiempo de ejecución de AWS Blu Age disponibles, incluidas las versiones administradas y no administradas, los objetivos de implementación y sus características respectivas, consulte la documentación. the section called “AWS Opciones de Blu Age Runtime”

Esta guía le ayudará a entender las diferencias entre las opciones de tiempo de ejecución y a elegir la más adecuada para su proyecto de modernización.

Temas

- AWS Lanzamientos de Blu Age
- AWS Conceptos de Blu Age Runtime
- Configurar la configuración de AWS Blu Age Runtime
- AWS Tiempo de ejecución Blu Age APIs
- Configurar AWS Blu Age Runtime (no gestionado)
- Modificación del código fuente con el IDE para desarrolladores de Blu Age
- AWS Preguntas frecuentes sobre Blu Age

AWS Lanzamientos de Blu Age

AWS El motor Blu Age tiene varias versiones entre las que puedes elegir. En esta página se ofrece información general sobre cómo funciona el control de versiones de AWS Blu Age, los cambios que se producen en cada versión, las instrucciones de actualización de las distintas versiones, cómo se comunican las actualizaciones de AWS Blu Age a los clientes y el ciclo de vida de estas versiones.

La página the section called “AWS Control de versiones de Blu Age” detalla información sobre las versiones y cómo se puede identificar cada una de ellas por versiones principales y secundarias. La the section called “AWS Notas de lanzamiento de Blu Age” página contiene notas de lanzamiento detalladas para cada versión principal y secundaria. the section called “AWS Vulnerabilidades de seguridad de la era”La página menciona cómo AWS Blu Age gestiona las vulnerabilidades y exposiciones comunes (CVE). the section called “Actualización de AWS Blu Age”detalla las instrucciones de actualización para las versiones de AWS Blu Age. the section called “AWS Ciclo vital de la era azul”Incluye todos los detalles sobre las fechas de finalización del ciclo de vida (EOL) de las principales versiones de AWS Blu Age Runtime.

Temas

- AWS Control de versiones de Blu Age
- AWS Opciones de Blu Age Runtime
- AWS Notas de lanzamiento de Blu Age
- AWS Vulnerabilidades de seguridad de la era
- Instrucciones de actualización para AWS Blu Age
- AWS Ciclo vital de la era azul

AWS Control de versiones de Blu Age

Los productos AWS Blu Age Transformation y Runtime se versionan mediante un esquema compatible con semver (control de versiones semánticas). Para implementar la aplicación, debe usar la versión de tiempo de ejecución correspondiente que sea compatible con el código modernizado. Si tienes dudas sobre qué versión utilizar, ponte en contacto con tu responsable de entrega de AWS Blu Age.

Versiones

Cada versión se identifica con un patrón **`[Major].[Minor].[Patch]`**. Por ejemplo, con la versión AWS Blu Age Runtime`4.1.0`, la versión principal es 4, la versión secundaria es 1 y la versión del parche es 0.

Tenemos la intención de lanzar nuevas versiones secundarias de AWS Blu Age Runtime mensualmente y nuevas versiones principales cuando se produzcan cambios importantes en el producto o en sus dependencias.

Para obtener más información sobre las nuevas características disponibles en cada versión, consulte the section called “AWS Notas de lanzamiento de Blu Age”.

Versiones preliminares alfa

Cada versión preliminar alfa se identifica con un patrón **`[Major].[Next_Minor].0-alpha.[pre-release]`**. Por ejemplo, en la versión preliminar `4.2.0-alpha.1`, los cambios disponibles en `alpha.1` se publicarán en la siguiente versión secundaria `4.2.0`.

Las versiones preliminares alfa son versiones frecuentes de corta duración que están pensadas y disponibles para su rápida iteración durante los proyectos de modernización. No existe una cadencia de publicación fija para las nuevas versiones preliminares de Alpha, y están disponibles a medida que se desarrollan y prueban.

Para obtener más información sobre el control de versiones, las actualizaciones y el soporte, consulte Ciclo de vida de componentes.

> **Important**
>
> Las versiones preliminares alfa solo deben utilizarse durante la fase del proyecto de modernización y no para cargas de trabajo críticas o de producción.

AWS Opciones de Blu Age Runtime

AWS Blu Age ofrece tres tipos de opciones de tiempo de ejecución para adaptarse a las diferentes etapas del proceso de modernización y a las necesidades operativas. Esta página describe cada opción, sus características, casos de uso y cómo acceder a ellos.

Tiempo de ejecución no gestionado

Con AWS Blu Age Runtime (no administrado), puede implementar su aplicación modernizada por su cuenta Cuenta de AWS , lo que le permite administrar su propia infraestructura. Esta opción ofrece versiones de lanzamiento y preliminares, lo que le brinda la flexibilidad de utilizar todos los componentes técnicos necesarios para ejecutar su aplicación modernizada de la manera que desee. Puede elegir entre versiones estables para entornos de producción o versiones preliminares para fines de prueba y desarrollo.

El tiempo de ejecución no administrado lo implementa y administra el cliente, lo que ofrece un mayor control sobre el entorno de ejecución. Proporciona capacidades de refactorización automatizadas y es adecuado para escenarios de implementación personalizados.

Cuándo se debe usar

El tiempo de ejecución no gestionado es adecuado para entornos de prueba y producción, y resulta especialmente útil cuando se requiere una personalización específica del entorno de ejecución.

Cómo acceder

Para solicitar acceso a los artefactos de tiempo de ejecución no gestionados, consulte Cómo incorporar AWS Blu Age Runtime.

Implementación

AWS Blu Age Runtime (no gestionado) está disponible para su implementación en:

- Amazon EC2
- Amazon ECS en Amazon EC2
- Amazon EKS en Amazon EC2
- Amazon ECS gestionado por AWS Fargate

La implementación en Amazon se EC2 puede realizar directamente en la instancia o mediante una aplicación contenerizada de Docker, que es la forma preferida cuando se utiliza Amazon ECS o Amazon EKS.

Para obtener instrucciones de implementación detalladas, consulte la documentación sobre cómo configurar AWS Blu Age Runtime (no administrado).

Tiempo de ejecución gestionado

Con la administración de AWS Blu Age Runtime, puede implementar su aplicación modernizada en un entorno administrado por AWS que simplifica su experiencia, por lo que no necesita administrar la infraestructura subyacente que ejecuta su aplicación modernizada.

El tiempo de ejecución gestionado incluye una infraestructura gestionada mediante actualizaciones AWS y parches automáticos. Ofrece operaciones y mantenimiento simplificados. Solo las versiones de lanzamiento están disponibles en el tiempo de ejecución gestionado, lo que garantiza la estabilidad y la fiabilidad de sus entornos de producción.

Cuándo se debe usar

El tiempo de ejecución gestionado es ideal para los entornos de producción y es el más adecuado para situaciones en las que se prefiere un enfoque de gestión del tiempo de ejecución sin intervención.

Cómo acceder

Para acceder a la versión gestionada por AWS Blu Age Runtime, simplemente necesita acceder a los siguientes AWS servicios:

- Amazon S3
- AWS Modernización del mainframe

Con los permisos de cuenta adecuados, puede interactuar sin problemas con el entorno de tiempo de ejecución gestionado. Este acceso simplificado garantiza que pueda aprovechar todas las capacidades de Blu Age Runtime y, al mismo tiempo, beneficiarse de los servicios gestionados AWS de Blu Age.

Implementación

Para obtener información sobre cómo configurar y utilizar el tiempo de ejecución gestionado, consulte la documentación sobre cómo configurar el tiempo de ejecución gestionado para AWS Blu Age.

Developer Runtime (BluInsights caja de herramientas)

Se puede acceder al Developer Runtime desde BluInsights Toolbox y está diseñado para las fases de desarrollo y prueba, y se actualiza con frecuencia con las funciones más recientes. Este tiempo de ejecución incluye tanto las versiones preliminares como las versiones preliminares de Alpha, lo que permite a los desarrolladores acceder a versiones estables y a funciones de última generación. Su objetivo principal es respaldar actividades específicas de prueba o desarrollo en un entorno de desarrollo local, normalmente desde un IDE. Es importante destacar que este tiempo de ejecución está limitado a 2 horas de uso, lo que lo hace adecuado para sesiones de desarrollo específicas en lugar de implementaciones a largo plazo o de producción.

Cuándo se debe usar

El Developer Runtime es ideal durante los proyectos iniciales de desarrollo y modernización, así como para realizar iteraciones y pruebas rápidas de aplicaciones modernizadas.

Cómo acceder

El acceso a la BluInsights caja de herramientas se proporciona como parte de su participación en el proyecto AWS Blu Age. El motor de ejecución para desarrolladores está disponible a través de las solicitudes de la caja de herramientas de AWS Blu Age. Una vez aprobado, tendrás acceso a compartimentos de S3 específicos: `s3://toolbox-dev-runtime-<region>`

Estos cubos están disponibles en las regiones us-east-1 y us-east-2. Para usar el depósito disponible, añada la región al nombre del depósito (por ejemplo,). `s3://toolbox-dev-runtime-us-east-1`

Para obtener instrucciones detalladas sobre cómo solicitar acceso y configurar los permisos necesarios, consulte la documentación especial y de desarrolladores de AWS Blu Age Runtimes.

Implementación

Para implementar el motor de ejecución para desarrolladores:

1. Enumere las versiones disponibles mediante AWS CLI:

```
aws s3 ls s3://toolbox-dev-runtime
```

2. Elija una versión específica y enumere su contenido:

```
aws s3 ls s3://toolbox-dev-runtime/[version]/
```

3. Descarga el artefacto de ejecución:

```
aws s3 cp s3://toolbox-dev-runtime/[version]/gapwalk-[version]-dev.tar.gz
```

4. Extraiga y configure el tiempo de ejecución de acuerdo con los requisitos de su proyecto.

Para obtener instrucciones de implementación y pautas de uso más detalladas, consulte la documentación especial y de desarrolladores de AWS Blu Age Runtimes.

> **Note**
>
> Asegúrese de tener los permisos de lectura de S3 necesarios Cuenta de AWS para acceder a estos buckets. Puede encontrar un ejemplo de política de IAM en la documentación especial y de desarrolladores de AWS Blu Age Runtimes.

AWS Notas de lanzamiento de Blu Age

Esta sección contiene las notas de la versión de AWS Blu Age Runtime and Modernization Tools a partir de la versión 3.5.0, primero la más reciente, organizadas por número de versión.

> **Note**
>
> Para ver las notas de la versión anteriores a este documento, ponte en contacto con los servicios de entrega de AWS Blu Age. Para obtener información sobre las últimas características de Blu Insights, consulte las versiones de Blu Insights.

Temas

- Notas de la versión 4.8.0
- Runtime, versión 4.8.0

- Motor de transformación AWS Blu Age 4.8.0
- Notas de la versión 4.7.0
- Runtime, versión 4.7.0
- Motor de transformación AWS Blu Age 4.7.0
- Notas de la versión 4.6.0
- Runtime, versión 4.6.0
- AWS Motor de transformación Blu Age 4.6.0
- Notas de la versión 4.5.0
- Runtime, versión 4.5.0
- AWS Motor de transformación Blu Age 4.5.0
- Notas de la versión 4.4.0
- Runtime, versión 4.4.0
- AWS Motor de transformación Blu Age 4.4.0
- Notas de la versión 4.3.0
- Tiempo de ejecución versión 4.3.0
- Herramientas de modernización versión 4.3.0
- Notas de la versión 4.2.0
- Tiempo de ejecución versión 4.2.0
- Herramientas de modernización versión 4.2.0
- Notas de la versión 4.1.0
- Tiempo de ejecución versión 4.1.0
- Herramientas de modernización versión 4.1.0
- Notas de la versión 4.0.0
- Tiempo de ejecución versión 4.0.0
- Herramientas de modernización versión 4.0.0
- Notas de la versión 3.10.0
- Tiempo de ejecución versión 3.10.0
- Herramientas de modernización versión 3.10.0
- Notas de la versión 3.9.0
- Tiempo de ejecución versión 3.9.0

- Herramientas de modernización versión 3.9.0
- Notas de la versión 3.8.0
- Tiempo de ejecución versión 3.8.0
- Herramientas de modernización versión 3.8.0
- Notas de la versión 3.7.0
- Tiempo de ejecución versión 3.7.0
- Herramientas de modernización versión 3.7.0
- Notas de la versión 3.6.0
- Tiempo de ejecución versión 3.6.0
- Herramientas de modernización versión 3.6.0
- Notas de la versión 3.5.0
- Tiempo de ejecución versión 3.5.0
- Herramientas de modernización versión 3.5.0

Notas de la versión 4.8.0

Fecha de lanzamiento: 23 de abril de 2025

Esta versión de AWS Blu Age Runtime and Transformation Engines presenta actualizaciones clave para mejorar las capacidades y el rendimiento de la transformación de las bases de datos:

- Transformación de bases de datos IBM IMS: se agregó soporte para transformar las bases de datos IMS de IBM en nuestra solución Blu Age personalizada JHDB.
- Función de caché en memoria: se agregó una función de caché en memoria que permite a los usuarios almacenar en caché datos de solo lectura en la memoria, lo que mejora el rendimiento de los programas con uso intensivo de datos.

Probamos esta versión del AWS Blu Age Runtime con la siguiente pila.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 22.11.0

	Npm 10.9.0
	Angular 19.1.3
Capa del servicio	Spring Boot 3.4.2
	Spring Core 6.2.2
	Sesión de primavera 3.4.1
	Spring Statemachine 4.0.0
Capa de persistencia	Motor PostgreSQL 15.10
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Runtime, versión 4.8.0

zOS

Nuevas características

- Se introdujo el soporte para definir encabezados personalizados para transacciones seguras a través de la propiedad YML. `gapwalk-application.security.customAllowedHeaders` Esta propiedad está relacionada con una `gapwalk-application.identity` propiedad con valor. `oauth`
- Se agregó una función para permitir a los clientes reconstruir los metadatos de Blusam en función de los registros reales de la tabla de la base de datos de Blusam
- Conexiones HDB seguras mediante la integración AWS Secrets Manager

Mejoras

- COBOL
  - Se ha mejorado la compatibilidad con la condición AT END en la declaración de archivo READ
  - Se ha mejorado la compatibilidad para mover literales numéricos firmados a campos alfanuméricos sin mover el signo

- Se ha mejorado la compatibilidad con la sentencia UNSTRING para gestionar las conversiones de campos no numéricos a numéricos y conservar los valores bajos cuando no se encuentra el delimitador y ningún valor se mueve a otras variables
- Se ha mejorado la operación de enlace de direcciones para entidades de enlace dinámico
- Se mejoraron las operaciones de concatenación de cadenas al incluir la compatibilidad con los bytes de entrada y salida (SOSI)
- Se ha mejorado la compatibilidad con la instrucción COBOL ACCEPT con la fuente DAY-OF-WEEK
- Se agregó soporte para SYSIN y SYSOUT como valores de NOMBRES ESPECIALES
- Se ha mejorado la compatibilidad con la concatenación de cadenas para gestionar el carácter del feed del formulario

- JCL - IDCAMS
  - Se agregó soporte para eliminar generaciones de GDG utilizando un nombre base sin el parámetro GDG
  - Se ha mejorado la compatibilidad para gestionar el parámetro VOLUME como una variante del parámetro VOLUMES estándar para la sentencia DEFINE CLUSTER
- JCL - ORDENAR
  - Se agregó soporte para SYMNAMES con el comando JOINKEYS
- JCL - IEBGENER
  - Se ha mejorado la compatibilidad con IEBGENER para gestionar contenido SYSIN no válido mediante la introducción de comprobaciones de validación y el recurso automático a ICEGENER, lo que garantiza un funcionamiento continuo incluso con instrucciones de control incorrectas
- JCL - INFUTILB
  - Gestión mejorada de las cláusulas INTO y FROM de descarga
- JCL - DSNUTILB
  - Se agregó soporte para el tipo de datos DOUBLE PRECISION
- JCL - Misc
  - Se mejoró la anulación de las configuraciones de archivos mediante nombres DD en el procedimiento secundario del trabajo
  - DSENTIAUL/INFUTILB: personalización mejorada de los bytes de relleno de SYSREC para la utilidad de descarga

- Se agregó soporte para la opción OUTDD para la redirección de sentencias DISPLAY
- Soporte mejorado para la escritura de archivos secuenciales en bloques fijos. Cuando la LRECL del programa COBOL difiere de la LRECL de la JCL, prevalece la LRECL de la JCL.
- Se agregó compatibilidad con los atributos DCB del catálogo para el archivo SORTOUT, pero solo para conjuntos de datos permanentes
- Soporte mejorado para la transformación gradual condicional de JCL
- Soporte mejorado para ACCEPT con SYSIN y líneas que superen el tamaño objetivo con espacios finales
- Se ha mejorado la compatibilidad con las sentencias SET para gestionar comandos sin incluir el nombre de la variable; el comando definido en el interior incluye un archivo miembro al que debería poder accederse desde la JCL y el comando contiene comentarios en línea
- Se ha añadido soporte para conservar los parámetros «ScriptContext» a nivel de trabajo en los metadatos de los puntos de control de JCL y el contexto del trabajo para su uso al reiniciar

- Blusam
  - Se ha mejorado la inserción de registros en los grandes conjuntos de datos indexados de Blusam al escribir después de cargar un conjunto de datos vacío
  - Se mejoró el rendimiento de los conjuntos de datos de gran tamaño al habilitar un mecanismo de calentamiento e introducir una ventana opcional de captura previa para los registros, así como un almacenamiento local para los índices y las páginas
  - Se ha mejorado la compatibilidad con Export Data Set para gestionar un conjunto de datos más grande con AWS Key Management Service
- SQL
  - Soporte mejorado para el tipo de datos JDBC BLOB
- Integración de colas de mensajes
  - Se agregó soporte para el identificador de usuario en el comando MQ PUT cuando XA está activado
  - Se agregó soporte para conexiones MQ simultáneas cuando XA está activado
- SIMS
  - Se agregó una propiedad YML `jhdb.keepParent` para controlar si las llamadas ISRT de IMS rellenan los campos _parent y _logicalparent de la base de datos cuando se establecen en true
  - Se mejoró la solidez al eliminar la cola de mensajes para el código de transacción no válido

AS400

Nuevas características

- Se agregó una función de caché en memoria que permite a los usuarios almacenar en caché datos de solo lectura en la memoria, lo que mejora el rendimiento de los programas con uso intensivo de datos.

Mejoras

- CL
  - Se ha mejorado la compatibilidad con el parámetro QRYSLT en el comando OPNQRYF para gestionar la palabra clave RANGE, analizar valores cuando las columnas de cadenas tienen valores puramente numéricos y gestionar cadenas vacías
  - Se ha añadido soporte para el comando FTP SENDPASV
  - Se ha mejorado el soporte para la formación de consultas en el comando DSPDBR ejecutado mediante Qcmdexc
  - Se mejoró la compatibilidad con la excepción MONMSG para propagarla correctamente en la pila de ejecución
  - Se mejoró la copia de registros entre tablas con diferentes recuentos de columnas y tipos de datos en el CPYF
  - Prioridad de monitoreo de mensajes mejorada: los mensajes de nivel de comando ahora tienen prioridad correcta sobre los de nivel de procedimiento
  - Se ha mejorado la compatibilidad con el comando CPYF para tener en cuenta la biblioteca definida por IVLIB BLU4
  - Se ha mejorado la compatibilidad con MonMsgs el comando CHKOBJ para gestionar los casos en los que el archivo no existe en la biblioteca
- Screen
  - Se mejoró el manejo de BigDecimal valores con más de 15 dígitos al generar el valor manualmente y luego convertirlo en un número para mostrarlo en lugar de tener que redondear el número
- Misc (Varios)
  - Se ha mejorado la compatibilidad UserSpace con Reemplazar como NO cuando UserSpace ya existe

- Funcionalidad mejorada de BLU4 IVLIB para admitir tanto la recuperación como la escritura de registros en bibliotecas anuladas
- Soporte mejorado para campos de matriz multidimensionales mediante operaciones DIM y OCURRES
- Soporte mejorado para los valores bajos que se transmiten como valores de campo clave en el funcionamiento de la DAO
- Se ha eliminado el conflicto en la instrucción MONMSG para los mensajes que pueden ser gestionados por un monitor local y uno global
- Se ha mejorado la configuración del FileStatus inserto de registros duplicados
- Se han reducido los conflictos de bloqueo del espacio de usuario y el riesgo de errores de creación al utilizar primero el registro para comprobar si el espacio de usuario existe
- Se ha mejorado el soporte para que readPriorEqual () omita registros después de eliminar ()
- Mecanismo mejorado de bloqueo de registros bajo control de compromiso
- Se ha mejorado el manejo de la sentencia RETURN de RPG en las subrutinas
- Se ha mejorado el soporte de la información de usuario en SharedContext

Motor de transformación AWS Blu Age 4.8.0

zOS

Mejoras

- COBOL
  - Generación mejorada del modo de impresora para varios archivos.
  - Se ha mejorado la compatibilidad con SPECIAL-NAMES para gestionar puntos finales y puntos múltiples opcionales.
  - Se ha mejorado el soporte de COPY con sentencias de reemplazo relativopath /REPLACE OFF/ COPY.
  - Se agregó soporte para COL como nombre de campo válido.
  - Se ha mejorado el análisis para gestionar los espacios después de SPECIAL-NAMES, X-COMPUTER y DECLARATIVES.
  - Se agregó soporte para S01 a S05 y C01 a C12 como nombres de párrafos válidos.
  - Se agregó soporte para la función LENG como alias de la función LENGTH.

  - Se ha mejorado el análisis de los libros de texto cuando el valor de la configuración de cobolMaxCode ancho es diferente del que utilizan en COBOL
  - Soporte mejorado para analizar expresiones con comas y operaciones, por ejemplo, MIN (A * B, (3 + 1)/2, 3 + 4).
  - Se agregó soporte para el PROGRAMA INICIAL IS en la cláusula PROGRAM ID.
  - Se ha añadido compatibilidad con la condición de nivel 88 en la sentencia SEARCH ALL.
  - Se ha añadido soporte para la sección de informes.
- SQL
  - Se agregó soporte para la transformación DDL desde la base de datos IMS de IBM.

AS400

Mejoras

- CL
  - Soporte mejorado de CL al establecer la transformación a JAVA como soporte predeterminado
  - Se mejoró la transformación a Groovy Script añadiendo el uso de un metamodelo
  - Se agregaron las palabras clave CONTINUE y FOR a la lista de palabras clave reservadas
- DDS
  - Se ha mejorado el soporte de los campos DSPF definidos en relación con el anterior
  - Se ha mejorado la compatibilidad con el campo de caracteres DDS con la palabra clave VARLEN
  - Se mejoró la transformación/generación de la entidad LF DAOs al generar la configuración explícita de los campos renombrados en los métodos de conversión y actualización de Vo. JPAFrom
- JUEGO DE ROL
  - Se ha mejorado la compatibilidad con el parámetro de palabra clave PREFIX para gestionar los casos en los que los valores no tienen comillas
  - Se ha mejorado el manejo de las llamadas a métodos anidados en índices matriciales
- COBOL400
  - Se mejoró la generación de entidades de registro importadas desde un DSPF con la directiva COPY DDS-ALL-FORMATS
- SQL

- Se mejoró la transformación de las cláusulas WHERE de SQL con condiciones OR no incluidas.

Notas de la versión 4.7.0

Fecha de lanzamiento: 12 de marzo de 2025

Esta versión de AWS Blu Age Runtime y Transformation Engines introduce actualizaciones clave en las dependencias principales: AWS Blu Age Runtime se ha actualizado para utilizar la última versión 3.4.2 de Spring Boot y las aplicaciones frontend modernizadas ahora utilizan la última versión 19 de Angular.

Hemos probado esta versión de AWS Blu Age Runtime con la siguiente pila. Es posible que otras versiones también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 22.11.0
	Npm 10.9.0
	Angular 19.1.3
Capa del servicio	Spring Boot 3.4.2
	Spring Core 6.2.2
	Sesión de primavera 3.4.1
Capa de persistencia	Spring Statemachine 4.0.0
	Motor de PostgreSQL 14
Oracle 21c	
Servidor de aplicaciones	Apache Tomcat 10.1.17

Runtime, versión 4.7.0

zOS

Nuevas características

- Se agregó una nueva propiedad YML `dataSimplifier.doubleFloatingType` para definir qué formato usar al codificar y decodificar los números de punto flotante. Los valores permitidos son `IEEE_754` (por defecto)/`HFP`(para el formato hexadecimal de punto flotante)
- Se ha añadido la compatibilidad con la posición P de escala decimal en la imagen numérica

Mejoras

- COBOL
  - Se ha mejorado la compatibilidad con la sentencia INSPECT con las cláusulas AFTER INITIAL y BEFORE INITIAL
  - Compatibilidad mejorada con tipos editados numéricos con conversión de bytes optimizada, lo que mejora la eficiencia general del manejo de datos
  - Se agregó soporte para FUNCTION MIN & MAX con comparación alfanumérica
  - Se ha mejorado la compatibilidad con FUNCTION INTEGER para valores negativos
- PL/I
  - Compatibilidad mejorada de la sentencia PUT con el conjunto de datos SYSPRINT de FBA/VBA
- JCL - ORDENAR
  - Se agregó compatibilidad con la opción SQZ de OUTREC con los subparámetros SHIFT, LEAD, TRAIL, MID, PREBLANK y PAIR
  - Se mejoró la catalogación de los atributos DCB para el archivo SORTOUT
- JCL - DSNUTILB
  - Soporte mejorado para el tipo de datos DECIMAL EXTERNAL
  - Se ha mejorado la compatibilidad con la condición NULLIF relacionada con los valores o la posición de otra columna
  - Soporte mejorado para las columnas BYTEA
- JCL - INFUTILB
  - Se ha mejorado el soporte para las palabras clave CURRENT [] DATE/TIME/TIMESTAMP en la operación UNLOAD

- JCL: compatibilidad mejorada con el código de retorno cualificado de las llamadas PROCs
- Procesamiento de datos
  - Se ha mejorado la lógica de comparación numérica al procesar campos con valores en blanco, redefinidos de alfanuméricos a numéricos, lo que garantiza una evaluación correcta con respecto a cero
- Blusam
  - Se ha mejorado la compatibilidad con el comando START para gestionar búsquedas parciales de claves mediante segmentos de la clave principal, lo que proporciona capacidades de recuperación de registros más flexibles
- SQL
  - Se ha añadido compatibilidad con el parámetro de entrada introducido en la cláusula ON
  - Se agregó soporte para la función REPEAT
  - Se ha mejorado la compatibilidad con la sentencia UPDATE con una asignación que incluye el nombre de la columna
  - Se ha mejorado el manejo de SQLCODE y SQLSTATE en las búsquedas de cursores de varias filas
  - Se agregó soporte para la función XML para el destino de la base de datos DB2 DB2
- Integración de colas de mensajes
  - Gestión de errores mejorada en las operaciones GET de MQ, que proporciona una indicación de estado más clara MQCC_FAILED para cuerpos de mensajes vacíos
  - Soporte mejorado para las operaciones PUT de MQ mediante el manejo del MQRFH2 encabezado en el contexto de la transacción XA
  - Soporte mejorado para los secretos de AWS Blu Age Runtime para gestionar propiedades adicionales de JMS MQ
  - Capacidad mejorada de configuración del conjunto de conexiones JMS MQ
- Misc (Varios)
  - Integración del sistema: se agregó compatibilidad con el entorno de esquemas en las llamadas al programa DFSRRC00
  - Se agregó compatibilidad para las transacciones IMS cuando Blusam está deshabilitado
  - Se ha mejorado la compatibilidad con la end-of-file condición de un archivo secuencial para ajustarla a la regla antigua «El EOF se considera una lectura incorrecta»

AS400

Nuevas características

- Se agregó compatibilidad con el comando RUNSQLSTM para ejecutar consultas SQL almacenadas en un archivo fuente externo

Mejoras

- CL
  - Se agregó un mecanismo de bloqueo de archivos mediante los comandos ALCOBJ/DLCOBJ
  - Se mejoró la funcionalidad del comando CPYF para mantener la integridad de la secuencia de registros durante la copia de archivos
  - Se ha mejorado el comando CPYF para admitir la salida como un archivo plano
  - Gestión de parámetros mejorada para ajustar automáticamente la longitud de los datos durante las llamadas al programa
  - Soporte de parámetros mejorado para gestionar la referencia de rango
  - Se agregó soporte para MONMSG CPF21 05 en el comando DLTDTAARA
  - Se agregó compatibilidad con la biblioteca QTEMP para el comando CHKOBJ en DATAAREA
  - Soporte mejorado para el comando ADDLFM, lo que mejora las capacidades de administración de miembros de archivos lógicos
  - Se agregó soporte para nombres de áreas de datos genéricos en el comando DLTDTAARA
  - Se agregó compatibilidad con SBMJOB para gestionar los conmutadores de paso mediante el parámetro SWS
  - Se mejoró la lógica de escritura del área de datos al colocar las llamadas de lectura y escritura en un bloque sincronizado
  - Gestión mejorada de la simultaneidad del área de datos para evitar escrituras simultáneas mediante la implementación de un mecanismo de bloqueo granular, lo que elimina los errores durante las operaciones paralelas
  - Se ha añadido compatibilidad con la palabra clave *ALL en el parámetro DTAARA del comando RTVDTAARA
- Screen
  - Se ha mejorado el color de un atributo blanco en la interfaz
- JUEGO DE ROL

  - Soporte mejorado para la operación TESTB para manejar mejor las diferentes codificaciones
  - Soporte mejorado para el funcionamiento de EDITC con los códigos de edición «Y» y «P»
  - Soporte mejorado para EDITCODE para los códigos K, Q y Z
  - Implemente la compatibilidad con la compatibilidad con las ExtProc palabras clave Prototype en el procedimiento local con parámetros
  - Se gestionaron los parámetros transferidos por valor para procedimientos externos
- COBOL
  - Se ha mejorado la sentencia SORT para gestionar la biblioteca de destino
  - Se truncó una partición al abrir la DAO asociada en el modo OUTPUT
  - Se agregó soporte para la palabra clave PREFIX
- DataQueue
  - Se agregaron opciones de configuración de almacén de claves y almacenes de confianza para los datos de RabbitMQ y las colas de mensajes híbridas, lo que permite una mensajería segura en las implementaciones de producción
  - Se optimizó el rendimiento de reinicio de los servidores al eliminar la redundante comprobación de la existencia de colas de datos
  - Arquitectura simplificada mediante la eliminación de componentes DataQueueRegistry DataQueueConnectionProvider
  - Eliminación simplificada de las colas de datos mediante la gestión directa sin validación del registro
  - Se mejoró el proceso de creación de colas al garantizar la creación y el enlace adecuados de los intercambios queue-to-exchange
- Misc (Varios)
  - Se agregó la validación del tamaño de las solicitudes de transacción para evitar la sobrecarga del sistema relacionada con las solicitudes de rango excesivo
  - Gestión mejorada de los bloqueos en la limpieza de los recursos de trabajo
  - Se ha mejorado el mecanismo de bloqueo de la DAO para evitar que se bloquee un registro cuando el archivo se abre en el modo INPUT
  - Rendimiento optimizado para el mapeo de entidades dinámicas
  - Manejo mejorado del contexto de persistencia de JPA para mejorar el rendimiento en las operaciones de DAO

  - Se mejoró el soporte del área de datos al agregar columnas de marca de tiempo para gestionar los metadatos de los registros
  - Se mejoró la confiabilidad del área de datos mediante la implementación de operaciones de lectura/escritura sincronizadas para evitar problemas de acceso simultáneo
  - Se ha mejorado la compatibilidad con #moveDate para facilitar las conversiones DateHelper date-to-timestamp
- DDS
  - Se ha mejorado la compatibilidad con las condiciones de selección/omisión, en combinación con operaciones sintéticas como SST o CONCAT
- SQL
  - Se optimizó el SQLExecutor Builder para gestionar la recuperación parcial de registros de forma más eficiente, especialmente cuando se recuperan varias filas con límites de recuperación superiores a uno

Motor de transformación AWS Blu Age 4.7.0

zOS

Mejoras

- COBOL
  - Se agregó soporte para la declaración CODE-SET
  - Se agregó soporte para la opción NOT END OF PAGE de la declaración WRITE
  - Se agregó soporte para una nueva sintaxis en ALTER
  - Se agregó la compatibilidad con el reemplazo de varias líneas para la instrucción COPY REPLACING
  - Se ha mejorado el análisis de la palabra clave CHARACTERS implícita en una definición de cláusula diferente
  - Se ha mejorado la compatibilidad con las claves alternativas incluidas en la sección de entrada y salida cuando los nombres de las claves están duplicados pero tienen padres diferentes
  - Se ha mejorado la compatibilidad con el estado IF al comparar campos grandes con espacios
  - Se ha mejorado la resolución estática de referencia de los archivos lógicos al hacer que no distinga entre mayúsculas y minúsculas

- Transformación de código mejorada para generar automáticamente campos implícitos de LINAGE-COUNTER para cada entrada de FD que contenga una cláusula LINAGE
- Se ha mejorado la compatibilidad con la definición del clúster IDCAMS con un nombre entre comillas simples
- Se ha mejorado la compatibilidad con la sentencia PERFORM para el formato Basic, con el formato de frase TIMES y el formato que incluye la calificación de la sección
- Se ha habilitado la transformación de la sentencia SORT con varios archivos GIVING y se ha añadido una excepción para hacer un seguimiento de este caso en el momento de la ejecución
- Se ha mejorado la transformación de RENAMES cuando se especifica en un grupo mediante la cláusula REDEFINES

- PL/1
  - Se agregó soporte para la opción DATA o LIST para PUT STRING
  - Se ha añadido compatibilidad con el programa Multiple Entry

AS400

Mejoras

- CL
  - Se mejoró el análisis de los parámetros de QCMDEXC para poder ejecutar ALCOBJ/DLCOBJ a través de QCMDEXC
  - Se ElseIf agregó soporte para la declaración sobre CL
  - Se agregó soporte para la función integrada %SWITCH para configurar los cambios de trabajo.
- DDS
  - Se ha mejorado la compatibilidad con el registro DSPF descrito por el programa para gestionar el registro ficticio en las especificaciones de entrada
  - Se han mejorado las condiciones de la cláusula WHERE para la generación de DAO en el caso de un archivo SQLDDS de CREATE INDEX
- JUEGO DE ROL
  - Se agregó soporte para la inicialización de ZEROES INZ (*ZEROS) en enteros sin signo
  - Se ha mejorado la compatibilidad con los comandos, EXCEPTO con un tipo de archivo de estación de trabajo y un índice SQL
  - Soporte mejorado del tipo de datos interno RPG para campos numéricos definidos externamente

- Se ha mejorado la compatibilidad con la función %ERROR integrada para obtener el estado actual del error tras una operación en un archivo
- Se ha mejorado la compatibilidad con la palabra clave COMMIT para gestionar archivos lógicos multiformato
- Procesamiento SELECT/OMIT mejorado para archivos lógicos multiformato
- Se mejoró el manejo de MOVEA para configurar una matriz con espacios en blanco y mover matrices fijas de paquetes, zonas y binarios a otra matriz del mismo tipo.
- Manejo de matrices mejorado para Z-ADD *ZEROS y MOVEA *ZEROS
- Se han mejorado las operaciones MOVE/MOVEL al cambiar de combinaciones numéricas y de caracteres
- Implemente la compatibilidad con la compatibilidad con las ExtProc palabras clave RPGLE Prototype en procedimientos locales con parámetros
- Manejo mejorado de la constante figurativa *HIVAL en las instrucciones de asignación
- Se ha mejorado la compatibilidad con la palabra clave PREFIX para gestionar el reemplazo de caracteres

- COBOL400
  - Soporte mejorado para las operaciones de REESCRITURA y ACTUALIZACIÓN en el modo de acceso dinámico
  - Se ha añadido compatibilidad con la cláusula ACCEPT Statement FOR en el CBLLE
  - Se ha mejorado la compatibilidad con la función %STATUS integrada para obtener el estado actual de un archivo después de una operación
- Misc (Varios)
  - Se ha mejorado la transformación de los programas que definen estructuras de datos que comparten el mismo nombre que uno de los campos del PF/LF especificado
  - Se mejoró la generación de acceso a matrices en variables kanji

Notas de la versión 4.6.0

Fecha de lanzamiento: 24 de enero de 2025

Hemos probado esta versión del AWS Blu Age Runtime con la siguiente pila. Es posible que otras versiones también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 22.11.0
	Npm 10.9.0
	Angular 18
Capa del servicio	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Runtime, versión 4.6.0

zOS

Mejoras

- COBOL
  - `WRITE ADVANCING`Capacidades mejoradas con una precisión mejorada para la escritura secuencial de líneas de archivos, la compatibilidad con múltiples contextos (`BEFORE>` y usos implícitos) y la implementación completa de sentencias `AFTER PAGE`
  - Soporte mejorado `FILLER` para los casos en los que una tabla anidada FILLER se usa como un grupo con una tabla como elemento secundario
  - Se ha mejorado el acceso a los hijos de padres ambiguos dentro de un segmento
  - Se agregó soporte para el tipo de edición numérica con picture='-----'
  - Se ha mejorado el manejo de la visualización de datos de tipo binario
- PL/I

  - Se ha mejorado la conversión de valores literales binarios en sentencias de asignación
- JCL — ORDENAR
  - Soporte mejorado para `OVERLAY` parámetros consecutivos en la misma declaración `OUTFIL`
- JCL — DSNUTILB
  - Mecanismos de carga optimizados, lo que resulta en tiempos de recuperación de datos un 25% más rápidos
  - Soporte mejorado para las transacciones de XA para fuentes de datos empresariales externas
- JCL — INÚTIL
  - UNLOAD: se agregó soporte para el tipo de datos FLOAT8
- JCL — IDCAMS
  - Manejo optimizado de los códigos de retorno de los comandos `IDCAMS`
  - Se agregó soporte para eliminar todas las generaciones de GDG en función del nombre base de GDG
  - Se agregó soporte para la eliminación de archivos sin parámetros `NONVSAM`
- JCL — Misc
  - Gestión mejorada de metadatos de Batch Restart para mejorar la administración del estado del flujo de trabajo durante el modo de reinicio
- Blusam
  - Se agregó soporte de TTL para la caché de Blusam en las implementaciones de Ehcache y Redis
  - Se ha mejorado la compatibilidad con el `DEPENDING ON` campo en la descripción del archivo COBOL para el archivo KSDS de Blusam `FD`
  - Seguridad de subprocesos mejorada en las operaciones de lectura de Redis Blusam para la ejecución simultánea de varios trabajos
  - Se mejoró la creación del esquema de Blusam para una mayor solidez en lo que respecta a los privilegios de los usuarios de la base de datos
  - Se ha mejorado el relleno a la derecha en el conjunto de datos de entrada concatenado en bloques variables `READ`
- BAC
  - Se agregó soporte para la creación de conjuntos de datos en modo multiesquema, incluida una nueva columna de «Esquema» para indicar la asociación de esquemas para cada conjunto de datos

- MFS
  - Se mejoró la propagación de la información del usuario desde el front-end al contexto compartido, lo que garantiza una propagación adecuada al contexto JHDB
  - Se agregó soporte para el encabezado de información IMS de IBM MQ en las transacciones de XA
- SQL
  - Se ha mejorado `SQLCODE` el manejo del valor 305 durante la búsqueda del cursor cuando todos los valores de las columnas son NULOS
  - Se agregó soporte para la `IN` cláusula que incluye un `OCCURS` parámetro para las condiciones `WHERE`
  - Se agregó soporte para declaraciones de tablas `DECLARE GLOBAL` temporales
  - Soporte ampliado de DB2 SQL para el formato de marca de tiempo DB2 específico de medianoche y 24 horas mediante conversiones específicas en el momento de la ejecución según el motor de base de datos específico
- Misc (Varios)
  - Se ha mejorado el juego de caracteres IBM93 0 para permitir que los caracteres Unicode U +2014 y U+2015 se correspondan con x'44x4A' en EBCDIC
  - TDQUEUE: implementación de SQS refactorizada para admitir subprocesos múltiples
  - Resolución de nombres de conjuntos de datos GDG mejorada para permitir al cliente archivar archivos con el mismo prefijo GDG (por ejemplo, es el archivo actual y es un archivo archivado) `A.B.C.G0002V00` `A.B.C.G0001V00.1236`
  - Mejorado `SQLConverter::toPgmDate/Time/Timestamp` para alinear el cálculo de la fecha de acuerdo con el formato anterior

AS400

Nuevas características

- Se agregó soporte para tablas AS4 00 creadas dinámicamente para archivos planos y entidades duplicadas, lo que permite el acceso a las tablas creadas mediante comandos CL como CRTPF, CRTDUPOBJ y CPYF
- Se agregó un servicio para admitir la lista de bibliotecas a través de un registro que gestiona la biblioteca predeterminada para cada tabla

Mejoras

- CL
  - CLRPFM: se ha mejorado el manejo del miembro cuando se invoca el comando de la biblioteca QTEMP
  - SMBJOB: soporte mejorado de parámetros para manejar argumentos construidos dinámicamente `PARM`
  - CPYFRMIMPF: se agregó soporte para los parámetros, y `TIMFMT` `ERRRCDFILE` `ERRRCDOPT`
  - CPYFRMIMPF: compatibilidad mejorada con valores alfanuméricos de bases de datos que contienen comillas simples
  - CPYF: se ha perfeccionado la construcción de consultas de comandos para archivos de varios miembros con `FROM` `TOMBR(*ALL)`
  - CPYF: soporte mejorado para gestionar los parámetros de `FMTOPT` `MAP DROP`
  - CPYTOIMPF: soporte mejorado del parámetro para manejar la tabla `FROMFILE` `MEMBER`
  - RTVUSRPRF: se agregó soporte para el parámetro `RTNUSRPRF`
  - DSPDBR: Modifique el comando para adaptarlo al comportamiento anterior esperado al imprimir información sobre las vistas que existen en una tabla, así como sobre la biblioteca y el miembro del que forman parte
  - DSPFD: compatibilidad mejorada con los parámetros `FILE`
  - DSPFD: soporte mejorado de la `TYPE MBR` salida de parámetros para incluir valores adicionales: mbfile, mblib, mbfcdt, mfccn
- Screen
  - Se ha mejorado la prioridad de posición del cursor para `DSPATR(PC)`
  - Se mejoró la validación de los campos de registro de los subarchivos al ignorar la validación inicial de los campos «protegidos»
  - Se ha mejorado el soporte para inicializar registros en estaciones de trabajo con varios campos de matriz que comparten nombres de componentes
  - Soporte mejorado para los indicadores de respuesta en `DSPF` palabras clave (`SFLMSGSFLMSGID`, `CHANGE` y teclas de comando)
- JUEGO DE ROL
  - Soporte mejorado del ciclo de programación para un mejor manejo de los campos leídos de los archivos primarios/secundarios

- Se agregó soporte para Split Control Field para la lectura de archivos primarios/secundarios
- Método %SUBST integrado mejorado para gestionar campos de doble byte en las declaraciones de comparación
- Soporte mejorado del indicador ZERO para el funcionamiento del MVR

- DDS
  - Se ha añadido la compatibilidad con archivos lógicos multiformato con formato de registro que hacen referencia al mismo registro físico
- DataQueue
  - Se ha mejorado la gestión de las interrupciones de los trabajos en espera de recibir mensajes de la cola de datos, ya que no hay problemas de consumo durante las interrupciones
  - Se migró de RabbitMQ a Spring-AMQP para mejorar la gestión de los canales y el escalado de subprocesos
- Misc (Varios)
  - SQLExecutorBuilder mejorado para admitir consultas con varios espacios en blanco y corchetes abiertos sin espacios iniciales
  - Se ha mejorado la compatibilidad con el DAO para gestionar correctamente la posición del cursor al cambiar la dirección de lectura
  - Se ha perfeccionado la inicialización de las claves tras las operaciones de recuperación y eliminación para garantizar la correcta eliminación de los registros relacionados antes de insertar los registros actualizados
  - Código generado por el mapeador DAO optimizado para mejorar el rendimiento de la ejecución a tiempo

AWS Motor de transformación Blu Age 4.6.0

zOS

Mejoras

- COBOL
  - Análisis mejorado de la RESERVE cláusula con un literal opcional AREA/AREAS
  - Compatibilidad mejorada con COBOL con DATA DIVISION declaración opcional, lo que permite simplificar los casos de prueba

  - Se mejoró el párrafo de nombres especiales al añadir soporte para `ALPHABETSYMBOLIC`, y `CLASS` cláusulas, modificadores y variables `FORMFEED`
  - Se agregó soporte para `SYSIN` un nombre mnemotécnico en las declaraciones `ACCEPT`
  - Se ha mejorado el soporte de `PICTURE` cláusulas para los símbolos «$», «0", «CR» y «DB» en `PIC` los cálculos de tamaño lógico
  - Transformación de `USE` sentencias mejorada para varios escenarios de archivos
  - Transformación mejorada de `ALTER` declaraciones para múltiples modificaciones
  - Se agregó soporte para las constantes `ZERO` `HIGH-VALUE` `LOW-VALUES` figurativas en la cláusula `delimited by`
- SQL
  - Se mejoró la transformación del valor predeterminado para que el destino de PostgreSQL maneje las comillas alrededor del valor predeterminado `CURRENT_TIMESTAMP`
  - `WITH CHECK OPTION`Cláusula Handle de las vistas de SQL

AS400

Mejoras

- DDS
  - Soporte mejorado de archivos lógicos multiformato que hacen referencia al mismo registro físico varias veces
- JUEGO DE ROL
  - Mejorado `MOVE` y `MOVEL` operaciones para manejar mejor los ceros rellenados
  - Gestión mejorada de las llamadas a funciones anidadas en las evaluaciones y condiciones
- COBOL400
  - Se agregó soporte para transformar la `IN` palabra clave en `SELECT` las declaraciones
  - Se ha mejorado la compatibilidad con los puntos que faltan en las entradas de descripción de los datos, adaptándose a la versión más reciente de COBOL, en la que se asumen los puntos cuando faltan
  - Posicionamiento mejorado del cursor en las operaciones `REWRITE`
  - Soporte mejorado para que la `START` declaración bloquee el registro en la posición actual del archivo

  - Soporte mejorado para la directiva del compilador `COPY DDS` para generar toda la estructura de datos de entrada/salida
- Misc (Varios)
  - StateMachines - Transformación mejorada para mejorar la declaración de estados compuestos de acuerdo con el paradigma stateless4j
  - Saneamiento mejorado para los archivos LF que contienen caracteres especiales
  - Se ha mejorado el soporte de valores figurativos `*ALL` con valores hexadecimales
  - Soporte `MOVE` operativo mejorado para la conversión implícita de tipos numéricos a caracteres
  - Se optimizó la generación de granos de informes para ordenarlos por el nombre de impresora asociado, lo que evita nombres duplicados o conflictivos
  - Se ha mejorado la compatibilidad con las palabras clave y se ha `EXTFILE` combinado con `USROPN` el manejo del valor literal y el formato `libname/filename`

Notas de la versión 4.5.0

Fecha de lanzamiento: 20 de diciembre de 2024

Esta versión de los motores AWS Blu Age Runtime y AWS Blu Age Transformation incluye las siguientes funciones clave.

- Compatibilidad con JCL: ahora es posible generar y ejecutar scripts JCL sobre la marcha en el contexto del tiempo de ejecución. Esta función añade flexibilidad y automatización al procesamiento de trabajos por lotes. Hemos actualizado la compatibilidad con las utilidades de JCL durante el tiempo de ejecución, con una serie de mejoras en SORT, ICETOOL, INFUTILB e IDCAMS (consulte los detalles en las siguientes secciones). Estas mejoras ofrecen capacidades de procesamiento de datos más sólidas y eficientes.
- Compatibilidad con directorios de enlace y grupos de activación para aplicaciones modernizadas del AS/400: los directorios de enlace mejoran la organización del sistema al gestionar las referencias de procedimientos exportadas, mientras que los grupos de activación optimizan la gestión del contexto de ejecución. Estas funciones mejoran la precisión y la fiabilidad, la sólida administración de los recursos y optimizan las interacciones del sistema. El resultado es un sistema más resistente, organizado y eficiente para aplicaciones AS4 00 modernizadas.
- Actualizaciones de dependencias: — Actualización de todos los marcos de interfaz (BAC/JAC y aplicaciones modernizadas) a las versiones de soporte a largo plazo (LTS). La actualización de Angular de la versión 17 a la versión 18 introduce un nuevo modelo de reactividad y una

gestión del estado simplificada, lo que reduce la complejidad y mejora el mantenimiento de las aplicaciones para los desarrolladores. Node.JS también se ha actualizado de la v20 a la v22.

Hemos probado esta versión de AWS Blu Age Runtime con la siguiente pila. Es posible que otras versiones también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 22.11.0
	Npm 10.9.0
	Angular 18
Capa del servicio	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Runtime, versión 4.5.0

zOS

Nuevas características

- JCL: se agregó la capacidad de invocar un trabajo por lotes desde programas en línea. Añadimos un servicio para gestionar el script JCL almacenado en un servidor específico TDQueue cuando un programa modernizado lo genera sobre la marcha. Este servicio permite reconstruir el mensaje de JCL, refactorizar este mensaje en un script genial y ejecutar este script genial.
- ADABAS: se agregó soporte para el programa ADABAS. Con este soporte, el motor de ejecución emula los comandos de ADABAS para el acceso a la base de datos (disponible solo para Oracle).

Mejoras

- COBOL
  - Soporte mejorado de la declaración DISPLAY al aprovechar la opción NO ADVANCED
  - Mayor precisión en la gestión de los signos de cambio, lo que permite al usuario beneficiarse de una estructura COBOL transformada más precisa
  - Soporte mejorado para la asignación de valores al mover un campo sin firmar a un campo firmado y viceversa
  - Se ha mejorado la compatibilidad con el tamaño de bloque de los archivos GDG y los archivos concatenados
- CICS
  - Se agregó soporte para OpenStatus y para los conjuntos de datos de Blusam EnableStatus
  - Se agregó soporte para el comando `SET DATASET`
- JCL — ORDENAR
  - Manejo mejorado del tamaño del registro del conjunto de datos
  - Se ha mejorado la compatibilidad con la `OUTFIL` sentencia para generar archivos de salida que contengan únicamente los registros de los archivos de entrada según los valores especificados en STARTREC y en las opciones `ENDREC`
  - Mejora del apoyo a las declaraciones `OVERLAY`
  - Se ha mejorado el soporte de la `OUTREC` declaración para gestionar una variante de la `EDIT` opción. Ahora apoyamos, `EDIT(...)` además de `EDIT=(...)`
  - Se agregó soporte para el patrón (p, m, f, OPERATOR, p2, m2, f2) en las operaciones aritméticas
  - Puede utilizar la cláusula de `DUMMY` archivo del `SORT` programa de un JCL para gestionar los archivos de entrada vacíos y beneficiarse de la generación de archivos vacíos
- JCL — ICETOOL
  - Mayor apoyo a la `SORT FIELDS=COPY` declaración a través del programa `SORT`
- JCL — INFUTILB
  - Se ha mejorado la compatibilidad con el cálculo del tamaño de los registros si no se especifica en la JCL y la propiedad DFSIGDCB está deshabilitada
  - Se mejoró la función UNLOAD con la cláusula INTO para DECIMAL al actualizar la precisión y la escala de acuerdo con los campos de la cláusula INTO
  - Método de formato mejorado en VarcharFormatter

  - Soporte mejorado con una nueva opción configurable que permite a los usuarios controlar cómo se manejan los campos VARCHAR durante la descarga de datos en lo que respecta al comportamiento de relleno, lo que garantiza flexibilidad y precisión en los procesos de extracción de datos.
- JCL — IDCAMS
  - Se ha mejorado la eliminación de archivos con un sufijo comodín y un nombre definido directamente entre paréntesis o entre comillas simples
  - Precisión mejorada para aprovechar el código de retorno de MAXCC
- JCL — IKJEFT01: se agregó un indicador de función `systin.encoding` (predeterminado `=ASCII`) para admitir la codificación específica del conjunto de datos de archivos SYSTSIN
- JCL: se ha mejorado la compatibilidad con la propiedad BDW para un archivo de salida generado en un paso de JCL y, en los pasos siguientes, se utiliza el mismo sistema de archivos como entrada y `DISP=PASS`
- MF
  - Soporte mejorado para el encabezado de 2 bytes para el archivo Record Sequential
  - Gestión mejorada de los códigos de retorno para el comando DELETE
  - Línea de avance de escritura mejorada para grabar un archivo secuencial
- Redis
  - Se ha mejorado la inicialización de la plantilla de Redis para los puntos de control de JCL y los Jics TSQueues
  - Mejoras en la accesibilidad y legibilidad de la información sobre el bloqueo de registros del conjunto de datos de Redis
- SQL
  - Se ha mejorado el análisis de la clave foránea con la cláusula REFERENCES
  - Se proporcionó una función de almacenamiento en caché ampliable para almacenar los tipos de gráficos originales heredados en la base de datos, lo que mejoró la trazabilidad de los datos y facilitó el cálculo gráfico
  - Soporte de análisis mejorado del patrón CASE WHEN de consultas SQL en todas las utilidades de tiempo de ejecución
  - Se ha mejorado la función integrada de SQL Postgres Blu Age, gwdecimal, en la que se basa el tiempo de ejecución para adaptarse a la función integrada DECIMAL. DB2
- Misc (Varios)

  - Soporte mejorado para usar el operando SIGN NumericEditedType
  - Generación mejorada de la configuración de la fuente de datos principal SpringBootLauncher en la aplicación modernizada
  - Flexibilidad mejorada para separar los registros de las aplicaciones de la ruta relacionada con el trabajo solicitado.
  - Se ha mejorado la compatibilidad con el valor en blanco al comparar campos de NumberUtils
- ARCHIVO: compatibilidad mejorada con conjuntos de datos de bloques variables en los archivos subyacentes
- MQ: gestión de conexiones MQ mejorada para una alta disponibilidad y preparada para entornos
- Se mejoró la compatibilidad con MQ Queue al agregar soporte para clientes que no son JMS para mejorar la codificación y el manejo del conjunto de caracteres
- Se ha mejorado la compatibilidad con los caracteres de control ANSI para el archivo Ebcdic

AS400

Nuevas características

- Se agregó soporte para los datos exportados dentro de los programas enlazados
- Se agregó soporte específico de ILE para la división por cero

Mejoras

- COBOL400
  - Compatibilidad mejorada con el EOF en el estado de los archivos
  - Aumente el soporte de precisión de la sentencia START de Cobol para admitir la palabra clave EQUAL en la cláusula KEY IS
- CL
  - Se agregó soporte para el comando UPDENVPARM
  - CRTPF: se agregó soporte para tablas a las que se accede con una partición
  - RCVF: soporte mejorado de archivos lógicos con anulación
  - FTP: soporte mejorado de archivos de E/S lógicos con OVRDBF y registro de SALIDA mejorado, además de soporte adicional para archivos de E/S en el directorio de trabajo
  - CPYFRMIMPF: se agregó soporte para los parámetros, `ERRRCDFILE` `TIMFMT` `ERRRCDOPT`

- CPYF: creación mejorada de particiones QTEMP
- CPYF: se agregó un mensaje de monitoreo cuando el archivo *FROM está vacío
- OVRPRTF: se agregó soporte para nuevos parámetros:`PAGESIZE`,,,,,,`OUTQ`, `DEV` `LIP` `CPI` `OVRFLOW` `LVLCHK` `FORMTYPE` `HOLD`
- Mayor precisión al utilizar el `FMTOPT` parámetro con `DROP` las opciones `MAP` y opciones del `CPYF` comando para permitir copiar datos de un archivo de origen con columnas adicionales a un archivo de destino
- Mayor precisión en la gestión del mapeo de los patrones comodín de las rutas del sistema de archivos en el comando `RMVLNK`
- El comando `RMVM` (Eliminar máquina virtual) se ha mejorado para gestionar las tablas de `DROP` particiones, lo que garantiza una limpieza completa de los recursos relacionados.
- OPNQRYF: compatibilidad mejorada con el parámetro *FILE para el comando
- Se implementó el manejo del CPF0000 para abarcar todos los mensajes CPFx
- CHGDTAARA: se agregó soporte para la palabra clave*ALL para cambiar toda el área de datos

- Screen
  - Se ha mejorado el cursor tables/subfile displaying by increasing accuracy for scrolling and position/priority
  - `CHECK(RB)`Funcionalidad mejorada `CHECK(RZ)` para campos no numéricos y no firmados
  - Se ha mejorado la compatibilidad de la función de pantalla de ayuda con las palabras clave `HLPARA`
- JUEGO DE ROL
  - Soporte mejorado de sistemas integrados `%SubDt`
  - Soporte mejorado para el procedimiento utilizando una estructura de datos local que se describe externamente
  - Se agregó soporte para el parámetro de código de error opcional`QMHSNDPM`, y `QMHRMVPM` `QMHRCVPM`
  - Compatibilidad mejorada con el método `%SUBST` integrado para gestionar mejor los campos de doble byte.
  - Se agregó compatibilidad con la versión integrada %TLOOKUP y sus variantes (%TLOOKUPGE, %TLOOKUPGT, %TLOOKUPLE, %TLOOKUPLT)
- Área de datos
  - Soporte mejorado para la operación OUT cuando el factor 1 está en blanco

  - Lecturas simultáneas mejoradas en la misma área de datos
  - Se agregó una variable de configuración `blu4iv.dtaara.library.disable` para deshabilitar las bibliotecas del área de datos
  - Soporte ampliado para aprovechar las bibliotecas designadas mediante operaciones de área de datos, lo que permite al usuario estructurar la ubicación del área de datos como desee.
- DataQueue
  - Uso mejorado del canal RabbitMQ
  - Se ha mejorado RabbitMQ Consumer para que solo intente cancelar al consumidor una vez
  - Se mejoró la recuperación de la cola de datos de RabbitMQ al intentar BasicGet solo cuando el tiempo de espera es 0
- Misc (Varios)
  - Espacio de usuario: comportamiento mejorado cuando varios trabajos intentan recuperar el mismo espacio de usuario simultáneamente
  - Se ha mejorado la compatibilidad con la eliminación no confirmada de registros bajo el control de compromisos
  - Entidad: compatibilidad mejorada con omisiones consecutivas, ya que OMIT tiene un significado implícito `AND`
  - Se ha añadido compatibilidad con el uso de la forma de camello en las entidades, mapeadores y configuradores para gestionar las costumbres denominadas definidas mediante una refactorización adicional
  - Se ha mejorado la propagación de la información del usuario a partir de las transacciones del entorno AS4 00 a través de toda la aplicación.
  - Se ha mejorado la precisión al finalizar un trabajo programado por Quartz en caso de interrupción.
  - Se ha mejorado el soporte de control de compromiso para adaptarlo al ámbito del programa

AWS Motor de transformación Blu Age 4.5.0

zOS

Mejoras

- JCL: generación mejorada de groovy para el conjunto de datos KSDS basada en el análisis LISTCAT

- COBOL
  - Se ha mejorado el análisis de la `COPY-REPLACING` sentencia para gestionar la sustitución del subcampo cualificado cuando existe ambigüedad en el nombre de este subcampo
  - Se ha mejorado el soporte para lo definido en la declaración `SYSOUT SPECIAL-NAMES`
  - Se ha mejorado el soporte de los CEROS figurativos en la declaración `ADD n TO ZERO`
  - Se ha mejorado la compatibilidad con una `REPLACE` declaración para gestionar la emisión de varias líneas al aplanar las teclas y los bloques de texto de varias líneas
  - Soporte mejorado para operaciones aritméticas con cláusulas ADD/SUBTRACT/MULTIPLY/ DIVIDE `GIVING`
  - Se inició el soporte de análisis de REPORT SECTION y sus acciones relacionadas (INICIAR, TERMINAR, GENERAR informes)
- Misc: mejore la generación y la solidez de los informes meteorológicos

AS400

Mejoras

- DDS
  - Se ha mejorado la compatibilidad con la longitud implícita del tipo DATE
  - Se ha mejorado el soporte del stop-zero-suppression carácter en la palabra clave EDITWORD
  - Se ha mejorado la compatibilidad con el nombre de columna DESC, ya que es una palabra reservada en la base de datos
- JUEGO DE ROL
  - Soporte mejorado para la versión integrada %TIME
  - Se ha mejorado la generación de sentencias EVALR para gestionar la asignación de un valor de cadena a una variable de menor longitud con un mejor ajuste a la derecha
  - Análisis SQL mejorado en torno a la configuración de opciones
  - Soporte mejorado para la inicialización de PSDS en los programas RPGLE de NOMAIN
  - Se ha mejorado la compatibilidad con la palabra clave LIKE para definir un campo numérico DDS como empaquetado, independientemente de su descripción externa
  - Se ha mejorado el saneamiento de los nombres de los archivos al sustituir «$» por «DL»
  - Se ha mejorado la compatibilidad con el %SUBST integrado para gestionar valores de doble byte

- COBOL400
  - Pantalla: compatibilidad mejorada con el registro DSPF en las operaciones de E/S
- CL
  - Se ha mejorado el cambio de nombre de los nombres de variables reservadas
  - Se ha mejorado la compatibilidad con las condiciones de selección/omisión para gestionar archivos de varios formatos
- Misc (Varios)
  - Se han reducido las entidades duplicadas en torno a las operaciones con archivos (EOF, FOUND, EQUAL)
  - Generación mejorada de archivos JRXML para QPRINT, una impresora estándar del AS/400. Cuando se utilice, el archivo JSON creado no contendrá ninguna referencia al programa o al archivo. Solo se genera un archivo JRXML (QPrint-QPrint.jrxml)
  - Se ha mejorado la visualización de información de mensajes adicional para los componentes que muestran los mensajes de la cola de programas

Notas de la versión 4.4.0

Fecha de lanzamiento: 13 de noviembre de 2024

Esta versión de AWS Blu Age Runtime and Transformation Engines se centra en actualizar las dependencias críticas y las tecnologías compatibles, a la vez que se mejora el rendimiento en múltiples funcionalidades. Entre los cambios y características clave de esta versión se incluyen:

- Actualizaciones de dependencias: las aplicaciones de consola (BAC y JAC) y las aplicaciones modernizadas ahora se ejecutan en Bootstrap 5. El AWS Blu Age Runtime ahora funciona con el framework Spring Boot 3.3.5.
- Rendimiento: se mejoró el rendimiento de la ejecución de las máquinas de estado (hasta 10 veces más rápido), gracias a una nueva implementación que supera la degradación del rendimiento tras actualizar la biblioteca Spring State Machine de la versión 2.5.1 a la 4.0.0. Esta actualización no era opcional, ya que la versión 2.5.1 ya no recibía mantenimiento y contenía versiones Critical y High. CVEs Incluye una implementación de máquina de estados en tiempo de ejecución en la plataforma para una nueva biblioteca, con una implementación de máquina de estados liviana y eficiente, sin CVE y con un mejor rendimiento general.
- Simplificación del acceso a la base de datos: se llevó a cabo una importante revisión de los componentes utilizados para acceder a la base de datos, incluidas las entidades JPA DAOs,

las entidades DDS DataSimplifier y los mapeadores. Este rediseño se debió a la necesidad de brindar un mejor soporte a la función OVRDBF (anular el archivo de base de datos) común en los proyectos de 00. AS4 Permite gestionar más casos con una arquitectura simplificada para el código generado.

Probamos esta versión del AWS Blu Age Runtime con la siguiente pila. Es posible que otras versiones de componentes también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 18.18
	Npm 9.8
	Angular 17
Capa del servicio	Spring Boot 3.3.5
	Spring Core 6.1.14
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Runtime, versión 4.4.0

zOS

Nuevas características

- COBOL: se agregó soporte para la declaración JSON GENERATE

- COBOL: se agregó soporte para bloques de control
- MF: se agregó soporte para la directiva del compilador FCDREG
- Blusam - Se agregó la función de conjuntos de archivos VSAM con una implementación basada en el esquema de la base de datos - Solo se admite PostgresSQL
- Blusam: se agregó soporte para el manejo del TTL (Time to live) para los elementos de datos en caché de Blusam (motor de caché Redis)
- JCL - IDCAMS: se agregó una nueva propiedad para forzar el uso del juego de caracteres para decodificar la tarjeta SYSIN `idcams.encoding.forced`
- JICS: se ha ampliado la `jics.db.dataScriptLocation` propiedad para que acepte una lista de `application-main.yml` rutas de archivos y carpetas. El orden de la lista es importante. El primer archivo SQL se ejecuta primero y así sucesivamente. Cuando se ejecuta una carpeta, los scripts SQL que contiene se ejecutan sin un orden definido.
- Se agregó soporte para la utilidad CEE3 ABD

Mejoras

- Blusam: se mejoró el tiempo de carga y el consumo de memoria de los grandes conjuntos de datos antiguos a Blusam para los clientes que utilizan el motor PostgreSQL (observamos un aumento de hasta 8 veces en la velocidad de carga de conjuntos de datos de gran tamaño)
- Blusam: API ToS3 mejorada con exportDataSet Credentials Support
- Blusam: se mejoró la carga de archivos LISTCAT para la creación de conjuntos de datos
- Blusam: soporte mejorado para la lectura dinámica mediante una clave explícita
- Blusam: se ha mejorado la lógica del mecanismo de escritura trasera
- JCL: soporte JES mejorado para mejorar el bloqueo de archivos en la ejecución paralela
- JCL: se agregó soporte para la declaración `INCLUDE MEMBER`
- JCL - DNSUTILB - Se ha mejorado la compatibilidad con la clave duplicada para gestionar mayúsculas y minúsculas cuando la clave principal contiene espacios
- JCL - DSNUTILB: mejorado para optimizar el rendimiento al cargar datos gráficos LoadTask
- JCL - INFUTILB - Se agregó soporte para cuando no está definido `fetchsize` `chunksize`
- JCL - INFUTILB - Soporte mejorado para consultas que devuelven un conjunto de resultados vacío
- JCL - INFUTILB: solidez mejorada al procesar datos en CHUNK
- JCL - INFUTILB: soporte mejorado para la descarga con campos anulables
- JCL - INFUTILB - Soporte mejorado para tipos numéricos

- JCL - INFUTILB: descarga mejorada para campos anulables
- JCL - SORT - Soporte mejorado para la sintaxis OUTREC
- JCL - SORT - Análisis mejorado de la declaración DATE1
- JCL - SORT - Soporte mejorado de la cláusula INREC PARSE con RDW
- JCL - SORT - Formato de campos mejorado mediante máscaras de edición
- JCL - SORT - Se ha mejorado el soporte de 'SubString' en OUTREC
- JCL - SORT - Soporte mejorado para tarjetas compatibles con MF
- JCL - UNLOAD - Soporte mejorado del tamaño de campo con Postgresql
- JCL - IDCAMS: rendimiento mejorado para la carga de archivos del conjunto de datos VSAM al introducir el modo masivo
- PL/1 - Mejora la compatibilidad con el NumericEditedType formato para evitar discrepancias de escala
- IMS: compatibilidad mejorada con la base de datos IMS (columna derecha) en NodeSorter
- CICS: comando mejorado `RECEIVE MAP` con `SET` y sin `INTO`
- BMS: soporte mejorado del valor inicial de campo
- SQL: DateTimeFormat análisis mejorado del patrón `ddMMMyy`
- COBOL: compatibilidad mejorada con el NumericEditedType valor cuando no se tiene en cuenta el punto decimal al obtener el valor
- Soporte mejorado para leer campos de longitud variable en archivos secuenciales de líneas
- Soporte mejorado para la herencia del tamaño de los registros del catálogo de conjuntos de datos para archivos GDG
- Soporte mejorado para la impresión de informes al permitir líneas de avance personalizables
- Se ha mejorado la inicialización de los datos de registro para los archivos de bloques variables (VB)

GS21

Nuevas características

- Pantalla: se agregó soporte para archivos PSAM
- Pantalla: se agregó soporte para ATTR2
- Se agregó soporte para el ecosistema AIM (Advanced Information Manager).
- Se agregó soporte PED en AIM

Mejoras

- Se han mejorado BitUtils las firmas para gestionar RangeReference
- Se ha mejorado la compatibilidad DummyFileConfiguration para añadir los atributos RecordSize, rdw, bdw, blksize y blkszlim
- Se ha mejorado la compatibilidad con la sentencia VPOINT para gestionar el caso de que no se encuentre un registro
- Se ha añadido robustez al acceder a la matriz de bytes del registro
- Se ha mejorado el mapeo de caracteres del juego de caracteres JEF
- Soporte mejorado para el manejo de matrices y condiciones en el mapeo JDBC
- Se ha mejorado la compatibilidad con las solicitudes SQL en las distintas sentencias de la NDB, lo que permite gestionar mejor las variaciones de las sintaxis SQL mediante constantes para cada parte de una consulta SQL.
- Se ha mejorado el soporte para que GS21 PackedType el último fragmento sea C, D o F para la validación numérica
- Pantalla: compatibilidad mejorada con ACSAPI y con SPA y DefaultPsamController ENTER
- Pantalla: compatibilidad mejorada con los verbos ACSAPI y NDB

AS400

Nuevas características

- Se agregó soporte para archivos de bases de datos con formato de registro múltiple
- Se rediseñó el marco de acceso a la base de datos AS4 00
  - Capacidades mejoradas en torno a la anulación de archivos
  - Se eliminaron los componentes obsoletos y se redujo la complejidad
  - Se simplificó el código generado a partir de los programas heredados
  - Integramos el componente DAOCycle Manager en el complemento Blu4iv, lo que nos permite aprovechar las AS4 00 funciones específicas de nuestro tiempo de ejecución personalizado.
- JOB: soporte mejorado para la gestión de tareas (Quartz) para añadir la posibilidad de interrumpir una tarea o un grupo de tareas. Se agregó un punto final de la API REST para interrumpir un trabajo con el identificador de ejecución especificado (único para cada trabajo, ya que es una clave principal). Si la interrupción se realiza correctamente, el motor de ejecución actualiza el estado del trabajo a «INTERRUMPIDO».

- Se agregó soporte para el programa de utilidad CEERAN0
- Se agregó soporte para el modo pasivo. Se agregó el YAML `configuration gapwalk-application.cl:ftpservice:passive` para habilitar el modo pasivo
- Se agregó una función para crear sesiones QTEMP y retrasar la limpieza de QTEMP
- Se agregó soporte para la función de compilación BNDDIR para definir dependencias explícitas entre programas
- Se agregó soporte para el mecanismo de grupos de activación

Mejoras

- CL: se ha mejorado el comando RMVMSG en la cola de mensajes del programa para gestionar la palabra clave *PREV
- CL: soporte mejorado para anulaciones en OPNQRYF
- CL - Se agregó soporte para los parámetros MSGLEN y SECLVLLEN del comando RTVMSG
- CL - Se ha mejorado la compatibilidad con CRTDUPOBJ para gestionar mayúsculas y minúsculas cuando no se pasa el archivo NEWOBJ y se ha añadido compatibilidad con nombres de tablas genéricas
- CL: compatibilidad mejorada con FTP para gestionar los parámetros GET, RMTSYS y BINARY
- CL: se mejoró el rendimiento de las consultas de CLRPFM y se agregó una opción para usar TRUNCATE en lugar de DELETE
- CL - Se ha mejorado el SBMJOB para gestionar correctamente el parámetro USER y utilizarlo como USUARIO cuando se envía un trabajo
- CL: soporte de comandos DLTOVR mejorado para gestionar el caso de*ALL
- Área de datos: soporte mejorado para Blu4 al agregar registros para DataArea el manejo de excepciones
- Área de datos: soporte mejorado para que Blu4 DataArea busque una nueva DataAreaDao instancia para cada hilo
- Área de datos: se mejoraron los bloqueos del área de datos, evitando bloqueos a nivel de registro y, en su lugar, utilizando el mecanismo de bloqueo recientemente implementado
- Área de datos: la operación de escritura del área de datos ahora continúa con la ejecución cuando no se adquiere un bloqueo y se proporciona un indicador de error
- Informe: se ha mejorado la compatibilidad con la convención de nomenclatura y ruta de salida de los informes para los informes impresos. Permitió a los clientes personalizar la ruta de salida

del informe y también el nombre. El cliente puede especificar su propia ruta y convención de nomenclatura sin afectar a ningún otro proyecto.

- TRABAJO: Soporte mejorado para la gestión de tareas (Quartz) para actualizar el estado del trabajo en caso de una rescisión anormal del trabajo, p. ej.: «cierre» o «cierre anormal» de Tomcat
- Pantalla: manejo mejorado del valor numérico en el campo al editar palabras con un signo menos
- Pantalla: ventana emergente de renderización mejorada con solo titleColorTop
- Pantalla: soporte mejorado para la recuperación de información de ayuda para gestionar mayúsculas y minúsculas cuando no se encuentra un elemento de ayuda general
- Pantalla: se ha mejorado la visualización de la pantalla de «información adicional» al pulsar F1 en la línea de mensajes del subarchivo
- Pantalla: visualización mejorada de los pies de página de los mensajes para SFLMSG
- Pantalla: interfaz mejorada para eliminar un registro en su totalidad cuando se superpone un registro nuevo
- Lista de espera: se ha mejorado la recuperación de mensajes de RabbitMQ para consumir menos recursos
- Creación de colas: se ha mejorado la implementación de la cola de datos de RabbitMQ para recuperar solo un mensaje a la vez.
- SQL: manejo mejorado de SQLCODE en SQLExecutor Builder para consultas dinámicas de tablas CREATE y DROP
- SQL: compatibilidad mejorada con OVRDBF en las consultas
- SQL: SQLExecutor generador mejorado para que las anulaciones de OVRDBF se apliquen a las sentencias preparadas
- RPG: compatibilidad mejorada con las especificaciones de entrada y salida de los archivos de disco descritos por el programa
- RPG: soporte mejorado para la lectura de archivos primarios y secundarios con el indicador MR (registros coincidentes). Se ha mejorado el orden de recuperación de un ciclo DAO con campos coincidentes.
- RPG: compatibilidad mejorada con los archivos principales y secundarios. Mejora en la actualización de los archivos principales y la generación de código de actualización/escritura de archivos secundarios de salida.
- RPG: Se agregó soporte para la declaración RETURN en formato libre
- RPG: transformación mejorada y manejo del tiempo de ejecución de las asignaciones decimales numéricas,

- RPG: generación mejorada de variables binarias
- RPG: soporte mejorado para EDITC
- RPG: manejo mejorado del área de datos local
- Se ha mejorado la compatibilidad con los campos DDS compartidos por varios tipos de dispositivos (DISK, WORKSATION, PRINTER)
- Se ha mejorado el manejo de las anulaciones para que las anulaciones activadas ya no afecten PFs LFs
- Se ha mejorado Blu4 para ivWebController no volver a establecer el nombre de usuario y el identificador de usuario a los valores predeterminados
- Se ha mejorado el ajuste del índice durante las lecturas de registros cuando cambia la dirección de lectura
- Se ha mejorado la ubicación del cursor en las lecturas de registros tras las operaciones de actualización/eliminación
- Se ha mejorado la compatibilidad con la lectura en una DAO de varias entidades cuando cambia la dirección de lectura
- Se ha mejorado la compatibilidad con los espacios de usuario para evitar que todos los subprocesos reutilicen la instancia en lugar de que cada subproceso tenga su propia instancia
- Se ha mejorado la compatibilidad con el acceso simultáneo a varios subprocesos en la lectura de registros
- Se mejoró el almacenamiento del nombre de usuario/ID de usuario mediante la configuración de YML SharedContext
- Versión mejorada de Locked Records con valores actualizados
- Se agregó soporte para el comportamiento específico del compilador OPM para la declaración NEXT SENTENCE

Capacidades transversales

Nuevas características

- Se agregó la nueva propiedad metadata.ini que `legacy.compilerto` especifica el compilador heredado de los artefactos que se van a transformar. El soporte de algunas sentencias de COBOL, como NEXT SENTENCE, es diferente según el valor que se establezca.
  - «ZOS» para un sistema z/OS heredado.

- «ILE» u «OPM» para AS4 un sistema 00. Predeterminado = «ILE» cuando `legacy.system` = «as400"

Mejoras

- Interfaz: se rediseñaron los componentes de los campos de la pantalla para ampliar la gama de tipos de campos compatibles. Esta mejora permite que el tiempo de ejecución se adapte a una variedad más amplia de requisitos de entrada y datos de los usuarios relacionados con 00. AS4
- Método mejorado `isValid()` para separar el byte de signo en ZonedType
- Soporte mejorado `StringConcatenationBuilder::withPointer` para la concatenación con CRLF
- Se ha mejorado la compatibilidad con codificaciones específicas de bytes dobles para que sean seguras para subprocesos
- Se mejoró el rendimiento de las máquinas de estado mediante la integración de un nuevo marco
- Algoritmo mejorado para la optimización de las asignaciones a fin de evitar reescrituras inesperadas

AWS Motor de transformación Blu Age 4.4.0

zOS

Mejoras

- LISTCAT: analizador mejorado para evitar entradas duplicadas
- LISTCAT: compatibilidad mejorada de ESDS con el sistema de archivos en JCL/Groovy
- CICS: soporte mejorado para la longitud de las declaraciones CICS

AS400

Mejoras

- Mejora de la generación de registros DDS
  - Se mejoró el soporte del registro DDS para generar entidades que se correspondan con la estructura de registros del DDS
  - Se proporcionó soporte para campos compartidos y funciones de mapeo que se adaptan mejor a las versiones anteriores

  - Se mejoró el manejo de los archivos descritos externamente y los descritos por el programa
- RPG: detección de juegos de rol mejorada para un módulo que solo tiene formato libre
- RPG: Se ha mejorado la compatibilidad con la sentencia COPY que permite ignorar la palabra clave `*LIBL/` como prefijo para localizar el cuaderno de notas de una aplicación
- RPG - PF: soporte mejorado para la especificación de entradas con registros físicos del perfil
- RPG: se agregó soporte para la declaración On-Exit
- RPG: soporte mejorado de palabras clave LikeRec
- RPG: mapeo mejorado de campos DSPF renombrados
- CL: resolución mejorada de nombres de campos
- COBOL: soporte mejorado para la conversión de hexadecimales a caracteres
- Soporte mejorado para la generación de tipos decimales
- Se ha mejorado la compatibilidad con el mensaje FIXME para el código heredado no compatible (mostrar toda la línea antigua)
- Rendimiento mejorado en AWS Transformation Engine (paso de análisis AS4 00)
- Se mejoró el soporte de la palabra clave LikeRec para alinearla con las especificaciones del archivo
- Se ha mejorado la compatibilidad con la función integrada %Diff
- Se ha añadido compatibilidad con el signo de moneda con caracteres especiales en la etiqueta DSPF

Notas de la versión 4.3.0

Fecha de lanzamiento: 16 de septiembre de 2024

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en ampliar las capacidades y la cobertura para modernizar las funcionalidades del mainframe. Entre los cambios y características clave de esta versión se incluyen:

- CICS: compatibilidad adicional para intercambiar datos desde los terminales y ejecutar transacciones con los datos entrantes al admitir el comando SEND MAP con Map Reference.
- JCL: nueva capacidad que permite reiniciar la ejecución más reciente de un trabajo por lotes a partir de un paso de JCL/PROC anteriormente fallido, o provocar un reinicio retrasado al omitir los

pasos ejecutados anteriormente. Esto proporciona un mayor control sobre el procesamiento por lotes mediante puntos de comprobación persistentes escalonados.

- AS400: compatibilidad adicional con bibliotecas, rendimiento y robustez mejorados de los comandos más utilizados, como CPYF, OVRDBF, SBMJOB y OPNQRYF, y muchos más.

Probamos esta versión del Blu Age Runtime con la siguiente pila. AWS Es posible que otras versiones de componentes también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 18.18
	Npm 9.8
	Angular 17
Capa del servicio	Spring Boot 3.2.5
	Spring Core 6.1.5
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 4.3.0

zOS

Nuevas características

- CICS: se ha agregado compatibilidad con Map Reference en el comando SEND MAP.

- CICS: se ha agregado compatibilidad con el comando RECEIVE y compatibilidad para ejecutar transacciones con datos de la pantalla `JicsTransactionRunner`.
- Se ha agregado compatibilidad con el encabezado IIH para los mensajes JMS.
- COBOL: se ha agregado compatibilidad con múltiples espacios incrustados en el pseudotexto para la instrucción REPLACING.
- COBOL: se ha agregado compatibilidad con la instrucción JSON PARSE.
- Blusam: se ha agregado compatibilidad con KMS para incluir la característica Exportar conjunto de datos.
- BAC: se agregó la configuración de `application-main.yaml` para definir el tamaño del registro y filtrar las máscaras cargadas que coincidan con este tamaño de registro.
- JCL: INFUTILB: se ha agregado compatibilidad con la palabra clave INTO como parte de la instrucción de control de BMC.
- GS21 - Se agregó el manejo de SOSI para la codificación JEF
- GS21 - JCL - Se agregó KDJBR14 como alias de IEFBR14
- GS21 - JCL - Se agregó KQCAMS como alias de IDCAMS
- MF: se ha agregado compatibilidad con archivos compatibles con COBOL MF según la compatibilidad de campo.
- MF: se ha agregado compatibilidad con el mecanismo SORT para archivos compatibles con COBOL MF.
- MF: se ha agregado compatibilidad para abrir archivos que faltan no opcionales compatibles con COBOL MF.

Mejoras

- JCL: DSNUTILB: se ha mejorado la operación LOAD con el tipo ZONED DECIMAL.
- JCL: DSNUTILB: se ha agregado compatibilidad con clave duplicada.
- JCL: DSNUTILB: se ha agregado compatibilidad con el mecanismo de reversión en el comando LOAD.
- JCL: INFUTILB: se ha mejorado UNLOAD con las nuevas propiedades FETCHSIZE y CHUNKSIZE.
- JCL - A - IKJEFT1 Se ha mejorado la lectura de archivos SYSTEM añadiendo el juego de caracteres actual
- JCL - DFSORT: se agregó soporte para la opción & DATE4 DATE5

- JCL: DFSORT: se ha agregado compatibilidad con el caso del tipo de bloque variable como entrada y el tipo de bloque fijo como salida.
- JCL: DFSORT: se ha agregado compatibilidad con ALTSEQ.
- JCL: se han mejorado los metadatos de puntos de comprobación con el identificador web de trabajos.
- JCL: se ha mejorado la purga de puntos de comprobación de reinicio por lotes para REDIS.
- IMS: se ha implementado la función EXPRESS para el comando PURGE.
- IMS: se ha agregado compatibilidad con las opciones PCBNAME y LIST para la instrucción PCB.
- COBOL: se ha agregado compatibilidad con la declaración GO TO sin objetivo.
- CICS: soporte mejorado para la declaración INTO en READQ TS RecordAdaptable
- CICS: se ha mejorado la compatibilidad con el comando INQUIRE TRANSACTION.
- CICS: se ha mejorado la compatibilidad con setBytes en el comando READNEXT.
- CICS: se ha mejorado la compatibilidad con el comando START sin la opción CHANNEL.
- CICS - Se agregó soporte para el tipo de referencia para Inquire TSQueue
- CICS: se ha mejorado la compatibilidad con el comando RECEIVE MAP cuando el mapa y el conjunto de mapas son de referencia.
- CICS: se ha mejorado la compatibilidad con las opciones FROM y LENGTH del comando RECEIVE MAP.
- CICS: se agregó soporte para el atributo RecordAdaptable
- CICS: se ha mejorado la compatibilidad con el comando RECEIVE para gestionar el desbordamiento.
- CICS: se ha agregado compatibilidad con la regla de segmentación en las instrucciones CICS.
- CICS: se ha mejorado la compatibilidad con las estructuras de vinculación DFHCOMMAREA y DFHEIBLK. El motor de transformación admite definiciones más implícitas
- CICS: se ha agregado compatibilidad con las opciones START, NEXT y END para el comando INQUIRE CONNECTION.
- CICS: se ha agregado compatibilidad con los tipos int y reference para la opción LENGTH del comando RECEIVE.
- CICS: se ha mejorado la compatibilidad para analizar el comando INQUIRE NETNAME.
- CICS: se agregó soporte para el nombre de grupo para JicsQueueBuilder
- Blusam: se ha agregado compatibilidad con archivos indexados que comiencen con una clave genérica.

- Blusam: se han mejorado los cargadores de Blusam.
- BAC: se ha mejorado la compatibilidad con la sincronización de datos en entornos de varias instancias cuando se utiliza Redis para centralizar los valores en caché, incluidos bloqueos y datos reales.
- BAC: interfaz de usuario mejorada (estilo, logotipo, casilla de verificación).
- BAC y JAC: se ha agregado la configuración de application-main.yaml para recuperar el nombre de usuario y la contraseña del usuario superadministrador predeterminado en el secreto de AWS Secrets Manager especificando el ARN.
- BAC y JAC: actualización de la dependencia a Bootstrap 5.
- Se han mejorado los puntos de control de JCL y la configuración de la plantilla JICS Redis TSQueues
- Se ha mejorado la compatibilidad con el tamaño del puntero en función de AMode
- Se agregó soporte para la comparación cero en NumericEditedType
- Se forzaron las propiedades MDC de Slf4j antes del registro.
- Se ha mejorado la compatibilidad de lectura de archivos para gestionar varias líneas vacías.
- MF: se ha mejorado la compatibilidad para inicializar variables de puntero para la directiva initPtr del compilador MF COBOL.
- Redis: función mejorada GwFileLock en el aspecto de la concurrencia mediante una implementación basada en Redisson

AS400

Nuevas características

- CL: se ha agregado compatibilidad con el comando CHGPF.
- RPG: se ha agregado compatibilidad con las funciones %HOURS, %MINUTES y %SECONDS.
- COBOL: se ha agregado la compatibilidad del archivo SORT con la arquitectura DAO Blu4IV.

Mejoras

- CL: se mejoró PgmClose para registrarse como un programa y aceptar una variedad de objetos como parámetro OPNID
- CL: se ha refactorizado RTVMBRD para gestionar múltiples bibliotecas y miembros.
- CL: se ha agregado compatibilidad con el parámetro TOLIB en el comando MOVOBJ.

- CL: se ha mejorado la compatibilidad de particiones en el comando CPYFRMSTMF.
- CL: se ha agregado compatibilidad con el parámetro TOUSR de SNDMSG.
- CL: se ha mejorado la compatibilidad del comando OVRDBF.
- CL: se ha mejorado el rendimiento del comando OVRDBF: actualización de los valores predeterminados para srcfile y member.
- CL: se ha mejorado la copia de archivos con el comando CPYF.
- CL: se ha rediseñado el comando CPYF para que sea más robusto y gestione mejor QTEMP, CRTFILE, FROMRCD y TORCD, MBROPT y FMTOPT (MAP Y DROP).
- CL: se ha mejorado la compatibilidad con el comando CPYF en los casos en que FROMFILE y TOFILE tienen columnas que no coinciden.
- CL: se ha mejorado en NOCHK de CPYF la gestión de columnas con nombres diferentes cuando se especifica REPLACE.
- CL: se ha agregado una implementación vacía para el comando CRTDUPOBJ en los archivos lógicos.
- CL: se ha gestionado el problema de indexación de subcadenas con el comando CHGDTAARA.
- CL: se ha mejorado la compatibilidad con el comando SBMJOB.
- CL - Creado y mapeado sin distinción entre mayúsculas OverrideManager y minúsculas OpnqryfHelper
- Pantalla: se ha mejorado el foco inicial del primer campo editable cuando no se especifica un cursor.
- Pantalla: se ha mejorado la posición de foco después de cerrar y al usar el menú de ayuda.
- Pantalla: se ha mejorado el foco del cursor después de pulsar la página hacia arriba o hacia abajo en un componente de tabla.
- Pantalla: se ha mejorado la compatibilidad con varios mensajes de error de campo y el foco.
- Pantalla: se ha mejorado el cálculo del número de línea para los campos de subarchivos
- Pantalla: se ha mejorado la compatibilidad con los subarchivos inicializados mediante SFLINZ.
- Pantalla: se ha mejorado la compatibilidad con la entrada solo numérica.
- Pantalla: se ha mejorado la gestión de la palabra clave WINDOW en DSPF con 3 parámetros.
- Pantalla: se ha mejorado la posición del pie de página para tablas con registros que contienen más de 1 línea.
- Pantalla: se ha mejorado la navegación de la página para que los mensajes de rotación se queden al subir o bajar la página.

- Se ha mejorado la funcionalidad de EDITC para el código de edición 3.
- Se ha mejorado el mecanismo de bloqueo del área de datos de Blu4iv para no hacer nada cuando no haya ningún bloqueo que desbloquear en lugar de lanzar una excepción.
- Se agregó soporte para devolver el número de filas afectadas StraightQueryBuilder
- Se ha mejorado el mecanismo de registro de QTEMP.
- Se ha mejorado DAOManager reads/writes/deletes para los casos de uso en los que un archivo ha sido anulado por un archivo o biblioteca diferente

Capacidades transversales

Nuevas características

- Se agregó una forma centralizada de administrar las propiedades del sistema relacionadas con SSL/TLS mediante la configuración, lo que permite el uso de AWS Secrets Manager
- Configuración mejorada de los recursos de IBMMQ con AWS Secrets Manager
- JCL: Se agregó la configuración de ubicación temporal para los archivos groovy resueltos en tiempo de ejecución mediante la propiedad YML tempFilesDirectory y se agregó la capacidad de especificar si se debe purgar el contenido de la carpeta de archivos temporales al iniciar la aplicación mediante la propiedad YML cleanTempFiles DirectoryAtStartup
- Se han agregado secretos de AWS para todas las credenciales de Redis.

Mejoras

- Se ha mejorado la conversión del tipo alfanumérico al tipo numérico editado.
- Se mejoró la comprobación: :isNumeric para DataUtils PackedType
- Se ha mejorado la marca temporal de los archivos de registro.
- Se gestionó el inicio de sesión por separado. ZonedType decodeAsString
- COBOL: se ha mejorado la compatibilidad con la instrucción INITIALIZE.
- Soporte mejorado de DataUtils. compareAlphInt para gestionar los espacios iniciales y finales de AS4 00 y ZOS
- SQL: se ha mejorado la validación implícita del tiempo de ejecución del cursor de solo lectura.
- SQL: se ha mejorado el mecanismo de almacenamiento en caché de metadatos.
- Se ha eliminado la conexión a la base de datos de Jics/Blusam de la aplicación de Gapwalk `application-main.yml`.

Herramientas de modernización versión 4.3.0

zOS

Nuevas características

- GS21 - Se ha añadido soporte para COBOL GS21 CONSTANT SECTION
- GS21 - Se agregó la codificación JEF a los conjuntos de caracteres disponibles

Mejoras

- CICS: se ha agregado compatibilidad para analizar el comando DOCUMENT CREATE.
- CICS: se ha agregado compatibilidad para analizar el comando CICS WEB EXTRACT.
- CICS: se ha agregado compatibilidad para analizar el comando WEB WRITE.
- CICS: se agregó soporte de transformación para DB2 CONN SIGNIN y PLAN
- CICS: se ha mejorado la compatibilidad para analizar el comando SEND MAP ignorando la opción TERMINAL.
- CICS: se ha mejorado la compatibilidad para analizar el comando RETURN ignorando la opción ENDACTIVITY.
- MFS: se ha mejorado la compatibilidad para generar archivos MFS con una extensión específica.
- COBOL: se ha mejorado la compatibilidad con la instrucción REPLACE.
- COBOL: se ha gestionado la ruta dinámica y la directiva del compilador de MF.
- COBOL: se ha mejorado la compatibilidad con el valor OMITTED en la instrucción CALL.
- COBOL: se han mejorado los campos multidimensionales para admitir valores firmados.
- COBOL: se ha agregado compatibilidad con la cláusula OF en la instrucción FILE STATUS.
- COBOL: análisis mejorado de la declaración RESULT-SET-LOCATOR
- JCL: IDCAMS: se ha agregado la compatibilidad con la abreviatura de RECORDS.

AS400

Nuevas características

- CL: se ha agregado la compatibilidad con variables definidas y basadas en punteros en la transformación de CL.

- CL: se ha agregado la compatibilidad con caracteres especiales en DCLF.
- Se ha agregado la compatibilidad con la API de la pila de llamadas (QWVRCSTK).

Mejoras

- RPG: se ha mejorado la transformación de los parámetros de procedimiento con la palabra clave `likeds`.
- RPG: se ha revisado la compatibilidad con la palabra clave EXTNAME.
- RPG: se ha mejorado la compatibilidad con el valor literal *ALL.
- RPG: se ha mejorado la compatibilidad con los archivos descritos por el programa y la especificación de salida.
- DDS: se ha mejorado la resolución de los campos DDS en un LF que hace referencia a un PF que hace referencia a un PF de diccionario.
- Pantalla: se han borrado los indicadores cuando se utiliza la instrucción CLEAR para borrar un registro de DSPF.
- CL: se ha mejorado la transformación/generación de los parámetros CL con listas de elementos.

Capacidades transversales

Mejoras

- SQL: se ha mejorado la generación de las consultas SQL que contienen N con carácter con tilde.
- COBOL: se ha mejorado la compatibilidad con la instrucción LENGTH OF para campos de grupo.
- COBOL: se ha mejorado la compatibilidad con campos REDEFINED mediante cuadernos.

Notas de la versión 4.2.0

Fecha de lanzamiento: 10 de julio de 2024

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en el rendimiento y la seguridad. Algunas funciones y cambios clave de esta versión son:

- Hemos mejorado el rendimiento de la transformación, especialmente para proyectos grandes con más de 30 millones de líneas de código. Hemos implementado una serie de mejoras y los resultados que obtuvimos mostraron una reducción del tiempo de más del 150 % y las ejecuciones

se completaron en cuestión de minutos en lugar de horas. La mejora clave que implementamos es la configuración de un mecanismo de tiempo de espera para limitar el tiempo máximo asignado al análisis con el fin de omitir los archivos con problemas detectados. Marcamos los archivos omitidos para que pueda investigarlos más adelante si es necesario.

- Hemos añadido soporte para un sistema de gestión de bloqueos distribuido para AS4 00 proyectos. En un entorno de alta disponibilidad (varios nodos) en el que varias instancias de la aplicación apuntan a la misma base de datos, mantener la coherencia de los datos durante todo el ciclo de vida de estas instancias resulta un desafío importante. Para abordar este desafío de manera efectiva, hemos agregado Redis como servidor de almacenamiento en caché externo y compartido para coordinar entre todas las instancias cuando se ejecutan por lotes.
- Hemos agregado una nueva característica de paginación dinámica para el componente de tabla. El objetivo de esta característica es mejorar el tiempo de respuesta y reducir el uso de memoria en las tablas con un gran número de filas. Esta característica permite que el componente de tabla solo cargue parte de los datos y busque más registros según la demanda conforme navega por las páginas. Para mejorar aún más la experiencia, la plataforma también admite la captura previa de datos. Esta nueva característica de paginación dinámica proporciona una experiencia de usuario más eficiente y con mayor capacidad de respuesta para aplicaciones con grandes conjuntos de datos.
- Para abordar un desafío clave que surge con frecuencia, hemos agregado compatibilidad con programas COBOL anidados. Anteriormente, la solución alternativa para modernizar los programas COBOL anidados consistía en separar manualmente los programas en diferentes archivos, vincularlos a través de la sección de enlaces y hacer que se llamaran entre sí con los argumentos necesarios. Este proceso no solo consumía mucho tiempo, sino que también era propenso a errores. Ahora puede modernizar los programas COBOL anidados sin necesidad de separarlos manualmente.

Probamos esta versión del AWS Blu Age Runtime con la siguiente pila. Es posible que otras versiones de componentes también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 18.18
	Npm 9.8

	Angular 17
Capa del servicio	Spring Boot 3.2.4
	Spring Core 6.1.5
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 4.2.0

zOS

Nuevas características

- DB2 - Se agregó soporte para la invocación de procedimientos almacenados sin calificador de esquema en la consulta SQL
- COBOL: se ha agregado compatibilidad con la función HEX-OF.
- COBOL: se ha agregado compatibilidad con programas anidados.
- COBOL: se agregó soporte para FUNCTION y TEST-DATE-YYYYMMDD TEST-DAY-YYYYDDD
- CICS: se ha agregado compatibilidad con la opción UCTRANST en el comando SET TERMINAL.
- CICS: se agregó soporte para el comando DB2 INQUIRE CONN
- BluSam - Se agregó soporte para la eliminación de claves en los VSAM a los que se accede dinámicamente
- IMS: se ha agregado compatibilidad con el comando TERM.
- BAC: se han agregado comprobaciones de autorización en todos los puntos de conexión REST de BAC.
- BAC: se ha agregado la configuración mediante `application-main.yaml` para definir un tamaño de registro para filtrar las máscaras cargadas que coincidan con ese tamaño de registro.

- BAC y JAC: se agregó una configuración `application-main.yaml` para recuperar el nombre de usuario y la contraseña del usuario superadministrador predeterminado en secreto `command` especificando el ARN

Mejoras

- JCL - SORT: compatibilidad mejorada con la cláusula OMIT para gestionar condiciones con mayúsculas y caracteres ShiftOut
- JCL: SORT: se ha mejorado la compatibilidad con el campo BDW.
- JCL: SORT: se ha mejorado la compatibilidad con varias concatenaciones con el campo BDW.
- JCL: DFSORT: se ha agregado compatibilidad con las cláusulas INREC PARSE STARTAFT/ STARTAT.
- JCL: IEBGENER: se ha mejorado la gestión de recordSize para los archivos de salida.
- JCL: INFUTILB: se ha deshabilitado NULL INDICATOR en función de YML- FIX GRAPHIC CASE.
- JCL: soporte mejorado para el manejo de constantes en el campo FormatterParser OUTREC
- JCL: se han mejorado los datos de carga para el tipo de gráfico en la utilidad del programa DSNUTILB.
- JCL: SORT: se ha mejorado la compatibilidad con el formato decimal zonificado.
- JCL - SORT: compatibilidad mejorada con la cláusula OMIT para gestionar condiciones con mayúsculas y caracteres ShiftOut
- MQ: se ha mejorado la gestión de la conexión MQ para adaptarse a varios flujos de trabajo empresariales.
- CICS: se ha mejorado la compatibilidad de la referencia del puntero para las instrucciones EXEC CICS READ SET (ptr-ref).
- COBOL: se ha mejorado la compatibilidad con el registro de la sección de enlace de ADDRESS OF.
- COBOL: se agregó soporte para las funciones EXP y 0 EXP1
- COBOL: se ha mejorado la compatibilidad con la instrucción REPLACE mediante un cuaderno.
- COBOL: se ha mejorado el acceso a los campos multidimensionales para admitir valores con signo.
- MF COBOL: se ha agregado compatibilidad con archivos secuenciales de formato variable.
- IMS: se ha mejorado la lectura de la configuración de los archivos YML de IMS para permitir el uso de variables de entorno.

- IMS: se han gestionado maneras adicionales de especificar el número de segmento.
- IMS: se ha agregado solidez cuando se llama a un programa IMS desde una transacción iniciada mediante programación.
- IMS: se ha mejorado los criterios de búsqueda creados por la SSA para tener en cuenta la longitud actual de la cláusula WHERE si no se proporciona la longitud del segmento implícita.
- IMS: se ha mejorado la lectura de la configuración de los archivos YML de IMS para permitir el uso de variables de entorno.
- Se ha mejorado el soporte para la cláusula VALUE en NumericEditedType
- Se ha mejorado la compatibilidad con la concatenación de cadenas para gestionar los casos en los que la primera cadena que se concatenará está vacía o en blanco, o tiene espacios.

AS400

Nuevas características

- Se ha agregado compatibilidad con la paginación dentro del componente Tabla; los proyectos pueden usar esta característica para reducir el tiempo y el tamaño de respuesta cuando se carga un componente Tabla con un gran número de filas.
- Se agregó compatibilidad con bibliotecas para consultas SQL en la aplicación AS4 00; dado que las bibliotecas se convierten en particiones en las aplicaciones modernas, adaptamos el tiempo de ejecución para reescribir las consultas en consecuencia
- RPG: se ha agregado compatibilidad con la biblioteca QTEMP para consultas SQL.
- RPG: se ha agregado codificación en la función CONVERT para gestionar valores de entrada vacíos.
- RPG: se ha agregado compatibilidad con las funciones %HOURS, %MINUTES y %SECONDS.
- CL: se ha agregado el comando CHGPFM.
- CL: se ha agregado compatibilidad con la palabra clave *FROMLIB en el comando CRTDUPOBJ.
- CL: se ha agregado compatibilidad con la creación de tablas y particiones para nombres de tablas que superen los 9 caracteres.
- CL: se ha agregado compatibilidad con la eliminación de archivos sin formato en subcarpetas para el comando DLTF.

Mejoras

- Pantalla: se ha mejorado ErrorMessage para vincular un campo específico y añadirlo a ArrayMessageLine
- Pantalla: se ha mejorado el cursor.
- Pantalla: mejorada ArrayMessageLine para no incluirla en el orden de pestañas
- Pantalla: visualización mejorada de las matrices de mensajes de error en la pantalla AS4 00
- SQL: se ha mejorado la compatibilidad con los cursores para confirmar la transacción al cerrarla y evitar así bloqueos en la creación de particiones.
- CL: se agregó soporte para el PgmCall comando y se mejoró el patrón QCMDEXC no compatible
- CL: se ha mejorado la compatibilidad con el comando CHKOBJ para gestionar OBJTYPE PGM.
- CL: se ha mejorado la compatibilidad con varias bibliotecas con CPYF y otros comandos de CL que se ocupan de bibliotecas y particiones.
- CL: se ha agregado compatibilidad para analizar una variable de nombre de programa en el comando CALL PGM.
- CL: se ha gestionado el caso para el tipo predeterminado de tipo Objeto.
- CL: se ha agregado compatibilidad con varias bibliotecas para el comando CRTDUPOBJ.
- CL: se ha mejorado la gestión de las conexiones a bases de datos en varios comandos.
- CL: se ha mejorado la compatibilidad con RMVLNK para gestionar los casos en los que no se encuentra un archivo o directorio y el mensaje de monitor CPF0000.
- CL: se ha mejorado CLRPFM para tener en cuenta la biblioteca al eliminar registros.
- CL - CPYF - Comando mejorado para admitir la biblioteca QTEMP, FmtOpt el parámetro (*) y el carácter de control NoChk
- CL: se ha corregido la gestión de las comillas y los parámetros que faltaban en los comandos RMVLNK y CPY.
- RPG: alcance variable mejorado; ahora DataArea está en el ámbito de trabajo en lugar del ámbito de enlace
- RPG: se han mejorado las consultas de lectura DAO para que se ejecuten sin transacciones y así evitar bloqueos.
- Se ha mejorado la búsqueda de mensajes MQ al agregar un recorte a MSGQ en la búsqueda en la base de datos.
- Se han eliminado las declaraciones de transacciones innecesarias en la compatibilidad con la conexión a la base de datos.
- Se ha mejorado la actualización del estado del trabajo de Quartz en caso de excepción.

- Se ha agregado compatibilidad para gestionar los casos en los que una matriz de indicadores no está inicializada.

Capacidades transversales

Nuevas características

- Redis: se ha agregado la configuración de Redis global para todas las cachés de Redis.
- Se ha agregado la funcionalidad de seguimiento de sesiones para poder almacenar la información de seguimiento de sesiones (ID de sesión, nombre de usuario asociado, marca de tiempo de creación e ID de nodo) mediante la conservación de los datos en Redis.
- JCL: se ha agregado la configuración de ubicación temporal para los archivos groovy resueltos del tiempo de ejecución mediante la propiedad `tempFilesDirectory` de YML. También se ha agregado la capacidad de especificar si se purgará el contenido de la carpeta de archivos temporales al iniciar la aplicación mediante la propiedad `cleanTempFilesDirectoryAtStartup` de YML.

Mejoras

- Se ha agregado compatibilidad para las propiedades de configuración de implementación de grupos de conexiones para los orígenes de datos de utilidad.
- Se ha mejorado la compatibilidad para el modo de impresora y el control de carro ANSI gracias al uso de las cláusulas ADVANCING y WRITE BEFORE.
- Se ha actualizado la versión de Angular en la aplicación front-end para proyectos modernizados.
- Construcción mejorada de la sintaxis de URL del administrador secreto para DB2
- Se mejoró el DataUtils. compareAlphInt método para añadir soporte a los espacios finales
- Se ha mejorado la compatibilidad con SQL para la salida de tipo blob.
- Se ha agregado solidez a los desencadenadores de trabajos mediante el punto de conexión post/script.

Herramientas de modernización versión 4.2.0

zOS

Nuevas características

- CICS: se ha agregado compatibilidad para analizar los comandos WEB CICS.
- CICS: se ha agregado compatibilidad con la transformación del comando MONITOR.
- CICS: se ha agregado compatibilidad para analizar el comando SEND MRO de CICS.
- COBOL: se ha agregado compatibilidad para analizar la instrucción NO REWIND.
- COBOL: se ha agregado compatibilidad con el tipo numérico de la opción UCTRANST en el comando SET TERMINAL de CICS.
- COBOL: se admite la adición de la cláusula MULTIPLE FILE en I-O-SECTION
- CSD: se ha agregado compatibilidad con la transformación de varios archivos CSD.
- CSD: se agregó soporte para la generación de jicsFileAix .json a partir de varios archivos CSD
- IDCAMS: se ha agregado compatibilidad con la creación de un conjunto de datos de registro relativo (RRDS).

Mejoras

- Se ha mejorado el rendimiento al calcular las máscaras SQL.
- COBOL: se ha mejorado el análisis de la cláusula RESERVE sin utilidad en FILE-CONTROL.
- COBOL: se ha mejorado el análisis de SECTION y CLASS.
- COBOL: se ha mejorado la gestión de DFHRESP.
- COBOL: se ha mejorado la compatibilidad con EXIT PARAGRAPH a través de perform.
- IMS: se ha mejorado la compatibilidad con los nombres de segmentos especificados mediante paréntesis dobles.
- IMS: se ha enriquecido la generación de códigos de estado cuando se invocan SCHD y TERM.
- COBOL: se ha mejorado la generación de campos DEPENDING ON.
- COBOL: transformación mejorada de la función incorporada TO_TIMESTAMP DB2

AS400

Nuevas características

- Se ha agregado compatibilidad para convertir campos alfanuméricos como CHAR en scripts SQL.
- COBOL400 - Se agregó soporte para los archivos de BASE DE DATOS descritos por el programa

Mejoras

- DDS: se ha mejorado la compatibilidad con el nombre ALIAS.
- Se ha mejorado la compatibilidad con el tipo float sin valor inicial.
- COBOL 400: se ha mejorado la computación del tamaño para tipos zonificados con signo.

Capacidades transversales

Mejoras

- Se han mejorado los informes de ID de error relacionados con el análisis de DDS y SQL.
- Se ha mejorado la generación de código en las ramas de condición.
- Se ha mejorado el rendimiento en la generación de informes meteorológicos.

Notas de la versión 4.1.0

Fecha de lanzamiento: 31 de mayo de 2024

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en el rendimiento y la seguridad. Algunas funciones y cambios clave de esta versión son:

- Transformación y rendimiento: para permitir que los proyectos con una base de código grande (más de 50 millones de líneas de código) se transformen correctamente, hemos optimizado el rendimiento y la huella de memoria de todo el mecanismo de transformación.
- BAC/JAC: La seguridad AWS es la máxima prioridad. Las aplicaciones modernizadas con AWS Blu Age deben cumplir con los estándares de seguridad. Hemos realizado algunas mejoras importantes en la Consola de BluSam administración (BAC) y la Consola de administración del JICS (JAC) para hacerlas más seguras:
  - Se ha actualizado la aplicación a Angular v17.
  - Además del soporte nativo para AWS Cognito, hemos añadido soporte genérico para ofrecer una mayor flexibilidad a fin de OAuth que los clientes puedan utilizar el proveedor de identidades que elijan.
  - Hemos configurado y ampliado las características de seguridad mediante los encabezados adecuados.
- AS400: Soporte de varios nodos para el mecanismo de bloqueo de bases de datos. Proporcionó la posibilidad de conectar un servidor de almacenamiento en caché externo y compartido (Redis) para ejecutar una aplicación por lotes en varias instancias, como la modernización gestionada AWS del mainframe.

Se ha probado esta versión del tiempo de ejecución de Blu Age con la siguiente pila. Es posible que otras versiones también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 18.18
	Npm 9.8
	Angular 16.1
Capa del servicio	Spring Boot 3.2.5
	Spring Core 6.1.5
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21c
Servidor de aplicaciones	Apache Tomcat 10.1.17

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 4.1.0

zOS

Nuevas características

- Se agregó una configuración para la gestión dinámica OAuth2 de los proveedores. Se introdujo SECRET_ OAUTH2 _PROVIDER_NAME_KEY para especificar el proveedor. Se ha actualizado el método de recuperación de secretos para gestionar varios proveedores. Se garantiza que los secretos se recuperen de forma segura de AWS Secrets Manager.

- Se ha añadido la compatibilidad AWS Secrets Manager con las propiedades de DB2 SSL para poder definir un certificado SSL (sslTrustStoreubicación) y una contraseña (sslTrustStorecontraseña) para desbloquear el archivo del almacén de claves.
- Se ha agregado compatibilidad con orígenes de datos empresariales externos.
- JCL: se ha agregado compatibilidad con el mecanismo de puntos de comprobación para el reinicio por lotes.
- JCL: se ha agregado compatibilidad con el tamaño del registro y RDW de los parámetros DCB.
- JCL: se ha agregado la configuración dinámica de nombres de carpetas para los archivos temporales generados.
- REDIS: se ha agregado una configuración de grupo en la configuración de Redis para JICS.
- REDIS: se ha agregado el índice de bases de datos en la configuración de Redis para catálogo y JICS.
- BatchScript - Se ha añadido la propagación del nombre del paso para ejecutar el programa.
- CICS: se ha agregado compatibilidad con el comando ADDRESS SET.
- CICS: se ha agregado compatibilidad con PURGE MESSAGE y JUSTIFY.

Mejoras

- JCL: INFUTILB: se ha mejorado la compatibilidad para deshabilitar el indicador nulo en función de la propiedad YML.
- JCL: INFUTILB: se ha mejorado la compatibilidad con el tipo de datos CHAR/BPCHAR.
- JCL: ICEGENER: se ha mejorado la compatibilidad para copiar secuencias de entrada multilínea en archivos.
- JCL: IEBGENER: se ha mejorado la compatibilidad para gestionar la conversión de archivos de bloque variable a bloque fijo.
- JCL: DFSORT: se ha mejorado la compatibilidad con parámetros de varios dígitos en DATE de operación.
- JCL: DFSORT: se ha agregado compatibilidad con la cláusula INCLUDE=ALL.
- JCL: se ha mejorado la compatibilidad con la utilidad SORT para gestionar el campo BDW en la salida.
- JCL: se ha mejorado la compatibilidad con la concatenación de DD.
- JCL: se ha mejorado la compatibilidad con la secuencia de entrada.

- JCL: DSNUTILB: se ha mejorado la compatibilidad con la instrucción NULLIF ().
- JCL: INFUTILB: se ha agregado compatibilidad para descargar datos con la opción NOPAD.
- JCL: INFUTILB: se ha mejorado la compatibilidad con la fecha actual en INFUTILB.
- JCL: se han agregado comprobaciones de tamaño y existencia de archivos antes de utilizar un archivo.
- JCL: GDG: se ha mejorado la gestión de los subdirectorios para GDG.
- MQ: se ha mejorado la apertura de conexión en la implementación de JMS.
- MQ: se ha mejorado la configuración de la longitud de datos del mensaje GET para el origen de datos XA.
- MQ: se ha descompuesto el cuaderno estándar CMQV para evitar errores de compilación y usos de refactorización.
- BluSam - Se ha mejorado el soporte para las solicitudes de eliminación de conjuntos de datos inexistentes.
- Se ha mejorado la compatibilidad con la instrucción ALLOCATE.
- Se ha mejorado la solidez de la nomenclatura de TS-QUEUE.
- BatchScript - Se ha mejorado la conservación del código de retorno del paso anterior al volver a ejecutar un trabajo.
- Conjunto de datos: se ha mejorado la comprobación de la existencia del archivo cuando existe un archivo y es temporal.
- Conjunto de datos: se ha mejorado la simultaneidad a la hora de localizar los archivos GDG que se eliminarán.
- Conjunto de datos: se ha agregado la compatibilidad para obtener el tamaño de registro del conjunto de datos de GDG.
- CICS: se ha mejorado la compatibilidad con la opción SUSPENDED del comando INQUIRE TASK LIST.
- CICS: se ha mejorado la compatibilidad con LOAD SET mediante la instrucción ADDRESS OF.
- CICS: se ha mejorado REMOTESYSTEM de argumentos de CICS no administrados cuando CICS INQUIRE.
- CICS: se ha mejorado la compatibilidad con el comando GETMAIN para gestionar la opción SET con un puntero definido con la palabra clave OF.
- JICS: se mejoró la solidez del método jics XAPrepare () al añadir la verificación del estado de la transacción.

- JICS XA: se ha agregado una comprobación del estado de la transacción y se mejorado la terminación del subproceso de transacciones.
- BAC: se ha mejorado la autenticación basada en roles en el cliente y se han refactorizado y centralizado todas las llamadas a la API.
- BAC: se ha implementado una característica para bloquear el acceso público a BAC y JAC según la configuración
- BAC: se han actualizado las dependencias: Angular 17.
- BAC: integración de seguridad mejorada con. OAuth2
- BAC: se ha mejorado el DDL generado por hibernación.
- BAC: se ha mejorado el mecanismo de exportación de conjuntos de datos.
- JAC: se ha actualizado a Angular 17 e informes sobre todos los trabajos específicos de BAC (ROLE, sadmin conf, XSRF, cierre de sesión).
- COBOL: se ha agregado compatibilidad con las funciones CHAR y ORD-MIN.
- Mejorado FileFactory para mantener el tamaño del registro del catálogo a disposición del Ministerio de Defensa.
- Se ha habilitado el registro mediante MDC para las transacciones de JICS.
- Se ha mejorado SQLCA > SQLSTATE producido para procedimientos almacenados que generan conjuntos de resultados ad hoc.
- Se ha mejorado la compatibilidad con la programación de tareas relacionada con la última actualización de Spring.

AS400

Nuevas características

- Se ha agregado compatibilidad con nodos múltiples para bloqueos de registros de bases de datos mediante Redis.
- Se ha agregado compatibilidad con BINARY CHARACTER para el tipo DDS.
- CL: se ha agregado compatibilidad con la generación de archivos de informes personalizados.
- RPG: se ha agregado compatibilidad con la palabra clave RENAME en los archivos principales/secundarios.

Mejoras

- Se ha mejorado la compatibilidad con la base de datos para gestionar la columna CTID con una cláusula JOIN.
- Se ha mejorado la posición del cursor para varios DSPATR(PC).
- Se ha mejorado el registro en la excepción de lectura.
- Se ha mejorado el registro de trabajos de Quartz para incluir las propiedades de los trabajos en MDC.
- Soporte mejorado para la pantalla de ayuda AS4 00.
- CL: se ha mejorado la compatibilidad con el comando RMVJOBSCDE para aceptar números de entrada con espacios finales.
- CL: se ha mejorado la compatibilidad con el comando RMVJOBSCDE para eliminar una programación de trabajo utilizando un nombre de trabajo genérico.
- CL: se ha mejorado la compatibilidad con el comando SAVOBJ para ordenar registros por clave de tabla.
- CL: se ha mejorado la compatibilidad con el comando CPYF para establecer una nueva conexión para las consultas de base de datos.
- CL: se ha mejorado la inserción de los mensajes de consulta en los mensajes de cola con SNDPGMMSG.
- CL: se ha mejorado la configuración de la cola de trabajos para especificar la cola de trabajos predeterminada.
- CL: se ha mejorado el comando CRTPF para que admita la biblioteca QTEMP y el parámetro RCDLEN.
- CL: se ha mejorado la compatibilidad con el comando CHKOBJ: comprobar si hay particiones con la biblioteca.
- CL: RTVMGS mejorado para enviar CPF24 07 y CPF2419 cuando no se encuentra el archivo/ID.
- CL: se ha mejorado la interpretación de los parámetros de formato heredados con CPYTOIMPF y CPYFRMIMPF.
- CL: se ha agregado compatibilidad con el parámetro USRDTA de OVRPRTF.
- CL: se ha mejorado el comando CL CPYTOIMPF para establecer una nueva conexión y evitar el cierre de los conjuntos de resultados existentes.
- CL: se ha mejorado CHGDTAARA para que ya no modifique la longitud del área de datos al actualizar el contenido.
- CL: manejo mejorado ClCommand de la conexión a la base de datos.

- Se ha optimizado la Interacción entre el frontend y el back-end.
- COBOL: se ha actualizado la transformación para gestionar FILLER en cuadernos.
- Se ha mejorado la visualización de información de mensajes adicional para los mensajes personalizados enviados al frontend.
- Se ha actualizado el valor predeterminado del selector en app.component.ts.
- Se ha mejorado la división del texto en la split-dynamic-field pantalla.
- Se ha mejorado la visualización de los mensajes de error con varias escrituras seguidas de una lectura.

Capacidades transversales

Nuevas características

Se agregó soporte para la configuración dinámica del secreto del OAuth2 proveedor.

Mejoras

- Impresión: se ha mejorado la compatibilidad con los parámetros QCMDEXC para gestionar las comillas y se ha mejorado la formación de los nombres de informes
- Se ha mejorado la compatibilidad con la sintaxis delimitada activada RecordAdaptable.
- Se ha mejorado InspectBuilder el registro de errores para añadir contexto a la cadena de origen.
- DataSimplifier - mayor robustez para ByteArray la afectación.
- Registro de MDC mejorado con nuevos atributos de tiempo de ejecución.

Herramientas de modernización versión 4.1.0

zOS

Nuevas características

- Se ha agregado compatibilidad con varias transformaciones de archivos CSD
- COBOL: se ha agregado compatibilidad con la instrucción ALLOCATE de CICS.
- COBOL: se ha agregado compatibilidad con ON SIZE ERROR en la instrucción ADD CORRESPONDING.
- COBOL: se ha agregado compatibilidad con EXIT PARAGRAPH.

Mejoras

- COBOL: se ha mejorado la compatibilidad con el cuaderno -INC.
- COBOL: se ha mejorado la compatibilidad con la inicialización de FILLER.
- COBOL: se ha mejorado la compatibilidad con la comparación de valores figurativos.
- COBOL: se ha mejorado la compatibilidad con WHEN ANY en cláusulas WHEN consecutivas que carecen de bloques de códigos intermedios.
- COBOL: se ha mejorado la compatibilidad con la constante figurativa.
- COBOL: se ha mejorado la compatibilidad con el cálculo de tamaño de tipo empaquetado.
- COBOL: se ha mejorado el argumento KEEP de CICS no gestionado para SPOOLCLOSE.
- COBOL: se ha mejorado la generación con la función TEST-NUMVAL.
- COBOL: se han mejorado los argumentos de generación de Java gracias a la compatibilidad con el marco INSPECT.
- CICS: se ha mejorado la compatibilidad para definir DFHCOMMAREA.

AS400

Nuevas características

- RPG: se ha agregado un mecanismo de detección de errores para generar el DDS (incompleto) y no bloquear la generación de programas.
- Se ha agregado compatibilidad con la palabra clave de especificación de descripción de archivo INCLUDE.

Mejoras

- RPG: se ha mejorado el análisis completamente libre.
- RPG: se ha agregado solidez con la detección de errores.
- RPG: se ha mejorado la inicialización campo/DS con la palabra clave de exportación.
- RPG: se ha mejorado la operación DAO para gestionar indicadores.
- RPG: se ha gestionado el valor predeterminado de PERRCD con CTDATA.
- RPG: se ha actualizado el analizador de Free-RPG para registrar un error único por regla de análisis.
- PRTF: se ha gestionado el conflicto de nombre entre PRTF y JRXML.

- COBOL: se ha mejorado la compatibilidad con la palabra clave LIKE.

Capacidades transversales

Mejoras

- Se ha agregado solidez para la API de ErrorID
- Se ha optimizado el rendimiento para la transformación de grandes proyectos. Por ejemplo: tiempo de espera para omitir archivos bloqueados, reutilización de la clasificación de Blu Insights y mejores asignaciones de memoria.
- Se optimizó el consumo de memoria durante la transformación COBOL/PL1 .
- Se corrigió la CVE sobre tercero (jQuery y bootstrap).
- Se han administrado las opciones de timeoutParser en TC.
- Se ha mejorado la reescritura de múltiples espacios en las consultas SQL.
- Se ha mejorado el cursor de solo lectura mejorado con atributo de sensibilidad.

Notas de la versión 4.0.0

Fecha de lanzamiento: 8 de abril de 2024

Para obtener instrucciones sobre cómo migrar de AWS Blu Age Runtime 3.10.0 a 4.0.0, consulte. the section called “Migración de 3.10.0 a 4.0.0”

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en actualizar las dependencias críticas y las tecnologías compatibles, a la vez que se mejora el rendimiento en múltiples funcionalidades. Algunas funciones y cambios clave de esta versión son:

- • Actualice de Spring Boot 2.7 a 3.2.4, de Spring Core 5.3 a 6.1.5 y de Tomcat 9.0 a 10.1.17 para mejorar la seguridad, el rendimiento y la facilidad de mantenimiento mediante el uso de versiones que están siendo parcheadas y mantenidas de manera activa.
  - La carga diferida en la aplicación frontend permite crear proyectos grandes más rápidos con más de 2000 pantallas y reducir la inicialización de la pantalla de 10 s a 300 ms.
  - Compatibilidad con la visualización de DBCS en la aplicación frontend para mejorar la compatibilidad con los caracteres de doble byte y proporcionar una nueva fuente que gestione caracteres de un byte y de doble byte, impida la entrada de un solo byte en un campo de doble byte y gestione campos con una combinación de caracteres de un byte y de doble byte.

- Función de monitoreo de subprocesos para que la aplicación AS4 00 Online ejecute la aplicación AS4 00 con paralelización.
- Se mejoró el rendimiento en el contexto y RunUnit la inicialización al agregar un mecanismo configurable para preinicializar el contexto del programa, lo que reduce el impacto de cargar estructuras complejas inherentes a la complejidad heredada.

Esta versión de AWS Blu Age Runtime se probó con la siguiente pila. Es posible que otras versiones también sean compatibles.

Componente	Versión probada
Java	Java 17
Capa de presentación	Node JS 18.18
	Npm 9.8
	Angular 16.1
Capa del servicio	Spring Boot 3.2.4
	Spring Core 6.1.5
	Spring Statemachine 4.0.0
Capa de persistencia	Motor de PostgreSQL 14
	Oracle 21
Servidor de aplicaciones	Apache Tomcat 10.1.17

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 4.0.0

zOS

Nuevas características

- Se ha agregado compatibilidad con la instrucción de inclusión -INC CPYNAME.
- CICS: se ha agregado compatibilidad con la instrucción PUSH/POP HANDLE.
- COBOL: se ha agregado compatibilidad con ASSIGN TO DYNAMIC.
- Se agregó soporte para DB2 UNLOAD mediante INFUTILB.
- Se ha agregado compatibilidad con la palabra clave SEQNUM en una OVERLAY de la instrucción INREC.

Mejoras

- SORT: se ha agregado compatibilidad con caracteres especiales (paréntesis y asteriscos) en los literales de cadena de ordenación C'... '.
- SORT: se ha mejorado la compatibilidad con el argumento OUTFIL NOMATCH- (..).
- SORT: se ha agregado compatibilidad con la definición de datos de SYMNAMES.
- SORT: se ha mejorado la gestión de los argumentos TO= y LENGTH=.
- SORT: se ha mejorado la gestión de la disposición de MOD.
- SORT: se ha agregado compatibilidad con el argumento HIT=NEXT.
- Se ha mejorado ICEGENER para agregar soporte para la codificación específica de archivos de salida.
- INFUTILB: se ha agregado la compatibilidad con la cláusula WITH UR.
- INFUTILB: soporte mejorado para descargar cuando es falso. writeNullIndicator
- DSNUTILB: se ha mejorado la solidez del paso de carga cuando la palabra clave NULLIF aparece después de una palabra clave SQL opcional.
- DSNUTILB: se ha mejorado la compatibilidad para aislar el nombre de la columna.
- DSNUTILB: se ha agregado compatibilidad para cargar un archivo vacío en una tabla.
- DNSUTILB: se ha agregado compatibilidad con la disposición de MOD para el archivo DNSUTILB SYSDISC.
- IDCAMS: se ha mejorado la compatibilidad con los comentarios.
- JCL: se agregó soporte para columnas con comillas dobles. LoadTask
- JCL: se ha mejorado la gestión de consultas SQL UNLOAD relacionadas con la eliminación de los espacios en blanco.
- JCL: se ha mejorado la respuesta del script de Groovy cuando se produce una excepción en el procesamiento para garantizar un formato JSON.

- JCL: se ha mejorado la comprobación de la disposición de archivos en el caso de DISP=NEW y DISP=OLD.
- JCL: se ha mejorado la compatibilidad para gestionar múltiples referencias de generación de GDG con caracteres especiales en el nombre base de GDG.
- JCL: se ha mejorado la compatibilidad para cargar un archivo ficticio.
- JCL: compatibilidad mejorada con el parámetro tempFilesDirectory YML.
- JCL: se ha mejorado el retorno de JSON cuando es necesario aplicar escape con comillas dobles dentro de un elemento de cadena.
- JCL: mejorado FileUtils para admitir el nombre base GDG.
- JCL: programa DSNTEP mejorado para la ejecución de múltiples consultas. DB2
- Se ha agregado compatibilidad con los bean de Spring.
- Mejorado SQLConverter para evitar la rectificación de fechas incorrectas.
- JicsTimeBuilder Manejo mejorado de YYYYDDD.
- Se ha permitido acceder a archivos JAR personalizados desde groovy.
- IMS: se ha mejorado la navegación entre los registros en la implementación de la base de datos de IMS.
- IMS: se ha mejorado CBLTDLI para poder iniciar la depuración de uso del programa.
- IMS: DFSRRC00 puede pasar los parámetros de groovy al programa de backend.
- Se ha agregado compatibilidad con el comando JICS que no se invocó a través de un TransactionRunner.
- JICS: se ha mejorado el rendimiento mediante el uso de una caché configurable.
- BluSam - Se ha añadido soporte para deshabilitar el calentamiento BluSam al abrirlo para mejorar el rendimiento en conjuntos de datos de gran tamaño.
- BluSam- Se ha mejorado el comportamiento de borrar/cambiar el nombre en conjuntos de datos normales. BluSam
- BluSam - Rendimiento mejorado en las operaciones de registro.
- Se ha mejorado el simplificador de datos para los métodos que determinan si una cadena tiene un valor bajo.
- Se ha mejorado la compatibilidad para problemas de orden de clasificación y decimales empaquetados.
- Configuración mejorada de DB2 una fuente de datos principal con AWS Secrets.

- FileSystem API mejorada para mostrar el estado del archivo.
- Entrada de flujo de DynamicFileBuilder lectura mejorada con LineSeparator.
- Simplificador de datos mejorado para los métodos que determinan si una cadena tiene un valor bajo cuando se trata de un juego de caracteres 0. CUSTOM93
- SQL: se ha mejorado el procesamiento de salida del procedimiento almacenado de SQL.
- SQL: se ha mejorado la asignación lambda para varias tablas con alias.
- COBOL: se ha mejorado la compatibilidad con la instrucción LENGTH OF.
- COBOL: se ha agregado compatibilidad con la instrucción TRANSFORM.
- COBOL: se ha agregado compatibilidad con 9 nuevas funciones matemáticas.
- COBOL: compatibilidad mejorada con FUNCTION. INTEGER-OF-DAY
- COBOL: se ha mejorado la compatibilidad con el nivel 88 que implica valores figurativos.
- COBOL: se ha mejorado la transformación para la instrucción SET ADDRESS.

AS400

Nuevas características

- Se han eliminado entidades indicadoras duplicadas.
- Se ha agregado compatibilidad con caracteres DBCS.
- Se ha introducido la gestión de la palabra clave HELP para el control de registros de subarchivos.
- Se ha agregado un parámetro de configuración para alternar el uso de mayúsculas en los nombres de las columnas y dividir el contenido de las columnas de comentarios en el carácter de barra vertical.
- Se ha agregado compatibilidad para usar 0x0c como el último nibble para los campos de tipo empaquetado.
- RPG: manipulaba prototipos declarados con ExtProc («sistema»).
- CL: se ha gestionado el parámetro CLEAR del cl-command RMVMSG + introducir colas de mensajes en la memoria que no son del programa.
- CL: se han gestionado las instrucciones genéricas que se pasan a las llamadas de SBMJOB CMD().
- CL: se han agregado los comandos STRCMTCTL y ENDCMTCTL. Se ha modificado el mecanismo de bloqueo y se han limpiado las transacciones y los bloqueos.
- CL: se ha agregado compatibilidad con el parámetro RCDDLM para el comando CPYTOIMPF.

- CL: se ha agregado la gestión de los ceros de relleno en el comando SAVOBJ.
- CL: se ha agregado el manejo de las bibliotecas incluidas en el nombre completo del parámetro OBJ para RTVOBJD.
- CL: se ha agregado compatibilidad con los parámetros de comando STRDLM, STRESCCHR y RMVBLANK.
- CL: RTVMGS mejorado para enviar CPF24 07 y CPF2419 cuando no se encuentra el archivo/identificador.
- CL: se ha mejorado el comando RCVF para recibir registros de cualquier biblioteca proporcionada en el parámetro DEV.

Mejoras

- Se han modificado los valores predeterminados del ejecutor de tareas Blu4iv para permitir un mejor escalado de manera predeterminada.
- Parameterhelper modificado para convertir una lista de cadenas y en una cadena. ElementaryRangeReference
- Se ha mejorado CTID para gestionar columnas que no existen en POSTGRE.
- Se ha agregado solidez para admitir la API de espacio de usuario QUSPTRUS.
- Se agregó soporte para los espacios APIs de usuario QUSRUSAT y QUSCUSAT.
- Se ha mejorado la compatibilidad con la API de espacio de usuario (QUSPTRUS) sin código de error.
- Se ha agregado compatibilidad con la programación de tareas de CRON mediante Quartz.
- Se ha mejorado la compatibilidad con el ciclo de programación de RPG.
- Se ha mejorado la administración de las transacciones de Blu4iv.
- Se ha mejorado el bloqueo de registros de archivos bajo control de compromiso dentro de la misma transacción.
- Se ha mejorado la gestión de la inicialización de subarchivos.
- Se ha mejorado la visualización de los indicadores de desplazamiento de las líneas de mensajes.
- Se ha impedido que aparezcan ceros finales en los números enviados a través de la cola de datos.
- Se ha mejorado la pantalla de información de mensajes adicional.
- Se han mejorado las operaciones de escritura de JPA para tener en cuenta la biblioteca actual.
- Se ha mejorado el comportamiento al ejecutar programas sin ProgramJobExecutor parámetros.

- Se ha agregado la funcionalidad para pasar directamente los argumentos de los enlaces frontend a los scripts de back-end.
- Se ha mejorado la gestión de las transacciones de los metadatos de trabajos.
- CL: se ha agregado compatibilidad con el parámetro SECLVL en RTVMSG.
- CL: se ha agregado una implementación vacía para CLRLIB.
- CL: se ha mejorado la compatibilidad con CPYFRMIMPF para copiar tanto de bases de datos como de CSV.
- CL: se ha mejorado la implementación de CPYFRMIMPF para ignorar las columnas adicionales.
- CL: se ha mejorado la interpretación de los parámetros de formato heredados con CPYTOIMPF y CPYFRMIMPF.
- CL: se agregó un parámetro removeDecimalPoint para formatear valores numéricos en SAVOBJ.
- CL: se ha mejorado el comando RCVF para gestionar correctamente la condición de EOF.
- CL: RTVSYSVAL: SYSVAL de implementación = QDATETIME.
- CL: se ha modificado el comando OVRDBF para obtener el campo como nombre de tabla predeterminado.
- CL: valor de RTVJOBA no disponible para el parámetro: USRLIBL.
- CL: se han gestionado las barras diagonales iniciales en el parámetro SNDPGMMSG MSGF.
- CL: se ha mejorado la compatibilidad con los caracteres comodín en el archivo de origen del comando DSPFFD.
- CL: se ha mejorado la gestión del parámetro PGMQ en RCVMSG y SNDPGMMSG.
- CL: se ha hecho que el parámetro MSG de RTVMSG fuera opcional para que estuviera en consonancia con los documentos heredados.

Capacidades transversales

Nuevas características

- Se ha mejorado la capacidad al pasar el parámetro en la cláusula USING del cursor OPEN.
- Rendimiento: se ha mejorado la preinicialización del contexto y el ajuste del rendimiento. RunUnit

Mejoras

- Se ha mejorado el mecanismo para volcar los valores bajos del comando UNLOAD del programa de utilidad INFUTILB.

- Se ha agregado compatibilidad con la opción de esquema actual en el administrador de secretos de orígenes de datos.
- Se ha mejorado el tiempo de ejecución para no tener en cuenta los parámetros que se pasan al abrir el cursor cuando no son necesarios.
- Se ha mejorado la validación del formato numérico para campos numéricos.
- Se ha mejorado el diagnóstico de SQL en un entorno de ejecución altamente paralelo.
- Se ha introducido Unicode para la secuencia de bytes de la página de códigos (FE FD).
- DataSimplifier optimización del rendimiento: instrucciones de asignación mejoradas.
- DataSimplifier optimización del rendimiento: mejore el valor predeterminado para la inicialización de tipos numéricos para evitar un BigDecimal uso inútil.

Herramientas de modernización versión 4.0.0

zOS

Nuevas características

- Se ha agregado compatibilidad con la gestión de Abend PROGRAM.
- Se ha mejorado la compatibilidad para generar conjuntos de datos de AIX.
- COBOL: se agregó soporte para la cláusula JUSTIFIED en ALPHANUMERIC/ALPHABETIC/ GRAPHIC los campos.

Mejoras

- Se ha mejorado la gestión de atributos de PURGETHRESH para las definiciones de recursos de TRANSCLASS.
- Se ha mejorado la compatibilidad con la definición de datos y la instrucción MOVE.
- CICS: se ha mejorado la compatibilidad con el comando DELAY en la opción MILLISECS.
- Se ha mejorado la asignación lambda de SQL para varias tablas con alias.
- Se ha mejorado la compatibilidad con la búsqueda de campos principales.
- Se ha mejorado el conjunto de sqlstate de SQLCA para las operaciones COMMIT y ROLLBACK.
- COBOL: se ha mejorado el análisis comentando los párrafos obsoletos.
- COBOL: se ha mejorado la compatibilidad con la cláusula REPLACING.
- COBOL: se ha agregado compatibilidad con las funciones matemáticas ASIN ACOS LOG TAN.

- COBOL: se ha agregado compatibilidad con varias instrucciones AFTER en PERFORM VARYING.
- COBOL: se ha mejorado la compatibilidad con los campos RENAMES (nivel 66).
- COBOL: se ha mejorado el método LENGTH OF para obtener la longitud de un índice específico en un campo de matriz.
- COBOL: se ha agregado compatibilidad con varias cláusulas AFTER en instrucciones PERFORM VARYING.
- COBOL: se ha mejorado la compatibilidad con la cláusula RENAMES.
- COBOL: se ha mejorado la compatibilidad con la palabra clave PICTURE.
- COBOL: se ha mejorado la compatibilidad con el análisis de campos de nivel 88.
- COBOL: se ha mejorado la condición dependiente goto con elementos de datos de tablas.

AS400

Nuevas características

- Se ha agregado la funcionalidad para pasar argumentos a las llamadas Java de frontend directas.
- CL: se ha mejorado la generación de %SST, incluida la compatibilidad con *LDA con CL→Java.
- RPG: se ha agregado compatibilidad con un registro descrito por el programa para archivos DISK.

Mejoras

- Se ha mejorado el archivo de visualización, se han resueltos los campos de referencia con la palabra clave REFFLD.
- Se ha mejorado la compatibilidad con la palabra clave SETOF-CSRLOC del archivo de visualización.
- Se han eliminado los archivos del control de compromiso después de cerrar.
- Se ha garantizado un comportamiento coherente para las operaciones simultáneas de lectura y escritura en una tabla cuando las realiza el mismo programa.
- Gestionó la asignación a la subcadena de SizePrefixedAlphanumericType.
- Se ha gestionado el paso de la estructura de datos al procedimiento con un parámetro de cadena de longitud variable.
- Se ha mejorado la retención de valores numéricos no válidos en el evento OnBlur y la creación de agentes de escucha de eventos solo para los campos válidos.

- Se han mejorado los mensajes de error en las pantallas y el resaltado de los campos con entradas no válidas.
- Se ha mejorado la gestión de los campos de pantalla condicionados a los indicadores.
- Se ha habilitado el desplazamiento con la rueda del ratón.
- Se ha agregado compatibilidad con las teclas de función de la pantalla Ayuda.
- Soporte mejorado para texto largo en el split-dynamic-field componente.
- Se ha mejorado la gestión de los archivos LF de varios registros al cambiar el nombre de los registros.
- CL: se ha mejorado el comando RTVJOBD para gestionar archivos LF (vistas).
- CL: se ha mejorado el comando OVRDBF al usarse en un LF de varios registros.
- RPG: se ha gestionado el escenario en el que el procedimiento define una variable con el mismo nombre que el parámetro que se ha cambiado de nombre.
- RPG: se ha mejorado la gestión de *ZEROS al inicializar un binaryInteger con signo.
- RPG: se ha mejorado la gestión de los punteros a variables (de referencia) no locales.
- RPG: manejo mejorado de las declaraciones de ELSEIF después IFxx de las declaraciones.
- RPG: se ha agregado compatibilidad con los campos definidos con LIKE en el prototipo.
- RPG: se ha mejorado la compatibilidad con la palabra clave LIKE de un campo creado por LIKEREC.
- RPG: se ha mejorado la generación de los operadores con valores figurativos.
- RPG: se ha mejorado el análisis de la expresión matricial xxx(*) y se ha admitido en %lookup.
- RPG: código de LookUp operación mejorado con indicadores altos e iguales (o bajos e iguales).
- RPG: se ha mejorado el análisis de formato libre.
- RPG: se ha mejorado el análisis de las constantes con nombre de l-card que siguen los formatos de registro de l-card.
- RPG: se ha mejorado la compatibilidad con los tipos INTEGER y UNSIGNED.
- COBOL: se ha agregado compatibilidad con la cláusula INDIC del formato DSPF en la instrucción COPY DDS.
- COBOL: se ha mejorado la gramática de las instrucciones DISPLAY y ACCEPT para desbloquear la transformación y la generación.
- COBOL: se ha agregado compatibilidad con archivos DISK.
- COBOL: se han mejorado los programas de compatibilidad para archivos de visualización DDS.

- COBOL: se ha agregado compatibilidad con la cláusula LIKE.
- COBOL: se ha agregado compatibilidad con el archivo DISK descrito por el programa.
- COBOL: se ha agregado compatibilidad con el nombre de archivo con sufijo.

Capacidades transversales

Nuevas características

- Se ha gestionado la carga diferida de los componentes de mapa de proyectos web.

Mejoras

- Se ha mejorado la generación en Java de los parámetros de indicadores SQL.
- Capacidad mejorada para manejar las variables involucradas en la DB2 sentencia SET.
- Se ha mejorado la emisión de error al final del cursor recuperado cuando la salida es una matriz de una sola entidad.
- Se ha administrado la ruta en Linux.
- Data Migrator administra las vulnerabilidades y elimina las dependencias no utilizadas.

Notas de la versión 3.10.0

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en las actualizaciones y mejoras básicas de todo el producto, con el objetivo de aumentar el rendimiento y la solidez en todos los pasos de transformación y ejecución. Algunas funciones y cambios clave de esta versión son:

- Actualización de la versión de Java 8 a Java 17, lo que aumenta la seguridad y el rendimiento y permite a los clientes implementar y ejecutar aplicaciones implementadas en un lenguaje más moderno y utilizar versiones recientes de marcos de terceros.
- Compatibilidad adicional para administrar grandes espacios de memoria compartida entre usuarios o trabajos y almacenar datos reutilizables tras el reinicio de la aplicación o la instancia.
- Acceso más rápido a grandes conjuntos de datos en Blusam mediante un mecanismo de paginación que permite recuperar un subconjunto de registros de forma incremental.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.10.0

Este tiempo de ejecución se basa en Java17, Spring2.7 y Angular16.

zOS

Nuevas características

- Blusam: se ha agregado compatibilidad con grandes conjuntos de datos a través de un mecanismo paginado en el que los índices se almacenan y se cargan mediante páginas.

Mejoras

- DataUtils.compare mejorado para gestionar la conversión de cadena a número de menor prioridad
- Se ha añadido soporte para comprobar si ByteRange se crea un no con valores incorrectos mediante la propiedad DataSimplifier de YML. byteRangeBoundsCompruebe
- RemoveSOSI () mejorado para admitir la inicialización de a GraphicAlphanumericType con un carácter vacío
- Se ha agregado solidez para la operación del trabajo y una lectura segura del estado de GDG.
- Blusam: Se agregó soporte para borrar Ehcache de los conjuntos de datos de Blusam mediante un nuevo método llamado .removeCache () CoreBluesamManager
- BluSam: se ha mejorado el comportamiento de eliminar/cambiar el nombre en los conjuntos de datos de BluSam normales.
- Redis: se ha mejorado la compatibilidad para desbloquear conjuntos de datos y borrar el bloqueo de registros.
- JICS: se ha mejorado el mensaje de error para las solicitudes con errores.
- JCL: se ha agregado compatibilidad con la concatenación de variables ControlM basada en carácter de punto.
- JCL: se ha agregado compatibilidad con la escritura de ADVANCING (ADV) para archivos de GDG.
- JCL: se ha mejorado la compatibilidad con el número de generación actual después de eliminar todos los archivos de GDG.
- JCL: se ha mejorado la compatibilidad con la lectura de rdw/recordSize del catálogo al crear conjuntos de datos.
- JCL: se agregó soporte para actualizar el objeto de recurso (from AbstractSequentialFile) al abrir el archivo con el tamaño del registro de salida de datos

- JCL: se ha mejorado el rendimiento de IDCAMS.
- JCL: se ha mejorado la compatibilidad con PRINT STATEMENT al agregar CHAR como alias de CHARACTER.
- SORT: se ha mejorado la compatibilidad con la operación de copia de un conjunto de datos de longitud fija de Blusam a un conjunto de datos con longitud variable.
- SORT: se ha mejorado la gramática de clasificación para gestionar algunas instrucciones específicas.

AS400

Nuevas características

- Se agregó soporte para los espacios de usuario y sus relacionados APIs
- Se ha agregado compatibilidad con el parámetro TOMSGQ de SNDPGMMSG y se han implementado colas de mensajes.
- CL: se ha agregado compatibilidad con los parámetros FILE y SPLFNAME para el comando OVRPRTF.
- CL: se ha agregado compatibilidad con la gestión de las bibliotecas de la tabla de particiones correspondiente con el comando CPYF.
- CL: se ha agregado compatibilidad para gestionar el comando CHGCURLIB y tener en cuenta la biblioteca actual al crear consultas.
- CL: se ha agregado compatibilidad para gestionar el comando cl como parte del stacktrace de llamada.

Mejoras

- Se ha mejorado MessageHandlingBuilder para gestionar mejor la entrada de seguimiento de la pila de llamadas
- Se ha mejorado la ejecución paralela de la característica contextPreconstruct.
- Se han mejorado los atributos de visualización cuando SFLINZ crea un registro.
- Se ha mejorado SAVOBJ para permitir la gestión de varios archivos de salida.
- Se mejoró el manejo de los programas geniales al agregarlos programCallStack cuando se invocan desde un programa Java
- Se ha mejorado la detección del posicionamiento superior del modal de ayuda.

- Se ha mejorado la funcionalidad de toPgmQ al proporcionar el parámetro toMsgQ para SNDPGMMSG.
- Se ha mejorado la recuperación de mensajes predefinidos y la funcionalidad del cargador de mensajes.
- Se ha mejorado la gestión por parte de CPYTOIMPF de los caracteres delimitadores en el contenido.
- Se ha mejorado el bloqueo de liberación en el registro READ.

Capacidades transversales

Nuevas características

- Se ha agregado una traducción para los mensajes del sistema en el frontend.
- Se agregó un nuevo método ExecutionContext para devolver la pila de llamadas del programa
- Establezca un separador de líneas (para simplificar los datos) independientemente del entorno real.
- Se ha agregado la posibilidad de configurar la ruta JSON del modelo SQL

Mejoras

- Se mejoró el método de comparación DataUtils. compareAlphInt() cuando se trata de relleno
- Creación de un indicador para permitir un comportamiento personalizado en caso de excepción en las consultas de cursor
- Se ha mejorado la conversión gráfica de db de LOWVALUES.

Tercero

- Actualización para mitigar los errores CVE-2024-21634, CVE-2023-34055, CVE-2023-34462, -JAVA-ORGSPRINGFRAMEWORK SECURITY-5905484, CVE-2023-46120, CVE-2023-6481, CVE-2023-6378, CVE-2023-5072) IN1

Herramientas de modernización versión 3.10.0

zOS

Mejoras

- COBOL: se ha agregado compatibilidad con la función ABS.
- JCL: se ha mejorado el alcance de variables: asociado a STEP en lugar de a JOB.
- Se ha mejorado la inyección de parámetros de cursor para valores bajos/altos.
- Se ha mejorado el análisis de CSD, especialmente en el caso de las TRANSACTIONS remotas.

AS400

Mejoras

- Se ha eliminado la comprobación de espacios en blanco del indicador de nivel de control.
- Se ha agregado compatibilidad con nombres externos para las palabras clave IMPORT/EXPORT.
- Se ha agregado compatibilidad con %LEN en los campos.
- CL: se ha agregado compatibilidad con nuevos operadores para el lenguaje CLLE.
- CL: se ha agregado compatibilidad con IF anidadas.
- COBOL: se ha mejorado la gestión del comando START cuando se usa con varias teclas.
- DSPF: se ha mejorado la gestión de la posición del cursor con un número de registro.
- DSPF: se ha mejorado el formato de los campos numéricos con signo, los campos solo numéricos y los campos con gran escala.
- DSPF: se ha mejorado la definición del título para la pantalla de ayuda general.
- DSPF: se ha mejorado la compatibilidad con las especificaciones de entrada/salida
- DSPF: se ha mejorado la gestión de los separadores de agrupación durante la validación de los campos numéricos.
- Se han mejorado los registros DDS/salida de asignación.
- Se ha mejorado la capacidad de la palabra clave REFFLT del archivo de impresora para resolver los campos de referencia.
- RPG: se ha mejorado la compatibilidad con las instrucciones TOTALMENTE gratuitas.
- RPG: se ha mejorado el análisis de condiciones y se ha agregado compatibilidad para gestionar CABXX sin etiqueta de resultado.
- RPG: se ha mejorado la gestión de las especificaciones de entrada de los campos numéricos.
- RPG: gestión mejorada de las llamadas a los procedimientos dentro IF/ELSEIF/WHEN de las condiciones
- RPG: se ha mejorado la gestión del comando READ cuando se llama en un archivo dspf.

- RPG: se ha mejorado la compatibilidad con los archivos que hacen referencia a un DDS inexistente.
- Se ha mejorado la gestión de REFFLD cuando se pasa un nombre en formato de registro físico.
- Se ha agregado compatibilidad para usar return como nombre de columna de base de datos.

Capacidades transversales

Nuevas características

- Oracle: se hizo posible la definición de usuarios distintos de SYS para almacenar las funciones integradas.

Mejoras

- Versión de Java actualizada de v8 a v17
- Se ha mejorado la condición SQL con el nombre de la columna de Clúster.
- Se ha agregado compatibilidad con las cláusulas ORDER BY desde la vista.

Notas de la versión 3.9.0

Esta versión de las herramientas de tiempo de ejecución y modernización de AWS Blu Age se centra en múltiples mejoras transversales en todo el producto, con el objetivo de aumentar el rendimiento en arquitecturas de alta disponibilidad, junto con nuevas capacidades para llevar la ejecución de los trabajos al siguiente nivel. Algunas funciones y cambios clave de esta versión son:

- Actualización de la versión de Angular 13 a Angular 16, lo que aumenta la seguridad y proporciona acceso a nuevas funciones que mejoran el rendimiento de las aplicaciones en línea de los clientes.
- Añada la compatibilidad con funciones multitarea a la versión AS4 00, con la ventaja principal de que los trabajos pueden enviar mensajes de consulta de forma sincronizada entre ellos, lo que permite disociarlos en los trabajos modernizados.
- Mejoras en el rendimiento del uso de Redis, como la optimización del conjunto de conexiones, la alta seguridad de la conexión y la mejora del mecanismo de bloqueo de los conjuntos de datos.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.9.0

zOS

Nuevas características

- Programa de clasificación: entradas VSAM actualizadas con longitud fija
- JHDB DB: incorporación de tiempo de espera configurable

Mejoras

- Compatibilidad mejorada con el separador de líneas para la transmisión si se usa en la concatenación de archivos
- Compatibilidad mejorada para abrir archivos secuenciales concatenados. Inicialice después DataSetIndex de abrir el archivo
- Compatibilidad mejorada con el separador decimal virtual cuando a NumericEditedType se ve afectada por un valor numérico
- Soporte mejorado para NumericEditedType valores no negativos
- IDCAMS: Las tarjetas SYSIN ahora se leen con la propiedad de «codificación» definida en .yml application-utility-pgm
- IDCAMS: se ha actualizado la gramática para admitir el argumento FILE(..) en la instrucción DEFINE CLUSTER
- INFUTILB: se ha agregado compatibilidad con el argumento DFSIGDCB para anular los parámetros DCB de DD SYSREC
- INFUTIL: se ha mejorado la compatibilidad para el parámetro “DFSIGDCB YES”
- Se ha mejorado SPLICE para gestionar archivos de entrada de gran tamaño
- DFSORT: se ha mejorado la gestión de los campos de comentarios
- DFSORT: se ha agregado compatibilidad con el formato numérico de formato libre (firmado o no firmado) (SFF/UFF)
- SORT: se ha agregado compatibilidad con el análisis de las sentencias OPTION PRINT y OPTION ROUTE
- SORT/ICEMAN: se ha agregado compatibilidad para operaciones de división incluidas (campo con operador DIV)
- Se ha mejorado la compatibilidad para CICS READ mediante una clave genérica

- Se corrigió la función StringUtils .chargraphic para eliminar el SOSI de un tipo de gráfico
- Mejore el rendimiento en. DataUtils isDoubleByteCodificación
- JCL: se ha mejorado la compatibilidad con el modo de disposición KEEP para un conjunto de datos temporal. El sistema cambia la disposición a PASS
- JCL: gestiona los parámetros DCB de forma dinámica
- JCL: salidas de SUM FIELDS mejoradas para valores incorrectos
- JCL: CommonDDUtils: :getContent ahora busca el RecordSize en el catálogo
- JCL: lee los atributos rdw/recordSize del catálogo al crear el conjunto de datos
- JCL: se ha agregado compatibilidad con DCB=.MYDD para copiar los parámetros DCB de un DD en otro en el mismo paso de trabajo
- JCL: mejora en el sistema de herencia de tamaño de registros
- JCL: se ha agregado un bloqueo de conjunto de datos exclusivo (Redis)
- Redis: se ha agregado compatibilidad SSL para el modo independiente
- Redis: se ha agregado el recuento sincronizado de Redis con bloqueo
- Redis: parámetros de grupo compatibles con el bloqueo de Redis
- Redis: actualización de metadatos optimizada con Redis
- Redis: compatibilidad mejorada con clústeres de Redis
- Mejora en los bloqueos abiertos con modo IO
- Mejora en el rendimiento de los bloqueos de conjuntos de datos y el borrado de bloqueos no utilizados
- Mejora en la ruta del conjunto de datos al anular el registro del archivo
- Mejora en la invalidación de la caché de la ventana de recuperación previa
- Se ha agregado compatibilidad para el uso seguro de subprocesos por proveedores de orígenes de datos de utilidades
- Mejora en la comprobación de nulidad de datasetState
- Mejora en la compatibilidad de la no reapertura de conjuntos de datos ya abiertos
- Mayor robustez para la operación final del trabajo
- Mejora en la compatibilidad para los índices y el orden de las claves, lo que permite duplicarlos
- Mejora en la compatibilidad para el orden de serialización de las listas de omisión
- Se ha agregado compatibilidad para la característica de descarga de depuración para ayudar a diagnosticar problemas con el orden de los índices

- Mejora en la compatibilidad con la actualización de metadatos
- Se ha mejorado la compatibilidad con la lectura por lotes de Blusam.

AS400

Nuevas características

- Crea un nuevo registro application-context
- Compatibilidad para palabras clave DSPF CLRL(NO) Compatibilidad con la supervisión de bloqueos de registros
- Support for keyed DataQueue
- Compatibilidad con mensajes INQUIRY para trabajos por lotes
- Se agregó soporte para el archivo de impresora descrito por el programa para 00 COBOL AS4
- Gestión del comando RMVJOBSCDE cl
- Mejora para RUNSQL/DLYJOB
- CHKOBJ: generación de un código de error heredado para el parámetro LIB
- SNDPGMMSG: compatibilidad con parámetros de cadena
- RTVDTAARA: subcadena mejorada en LDA
- DSPFD: se ha agregado un parámetro FILE compatible para el nombre de archivo específico
- RUNQRY: compatibilidad para archivos sql en QRY PARAM
- CRTDUPOB: compatibilidad para copiar los datos entre áreas de datos
- SBMJOB: Convierte la instrucción en uso JobQueueManager
- OPNQRYF: se ha agregado compatibilidad para la biblioteca Qtemp
- CRTDUPOBJ: lógica mejorada para copiar el contenido de la partición
- CRTDUPOBJ: se ha agregado compatibilidad para Qtemp para las vistas
- RTVSYSVAL: compatibilidad para el valor SYSVAL, QDATFMT en el comando CL
- CHKOBJ: se ha agregado compatibilidad para OUTQ
- RTVJOBA: compatibilidad con el parámetro SWS
- SNDPGMMSG y RCVMSG: compatibilidad de parámetros adicionales como MSGF, MSGFLIB, MSGDTA, MSGTYPE, KEYVAR, MSGKEY, MSGID

Mejoras

- Mejora de la compatibilidad con tarjetas WORKSTATION I/O
- Mejora de la gestión del mensaje configurado superpuesto al mensaje anterior
- Compatibilidad con información de mensajes adicional en array-messageline
- Mejora del acceso al contenedor de matrices independiente dentro de EVAL, SortA y figuratives
- Mejore la DAOs limpieza cuando finalice la solicitud en línea
- Se ha agregado compatibilidad para formatos de fecha adicionales y se ha mejorado la gestión de las entradas de cadenas
- Se mejoró el manejo CVTDAT de SYSVAL al agregar valores del sistema, la clase auxiliar Decode y compilar parámetros a partir del comando CL SbmJob
- Se ha eliminado el paquete com.netfective.bluage.gapwalk.rt.blu4iv del escaneo de componentes gapwalk-cl-command
- Se ha mejorado la compatibilidad de mensajes predefinidos para la API de cola de mensajes
- Se mejoró el soporte para registros escritos en otro programa retrieveSubfileRecord
- Se ha mejorado la compatibilidad de mensajes inmediatos para la API de cola de mensajes
- Se ha mejorado la gestión del área de datos local al enviar un trabajo
- Se inicia JobQueues automáticamente cuando se inicia el servidor
- Utiliza la configuración de applicationContext para descodificar los parámetros de SBMJOB
- Mejora de los mensajes de error proporcionados por el sistema
- Permite a RTVMSG buscar archivos .properties en subdirectorios anidados
- Gestiona el restablecimiento de entidades vinculadas a punteros incorrectos o no válidos
- Se ha mejorado MessageHandlingBuilder la visualización del MsgID y el MsgFile nombre como cadenas para RCVMSG
- Método de withMsgFile nombre mejorado de la API de cola de mensajes
- Mejora en el mecanismo de bloqueo del área de datos
- RTVMBRD: compatibilidad para minúsculas y mayúsculas para el parámetro FILE
- CRTDUPOBJ: mejora en la gestión de las vistas
- CPYTOSTMF: mejora en la gestión de la conexión
- CPYF: mejora en la gestión del nombre del directorio al copiarlo desde un archivo plano
- RCVF: gestión correcta de los parámetros DEV/RCDFMT y la transformación de RCDFMT para groovy y java

- RCVF: gestiona las llamadas posteriores y evita restablecer el cursor
- CPYF: se ha agregado compatibilidad para escribir desde archivos planos
- CRTDUPOBJ: se ha agregado la gestión del nuevo obj con la biblioteca Qtemp
- CHGDTAARA: se ha aumentado la longitud máxima del área de datos de 256 a 2000
- SAVOBJ: se asegura de que los registros guardados estén en orden de inserción
- RTVDTAARA: valores recuperados (no deben recortarse)
- CHKOBJ: devuelve los mensajes de supervisión correctos cuando el miembro no existe
- RTVDTAARA: se ha agregado compatibilidad para la subcadena LDA
- RTVDTAARA: devuelve espacios en blanco de hasta la longitud de la variable especificada en el parámetro RTNVAR
- RTVDTAARA: admite parámetros enteros para el inicio y la longitud y admite el formato de transformación más reciente
- CHGDTAARA: se ha agregado compatibilidad para parámetros que incluyen límites inferior y superior
- CHKOBJ: gestiona el valor VIEW del tipo de objeto de parámetro
- CHKOBJ: el resultado se establece en true independientemente del miembro si la vista existe

Capacidades transversales

Nuevas características

- Gestiona la generación de informes en archivos .txt
- Se ha agregado la propiedad de origen de datos currentSchema XA al administrador de secretos
- Se ha agregado la propiedad YAML database.cursor.raise.already.opened.error para permitir que el marco genere el error 502 de SQLCODE cuando se esté abriendo un cursor que ya esté abierto.

Mejoras

- Se agregaron pompones Gapwalk a los empaques de AWS Blu Age en Amazon EC2
- Utiliza el nuevo paradigma de controlador de señales de forma predeterminada.
- Se ha agregado compatibilidad para el bloqueo cuando la disposición sea MOD u OLD
- Se ha agregado una caché para almacenar los patrones de fecha y hora de la base de datos
- Función de comprobación mejorada de PackedType

- Mejore las funciones DataUtils de .setTo para registros con VariableSizeArray
- Gestiona la opción MQ SYNCPOINT como si fuese una unidad de ejecución
- Marco habilitado para configurar SQLCODE en una transacción de reversión
- Se ha agregado el nombre de clase de controlador automático según el secreto de la clave del motor
- Tiempo de espera del programa/transacción
- Se restaura la posición del cursor después de la reversión al acceder al cursor

Tercero

- Actualice SnakeyAML, Redisson y Amazon SDK, elimine YamlBeans (mitigue CVE-2023-24621, CVE-2023-42809, CVE-2023-44487)

Herramientas de modernización versión 3.9.0

zOS

Mejoras

- Compatibilidad mejorada para XML-TEXT como origen para el destino de tipo String
- Se ha mejorado el flujo de trabajo de STM a UML para admitir el patrón de división X/(Y/Z)
- JHDB DB: acepta la llamada ROLLBACK antes de cualquier actualización de la base de datos
- JHDB DB: acepta el ROLLBACK incluso si la transacción está terminada (NOP)
- JCL: función de validación por pasos mejorada
- SORT: gestiona la función SUM con valores negativos decimales de zona
- COBOL: agrega soporte para el escape de comillas simples o dobles en literales de cadena

AS400

Mejoras

- Se ha mejorado la función integrada %editc, que gestiona el código de edición X agregando ceros a la izquierda.
- Se ha mejorado la gestión del valor inicial de los campos solo de entrada
- Se han agregado teclas de acción para facilitar los cuadros de diálogo

- Registro de pie de página de la tabla dinámica que aparece en la parte inferior
- Se gestiona el comando START sin KEY PHASE para los archivos que especifiquen una RECORD-KEY real
- Se agregó un valor predeterminado para los tipos float y NumberUtils: :pow
- Se ha agregado compatibilidad para definir una variable usando LIKE(IN)
- Se ha actualizado la gestión de bucles FOR para permitir la omisión de elementos opcionales
- Se ha actualizado el análisis de RPG para asociar los registros con el nombre de la matriz CTDATA
- Manejo mejorado de los indicadores para las declaraciones CABxx
- Compatibilidad del parámetro opcional de la palabra clave COMMIT
- Compatibilidad mejorada con la palabra clave FORMAT en LF
- Código de operación LOOKUP administrado con indicadores altos e iguales (o bajos e iguales)
- Se gestiona el nombre de clave PF declarado entre comillas dobles
- Se ha mejorado la gestión de EDTCDE X para no suprimir los ceros a la izquierda
- Se ha mejorado la compatibilidad con MSGCON en archivos de impresora que no generan etiquetas sin nombre
- El CONTENT del campo lo comparten varias estructuras de datos
- Se gestiona el parámetro ERRSFL en combinación con SFLMSG/SFLMSGID
- Se ha mejorado el código principal antes del ámbito de declaración de proceso del rpg libre completo
- Se ha agregado la especificación de control condicionado al análisis
- Se ha mejorado el soporte para el método setErrSfl () en dataholdermapper
- Se ha mejorado la resolución de tipos para las variables creadas internamente
- Se ha mejorado la compatibilidad para el código de operación Z-ADD
- Se ha mejorado la gestión del campo constante con un valor DFT
- Se ha mejorado la compatibilidad con campos enteros dentro del estado del programa ds
- Se gestiona la asignación de indicadores en los parámetros ENTRY
- Se ha mejorado el filtro de palabras clave propagadas a través de la palabra clave ref/reffield
- Estructura de datos sin nombre compatible DataArea
- Se ha mejorado la gestión del tipo de datos del puntero

- Los elementos gestionados de la matriz utilizados para definir variables con compatibilidad con la palabra clave LIKE admiten el acceso a la matriz en el campo de salida
- Se ha mejorado la compatibilidad para campos numéricos firmados, que solo muestran dígitos
- Compatibilidad con la relación lógica en la tarjeta O
- Caso de prueba para %CHAR en formato alfanumérico
- Compatibilidad con la entidad principal de la palabra clave de especificación de control
- EDTCDE con dos parámetros en el archivo de impresión
- Análisis de juegos de FullFree rol mejorado
- Se ha mejorado la tabla dinámica para garantizar que el pie de página esté colocado correctamente
- Se ha agregado compatibilidad para inicializar tipos numéricos con TODAS las constantes figurativas
- Se ha mejorado la gestión de varios archivos lógicos RPG que hacen referencia al mismo archivo físico
- Se ha mejorado la detección de campos modificados en una pantalla moderna
- Sincronización modal con campos dinámicos
- Se ha mejorado la gestión del campo numérico firmado de solo salida
- Mejora de la compatibilidad con tarjetas WORKSTATION I/O

Capacidades transversales

Nuevas características

- Herramienta de migración de datos: se agregó la propiedad ebcdicFilesWith VarcharIn VB para permitir tener en cuenta la longitud de 2 bytes de VARCHAR al leer los bytes
- Se ha implementado una API común para registrar los errores
- Implementación BluAgeErrorDictionaryUtils y uso de una API común para registrar errores y/o información en COBOL2 Model, Builder, Definitions2Model y RPGCycle FieldsProcessor
- Se ha mejorado la gramática SQL para admitir diferentes definiciones de cláusulas de aislamiento

Mejoras

- Se ha actualizado la versión de Angular a la v16
- Angular: se ha mejorado la versión ajv de la 6 a la 8.9

Tercero

- Se ha actualizado Groovy a la versión 2.4.15

Notas de la versión 3.8.0

Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en múltiples mejoras transversales en todo el producto para mejorar su calidad y seguridad, además de mejorar el rendimiento del almacenamiento en caché y la unificación de los soportes de comandos en una sola distribución. Algunas funciones y cambios clave de esta versión son:

- Actualización de la versión de Spring 2.5 a Spring 2.7, lo que aumenta el soporte de mantenimiento, el rendimiento y la seguridad de la plataforma.
- La unificación de más de 82 comandos CL es compatible con la over-the-counter distribución para facilitar el uso y el despliegue de aplicaciones modernizadas que anteriormente utilizaban scripts CL.
- Están APIs disponibles nuevas opciones para operar e interactuar mejor con los conjuntos de datos de BluSAM, como la importación integrada al servicio gestionado y la capacidad de enumerar la información de los metadatos de los conjuntos de datos.
- Mejoras en el rendimiento y ampliación del uso de Redis, incluida la disponibilidad en modo clúster, la recuperación de datos de alta disponibilidad y la estandarización del uso de datos secretos.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.8.0

zOS

Nuevas características

- Manejar la definición clave como una cadena para DynamicFileBuilder
- DFSORT: Se agregó soporte para varios elementos en la inicialización gramatical de OUTFIL TRAILER1 + DFSORT
- DDUtils Herramienta común: gestionar el tamaño de los registros en los datos in-stream
- Archivo indexado: gestión de la opción GENKEY

Mejoras

- Servicios de carga de BluSAM externalizados en un contenedor separado
- Se ha agregado compatibilidad para configurar la ubicación a fin de almacenar archivos temporales
- Mecanismos de caché compartida mejorados para casos con varios nodos
- Uso de caché compartida: los IDCAMS verifican la optimización
- Mejora de la inyección de ROWID para la selección integrada
- JCL: cada procedimiento de trabajo continuo se genera ahora en un archivo Groovy distinto
- Garantice una cobertura del card-demo-v 2% en las tarjetas IDCAMS JCL
- BluSAM: evite duplicar el calentamiento cuando utilice varias instancias
- Menor consumo de memoria al hidratar la caché
- Compatibilidad con la configuración de Jedis Pool
- Se ha agregado un separador de líneas a la transmisión si se usa en la concatenación de archivos
- Compatibilidad con tarjetas EBCDIC + bloquear comentarios (/*.../) en la utilidad IDCAMS
- Consulta de compatibilidad con bases de datos: compatibilidad con cadenas de doble byte en la conversión del nivel 49 a SQL
- Gramática DFSORT: implementa 17 instrucciones de control más la integración de 2 de ellas (OMIT/INCLUDE)
- Mejora de las columnas gráficas, búsqueda de INFUTILB
- Compatibilidad para leer archivos con tabla de tamaños variables
- Support para ZonedType con signo nibble donde el primer bit del último byte es 'E'
- DFSORT/ICETOOL añade compatibilidad con el argumento NOMATCH =(..) si un registro no coincide con ninguna de las constantes de búsqueda de CHANGE
- Compatibilidad con clústeres de Redis
- Gestión del estado del trabajo (fallido) según el código de salida groovy
- Compatibilidad mejorada con CICS SYNCPOINT ROLLBACK.
- Ventana de recuperación previa para optimizar el uso de la caché de Redis
- JCL/GROOVY: hereda la propiedad isRDW del conjunto de datos del paso anterior cuando DISP =(, PASS)
- Gestión de copias parciales de datos con una matriz de tamaño variable

AS400

Nuevas características

- Compatibilidad con tarjetas de E/S para archivos de visualización
- Soporte para información de mensajes adicional para las palabras clave DSPF ERRMSGID y CHKMSGID
- Compatibilidad con múltiples mensajes de error en la pantalla de la interfaz
- Se agregó o mejoró el soporte de 82 comandos CL dentro de la gapwalk-cl-command aplicación

Mejoras

- Compatibilidad mejorada para DELETE y READ bajo control de compromiso
- ConvertDate dentro de la versión integrada %dec
- Encabezados de seguridad XSS forzados
- Mejora de la solidez y la coherencia de la generación de STM (mejor gestión de: línea de continuación en formato rpg libre, comas para la parte decimal, bloques de formato libre en la definición o declaración)
- Generación mejorada DataHolderMapper
- Mayor robustez y margen de cambio en DataAreaFactory
- Se ha mejorado el cambio de enfoque en la tecla de tabulación
- Rendimiento mejorado en la generación de informes de Jasper
- Visualización de decimales mejorada con un relleno de 0
- Compatibilidad mejorada para el campo ROW/COL en INFDS
- Mejora de la compatibilidad con los campos modificados desde la pantalla
- Se han agregado captadores para el nombre y la ruta del informe generado
- Se ha mejorado la longitud de Dataqueue
- Configuración automática mejorada de Job Queues para adaptarla a los nuevos estándares de Spring Boot 2.7
- Actualizaciones de estaciones de trabajo mejoradas para múltiples sesiones simultáneas

Capacidades transversales

Nuevas características

- Compatibilidad con tolerancia a datos no válidos para empaquetados
- Se ha agregado paginación o filtrado para enumerar los puntos de conexión del conjunto de datos

Mejoras

- Estrategia mejorada de transformación de consultas de ORACLE en la comparación de columnas con cadenas vacías
- Manejo de BLOB DB2 con los programas de utilidad DSNTEP e INFUTILB. Los BLOB ahora están modernizados DB2 a postgres de tipo BYTEA.
- Mejora de la eliminación del último elemento del cursor
- Compatibilidad con la eliminación de archivos RRDS mejorada
- AWS Se ha mejorado el rendimiento secreto de Blusam
- Gestión mejorada de las conexiones de bases de datos en el marco SQL
- Claves de administrador secreto estandarizadas AWS de múltiples fuentes de datos
- Correcciones de regresión del rendimiento
- Función de comprobación mejorada para PackedType
- Manejo mejorado de LOW-VALUE para PackedType
- Paquete de seguridad Spring mejorado para la conexión de Cognito
- No se aplica la codificación y decodificación de puntos de cambio de código en las bases de datos de destino DB2

Tercero

- Actualización de Spring Boot de 2.5 a 2.7

Herramientas de modernización versión 3.8.0

zOS

Nuevas características

- JCL: gestión del flujo con retorno de carro “\ r”

Mejoras

- Se ha mejorado el registro para evitar la división por cero al modernizar una cláusula DIVIDE con ON SIZE ERROR
- JCL: compatibilidad mejorada para llamar a un procedimiento dentro de un procedimiento
- Compatibilidad con la palabra clave OF en el comando FORMATTIME CICS cuando hay campos ambiguos
- JCL: compatibilidad con el carácter Â¥ en variables
- JCL: cálculo de RC en función de los pasos anteriores
- Comparar bytes en lugar de cadenas cuando se usa SUBSTR PL1
- Mejora de la inicialización de matrices multidimensionales desde un solo origen
- Se ha mejorado el análisis de COBOL cuando se trata de una única consulta SQL en un bloque IF

AS400

Nuevas características

- Compatibilidad con la instrucción IF anidada en CL
- Compatibilidad mejorada para la instrucción ENDDO en formato libre de RPG

Mejoras

- Compatibilidad mejorada para el acondicionamiento del nivel de control
- El prototipo mejorado regresa con LIKE
- Compatibilidad mejorada para gestionar las funciones %months, %year, %days
- Compatibilidad con la función de ayuda para toda la pantalla
- Gestión de los BLANKS figurativos pasados como parámetro
- Mejora de la expresión EVAL con el operador ""
- Gestión del comando START sin KEY PHASE
- Mejora en la gestión de la palabra clave LIKEREC
- Mejora en los subcampos sin nombre
- Mejora en el procedimiento de devolución de un tipo sin signo
- Se ha mejorado la compatibilidad con el funcionamiento de RESET (RPG gratuito) y las funciones integradas %CHAR y %DEC.

- Mejora de la función integrada %LOOKUPXX
- Compatibilidad mejorada para la palabra clave LIKEDS en un procedimiento sin prototipo
- Gestión del tipo de matriz de palabras clave Dim (VAR, AUTO)
- Compatibilidad mejorada para XFOOT
- COBOL: compatibilidad mejorada para los campos RENAME
- CL: admite la condición while(true)
- Se ha mejorado la gestión de matrices independientes con la palabra clave LIKE
- Mejora de la función integrada %INT
- Análisis RPG Full Free mejorado
- Compatibilidad mejorada para la matriz en el enlace
- CL2Declaración de GROOVY: Support Select
- Mejora en la palabra clave del DSPF “ERRMSGID”
- Se ha mejorado la gestión de la inicialización de bytes con ceros a la izquierda
- Mejora de authorizedValues para los campos numéricos
- Gestión del extensor H para una instrucción EVAL de formato libre
- CL para Groovy: compatibilidad con la subcadena de LDA
- Compatibilidad mejorada para RESET en un registro
- Se ha mejorado la gestión de EDTCDE y EDTWRD con referencias
- Se ha mejorado el mapeo de campos de entrada con campos DDS
- Se ha mejorado la compatibilidad para caracteres MOVEA en la matriz IN
- Mejora en el prototipo con la palabra clave LIKEDS
- Se ha mejorado la compatibilidad con la palabra clave DSPATR del DSPF
- Análisis mejorado de la tarjeta D con +/-
- Mayor robustez en las llamadas de los programas
- Mayor solidez en el proceso de resolución de campo

Capacidades transversales

Mejoras

- FrontEnd: Simula un evento de pegado para la entrada de IME

Tercero

- Actualización de Spring Boot de 2.5 a 2.7

Notas de la versión 3.7.0

Esta versión de las herramientas de modernización y tiempo de ejecución de AWS Blu Age incluye principalmente mejoras para una mejor compatibilidad con los comandos y las utilidades, capacidades para integrarse con AWS Secrets Manager y nuevas funciones de supervisión. Algunos de los cambios clave de esta versión son:

- Varios componentes de tiempo de ejecución ahora pueden usar AWS Secrets Manager para aumentar la configuración de seguridad de las aplicaciones modernizadas, principalmente relacionadas con las fuentes de datos de las utilidades, las colas de Redis para TS, la BluSam memoria caché y los bloqueos.
- Punto de conexión que permite recuperar métricas de transacciones, lotes y JVM para optimizar el uso de los recursos y gestionar las operaciones, como el estado, la duración, el volumen y otras.
- Nuevas funciones para admitir las llamadas de IBM MQ en RPG y una mayor cobertura de transformación de JCL SORT e IDCAMS.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.7.0

Temas

- zOS
- AS400
- Capacidades transversales

zOS

Nuevas características

- Mejora del análisis de las consultas incluidas en la aplicación de utilidades del programa mediante el uso de una gramática similar a la de SQL. (V7-9401)

- Gestión de una matriz de tamaño variable indexada cuando está desplazada (V7-9904)
- Support INSERT SQL TIME en DB2 formato de 24:00:00 horas (V7-10023)
- Compatibilidad con consultas INSERT SQL desde matrices con las opciones FOR ROWS y ATOMIC (V7-10105)
- JCL SORT: mejorado para admitir OUTREC con IFTHEN (TranscodeTool V7-10124)
- JCL SORT: agregue compatibilidad con la palabra clave DATE en el comando OUTREC (V7-10125)
- JCL: añada compatibilidad con los procedimientos In-Stream (V7-10223)

Mejoras

- Debería haber un conjunto de datos marcado con la disposición “PASS” disponible en todas las etapas del trabajo (V7-9504)
- Compatibilidad del atributo SCHENV de JCL (V7-9570)
- Compatibilidad de SEND con la opción CTLCHAR (V7-9714)
- COBOL: gestione diferentes conjuntos de caracteres separadores de líneas en las instrucciones ACCEPT (V7-9875)
- Evite la reversión múltiple (V7-9958)
- Permita el uso de la disposición MOD para anexar al final de los archivos GDG (V7-10031)
- Optimización: refactorización de putAll (V7-10063)
- PutAll refactorización: añadir paginación (V7-10063)
- Haga que el tiempo de espera de lectura del cliente Jedis sea configurable (V7-10063)
- UseSsl soporte para modo independiente (V7-10114)
- Compatibilidad con EIBDS después de abrir el archivo correctamente (V7-10147)
- Compatibilidad con EIBDS tras una solicitud de control de archivos (V7-10147)
- Mejore la compatibilidad con CICS SYNCPOINT (V7-10187)
- BluesamRedisSerializer: problema con MetadataPersistence (V7-10202)
- Compatibilidad con AWS Secrets Manager de Redis para colas TS (V7-10204)
- Compatibilidad con JCLBCICS en la personalización del tamaño del nombre DD (V7-10224)
- Añade compatibilidad con la ruta absoluta en la instrucción DELETE de IDCAMS (V7-10308)

AS400

Nuevas características

- Implementación de la función de ayuda para pantallas AS4 00 (V7-9673)

Mejoras

- Número de registros en el INFDS (V7-9377)

Capacidades transversales

Nuevas características

- Support for Runtime on EC2 para enviar registros a Amazon CloudWatch (D87990246)
- Se ha agregado un nuevo punto de conexión para recuperar métricas sobre lotes, transacciones y JVM (D88393832)

Mejoras

- Compatibilidad con AWS Secrets Manager de orígenes de datos para la utilidad pgm (V7-9570)
- Se ha agregado compatibilidad con Db2 para DSNUTILB DISCARD (V7-9798)
- Compatibilidad para escribir en el registrador en lugar del flujo de salida predeterminado del sistema en los archivos SYSPRINT y SYSPUNCH predeterminados (V7-10098)
- Support BluSam Redis almacena en caché y bloquea las propiedades de conexión en AWS Secrets Manager (V7-10238)
- Compatibilidad para conexión SSL en secreto de AWS de Db2 XA (V7-10258)
- Metadatos actualizados para IDCAMS REPRO y VERIFY (V7-10281)
- Administración mejorada de códigos de devolución de IDCAMS Abend (V7-10307)

Herramientas de modernización versión 3.7.0

Temas

- zOS
- AS400

- Capacidades transversales

zOS

Nuevas características

- PLI: asignación mejorada para matrices bidimensionales y de sección transversal (V7-9830)

AS400

Nuevas características

- Gestión de los indicadores de nivel de control (V7-9227)
- Compatibilidad con el parámetro EXTNAME con valor *INPUT (V7-9897)
- Reescritura mejorada de Goto: compatibilidad con etiquetas ubicadas en instrucciones SELECT OTHER (V7-9973)
- Compatibilidad con la palabra clave REFSHIT en el DSPF (V7-10049)

Mejoras

- Mejora en el manejo de la palabra clave de descripción del fichero EXTIND (*INUx) (V7-7404)
- Transformación mejorada de archivos SQLDDS (V7-7687)
- Ya no se generan objetos de archivo para AS4 00 archivos (V7-9062)
- Se ha mejorado la gestión de la palabra clave de descripción del archivo EXTDESC (V7-9268)
- Se ha mejorado la gestión de la versión integrada %CHAR (V7-9311)
- Se ha mejorado la compatibilidad con la reducción de páginas en el último registro sin SFLEND (V7-9322)
- Compatibilidad mejorada con estructuras de datos con prefijos (V7-9436)
- Compatibilidad con dimensiones definidas con %SIZE (V7-9472)
- Compatibilidad para gestionar el nombre de campo PF declarado entre comillas dobles (V7-9557)
- Funcionamiento de archivos mejorado: no se distingue entre mayúsculas y minúsculas (V7-9785)
- Compatibilidad con el campo inicializado como *USER (V7-9806)
- Support para el tipo COMP en AS4 00 (V7-9840)
- Análisis COBOL4 00 mejorado en (Not) (V7-9922) InvalidKey

- Gestión mejorada de la operación SCAN (V7-9971)
- Compatibilidad mejorada con el código de operación GOTO (V7-9973)
- Gestión mejorada de la operación EXCEPT (V7-9977)
- Compatibilidad con prefijos mejorada (V7-10000)
- Compatibilidad con llamadas MQ en RPG (V7-10007)
- %LOOKUP integrado mejorado (estructura de datos de matriz con claves) (V7-10022)
- Compatibilidad con Close *All operation (V7-10036)
- Compatibilidad con la instrucción UPDATE AS ROW CHANGE SQLDDS (V7-10051)
- Mejora para gestionar el tipo de valor literal Long (V7-10073)
- Mejora en la gramática de RPG (uso de la palabra clave INZ como nombre de la subrutina) (V7-10074)
- Se ha mejorado la gramática en RPG para admitir valores numéricos con una parte fraccionaria vacía (V7-10077)
- Compatibilidad mejorada para los campos compartidos entre CL y un archivo externo (V7-10081)
- Compatibilidad mejorada para los indicadores condicionales DDS (V7-10084)
- Compatibilidad con el tipo binario DDS con programas COBOL (V7-10100)
- Se ha mejorado la colisión de nombres con el enlace (V7-10109)
- Compatibilidad para mezclar procedimientos principales y de exportación (V7-10112)
- Soporte mejorado para un subprocedimiento (DataStructure V7-10113)
- Compatibilidad mejorada de CLEAR (V7-10126)
- Compatibilidad mejorada del bucle DO (V7-10134)
- Compatibilidad con SQLTYPE en RPG completamente gratuito (V7-10151)
- Se ha mejorado el análisis de las condiciones de la palabra clave DDS (V7-10155)
- Generación de DSL mejorada (V7-10163)
- Mejora de processIndicators cuando la condición es una expresión binaria. (V7-10164)
- Mejorado GOTOs con el estado Else (V7-10168)
- Compatibilidad con los tipos Time y Timestamp en DSPF (V7-10173)
- Análisis mejorado de la línea de continuación para el DDS (V7-10183)
- Compatibilidad con COBOL para RENAMES FLD OF RECORD (V7-10195)
- Se ha mejorado el análisis de indicadores condicionales en los campos DSPF (V7-10221)

- Compatibilidad para el análisis de la palabra clave DDS NOALTSEQ (V7-10288)
- Compatibilidad con el menú de ayuda y campos ocultos (V7-10314)
- Comprobación mejorada del estado de las palabras clave de ayuda del DSPF (V7-10328)
- Ya no se propagan todas las palabras clave en el campo Ref (V7-10347)

Capacidades transversales

Nuevas características

- Migrador de datos: gestión de datos CLOB (V7-9665)

Mejoras

- Propagación de la propiedad SCHENV de JCL desde la definición de JOB a la de PROC GROOVY (V7-10225) JobContext
- FrontEnd - Ajustar el tamaño de la ventana en caso de que no haya bordes (V7-10358)

Notas de la versión 3.6.0

Esta versión de las herramientas de tiempo de ejecución y modernización de AWS Blu Age ofrece nuevas funciones tanto para las migraciones de zOS como para las migraciones anteriores, orientadas principalmente a ampliar los mecanismos de soporte del CICS, complementar las capacidades de JCL, optimizar el rendimiento en funciones simultáneas y de gran volumen y añadir funciones. AS4 multi-data-source Algunos de los cambios clave de esta versión son:

- Mejora de la gestión dinámica de archivos de JCL, expansión de las instrucciones actuales y administración de conjuntos de datos concatenados, ejecución de múltiples instrucciones en un solo bloque y transferencia de datos de lotes a programas.
- Compatibilidad mejorada con varios comandos del CICS, incluida la consulta de varios tipos de recursos del CICS.
- La capacidad de tener diferentes bases de datos cuando se utilizan las utilidades de Blu Age Runtime, que resulta ideal para situaciones en las que los datos empresariales se distribuyen en varios orígenes.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.6.0

Temas

- zOS
- AS400
- Capacidades transversales

zOS

Nuevas características

- JCL - - DynamicFileBuilder Gestión mejorada de los gestores de archivos (V7-9408)
- Conversión de formato mejorada en algunas DB2 funciones SQL integradas al llamar a la utilidad INFUTILB UNLOAD (V7-9554)
- Asignaciones de matrices multidimensionales PLI mejoradas (V7-9592)
- Gestión de la redirección de sysout al archivo (V7-9992)

Mejoras

- Se agregó la activación de los procedimientos almacenados para el RDBMS (V7-9155) DB2
- SORT gestiona la conversión a formato PDF (V7-9286)
- JCL/GROOVY: mejore la instrucción REPRO para que sea compatible con conjuntos de datos DUMMY (V7-9424)
- Mejora de la compatibilidad con CICS UNLOCK (V7-9606)
- Gestione el tamaño del valor predeterminado para Union (V7-9648)
- JCL/GROOVY handle different termination/dispositionen conjuntos de datos concatenados (V7-9653)
- Haga que pageSize sea configurable para conjuntos de datos Blusam (V7-9680).
- DSNUTIL: permite cargar las 24:00:00 como hora válida en LUW (V7-9697) DB2
- Support la comparación de VALORES ALTOS (0xff) en NumberUtils .ne ()/ NumberUtils.eq () (V7-9731)
- JCL/GROOVY: compatible con DO... A continuación, utilice palabras clave en las IF-THEN-ELSE cláusulas IDCAMS para ejecutar varias sentencias en un solo bloque (V7-9750)
- Programa JHDB no válido llamado ajeno a Runner (V7-9782) JHDBBatch

- Compatibilidad para espacios en blanco en la tarjeta de control SORT OUTFIL (V7-9808)
- Mejora de la compatibilidad con CICS READ PREV (V7-9845)
- Mejora del acceso simultáneo a los índices de los conjuntos de datos (V7-9864)
- Mejora de la compatibilidad con CICS REWRITE (V7-9873)
- COBOL: compatibilidad con SYSIN multilínea en las instrucciones ACCEPT para pasar datos de un lote (JCL) a un programa (COBOL) (V7-9875)
- Groovy: mejor manejo de los archivos en la etapa de creación (V7-9876 ConcatenatedFileConfiguration)
- IDCAMS UTILITY: gestión de la instrucción DEFINE PATH (V7-9878)
- SORT BUILD: ajuste la opción TRAN y gestione los espacios en blanco implícitos (V7-9925)
- Mejora de CICS DELETE con la compatibilidad con la opción GENERIC (V7-9939)
- Mejora de la compatibilidad con CICS STARTBR y ENDBR (V7-9952)
- Mejora del rendimiento cercano en el acceso simultáneo (V7-9953)
- Mejora de la gestión del estado de los archivos al inicio (V7-9991)
- Groovy: permite la llamada a getDisposition ()/()/() el () (getNormalTerminationV7-10012) getAbnormalTermination ConcatenatedFileConfiguration

AS400

Nuevas características

- Compatibilidad con indicadores externos en palabras clave COMMIT (V7-6035)
- Restablezca el bucle ReadC después de escribir en SFLCTL (V7-8061)
- Compatibilidad con el indicador LR en CALL (V7-9250)
- Añada un nuevo tipo de campo dinámico (dividido) para gestionar el campo de entrada en varias líneas (V7-9370)
- Compatibilidad con archivos primarios/secundarios (V7-9390)
- El área de datos local ahora se transfiere al trabajo solicitado al enviar un trabajo (V7-9775)
- Compatibilidad de QTEMP para el área de datos y compatibilidad para la creación de valor del área de datos. (V7-9916)
- Control de compromisos: compatibilidad para activar/desactivar el control de compromisos (V7-9956)
- Compatibilidad con los indicadores externos en las palabras clave COMMIT

Mejoras

- Mejora de la visualización del valor 0 y EDTWRD (V7-8933)
- Compatibilidad con la palabra clave “CHKMSGID” del DSPF (V7-9125)
- Transacción de confirmación de SQL al finalizar el lote (V7-9232)
- Mejora de la compatibilidad de las palabras clave EXPORT e IMPORT para el campo y la estructura de datos (V7-9265)
- Support minúscula en forma de entrada DateHelper (V7-9461)
- Admite la conversión de *CYMD a *ISO (numérico) (V7-9488)
- Mejora de la gestión del %len incorporado para un campo variable (lado izquierdo y derecho de una expresión) (V7-9733)
- Mejora de la compatibilidad con las funciones integradas '%LOOKUPXX' XX ("LE","LT","GE","GT") (V7-10064)

Capacidades transversales

Nuevas características

- CICS: mejora de la transacción Inquire para comprobar el estado de la opción (V7-9712)
- JCL: mejora de la carga de sysprint con el archivo de salida del sistema (V7-9797)
- CICS: mejora de INQUIRE TSQUEUE (V7-9823)
- CICS: mejora del terminal Inquire para el identificador de usuario de la opción (V7-9906)

Mejoras

- Mejora de la gestión de la comparación con blank (V7-8047)
- Mejora del registro para Jics y Blusam (V7-8847)
- Compatibilidad con atributos extendidos SOSI y símbolo programado F8 para campos dinámicos en BMS (V7-8857)
- Controle el desbordamiento del búfer en el parámetro del programa (V7-9138)
- Mejora de la simultaneidad de escritura de subprocesos para el registro de bloqueos de Blusam (V7-9505)
- Compatibilidad para la configuración de múltiples orígenes de datos para Utility-pgm (V7-9570)
- Modo de bloqueo de nivel de registro de Blusam únicamente (V7-9626)

- Asegúrese de que la persistencia de los metadatos se resista al reinicio del servidor (V7-9748)
- Mejora de la limpieza de DAO en caso de excepción (cierre del navegador) (V7-9790)
- Soporte DummyFile para INFUTILB SYSPUNCH (V7-9799)
- Mejore la compatibilidad con valores negativos en el (V7-9935) NumericEditedType

Herramientas de modernización versión 3.6.0

Temas

- zOS
- AS400
- Capacidades transversales

zOS

Nuevas características

- JCL: mejore el registro al final del procedimiento (V7-8509)
- PL1 - Mejore la generación de bolsas según el tipo PakedLong de datos (V7-8917)
- JCL: mejore el registro al final del procedimiento cuando el archivo contenga el marcador de “fin” // (V7-9509)
- PL1 - Mejore la compatibilidad con GET EDIT con Fixed-point y SYSIN Stream (V7-9593)
- DB2 - Mejore la compatibilidad con el tipo VARGRAPHIC (V7-9809) DB2
- CICS: mejora del comando QUERY SECURITY para la opción LOGMESSAGE (V7-9969)
- PL1 - Mejore la generación de bolsas para Charg/Chargraphic integrado (V7-9989)

Mejoras

- PL1- Mejore la compatibilidad con la palabra clave INCLUDEX (V7-9588)
- PL/I- Maneje la palabra clave CHARGRAPHIC como un parámetro válido de cualquier llamada a un método (V7-9589)
- Mejora de la resolución de la variable de PL1 host cuando se nombra con caracteres específicos @ # $ §. (V7-9654)
- COBOL: compatibilidad de las palabras clave C01... C12 y S01... S05 como parámetro de la instrucción WRITE ADVANCING en la etapa de análisis (V7-9669)

AS400

Nuevas características

- Compatibilidad para la transformación de SQL-DDS en Analyzer (V7-7687)
- Automatice la detección de archivos SQL-DDS (V7-7687)
- Implementación del preprocesamiento de SQL-DDS (V7-7687)
- Admite la palabra clave ALIGN (V7-9254)
- Support ExtName para DSPF y arreglos multidim (V7-9663)
- InvalidKey Declaraciones de apoyo sobre COBOL WRITE (V7-9793)

Mejoras

- Mejora del código de operación TESTB (V7-8865)
- Mejora de la compatibilidad con DECFMT en el enfoque (V7-8933)
- Gestión del indicador resultante en MOVE (V7-9224)
- Mejora de la compatibilidad de la palabra clave TEMPLATE para el campo y la estructura de datos (V7-9278)
- Mejora de LIKEDS (el DS definido con LIKEDS se califica automáticamente) (V7-9302)
- COBOL: mejora de la estructura de generación de indicadores (V7-9423)
- El parámetro constante del prototipo no es de solo lectura (V7-9437)
- Mejora de la palabra clave EDTCDE con el código de edición “Y” (V7-9443)
- Compatibilidad para la generación del campo *ROUTINE en PSDS e INFDS (V7-9487)
- Mejora del campo de reescritura XXX para que sea independiente (el valor predeterminado se pierde al reescribir) (V7-9522)
- Mejora de la compatibilidad con palabras clave del DSPF (V7-9658)
- Gestión del valor predeterminado ZEROES en formato binario (V7-9666)
- Compatibilidad con puntero implícito (V7-9719)
- Mejora de la gestión del %size de llamadas integrado con un parámetro (V7-9730)
- Mejora del manejo de las referencias a la estructura de datos en las llamadas integradas (%ELEM) (V7-9736)
- Mejora de la gestión de la longitud firmada de los campos con referencia LIKE en la especificación de definición (V7-9738)

- Mejora en REWRITE (V7-9791)
- Mejora de la generación de índices a partir de archivos DDS (V7-9803)
- Mejora de la solidez de los mapeadores con un valor numérico no válido (V7-9813)
- Mejora SQLModel y completa la generación de archivos Indexes (V7-9818)
- Mejora de la compatibilidad cualificada de DS (V7-9863)
- Mejora de la compatibilidad con LOOKUP (con un campo independiente como un parámetro DS) (V7-9961)
- Mejora de LIKE en el indicador (V7-9985)
- Gestión del indicador resultante en MVR (V7-9995)
- Compatibilidad con el carácter N con tilde (V7-10021)
- Mejora de la generación moderna de archivos DDL a partir de archivos antiguos de SQLDDS (V7-10067)

Capacidades transversales

Nuevas características

- Personalización de la ubicación de los recursos con una propiedad yml (D88816105)
- COBOL: compatibilidad de la instrucción EXIT PERFORM para salir de un PERFORM en línea sin utilizar un GO TO / PERFORM ... HASTA (V7-9582)
- Especificación de la codificación heredada predeterminada para considerarla en los metadatos globales. (V7-9883)

Mejoras

- Mejora en la generación de máscaras (V7-9602)
- Mejora de la preparación anticipada del contexto (V7-9621)
- Haga que el hilo Charset 0 sea seguro. CUSTOM93 (V7-9674)
- Mejora en MOVEA (V7-9773)

Notas de la versión 3.5.0

Esta versión de las herramientas de tiempo de ejecución y modernización de AWS Blu Age ofrece nuevas funciones para las migraciones antiguas de zOS y AS4 00, orientadas principalmente a la

optimización de los conjuntos de datos y la mensajería, así como capacidades ampliadas de Java como una ventaja del proceso de transformación. Algunos de los cambios clave de esta versión son:

- Capacidad de migrar programas de CL a Java, además de la característica preexistente de groovy scripts, para facilitar su integración con otros programas modernizados y simplificar la curva de aprendizaje de los clientes mediante la unificación del lenguaje de programación resultante.
- Reducción del tiempo y optimización del rendimiento de las cargas de conjuntos de datos en Redis con la nueva característica de almacenamiento masivo de datos.
- Capacidad para operar y transferir conjuntos de datos según las etapas del trabajo a fin de modernizar los comportamientos tradicionales de los conjuntos de datos.
- Extensión de la migración a SQL para admitir archivos de entrada de VB y migración simplificada a Java 11.
- Varios mecanismos nuevos para una integración más rápida con IBM MQ, incluidos encabezados adicionales, compatibilidad ampliada para GET/PUT y recuperación automática de los metadatos de las colas.
- REST Endpoint para metadatos de conjuntos de datos e importación de conjuntos de datos desde buckets de S3.

Para obtener más información acerca de los cambios incluidos en esta versión, consulte las siguientes secciones.

Tiempo de ejecución versión 3.5.0

Temas

- zOS
- AS400
- Capacidades transversales

zOS

Nuevas características

- JCL SORT: gestiona la superposición de nuevas palabras clave (V7-9409)
- ZOS COBOL: mejora la compatibilidad con caracteres flotantes (V7-9404)
- Puerto de RedisJics TSQueue a RedisTemplate & ListOperations (V7-9212)

- ZOS JCL: mejore la ruta del directorio temporal con el directorio de archivos si está definido mediante (V7-9012) UserDefinedParameters
- Gestión de la FUNCIÓN ORD-MAX con ALL (todos los elementos de la matriz) (V7-9366)
- Ahora se utilizan claves prefijadas y en lenguaje natural al almacenar TS Queues en Redis (V7-9212)
- Se ha agregado el punto de conexión del obtención del conjunto de datos para la API de Blusam.
- JCL: compatibilidad con ADD para trabajos por lotes con un nombre que incluya un carácter especial como # (V7-9136)
- TSModel la búsqueda ahora se realiza de forma robusta bajo demanda (V7-9212)

Mejoras

- Compatibilidad con INCLUDE sin versiones en archivos LNK (V7-6022)
- MQ: mejora de la compatibilidad de codificación (V7-9652)
- Mejora de la compatibilidad con bytes dobles o conjuntos de caracteres mixtos para distintos tipos de caracteres (V7-9596)
- JCL: la compatibilidad de configuración de filesDirectory en IDCAMS borra instrucciones NONVSAM (V7-9609)
- Compatibilidad con el modo masivo para la carga de conjuntos de datos ESDS y RRDS desde archivos (V7-8639)
- Gestión de la apertura de un ESDS vacío en el modo de entrada. (V7-9287)
- Mejora de la instrucción DEFINE CLUSTER con la compatibilidad para abreviaturas ORD/UNORD (V7-9451)
- Se ha mejorado el rendimiento del bloqueo de Redis de Blusam (V7-8639).
- Mejora de la instrucción DEFINE CLUSTER para que sea compatible con el RECORDSIZE proporcionado en el ámbito del argumento DATA() (V7-9337)
- Añade compatibilidad con los atributos BUFFERSPACE/UNIQUE en las instrucciones DEFINE CLUSTER (V7-9419)
- Se ha mejorado la operación de lectura de Blusam para un conjunto de datos de registros de longitud variable. (V7-9391)
- La CICS ADDRESS representa correctamente la CWA que falta como nulo (V7-9491)
- Eliminación de la escritura innecesaria en los bloqueos de los extremos (V7-8639)
- Gestión de la inyección de plantillas de caché de Redis en la caché (V7-9510)

- Decodificación correcta del parámetro BPXWDYN (V7-9417)
- Mejora del consumo de exportación de LISTCAT (V7-9201)
- Se admiten caracteres no imprimibles en el nombre de las colas de Blusam TS (V7-9212).
- Gestión de la creación de mapas de recepción para un campo con el mapset nulo (V7-9486)
- Mejore la operación de BluesamRelativeFile eliminación y reescritura para el modo de acceso dinámico. (V7-8989)

AS400

Nuevas características

- Añadida una función para generar archivos CL como programas Java mediante un sistema dinámico estándar DS/STM (V7-9427)
- Compatibilidad para el archivo de entrada con modo ADD (V7-9378)
- Se ha mejorado el orden de clasificación y la gestión de las recuperaciones para admitir el comando cl OPNQRYF (Abrir archivo de consulta) y se ha añadido la compatibilidad con el parámetro SHARE en. OverrideItem (V7-9364)

Mejoras

- Support SFLNXTCHG en (V7-8061) UpdateSubfile
- Modificación del alcance del contexto CL al ejecutar el comando CL (V7-9624)
- Gestión del código de retorno del programa BPXWDYN (V7-9417)
- Borrado de los monitores locales. (V7-9624)
- Compatibilidad con la palabra clave RTNCSRLOC del DSPF (V7-9389)
- setOnGreaterOrEqual() no se establece igual a 1 (V7-9342)
- Actualice la caché de campos en UpdateSubfileRecord (V7-9376)
- Mejora de la compatibilidad con SFLNXTCHG (V7-8061)

Capacidades transversales

Nuevas características

- Se ignora el prefijo G en la cadena gráfica literal. (V7-9420)

- ZOS COBOL: mejora de la compatibilidad de Fiedl.initialize() para algunas estructuras especiales (V7-9485)
- Se permite la inicialización del contexto de forma asíncrona para mejorar el rendimiento del inicio del programa (V7-9446)
- SQL publique explícitamente la sentencia de preparación abierta y. ResulSet (V7-9422)
- Mejore el JMS MQ: MQRFH2 compatibilidad con MQ PUT/V7-7085, compatibilidad con el administrador de colas predeterminado (V7-9400)
- Administración de SQL: se habilitan las conversiones de Lambda en los parámetros de los comandos SET (V7-9492)
- ZOS MQ JMS: se añade compatibilidad con MQCOMIT y MQBACK (V7-9399)
- ZOS IBMMQ: mejora de la compatibilidad con MQINQ (V7-9544)
- Gestión de la operación CONCAT con bytes en lugar de cadenas cuando se utiliza la codificación de doble byte. (V7-8932)
- ZOS IBMMQ: mejora de la compatibilidad con el comando PUT con las opciones SET_ALL_CONTEXT (V7-9544)

Mejoras

- Gestión de los nombres de los archivos gdg con el carácter $ (V7-9066)
- El diagnóstico de SQL devuelve 1 como cláusula NUMBER cuando la sentencia SQL anterior es correcta. (V7-9410)
- Esquema de un campo con una longitud no nula (V7-7536)
- Support integrado PL1 GRAPHIC (V7-9245)
- MQ: se ha agregado compatibilidad para la configuración de campos de MQGMO (V7-9500)
- JMS MQ GET: mejora de la longitud de datos del mensaje devuelto (V7-9502)
- Definición de sqlerrd (3) con el número de elementos recuperados en el contexto de ROWSET. (V7-9371)

Herramientas de modernización versión 3.5.0

Temas

- zOS
- AS400

- Capacidades transversales

zOS

Nuevas características

- ZOS PLI: compatibilidad del índice de asteriscos en la asignación con expresión binaria (V7-9178)
- De JCL a BatchScript - A «//» indica el final de la ejecución del trabajo (V7-9304)
- ZOS PLI: mejora la compatibilidad con caracteres flotantes y el tipo de registro editado numéricamente (V7-8982)
- COBOL: compatibilidad con la función SUM integrada (V7-9367)
- JCL: opcionalmente, se puede comentar el código inactivo después de una sentencia nula (//) (V7-9202)
- JCL: compatibilidad del operador '|' en la instrucción de estado (V7-9499)
- PL/I- Comentario sobre las directivas de precompilación en la etapa de preprocesamiento para evitar excepciones de análisis (V7-9507)

Mejoras

- Gestión de la definición de flujo con un delimitador (V7-9615)
- Mejora de la gestión de las exportaciones de LISTCAT. (V7-9201)
- PL/I- Mejora para admitir argumentos «nulos» implícitos (V7-9204)

AS400

Nuevas características

- Compatibilidad de la palabra clave CONCAT de DDS (V7-9439)
- Refactorización del código java generado para las palabras clave del DSPF. (V7-7700)
- Se admite variar las palabras clave en los campos de una definición de estructura de datos (V7-9029)

Mejoras

- Mejora del análisis de la relación lógica Y/O (V7-9352)

- COBOL Mejora del mapeo entre vo y dEntity (V7-9449)
- Se muestra un valor vacío si la entrada numérica está enfocada (V7-9374)
- Variable local en SQL Declare Cursor (V7-9456)
- Problema de alcance con un DS vacío (V7-9466)
- Truncamiento de la columna 80 antes del análisis (V7-9632)
- Mejora de la gestión de las referencias de campo y las llamadas integradas en las palabras clave (DIM, LIKE,...) en la especificación de definiciones (V7-9358)
- Comentarios sobre compatibilidad con SQL (--) (V7-9632)
- FullFree análisis, tipo Date/Time/Timestamp (V7-9542)
- Incluya el SQLCA del análisis (V7-9333) FullFree
- Mejora de la compatibilidad con el nivel de control. (V7-9610)
- Gestión de la comparación de DS con *BLANKS (V7-9668)
- Mejora de la compatibilidad con varios indicadores en el DDS (V7-9318)
- Mejora de la compatibilidad con varios programas de DSPF (V7-9657)
- Mejora de la gestión del campo con LIKE (caso de estructura de datos similar y caso de estructura de datos similar en una matriz) (V7-9213)
- RPG libre, gestión de continuación en literales (V7-9686)
- Mejora de la compatibilidad con los registros de fin de programa (V7-9452)
- Compatibilidad de la frase LINKAGE en la instrucción CALL. (V7-9685)
- Código de operación CASXX (CASBB sin grupo CASXX) (V7-9357)
- Mejore el análisis de los juegos de rol (V7-9457) FullFree
- El %LEN incorporado no admite DS como argumento (V7-9267)
- Mejoras en MOVEA cuando el factor 2 es *ALL'X... ' (V7-9228)
- Admite la asignación con el campo RENAME (V7-9385)

Capacidades transversales

Nuevas características

- Herramienta SQL Migrator: se ha agregado la opción OID para una longitud de registro variable en el paso de carga de ebcdic. (V7-9380)
- Herramienta SQL Migrator: compatibilidad con Java 11 en la opción OID (V7-9599)

Mejoras

- Mejora de la compatibilidad con matrices anidadas (V7-9595)
- Se sustituye el carácter Â¬ por ! en caso de que Â¬ sea compatible con la codificación original. (V7-9465)
- JCL: compatibilidad con la terminación normal PASS para compartir conjuntos de datos entre las etapas del trabajo (V7-9504)
- se aplica ON NULL a la definición de columnas en ORACLE cuando se trate de columnas tipo VARCHAR y bases de datos que admiten valores NULL. (V7-9681)
- Mejora de la conformidad con la inyección de Spring (V7-9635)

AWS Vulnerabilidades de seguridad de la era

Vulnerabilidades y exposiciones comunes (CVE) es una lista de referencia de vulnerabilidades de ciberseguridad conocidas públicamente. Cada entrada contiene un número de identificación, una descripción y al menos una referencia pública.

Le recomendamos que actualice siempre a la última versión de AWS Blu Age para protegerse contra las vulnerabilidades conocidas. Amazon Inspector realiza análisis de seguridad de forma continua y los resultados se clasifican según su gravedad en NIST.

La siguiente lista detalla CVEs las correcciones en cada versión secundaria disponible que resultan del uso de dependencias:

Versión	CVE
4.8.0	CVE-2025-22228
4.7.0	CVE-2025-25193, CVE-2025-24970, CVE-2024-47535, CVE-2024-38820, CVE-2024-38809, CVE-2024-22262, CVE-202245868
4.6.0	CVE-2024-12801, CVE-2024-12798, CVE-2024-50379, CVE-2024-56337

Versión	CVE
4.5.0	CVE-2024-47535, CVE-2024-52316, CVE-2024-47535, CVE-2024-38827
4.4.0	CVE-2024-38820, CVE-2024-38821, CVE-2024-38809, CVE-2024-38816, CVE-2024-47554, CVE-2024-6484, CVE-2024-6485
4.3.0	CVE-2024-43788, CVE-2022-25898, CVE-2021-30246, CVE-2024-21484, CVE-2024-34750
4.2.0	CVE-2020-11023, CVE-2023-26364 , CVE-2019-11358, CVE-2020-11022, CVE-2021-23358, CVE-2017-18214, CVE-2022-24785, CVE-2022-31129, CVE-2023-48631
4.1.0	CVE-2024-29025, CVE-2024-23080, CVE-2024-22262, CVE-2024-30171, CVE-2024-29857, CVE-2024-30172
4.0.0	CVE-2016-1000027, CVE-2022-1471, CVE-2024-1597, CVE-2024-22243, CVE-2024-22233, CVE-2024-22234, CVE-2024-22259, CVE-2024-22257, CVE-2024-29131, CVE-2024-29133

> **Note**
>
> Para obtener más información sobre las versiones corregidas en versiones anteriores CVEs , ponte en contacto con tu administrador de entregas de Blu Age AWS

Instrucciones de actualización para AWS Blu Age

Esta página contiene instrucciones para actualizar la versión AWS Blu Age.

Actualizaciones comunes

En la mayoría de los casos, al actualizar la versión (no gestionada) de AWS Blu Age RuntimeWARs, debe sustituir los elementos (archivos de configuración, scripts, etc.) de la versión anterior por los de la nueva y reiniciar la aplicación. Asegúrese de realizar exhaustivas pruebas de regresión de las aplicaciones modernizadas una vez que actualice. También puedes ponerte en contacto con tu responsable de entregas de AWS Blu Age para obtener instrucciones específicas aplicables a tu aplicación.

Para actualizar la versión (gestionada) de AWS Blu Age Runtime, consulteEntornos de tiempo de ejecución administrados.

Algunas actualizaciones pueden requerir configuración adicional para garantizar la compatibilidad. En ese caso, siga las instrucciones de esa actualización específica.

Migración de 3.10.0 a 4.0.0

El cambio principal de la versión 4.0.0 es la migración de Spring Boot 2.7 a Spring Boot 3.2 y de Tomcat 9 a Tomcat 10.

Cambios de código

En esta sección se enumeran los cambios necesarios para que el código modernizado sea compatible con AWS Blu Age Runtime 4.0.0. Puede omitir esta sección si decide iniciar una nueva generación con la versión 4.0.0 en Blu Insights (Transformation Center).

Cambios en POM

Grupo	ArtifactId	Cambio
org.slf4j	slf4j-api	Eliminación (es una dependencia transitiva)
org.yaml	snakeyaml	Eliminación (es una dependencia transitiva)

Grupo	ArtifactId	Cambio
org.springframework.boot	spring-boot-starter-web	- Actualice spring.boot.versio n a la versión 3.2.4 - Elimine la exclusión de log4 4j j-to-slf
org.springframework.boot	spring-boot-starter-jta-ato mikos	Cambiar a com.atomikos: 3-starter:6.0.0 transactions-sprin g-boot
org.apache.commons	commons-dbcp2	Actualización a 2.10.0
org.postgresql	postgreql	Actualización a 42.7.2
com.microsoft.sqlserver	mssql-jdbc	Actualización a 12.4.2.jre11
com.oracle.database.jdbc	ojdbc8	Cambio a ojdbc11 versión 23.3.0.23.09

Migración de Javax a Yakarta

La actualización de Tomcat incluye una migración del paquete Java Javax a Jakarta. Asegúrese de actualizar las importaciones como corresponde de javax.* a jakarta.*.

Casi todas las clases antiguas de referencia en el paquete Javax se encuentran en Jakarta. Las excepciones conocidas para esto son los paquetes `javax.sql` y `javax.xml`, que siguen sin cambios.

Cambio de Atomikos

Debido al cambio de dependencia mencionado anteriormente, las referencias a `org.springframework.boot.jta.atomikos.AtomikosDataSourceBean` deben cambiarse a `com.atomikos.spring.AtomikosDataSourceBean`.

Eliminación del dialecto de PostgreSQL

Se elimina la clase personalizada `PostgreSQLDialect.java`. También se deben eliminar las referencias a ella en el iniciador principal.

Implementación (Blu Age Runtime (no administrado)AWS

Tomcat

Esta versión es compatible con Tomcat `10.1.17`. Es necesario actualizar el servidor de Tomcat a esta versión para ejecutar el tiempo de ejecución de Blu Age `4.0.0`. Asegúrese de transferir los cambios de configuración anteriores (especialmente las propiedades de Catalina).

Dependencias compartidas

La carpeta compartida del entorno de ejecución contiene las up-to-date dependencias.

Dependencias adicionales

Si ha usado dependencias adicionales (no incluidas en el tiempo de ejecución), es posible que tenga que actualizarlas. El archivo readme de la carpeta adicional muestra las versiones admitidas.

AWS Ciclo vital de la era azul

En esta sección se definen las fechas de fin de vida útil (EOL) de las principales versiones del AWS Blu Age Runtime. Esto le permite planificar las actualizaciones de las versiones para mantenerse al día con las características y el mantenimiento más recientes. Para actualizar la versión, consulte the section called “Actualización de AWS Blu Age”.

Le recomendamos que compruebe si hay nuevas versiones cada 3 meses y que las actualice a las recientes con frecuencia. Para cada actualización, debe realizar pruebas de no regresión de las aplicaciones modernizadas antes de las implementaciones críticas o de producción.

> **Note**
>
> Las fechas de fin de vida útil pueden estar sujetas a cambios debido a correcciones de seguridad críticas. Para obtener más información, consulta Ciclo de vida de componentes.

AWS Fin de la vida útil de Blu Age Runtime (EOL)

En la siguiente tabla se resume la fecha de fin de vida útil de cada versión principal.

Versión principal	Fecha de fin de vida útil
Versión 3	8 de julio de 2024

Versión principal	Fecha de fin de vida útil
Versión 4	Aún no se ha publicado

> **Note**
>
> La fecha de fin de vida útil de la versión principal 4 se ajustará a la disponibilidad de la próxima versión principal.

Para comprender el modelo de soporte de las versiones secundarias, consulte Ciclo de vida de componentes.

AWS Conceptos de Blu Age Runtime

Comprender los conceptos básicos del tiempo de ejecución de AWS Blu Age puede ayudarle a comprender cómo se modernizan sus aplicaciones mediante la refactorización automatizada.

Temas

- AWS Arquitectura de alto nivel Blu Age Runtime
- AWS Estructura de Blu Age de una aplicación modernizada
- ¿Qué son los simplificadores de datos en AWS Blu Age?
- AWS Blu Age Blusam
- Programas disponibles en la aplicación web de utilidad
- AWS Consola de administración Blu Age Blusam

AWS Arquitectura de alto nivel Blu Age Runtime

Como parte de la solución de AWS Blu Age para modernizar los programas antiguos a Java, el AWS Blu Age Runtime proporciona un punto de entrada unificado y basado en REST para las aplicaciones modernizadas y un marco de ejecución para dichas aplicaciones, a través de bibliotecas que proporcionan construcciones heredadas y una estandarización de la organización del código de los programas.

Estas aplicaciones modernizadas son el resultado del proceso de refactorización automatizada de AWS Blu Age para modernizar los programas de mainframe y rango medio (denominados «heredados» en el siguiente documento) a una arquitectura basada en la web.

Los objetivos de AWS Blu Age Runtime son la reproducción del comportamiento de los programas heredados (isofuncionalidad), el rendimiento (con respecto al tiempo de ejecución de los programas y el consumo de recursos) y la facilidad de mantenimiento de los programas modernizados por parte de los desarrolladores de Java, mediante el uso de entornos y modismos familiares como tomcat, Spring, getters/setters, fluent APIs.

Temas

- AWS Componentes de ejecución de Blu Age
- Entornos de ejecución
- Ausencia de estado y gestión de sesiones
- Alta disponibilidad y ausencia de estado

AWS Componentes de ejecución de Blu Age

El entorno de ejecución de AWS Blu Age se compone de dos tipos de componentes:

- Un conjunto de bibliotecas java (archivos jar), a las que a menudo se hace referencia como “la carpeta compartida”, y que proporcionan construcciones e instrucciones heredadas.
- Un conjunto de aplicaciones web (archivos war) que contienen aplicaciones web basadas en Spring y que proporcionan un conjunto común de marcos y servicios para los programas modernizados.

En las siguientes secciones se detalla el rol de estos dos componentes.

AWS Bibliotecas de Blu Age

Las bibliotecas AWS Blu Age son un conjunto de archivos jar almacenados en una `shared/` subcarpeta que se añade a la ruta de clases estándar de Tomcat, para que estén disponibles para todos los programas Java modernizados. Su objetivo es proporcionar características que no están disponibles de forma nativa ni fácilmente disponibles en el entorno de programación Java, sino que son típicas de los entornos de desarrollo tradicionales. Estas funciones están expuestas de una forma que resulta lo más familiar posible para los desarrolladores de Java (captres/setters, basadas en clases, fluidas). APIs Un ejemplo importante es la biblioteca Data Simplifier, que proporciona a los

programas Java estructuras antiguas de diseño y manipulación de la memoria (que se encuentran en los lenguajes COBOL o RPG). PL1 Estos archivos jar son una dependencia fundamental para el código Java modernizado generado a partir de programas heredados. Para obtener más información sobre Data Simplifier, consulte ¿Qué son los simplificadores de datos en AWS Blu Age?.

Aplicación web

Los archivos de aplicaciones web (WARs) son una forma estándar de implementar código y aplicaciones en el servidor de aplicaciones Tomcat. Los que se proporcionan como parte del entorno de ejecución de AWS Blu Age tienen como objetivo proporcionar un conjunto de marcos de ejecución que reproduzcan los entornos y los monitores de transacciones tradicionales (lotes de JCL, CICS, IMS...) y los servicios necesarios asociados.

El más importante es `gapwalk-application` (a menudo abreviado como “Gapwalk”), que proporciona un conjunto unificado de puntos de entrada basados en REST para activar y controlar la ejecución de transacciones, programas y lotes. Para obtener más información, consulte AWS Tiempo de ejecución Blu Age APIs.

Esta aplicación web asigna subprocesos y recursos de ejecución de Java para ejecutar programas modernizados en el contexto para el que fueron diseñados. En la siguiente sección se detallan ejemplos de dichos entornos reproducidos.

Otras aplicaciones web añaden al entorno de ejecución (con más precisión, al “registro de programas”, que se describe a continuación) programas que emulan los programas antiguos disponibles y a los que se puede acceder desde ellos. Dos categorías importantes son las siguientes:

- Emulación de programas proporcionados por el sistema operativo: los lotes impulsados por JCL esperan poder llamar a una variedad de programas de manipulación de archivos y bases de datos como parte de su entorno estándar. Entre los ejemplos se incluyen `SORT`/`DFSORT` y `IDCAMS`. Para ello, se proporcionan programas Java que reproducen dicho comportamiento y se pueden invocar utilizando las mismas convenciones que los programas antiguos.
- Los “controladores”, que son programas especializados proporcionados por el marco de ejecución o el middleware como puntos de entrada. Un ejemplo es `CBLTDLI`, de qué dependen los programas COBOL que se ejecutan en el entorno IMS para acceder a los servicios relacionados con el IMS (IMS DB, diálogo con el usuario a través de MFS, etc.).

Registro de programas

Para participar y aprovechar esas constructos, marcos y servicios, los programas Java modernizados a partir de los antiguos se adhieren a una estructura específica documentada en AWS Estructura de Blu Age de una aplicación modernizada. Al iniciarse, el motor de ejecución de AWS Blu Age recopilará todos estos programas en un «registro de programas» común para poder invocarlos (y llamarlos entre sí) posteriormente. El registro de programas ofrece un acoplamiento flexible y posibilidades de descomposición (ya que los programas que se llaman entre sí no tienen que modernizarse simultáneamente).

Entornos de ejecución

Están disponibles los entornos y coreografías tradicionales que se encuentran con más frecuencia:

- Los lotes impulsados por JCL, una vez modernizados a programas Java y scripts Groovy, se pueden iniciar de forma sincrónica (bloqueo) o asincrónica (separada). En este último caso, su ejecución se puede supervisar a través de puntos de conexión REST.
- Un subsistema AWS Blu Age proporciona un entorno de ejecución similar al CICS mediante:
  - un punto de entrada utilizado para iniciar una transacción del CICS y ejecutar los programas asociados, respetando la coreografía de los “niveles de ejecución” del CICS,
  - un almacenamiento externo para las definiciones de recursos,
  - un conjunto homogéneo de sentencias que se APIs `EXEC CICS` reproducen con fluidez en Java,
  - un conjunto de clases conectables que reproducen los servicios del CICS, como las colas de almacenamiento temporal, las colas temporales de datos o el acceso a los archivos (normalmente hay varias implementaciones disponibles, como Amazon Managed Service para Apache Flink, Amazon Simple Queue Service o RabbitMQ para TD Queues),
  - para las aplicaciones orientadas al usuario, el formato de descripción de pantalla BMS se moderniza para convertirse en una aplicación web angular y se admite el correspondiente cuadro de diálogo “pseudoconversacional”.
- Del mismo modo, otro subsistema proporciona una coreografía basada en mensajes IMS y admite la modernización de las pantallas de interfaz de usuario en formato MFS.
- Además, un tercer subsistema permite la ejecución de programas en un entorno similar al de iSeries, incluida la modernización de las pantallas especificadas por el DSPF (Display File).

Todos estos entornos se basan en servicios comunes a nivel de sistema operativo, tales como:

- la emulación de la asignación y el diseño de la memoria tradicionales (Data Simplifier),
- reproducción basada en subprocesos de Java del mecanismo de ejecución y paso de parámetros de las “unidades de ejecución” de COBOL (declaración `CALL`),
- emulación de archivos sin formato, concatenados, VSAM (a través del conjunto de bibliotecas de Blusam) y organizaciones de conjunto de datos de GDG,
- acceso a almacenes de datos, como RDBMS (declaraciones `EXEC SQL`).

Ausencia de estado y gestión de sesiones

Una característica importante del tiempo de ejecución de AWS Blu Age es permitir escenarios de alta disponibilidad (HA) y escalabilidad horizontal al ejecutar programas modernizados.

La piedra angular de esto es la ausencia de estado, un ejemplo importante de lo cual es la gestión de las sesiones HTTP.

Gestión de sesiones

Como Tomcat se basa en web, un mecanismo importante para ello es la gestión de sesiones HTTP (tal como lo proporcionan Tomcat y Spring) y el diseño sin estado. El diseño de la ausencia de estado se basa en lo siguiente:

- los usuarios se conectan a través de HTTPS,
- los servidores de aplicaciones se implementan detrás de un equilibrador de carga,
- cuando un usuario se conecte por primera vez a la aplicación, se autenticará y el servidor de la aplicación creará un identificador (normalmente dentro de una cookie)
- este identificador se utilizará como clave para guardar y recuperar el contexto del usuario hacia/desde una caché externa (almacén de datos).

La gestión de las cookies se realiza automáticamente mediante el marco AWS Blu Age y el servidor Tomcat subyacente, de forma transparente para el usuario. El navegador de Internet del usuario lo gestionará automáticamente.

La aplicación web Gapwalk puede almacenar el estado de la sesión (el contexto) en varios almacenes de datos:

- Amazon ElastiCache (Redis OSS)

- Clúster de Redis
- en un mapa de memoria (solo para entornos de desarrollo e independientes, no apto para HA).

Alta disponibilidad y ausencia de estado

En términos más generales, uno de los principios de diseño del marco de la era AWS azul es la apatridia: la mayoría de los estados no transitorios necesarios para reproducir el comportamiento de los programas heredados no se almacenan en los servidores de aplicaciones, sino que se comparten a través de una «fuente única de información» externa y común.

Algunos ejemplos de estos estados son las colas de almacenamiento temporal o las definiciones de recursos del CICS, y los almacenamientos externos típicos de esos estados son los servidores o las bases de datos relacionales compatibles con Redis.

Este diseño, combinado con el equilibrador de carga y las sesiones compartidas, permite que la mayor parte del diálogo orientado al usuario (OLTP, “procesamiento transaccional en línea”) se pueda distribuir entre varios “nodos” (en este caso, instancias de Tomcat).

De hecho, un usuario puede ejecutar una transacción en cualquier servidor sin importarle si la siguiente llamada a la transacción se realiza en un servidor diferente. Luego, cuando se genera un nuevo servidor (debido al escalado automático o para reemplazar un servidor que no esté en buen estado), podemos garantizar que cualquier servidor accesible y en buen estado pueda ejecutar la transacción según lo esperado con los resultados adecuados (valor devuelto esperado, cambio de datos esperado en la base de datos, etc.).

AWS Estructura de Blu Age de una aplicación modernizada

Este documento proporciona detalles sobre la estructura de las aplicaciones modernizadas (mediante las herramientas de refactorización de la modernización del AWS mainframe), de modo que los desarrolladores puedan realizar diversas tareas, como las siguientes:

- Navegar por las aplicaciones sin problemas.
- Desarrollar programas personalizados a los que se pueda acceder desde las aplicaciones modernizadas.
- Refactorizar de forma segura las aplicaciones modernizadas.

Damos por supuesto que ya tiene conocimientos básicos sobre lo siguiente:

- conceptos de codificación comunes heredados, como los registros, los conjuntos de datos y sus modos de acceso a los registros (indexados, secuenciales), VSAM, unidades de ejecución, scripts jcl, conceptos de CICS, etc.
- codificación Java utilizando el marco de Spring
- A lo largo del documento, utilizamos `short class names` para facilitar la lectura. Para obtener más información, consulte AWS Asignaciones de nombres totalmente calificadas de Blu Age para recuperar los nombres completos correspondientes a los elementos de tiempo de ejecución de AWS Blu Age y Asignaciones de nombres totalmente cualificadas de terceros para recuperar los nombres completos correspondientes para los elementos de terceros.
- Todos los artefactos y muestras se toman de los resultados del proceso de modernización de la aplicación CardDemo COBOL/CICS de muestra.

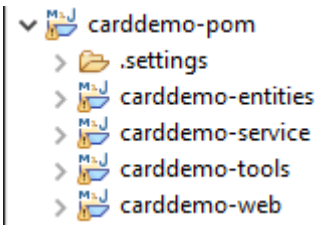
Temas

- Organización de artefactos
- Ejecutar y llamar a programas
- Escriba su propio programa
- Asignaciones de nombres totalmente cualificadas

Organización de artefactos

AWS Las aplicaciones modernizadas de Blu Age se empaquetan como aplicaciones web java (.war), que se pueden implementar en un servidor JEE. Normalmente, el servidor es una instancia de Tomcat que incorpora el tiempo de ejecución de AWS Blu Age, que actualmente se basa en los marcos Springboot y Angular (para la parte de la interfaz de usuario).

La aplicación war agrega varios artefactos de componentes (.jar). Cada artefacto jar es el resultado de la compilación (con la herramienta maven) de un proyecto Java específico cuyos elementos son el resultado del proceso de modernización.



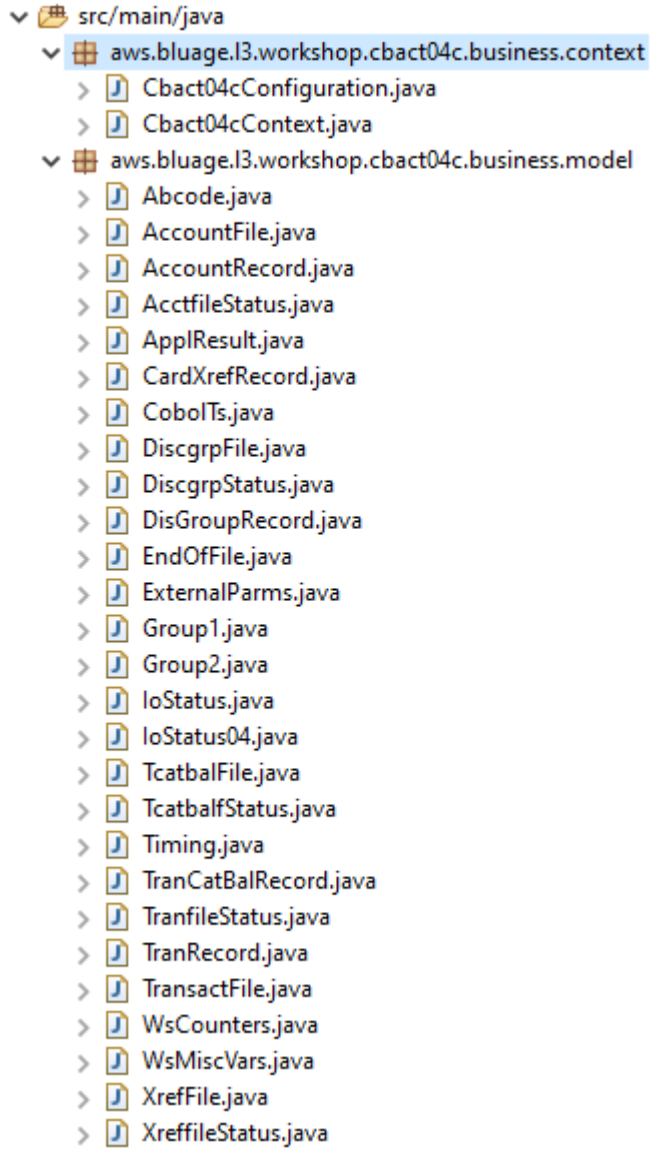
La organización básica se basa en la siguiente estructura:

- Proyecto de entidades: contiene elementos del modelo de negocio y del contexto. El nombre del proyecto generalmente termina con “-entities”. Normalmente, en el caso de un programa COBOL anterior, esto corresponde a la modernización de la sección de E/S (conjuntos de datos) y de la división de datos. Puede tener más de un proyecto de entidades.
- Proyecto de servicio: contiene elementos de modernización de la lógica empresarial heredados. Por lo general, la división de procedimientos de un programa COBOL. Puede tener más de un proyecto de servicio.
- Proyecto de utilidad: contiene herramientas y utilidades comunes compartidas, utilizadas por otros proyectos.
- Proyecto web: contiene la modernización de los elementos relacionados con la interfaz de usuario cuando corresponde. No se utiliza para proyectos de modernización únicamente por lotes. Estos elementos de la interfaz de usuario pueden provenir de mapas BMS de CICS, componentes de IMS MFS y otras fuentes de interfaz de usuario de unidad central. Puede tener más de un proyecto web.

Contenido de proyecto de entidades

> **Note**
>
> Las siguientes descripciones solo se aplican a las salidas de modernización de COBOL y PL/I. Las salidas de la modernización de RPG se basan en un diseño diferente.

Antes de cualquier refactorización, la organización de los paquetes en el proyecto de la entidad está vinculada a los programas modernizados. Puede lograr esto de dos maneras diferentes. La forma preferida es utilizar la caja de herramientas de refactorización, que funciona antes de activar el mecanismo de generación de código. Se trata de una operación avanzada, que se explica en las capacitaciones. BluAge Para obtener más información, consulte el taller Refactorings. Este enfoque le permite conservar la capacidad de volver a generar el código Java más adelante, para beneficiarse de nuevas mejoras en el futuro, por ejemplo). La otra forma es realizar refactorización de Java regular, directamente en el código fuente generado, utilizando cualquier enfoque de refactorización de Java que desee aplicar, bajo su propia responsabilidad.



Clases relacionadas con el programa

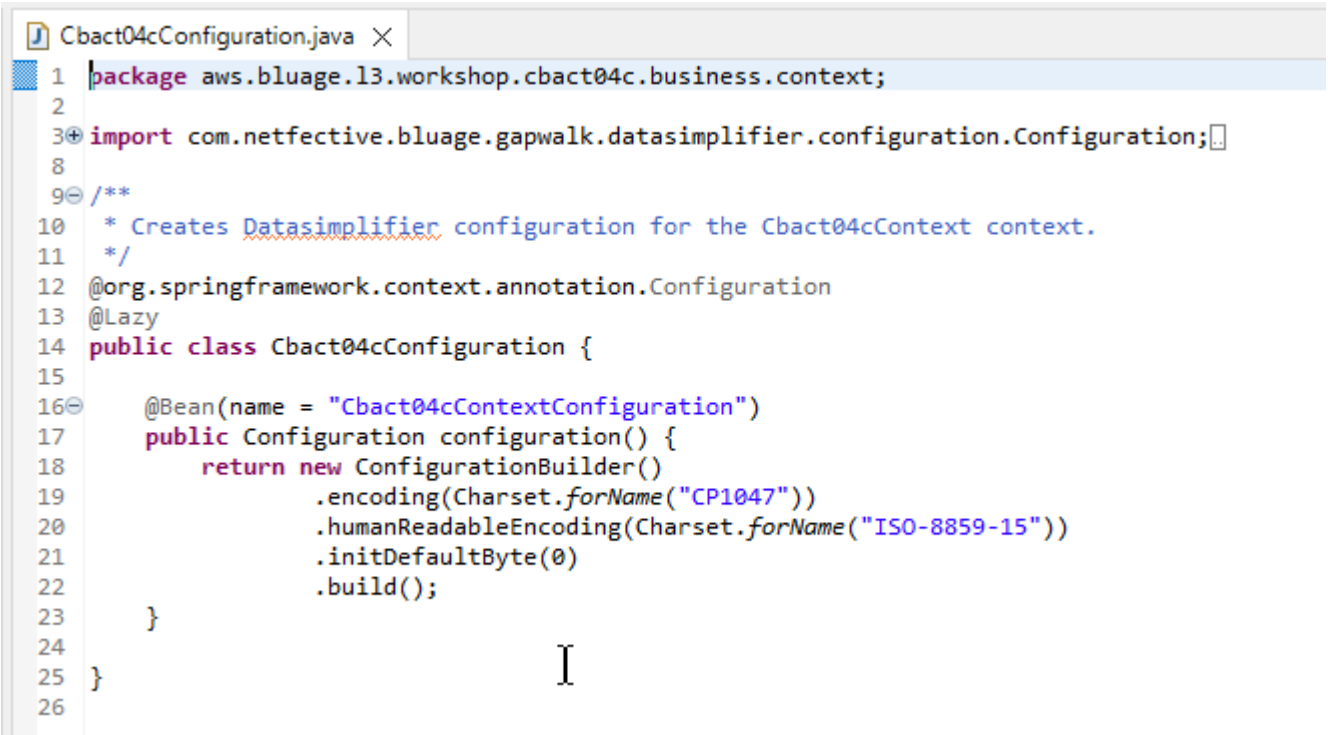
Cada programa modernizado está relacionado con dos paquetes, un paquete business.context y un paquete business.model.

- *`base package`*`.`*`program`*`.business.context`

  El subpaquete business.context contiene dos clases, una clase de configuración y una clase de contexto.

  - Una clase de configuración para el programa, que contiene detalles de configuración específicos para el programa en cuestión, como el juego de caracteres que se utilizará para representar los elementos de datos basados en caracteres, el valor

de bytes predeterminado para rellenar los elementos de la estructura de datos, etc.
El nombre de la clase termina con “Configuration”. Está marcado con la anotación `@org.springframework.context.annotation.Configuration` y contiene un único método que debe devolver un objeto `Configuration` configurado correctamente.



- Una clase de contexto, que sirve de puente entre las clases de servicios del programa (véase más abajo) y las estructuras de datos (`Record`) y los conjuntos de datos (`File`) del subpaquete del modelo (véase más abajo). El nombre de la clase termina con "Context" y es una subclase de la clase `RuntimeContext`.

Cbact04cContext.java

```
@Component("aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext")
@Import({
    aws.bluage.l3.workshop.cbact04c.business.model.TcatbalFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.XrefFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.AccountFile.class
,   aws.bluage.l3.workshop.cbact04c.business.model.TransactFile.class
})
@Lazy
@Scope("prototype")
public class Cbact04cContext extends JicsRuntimeContext {

    @Autowired
    private TcatbalFile tcatbalFile;

    @Autowired
    private XrefFile xrefFile;

    @Autowired
    private DiscgrpFile discgrpFile;

    @Autowired
    private AccountFile accountFile;

    @Autowired
    private TransactFile transactFile;

    private IndexedFile tcatbalFileFile;

    private IndexedFile xrefFileFile;

    private IndexedFile discgrpFileFile;

    private IndexedFile accountFileFile;

    private SequentialFile transactFileFile;

    private TranCatBalRecord tranCatBalRecord;
    private TcatbalfStatus tcatbalfStatus;
    private CardXrefRecord cardXrefRecord;
```

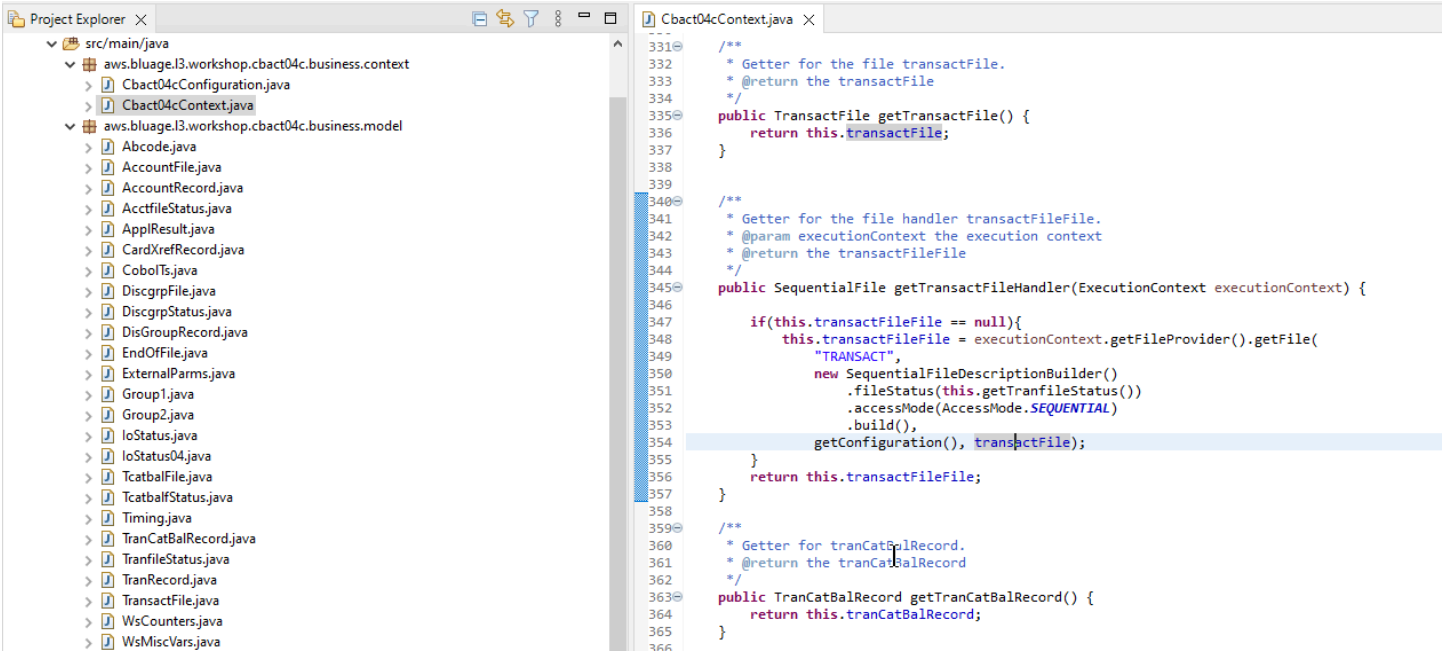
- *base package*.*program*.business.model

  El subpaquete del modelo contiene todas las estructuras de datos que puede utilizar el programa en cuestión. Por ejemplo, cualquier estructura de datos COBOL de nivel 01 corresponde a una clase del subpaquete del modelo (las estructuras de datos de nivel inferior son propiedades de la estructura de nivel 01 que poseen). Para obtener más información sobre cómo modernizamos estructuras de datos 01, consulte ¿Qué son los simplificadores de datos en AWS Blu Age?.

DiscgrpFile.java

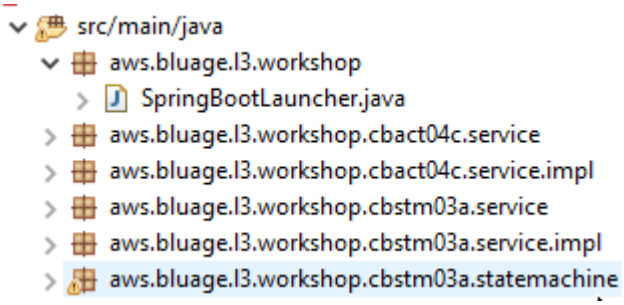
```java
package aws.bluage.l3.workshop.cbact04c.business.model;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary;
import com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group;
import com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference;
import com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType;
import com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

/**
 * Data simplifier file DiscgrpFile.
 *
 * <p>About 'fdDiscgrpRec' field, <br>uml entity: aws.bluage.l3.workshop.cbact04c.business.model.FdDiscgrpRec
 * <br></p>
 *
 */
@Component("aws.bluage.l3.workshop.cbact04c.business.model.DiscgrpFile")
@Lazy
@Scope("prototype")
public class DiscgrpFile extends RecordEntity {

    private final Group root = new Group(getData());
    private final Group fdDiscgrpRec = new Group(root);
    private final Group fdDiscgrpKey = new Group(fdDiscgrpRec);
    private final Elementary fdDisAcctGroupId = new Elementary(fdDiscgrpKey,new AlphanumericType(10));
    private final Elementary fdDisTranTypeCd = new Elementary(fdDiscgrpKey,new AlphanumericType(2));
    private final Elementary fdDisTranCatCd = new Elementary(fdDiscgrpKey,new ZonedType(4, 0, false));
    private final Elementary fdDiscgrpData = new Elementary(fdDiscgrpRec,new AlphanumericType(34));
```

Todas las clases amplían la clase `RecordEntity`, que representa el acceso a una representación de un registro empresarial. Algunos de los registros tienen un propósito especial, ya que están vinculados a un `File`. El enlace entre a `Record` y a `File` se realiza en los FileHandler métodos * correspondientes que se encuentran en la clase de contexto al crear el objeto de archivo. Por ejemplo, en la siguiente lista se muestra cómo TransactfileFile `File` está enlazado al TransactFile `Record` (del subpaquete del modelo).



Contenido del proyecto de servicio

Cada proyecto de servicio incluye una aplicación Springboot dedicada, que se utiliza como columna vertebral de la arquitectura. Esto se materializa a través de la clase denominada `SpringBootLauncher`, ubicada en el paquete base de los orígenes Java del servicio:



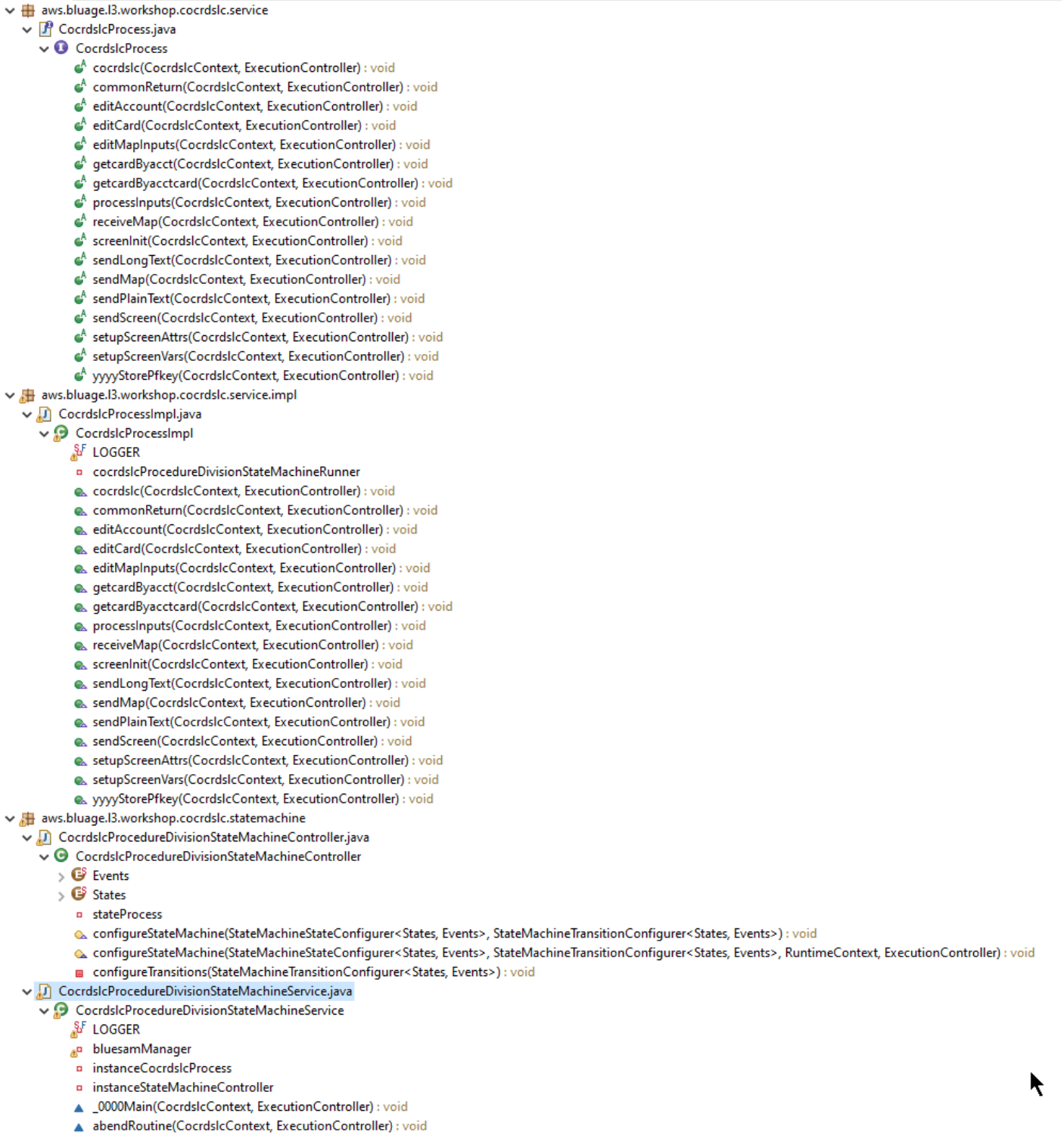
Esta clase es, en particular, responsable de:

- Hacer de enlace entre las clases de programas y los recursos administrados (orígenes de datos / administradores de transacciones / asignaciones de conjuntos de datos / etc...).
- Proporcionar un `ConfigurableApplicationContext` a los programas.
- Descubrir todas las clases marcadas como componentes de Spring (`@Component`).
- Garantizar que los programas están registrados correctamente en el `ProgramRegistry`. Consulte el método de inicialización responsable de este registro.

```
/**
 * Initialization method called when the spring application is ready.
 * Register all programs and services to the gapwalk shared context.
 * @param event the application ready event
 */
@EventListener
public void initialize(ApplicationReadyEvent event) {
    Map<String, ProgramContainer> programContainers = event.getApplicationContext().getBeansOfType(ProgramContainer.class);
    programContainers.values().forEach(ProgramRegistry::registerProgram);
    Map<String, ServiceContainer> serviceContainers = event.getApplicationContext().getBeansOfType(ServiceContainer.class);
    serviceContainers.values().forEach(ServiceRegistry::registerService);
}
```

Artefactos relacionados con el programa

Sin necesidad de una refactorización previa, las salidas de la modernización de la lógica empresarial se organizan en dos o tres paquetes por programa tradicional:



El caso más exhaustivo constará de tres paquetes:

- *base package*`.`*program*`.service`: contiene una interfaz denominada ProgramProcess, que cuenta con métodos empresariales para gestionar la lógica empresarial y preservar el flujo de control de ejecución tradicional.
- *base package*`.`*program*`.service.impl`: contiene una clase denominada ProgramProcessImpl, que es la implementación de la interfaz de procesos descrita anteriormente. Aquí es donde las declaraciones heredadas se «traducen» a declaraciones de Java, basándose en el marco de AWS Blu Age:

CocrdslcProcessImpl.java

```
210     /**
211      * Process operation sendScreen.
212      *
213      * @param ctx
214      * @param ctrl
215      */
216     @Override
217     public void sendScreen(final CocrdslcContext ctx, final ExecutionController ctrl) {
218         ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
219         ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
220         ctx.getCarddemoCommarea().setCdemoPgmReenter(true);
221         SendMapBuilder.newInstance(ctx.getDfheiblk(), ctx)
222         .withMap(ctx.getCcWorkAreas().getCcardNextMap())
223         .withMapset(ctx.getCcWorkAreas().getCcardNextMapset())
224         .withData(ctx.getGroup1().getCcrdslaoReference())
225         .withCursor()
226         .withErase()
227         .withFreeKB()
228         .execute();
229         ctx.getWsMiscStorage().setWsRespCd(ctx.getDfheiblk().getEibresp());
230     }
231
232     /**
233      * Process operation processInputs.
234      *
235      * @param ctx
236      * @param ctrl
237      */
238     @Override
239     public void processInputs(final CocrdslcContext ctx, final ExecutionController ctrl) {
240         receiveMap(ctx, ctrl);
241         editMapInputs(ctx, ctrl);
242         ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
243         ctx.getCcWorkAreas().setCcardNextProg(ctx.getWsLiterals().getLitThispgm());
244         ctx.getCcWorkAreas().setCcardNextMapset(ctx.getWsLiterals().getLitThismapset());
245         ctx.getCcWorkAreas().setCcardNextMap(ctx.getWsLiterals().getLitThismap());
246     }
247
```

- *base package*`.`*program*`.statemachine`: es posible que este paquete no esté siempre presente. Es necesaria cuando la modernización del flujo de control tradicional requiere utilizar un enfoque de máquina de estados (es decir, utilizar el StateMachine marco Spring) para cubrir adecuadamente el flujo de ejecución tradicional.

En ese caso, el subpaquete statemachine contiene dos clases:

- *Program*`ProcedureDivisionStateMachineController`: una clase que amplía una clase que implementa las interfaces `StateMachineController` (define las operaciones necesarias para controlar la ejecución de una máquina de estados) y `StateMachineRunner` (define las operaciones necesarias para ejecutar una máquina de estados), utilizadas para controlar la mecánica de la máquina de estados de Spring; por ejemplo, `SimpleStateMachineController` como en el caso de muestra.

CocrdslcProcedureDivisionStateMachineController.java

```
package aws.bluage.l3.workshop.cocrdslc.statemachine;

import aws.bluage.l3.workshop.cocrdslc.business.context.CocrdslcContext;

/**
 * Controller managing the state machine "CocrdslcProcedureDivisionStateMachine" execution.
 */
@Component("aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineController")
@Import({
aws.bluage.l3.workshop.cocrdslc.statemachine.CocrdslcProcedureDivisionStateMachineService.class
})
@Lazy
public class CocrdslcProcedureDivisionStateMachineController extends SimpleStateMachineController<States, Events> {

    /**
     * State machine states.
     */
    public enum States {
        _0000_MAIN_1, _0000_MAIN, ABEND_ROUTINE, FINAL, LOCAL_FINAL
    }

    /**
     * State machine events.
     */
    public enum Events {
        TO__0000_MAIN_1, TO__0000_MAIN, TO_ABEND_ROUTINE, TO_FINAL, TO_LOCAL_FINAL
    }

    /**
     * State machine state process service provider.
     */
    @Autowired
    @Lazy
    private CocrdslcProcedureDivisionStateMachineService stateProcess;

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        throw new UnsupportedOperationException("Please use the four arguments configureStateMachine method instead: configureStateMachine(StateMachineStateConfigurer<States, Events> states, "
                + "StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl)");
    }

    @Override
    protected void configureStateMachine(StateMachineStateConfigurer<States, Events> states, StateMachineTransitionConfigurer<States, Events> transitions, RuntimeContext ctx, ExecutionController ctrl) throws Exception {
        StateConfigurer<States, Events> configurer = states.withStates();
        configurer.initial(States._0000_MAIN_1).end(States.FINAL);
        configurer.state(States._0000_MAIN_1);
        configurer.state(States.FINAL);

        StateConfigurer<States, Events> subConfigurer = states.withStates().parent(States._0000_MAIN_1);
        subConfigurer.initial(States._0000_MAIN).end(States.LOCAL_FINAL);
        CocrdslcContext lctx = (CocrdslcContext) ctx;
        subConfigurer.state(States._0000_MAIN, buildAction(() -> {stateProcess._0000Main(lctx, ctrl);}), null);
        subConfigurer.state(States.ABEND_ROUTINE, buildAction(() -> {stateProcess.abendRoutine(lctx, ctrl);}), null);

        configureTransitions(transitions);
    }

    /**
     * Declare state machine transitions.
     * @param transitions the transitions configuration helper
     */
    private void configureTransitions(StateMachineTransitionConfigurer<States, Events> transitions) throws Exception {
        transitions.withLocal().source(States._0000_MAIN_1).target(States.ABEND_ROUTINE).event(Events.TO_ABEND_ROUTINE);
        transitions.withExternal().source(States.ABEND_ROUTINE).target(States.FINAL).event(Events.TO_FINAL);
    }
}

```

El controlador de la máquina de estados define los diferentes estados posibles y las transiciones entre ellos, lo que reproduce el flujo de control de ejecución heredado para el programa dado.

Al crear la máquina de estados, el controlador hace referencia a los métodos que se definen en la clase de servicio asociada ubicada en el paquete de la máquina de estados y que se describen a continuación:

```
subConfigurer.state(States._0000_MAIN, buildAction(() ->
 {stateProcess._0000Main(lctx, ctrl);}), null);
subConfigurer.state(States.ABEND_ROUTINE, buildAction(() ->
 {stateProcess.abendRoutine(lctx, ctrl);}), null);
```

- *Program*ProcedureDivisionStateMachineService: esta clase de servicio representa una parte de la lógica empresarial que debe estar vinculada a la máquina de estados que crea el controlador de la máquina de estados, como se describió anteriormente.

  El código de los métodos de esta clase utiliza los eventos definidos en el controlador de la máquina de estados:

CocrdslcProcedureDivisionStateMachineService.java

```
59    /**
60     * State process operation _0000Main.
61     *
62     * @param ctx
63     * @param ctrl
64     */
65    void _0000Main(CocrdslcContext ctx, ExecutionController ctrl) {
66        ctx.getDfheiblk().bind(ArgUtils.get(ctx, 0));
67        ctx.getDfhcommarea().bind(ArgUtils.get(ctx, 1));
68
69        /*
70        *****************************************************************
71        Program:     COCRDSLC.CBL                                      *
72        Layer:       Business logic                                    *
73        Function:    Accept and process credit card detail request     *
74        *****************************************************************
75        Copyright Amazon.com, Inc. or its affiliates.
76        All Rights Reserved.
77        Licensed under the Apache License, Version 2.0 (the \"License\").
78        You may not use this file except in compliance with the License.
79        You may obtain a copy of the License at
80        http://www.apache.org/licenses/LICENSE-2.0
81        Unless required by applicable law or agreed to in writing,
82        software distributed under the License is distributed on an
83        \"AS IS\" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND,
84        either express or implied. See the License for the specific
85        language governing permissions and limitations under the License
86        *****************************************************************
87        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:16:00 CDT */
88        instanceStateMachineController.registerSignalHandler(Events.TO_ABEND_ROUTINE, "!ABEND");
89        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).execute().handleException();
90        ctx.getCcWorkAreas().getCcWorkAreaReference().getField().initialize();
91        ctx.getWsMiscStorage().getField().initialize();
92        DataUtils.initialize(ctx.getWsCommarea().getWsCommareaReference());
93
```

CocrdslcProcedureDivisionStateMachineService.java

```
     *
     * @param ctx
     * @param ctrl
     */
    void abendRoutine(CocrdslcContext ctx, ExecutionController ctrl) {
        if (DataUtils.isLowValue(ctx.getAbendData().getAbendMsgReference())) {
            ctx.getAbendData().setAbendMsg("UNEXPECTED ABEND OCCURRED.");
        }
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        SendTextBuilder.newInstance(ctx.getDfheiblk(), ctx)
        .withData(ctx.getAbendData())
        .withLength(134)
        .execute();
        HandleAbendBuilder.newInstance(ctx.getDfheiblk(), ctx).cancel().execute().handleException();
        AbendBuilder.newInstance(ctx.getDfheiblk(), ctx).withAbendCode("9999").execute().handleException();

        /*
        Ver: CardDemo_v1.0-15-g27d6c6f-68 Date: 2022-07-19 23:12:33 CDT */
        instanceStateMachineController.sendEvent(Events.TO_FINAL);

    }
}

```

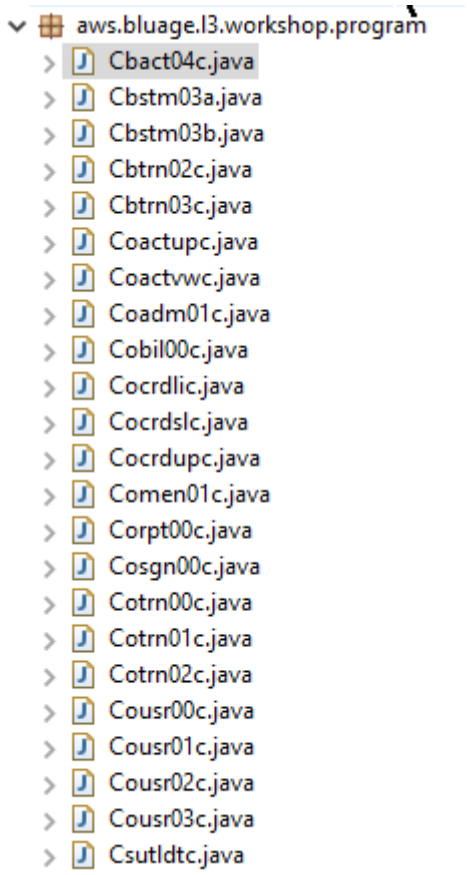
El servicio statemachine también realiza llamadas a la implementación del servicio de procesos descrita anteriormente:

CocrdslcProcedureDivisionStateMachineService.java

```
        /*
        ************************************************************
        COMING FROM CREDIT CARD LIST SCREEN
        SELECTION CRITERIA ALREADY VALIDATED
        ************************************************************ */
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter() && DataUtils.compare(ctx.getCarddemoCommarea().getCdemoFromProgramReference(), ctx.getWsLiterals().getLitCclistpgmReference()) == 0) {
        ctx.getWsMiscStorage().setInputOk(true);
        ctx.getCcWorkAreas().setCcAcctIdN(ctx.getCarddemoCommarea().getCdemoAcctId());
        ctx.getCcWorkAreas().setCcCardNumN(ctx.getCarddemoCommarea().getCdemoCardNum());
        instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmEnter()) {

        /*
        ************************************************************
        COMING FROM SOME OTHER CONTEXT
        SELECTION CRITERIA TO BE GATHERED
        ************************************************************ */
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    } else if (ctx.getCarddemoCommarea().isCdemoPgmReenter()) {
        instanceCocrdslcProcess.processInputs(ctx, ctrl);
        if (ctx.getWsMiscStorage().isInputError()) {
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        } else {
            instanceCocrdslcProcess.getcardByacctcard(ctx, ctrl);
            instanceCocrdslcProcess.sendMap(ctx, ctrl);
            instanceCocrdslcProcess.commonReturn(ctx, ctrl);
        }
    } else {
        ctx.getAbendData().setAbendCulprit(ctx.getWsLiterals().getLitThispgm());
        ctx.getAbendData().setAbendCode("0001");
        DataUtils.setToBlank(ctx.getAbendData().getAbendReasonReference());
        ctx.getWsMiscStorage().setWsReturnMsg("UNEXPECTED DATA SCENARIO");
        instanceCocrdslcProcess.sendPlainText(ctx, ctrl);
    }

    /*
    If we had an error setup error message that slipped through
    Display and return */
    if (ctx.getWsMiscStorage().isInputError()) {
        ctx.getCcWorkAreas().setCcardErrorMsg(ctx.getWsMiscStorage().getWsReturnMsg());
        instanceCocrdslcProcess.sendMap(ctx, ctrl);
        instanceCocrdslcProcess.commonReturn(ctx, ctrl);
    }
    instanceCocrdslcProcess.commonReturn(ctx, ctrl);
```

Además, un paquete *`base package`*`.program` desempeña un papel importante, ya que reúne una clase por programa, que servirá como punto de entrada al programa (más detalles sobre esto

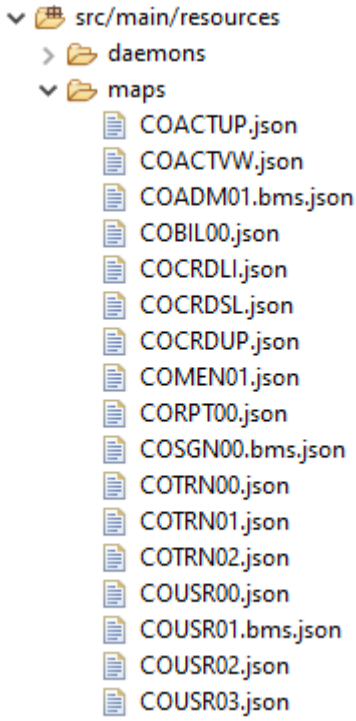
más adelante). Cada clase implementa la interfaz de `Program`, marcador del punto de entrada del programa.



Otros artefactos

- Compañeros de BMS MAPs

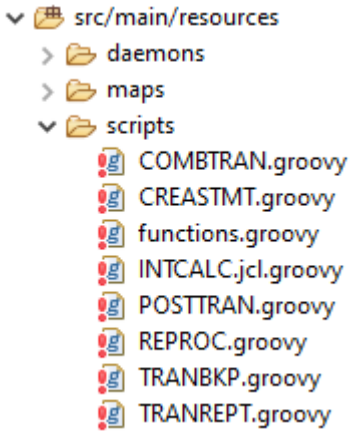
  Además de los artefactos relacionados con el programa, el proyecto de servicio puede contener otros artefactos para diversos fines. En el caso de la modernización de una aplicación en línea del CICS, el proceso de modernización produce un archivo json y lo coloca en la carpeta de mapas de the /src/main/resources cada carpeta:



El motor de tiempo de ejecución de Blu Age consume esos archivos json para vincular los registros utilizados por la instrucción SEND MAP con los campos de la pantalla.

- Scripts groovy

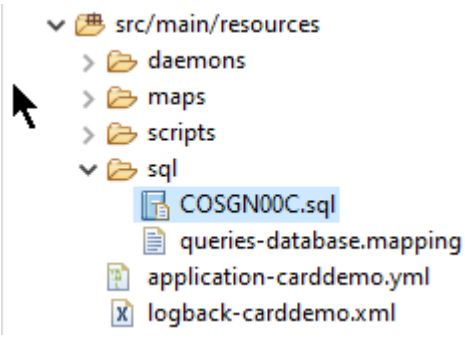
  Si la aplicación anterior tenía scripts JCL, estos se han modernizado como scripts geniales y se guardan en una the /src/main/resources/scripts carpeta (hablaremos de esa ubicación específica más adelante):



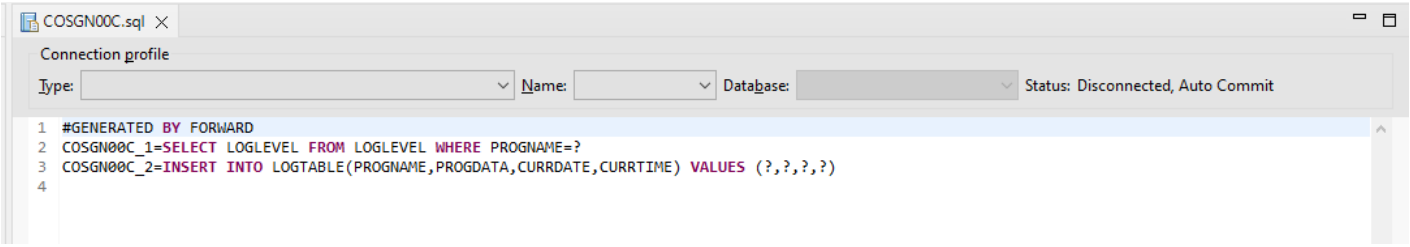
  Estos scripts se utilizan para lanzar trabajos por lotes (cargas de trabajo de procesamiento de datos dedicadas, no interactivas y con un uso intensivo de la CPU).

- Archivos SQL

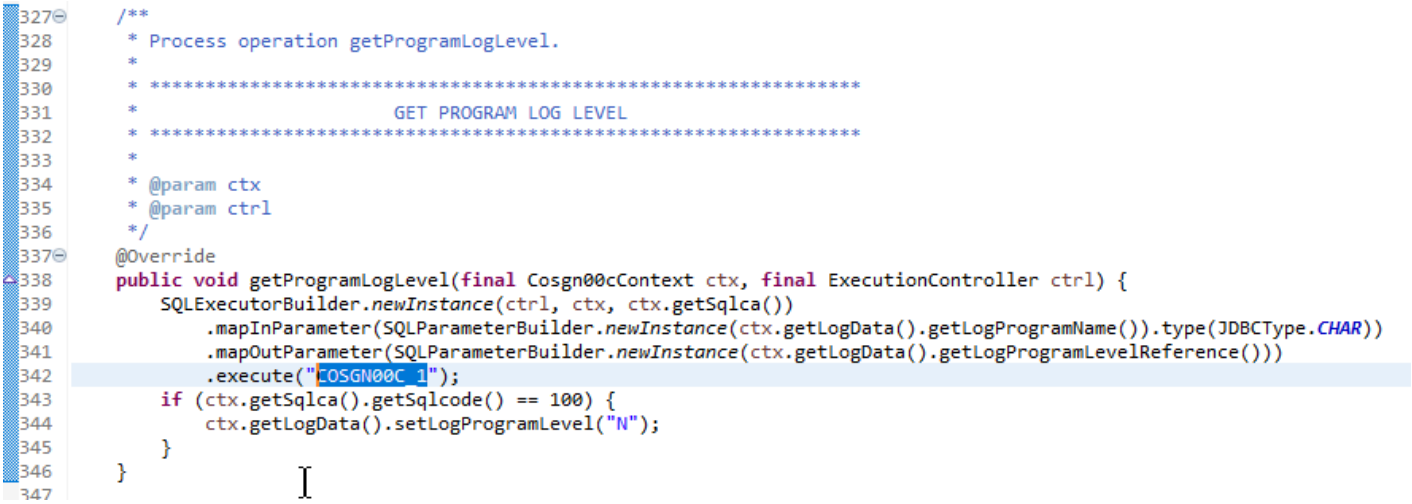
Si la aplicación antigua utilizaba consultas SQL, las consultas SQL modernizadas correspondientes se habrán recopilado en archivos de propiedades dedicados, con el patrón de nomenclatura program.sql, donde program es el nombre del programa que utiliza esas consultas.



El contenido de esos archivos sql es una colección de entradas (clave=consulta), donde cada consulta está asociada a una clave única, que el programa modernizado utiliza para ejecutar la consulta en cuestión:



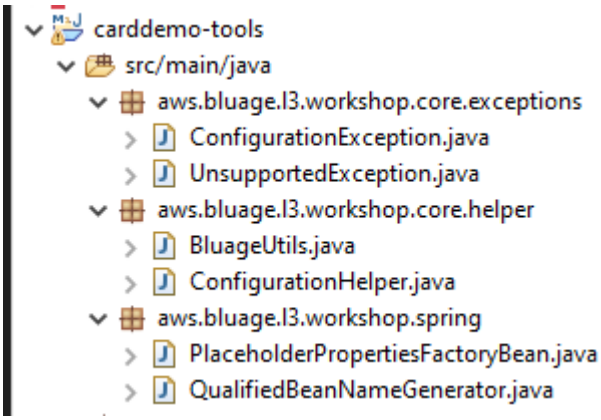
Por ejemplo, el programa COSGN00C ejecuta la consulta con la clave "COSGN00C_1" (la primera entrada del archivo sql):



```
327    /**
328     * Process operation getProgramLogLevel.
329     *
330     * ****************************************************************
331     *                       GET PROGRAM LOG LEVEL
332     * ****************************************************************
333     *
334     * @param ctx
335     * @param ctrl
336     */
337    @Override
338    public void getProgramLogLevel(final Cosgn00cContext ctx, final ExecutionController ctrl) {
339        SQLExecutorBuilder.newInstance(ctrl, ctx, ctx.getSqlca())
340            .mapInParameter(SQLParameterBuilder.newInstance(ctx.getLogData().getLogProgramName()).type(JDBCType.CHAR))
341            .mapOutParameter(SQLParameterBuilder.newInstance(ctx.getLogData().getLogProgramLevelReference()))
342            .execute("COSGN00C_1");
343        if (ctx.getSqlca().getSqlcode() == 100) {
344            ctx.getLogData().setLogProgramLevel("N");
345        }
346    }
347
```

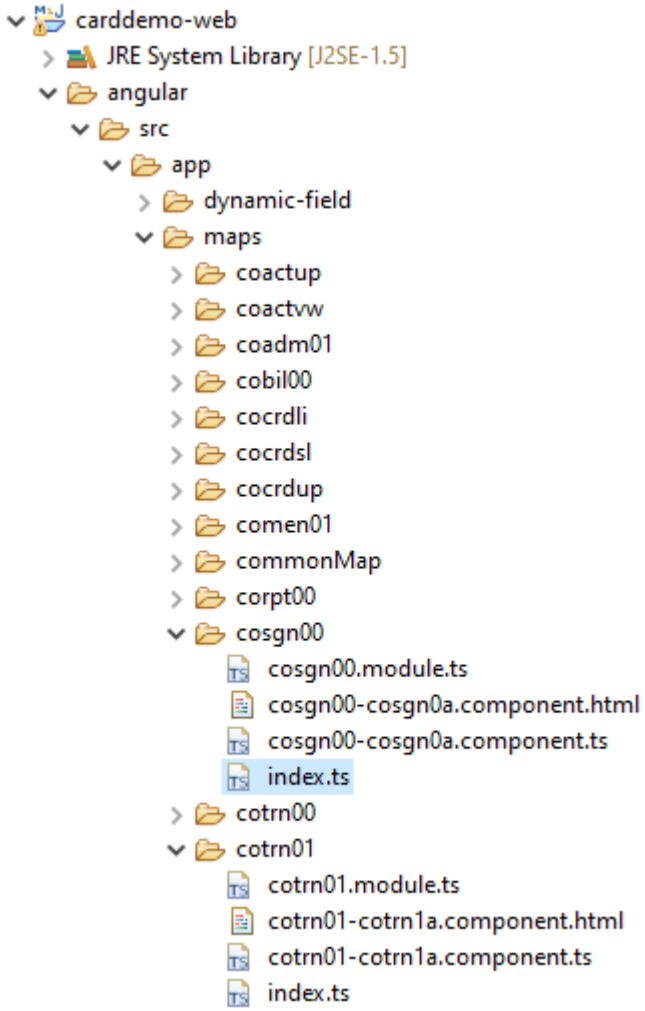
Contenido de proyecto de utilidades

El proyecto de utilidades, cuyo nombre termina con "-tools", contiene un conjunto de utilidades técnicas que podrían ser utilizadas por todos los demás proyectos.



Contenido de los proyectos web

El proyecto web solo está presente al modernizar los elementos de la interfaz de usuario heredados. Los elementos de la interfaz de usuario moderna que se utilizan para crear la interfaz de usuario modernizada de la aplicación se basan en Angular. La aplicación de ejemplo utilizada para mostrar los artefactos de modernización es una aplicación COBOL/CICS que se ejecuta en una unidad central. El sistema CICS se utiliza MAPs para representar las pantallas de la interfaz de usuario. Los elementos modernos correspondientes serán, para cada mapa, un archivo html acompañado de archivos TypeScript:



El proyecto web solo se ocupa del aspecto de la interfaz de la aplicación. El proyecto de servicio, que se basa en los proyectos de utilidades y entidades, proporciona los servicios de backend. El enlace entre el front-end y el backend se realiza a través de la aplicación web llamada Gapwalk-Application, que forma parte de la distribución de tiempo de ejecución estándar de Blu Age. AWS

Ejecutar y llamar a programas

En los sistemas heredados, los programas se compilan como ejecutables independientes que pueden autodenominarse mediante un mecanismo CALL, como la sentencia CALL de COBOL, pasando argumentos cuando es necesario. Las aplicaciones modernizadas ofrecen la misma capacidad, pero utilizan un enfoque diferente, ya que la naturaleza de los artefactos involucrados difiere de la de los antiguos.

Desde el punto de vista modernizado, los puntos de entrada de los programas son clases específicas que implementan la interfaz de `Program`, son componentes de Spring (@Component) y se encuentran en proyectos de servicio, en un paquete denominado *`base package`*`.program`.

Registro de programas

Cada vez que se inicia el servidor Tomcat que aloja las aplicaciones modernizadas, también se inicia la aplicación Springboot del servicio, lo que desencadena el registro del programa. Un nombre de registro específico `ProgramRegistry` se rellena con entradas de programa, y cada programa se registra con sus identificadores, una entrada por cada identificador de programa conocido, lo que significa que si un programa es conocido por varios identificadores diferentes, el registro contiene tantas entradas como identificadores.

El registro de un programa determinado se basa en la colección de identificadores devueltos por el método (): getProgramIdentifiers

Cbact04c.java

```java
package aws.bluage.l3.workshop.program;

import aws.bluage.l3.workshop.SpringBootLauncher;

/**
 * Reference the spring application of program CBACT04C.
 *  Provides an access to the contained program for the run unit.
 */
@Component
@Import({
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cConfiguration.class,
    aws.bluage.l3.workshop.cbact04c.business.context.Cbact04cContext.class,
    aws.bluage.l3.workshop.cbact04c.service.impl.Cbact04cProcessImpl.class
})
public class Cbact04c implements Program {
    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("CBACT04C").collect(Collectors.toSet()));

    /**
     * Main program identifier for the contained program.
     */
    private static final String programIdentifier = "CBACT04C";
    @Autowired
    PlatformTransactionManager transactionManager;

    @Autowired
    Map<String, DataSource> datasources;
    @Autowired
    BeanFactory beanFactory;
    /**
     * {@inheritDoc}
     */
    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public void updateExecutionContext(ExecutionContext executionContext) {
        executionContext.setDatasources(datasources);
        executionContext.setDatabaseSupport(ExecutionContext.DatabaseSupport.POSTGRE);
        executionContext.setSqlcaVersion(ExecutionContext.SqlcaVersion.getEnum("ansi-comp5"));
        executionContext.setTransactionManager(transactionManager);
        executionContext.setUseSQLDateNewParadigm(true);
        executionContext.setUseSQLTrimStringType(false);
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

En este ejemplo, el programa se registra una vez, con el nombre “CBACT04C” (mire el contenido de la colección de programIdentifiers). Los registros tomcat muestran todos los registros del programa. El registro del programa solo depende de los identificadores del programa declarados y no del nombre de la clase del programa en sí (aunque normalmente los identificadores y los nombres de las clases del programa se corresponden).

El mismo mecanismo de registro se aplica a los programas utilitarios incluidos en las distintas aplicaciones web utilitarias de AWS Blu Age, que forman parte de la distribución en tiempo de ejecución de AWS Blu Age. Por ejemplo, la Gapwalk-Utility-Pgm aplicación web proporciona los

equivalentes funcionales de las utilidades del sistema z/OS (IDCAMS, ICEGENER, SORT, etc.) y puede invocarse mediante programas o scripts modernizados. Todos los programas de utilidades disponibles que se registran al iniciar Tomcat se registran en los registros de Tomcat.

Registro de scripts y daemons

En el momento de iniciar Tomcat, se lleva a cabo un proceso de registro similar para los scripts groovy que se encuentran en la jerarquía de carpetas. the /src/main/resources/scripts Se recorre la jerarquía de carpetas de scripts y todos los scripts groovy que se descubren (excepto el script reservado functions.groovy especial) se registran en el `ScriptRegistry`, utilizando su nombre abreviado (la parte del nombre del archivo de script que aparece antes del primer punto) como clave para la recuperación.

> **Note**
>
> - Si varios scripts tienen nombres de archivo que dan como resultado la misma clave de registro, solo se registra la última, lo que sobrescribe cualquier registro que se haya encontrado anteriormente para esa clave determinada.
> - Teniendo en cuenta lo anterior, preste atención al utilizar subcarpetas, ya que el mecanismo de registro aplana la jerarquía y podría provocar sobrescrituras inesperadas. La jerarquía no cuenta en el proceso de registro: .groovy. typically /scripts/A/myscript.groovy and /scripts/B/myscript.groovy will lead to /scripts/B/myscript.groovy overwriting /scripts/A/myscript

Los scripts groovy de la the /src/main/resources/daemons carpeta se gestionan de forma un poco diferente. Se siguen registrando como scripts normales, pero además, se lanzan una vez, directamente al iniciar Tomcat, de forma asíncrona.

Una vez registrados los scripts en el `ScriptRegistry`, se pueden ejecutar mediante una llamada REST, utilizando los puntos de conexión dedicados que expone la aplicación Gapwalk. Para obtener más información, consulte la documentación correspondiente.

Programas que llaman a programas

Un programa puede llamar a otro programa como subprograma, pasándole parámetros. Para ello, los programas utilizan una implementación de la interfaz de `ExecutionController` (la mayoría de las veces, se trata de una instancia de `ExecutionControllerImpl`), junto con un mecanismo de API fluido llamado por el `CallBuilder` para crear los argumentos de llamada al programa.

Todos los métodos de los programas utilizan `RuntimeContext` y `ExecutionController` como argumentos de método, por lo que `ExecutionController` siempre está disponible para llamar a otros programas.

Véase, por ejemplo, el siguiente diagrama, que muestra cómo el programa CBST03A llama al programa CBST03B como subprograma y le pasa los parámetros:

Cbstm03aProcessImpl.java

```
67      /**
68       * Process operation xreffileGetNext.
69       *
70       * ----------------------------------------------------------------*
71       *
72       * @param ctx
73       * @param ctrl
74       */
75      @Override
76      public void xreffileGetNext(final Cbstm03aContext ctx, final ExecutionController ctrl) {
77          ctx.getWsM03bArea().setWsM03bDd("XREFFILE");
78          ctx.getWsM03bArea().setM03bRead(true);
79          DataUtils.setToZeroes(ctx.getWsM03bArea().getWsM03bRcReference());
80          DataUtils.setToBlank(ctx.getWsM03bArea().getWsM03bFldtReference());
81          ctrl.callSubProgram("CBSTM03B", CallBuilder.newInstance()
82              .byReference(ctx.getWsM03bArea())
83              .getArguments(), ctx);
84          if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "00") == 0) {
85
86              /*
87              Do nothing */
88          } else if (DataUtils.compare(ctx.getWsM03bArea().getWsM03bRcReference(), "10") == 0) {
89              ctx.getMiscVariables().setEndOfFile("Y");
90          } else {
91              if (LOGGER.isInfoEnabled()) LOGGER.info("ERROR READING XREFFILE");
92              if (LOGGER.isInfoEnabled()) LOGGER.info("{}{}" , "RETURN CODE: " , ctx.getWsM03bArea().getWsM03bRc());
93              abendProgram(ctx, ctrl);
94          }
95          ctx.getCardXrefRecord().setBytes(ctx.getWsM03bArea().getWsM03bFldtReference().getBytes());
96      }
```

- El primer argumento de `ExecutionController.callSubProgram` es un identificador del programa al que se va a llamar (es decir, uno de los identificadores utilizados para el registro del programa; consulte los párrafos anteriores).
- El segundo argumento, que es el resultado de la creación de `CallBuilder`, es una matriz de `Record`, que corresponde a los datos que se transmiten del programa que hace la llamada al programa que recibe la llamada.
- El tercer y último argumento es la instancia `RuntimeContext` del programa que hace la llamada.

Los tres argumentos son obligatorios y no pueden ser nulos, pero el segundo argumento puede ser una matriz vacía.

El programa que llama solo podrá procesar los parámetros transmitidos si se ha diseñado originalmente para ello. En el caso de un programa COBOL antiguo, esto significa disponer de una

sección LINKAGE y una cláusula USING para que la división de procedimientos pueda utilizar los elementos de LINKAGE.

Por ejemplo, consulte el archivo de origen de COBOL CBSTM03B.CBL correspondiente:

github.com/aws-samples/aws-mainframe-modernization-carddemo/blob/main/app/cbl/CBSTM03B.CBL

```
 98
 99          LINKAGE SECTION.
100          01  LK-M03B-AREA.
101              05  LK-M03B-DD          PIC X(08).
102              05  LK-M03B-OPER        PIC X(01).
103                88  M03B-OPEN       VALUE 'O'.
104                88  M03B-CLOSE      VALUE 'C'.
105                88  M03B-READ       VALUE 'R'.
106                88  M03B-READ-K     VALUE 'K'.
107                88  M03B-WRITE      VALUE 'W'.
108                88  M03B-REWRITE    VALUE 'Z'.
109              05  LK-M03B-RC          PIC X(02).
110              05  LK-M03B-KEY         PIC X(25).
111              05  LK-M03B-KEY-LN      PIC S9(4).
112              05  LK-M03B-FLDT        PIC X(1000).
113
114          PROCEDURE DIVISION USING LK-M03B-AREA.
115
```

Por lo tanto, el programa CBSTM03B toma un solo `Record` como parámetro (una matriz de tamaño 1). Esto es lo que `CallBuilder` está creando mediante el encadenamiento de los métodos byReference() y getArguments().

La clase de API fluida `CallBuilder` tiene varios métodos disponibles para rellenar la matriz de argumentos y pasarlos al programa que hace la llamad:

- asPointer (RecordAdaptable): agrega un argumento del tipo puntero, como referencia. El puntero representa la dirección de una estructura de datos de destino.
- byReference (RecordAdaptable): añade un argumento por referencia. El programa que hace la llamada verá las modificaciones que realiza el programa que recibe la llamada.
- byReference (RecordAdaptable): variante varargs del método anterior.
- byValue(Object): agrega un argumento, transformado en `Record`, por valor. El programa que hace la llamada no verá las modificaciones que realiza el programa que recibe la llamada.

- byValue (RecordAdaptable): igual que el método anterior, pero el argumento está disponible directamente como. `RecordAdaptable`
- byValueWithBounds (Object, int, int): agrega un argumento, transformado en a`Record`, extrayendo la parte de la matriz de bytes definida por los límites dados, por valor.

Por último, el método getArguments recopilará todos los argumentos agregados y los devolverá como una matriz de `Record`.

> **Note**
>
> Es responsabilidad del programa que hace la llamada asegurarse de que la matriz de argumentos tenga el tamaño requerido, que los elementos estén ordenados correctamente y que sean compatibles, en términos de diseño de memoria, con los diseños esperados para los elementos de enlace.

Scripts que llaman a programas

Para llamar a programas registrados desde scripts groovy, es necesario usar una instancia de clase que implemente la interfaz de `MainProgramRunner`. Por lo general, obtener una instancia de este tipo se logra mediante el uso de Spring: ApplicationContext

REPROC.groovy

```groovy
// Import
import com.netfective.bluage.gapwalk.rt.provider.ScriptRegistry
import com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
import com.netfective.bluage.gapwalk.io.support.FileConfigurationUtils
import com.netfective.bluage.gapwalk.rt.job.support.DefaultJobContext
import com.netfective.bluage.gapwalk.rt.utils.GroovyUtils
import com.netfective.bluage.gapwalk.rt.io.support.FileConfiguration
import com.netfective.bluage.gapwalk.rt.shared.AbendException
import com.netfective.bluage.gapwalk.rt.call.exception.GroovyExecutionException
// Variables
mpr = applicationContext.getBean("com.netfective.bluage.gapwalk.rt.call.ExecutionController", MainProgramRunner.class)
TreeMap mapTransfo = [:]
```

Cuando haya una interfaz de `MainProgramRunner` disponible, utilice el método runProgram para llamar a un programa y pasar el identificador del programa de destino como parámetro:

REPROC.groovy

```
//*******************************************************************
//*                              STEPS                              *
//*******************************************************************
// STEP PRC001 - PGM - IDCAMS*****************************************
def stepPRC001(Object shell, Map params, Map programResults){
    shell.with {
        if (checkValidProgramResults(programResults)) {
            return execStep("PRC001", "IDCAMS", programResults, {
                mpr
                    .withFileConfigurations(new FileConfigurationUtils()
                        .systemOut("SYSPRINT")
                        .output("*")
                        .build()
                        .bluesam("FILEIN")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .bluesam("FILEOUT")
                        .dataset("NULLFILE")
                        .disposition("SHR")
                        .build()
                        .fileSystem("SYSIN")
                        .path("&CNTLLIB(REPROCT)")
                        .disposition("SHR")
                        .build()
                        .getFileConfigurations(fcmap))
                    .withParameters(params)
                    .runProgram("IDCAMS")
                })
        }
    }
}
```

En el ejemplo anterior, un paso de trabajo llama a IDCAMS (programa de utilidades de gestión de archivos) y asigna las definiciones reales de los conjuntos de datos a sus identificadores lógicos.

Cuando se trata de conjuntos de datos, los programas antiguos utilizan principalmente nombres lógicos para identificar los conjuntos de datos. Cuando se llama al programa desde un script, el script debe hacer corresponder los nombres lógicos con los conjuntos de datos físicos reales. Estos conjuntos de datos pueden estar en el sistema de archivos, en un almacenamiento de Blusam o incluso estar definidos por una secuencia en línea, la concatenación de varios conjuntos de datos o la generación de un GDG.

Utilice el withFileConfiguration método para crear un mapa lógico a físico de conjuntos de datos y ponerlo a disposición del programa llamado.

Escriba su propio programa

Escribir un programa propio para ejecutar scripts u otros programas modernizados es una tarea habitual. Por lo general, en los proyectos de modernización, se escriben programas propios cuando un programa ejecutable antiguo está escrito en un lenguaje que el proceso de modernización no admite, o cuando los orígenes se han perdido (sí, eso puede suceder), o cuando el programa es una utilidad cuyos orígenes no están disponibles.

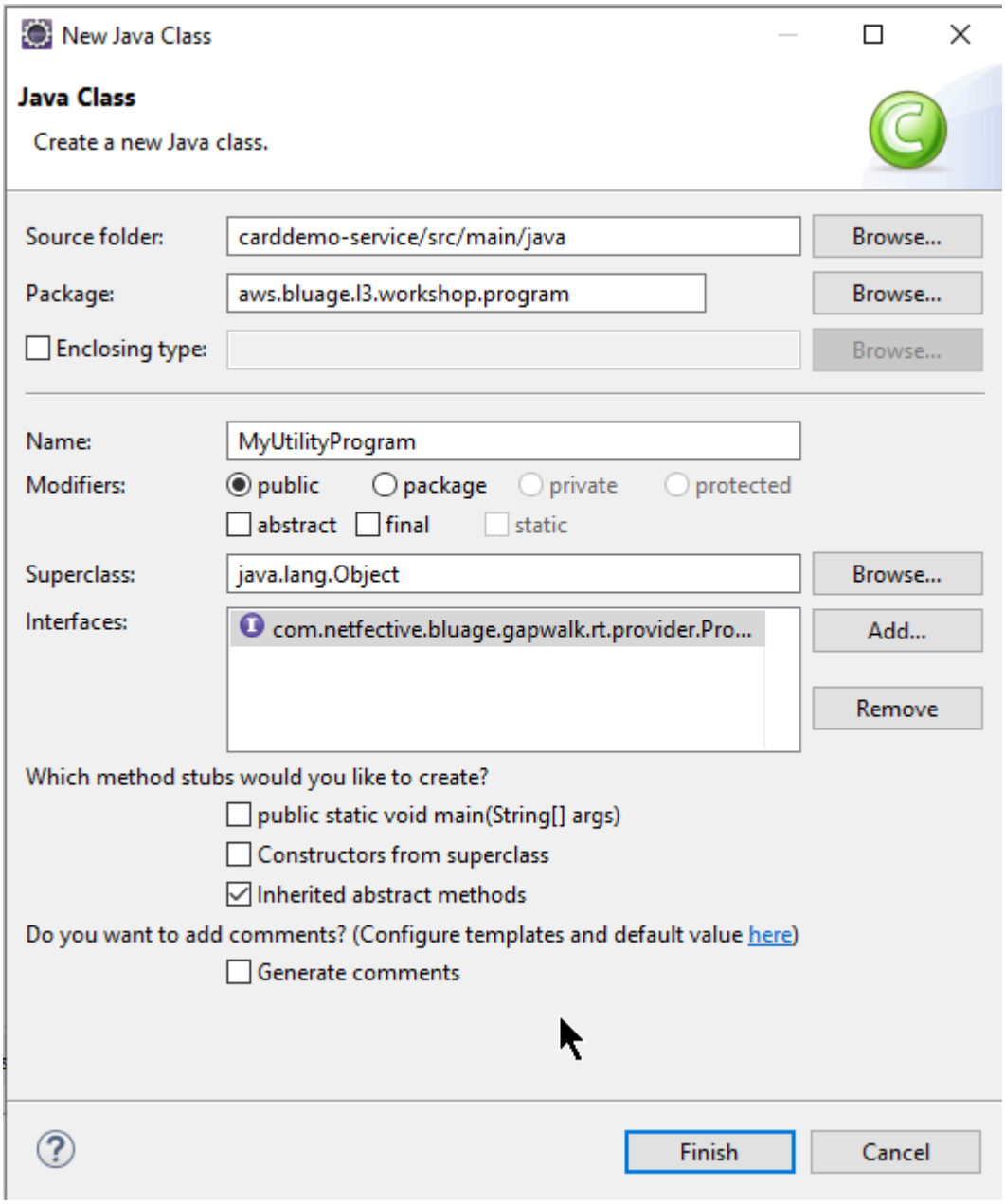
En ese caso, puede que tenga que escribir el programa que falta, en Java, usted mismo (suponiendo que tenga suficiente conocimiento sobre cuál debería ser el comportamiento esperado del programa, el diseño de memoria de los argumentos del programa, si los hubiera, etc.) Su programa Java debe cumplir con la mecánica del programa descrita en este documento, de modo que otros programas y scripts puedan ejecutarlo.

Para asegurarse de que el programa se puede utilizar, debe completar dos pasos obligatorios:

- Escriba una clase que implemente la interfaz de `Program` correctamente, de modo que se pueda registrar y llamar a ella.
- Asegúrese de que el programa esté registrado correctamente, de modo que sea visible desde otros programas/scripts.

Escribir la implementación del programa

Use su IDE para crear una nueva clase Java que implemente la interfaz de `Program`:



La siguiente imagen muestra el IDE de Eclipse, que se encarga de crear todos los métodos obligatorios que deben implementarse:

MyUtilityProgram.java

```java
package aws.bluage.l3.workshop.program;

import java.util.Set;

public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public Context getContext() {
        // TODO Auto-generated method stub
        return null;
    }

    @Override
    public void run(ExecutionController ctrl) {
        // TODO Auto-generated method stub

    }

}
```

Integración con Spring

En primer lugar, la clase debe declararse como un componente de Spring. Anote la clase con la anotación `@Component`:

```java
import org.springframework.context.ConfigurableApplicationContext;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.rt.call.ExecutionController;
import com.netfective.bluage.gapwalk.rt.context.Context;
import com.netfective.bluage.gapwalk.rt.provider.Program;

import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
public class MyUtilityProgram implements Program {
```

A continuación, implemente los métodos requeridos correctamente. En el contexto de este ejemplo, añadimos el paquete `MyUtilityProgram` que ya contiene todos los programas modernizados. Esa

ubicación permite al programa usar la aplicación Springboot existente para proporcionar lo necesario `ConfigurableApplicationContext` para la implementación del getSpringApplication método:

```
public class MyUtilityProgram implements Program {

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }
```

Puede elegir una ubicación diferente para su programa. Por ejemplo, puede ubicar el programa en otro proyecto de servicio dedicado. Asegúrese de que el proyecto de servicio en cuestión tenga su propia aplicación Springboot, que permita recuperar el ApplicationContext (que debería ser un`ConfigurableApplicationContext`).

Dar una identidad al programa

Para que otros programas y scripts puedan invocarlo, el programa debe tener al menos un identificador, que no debe colisionar con ningún otro programa registrado existente en el sistema. La elección del identificador puede estar motivada por la necesidad de sustituir un programa antiguo existente; en ese caso, tendrá que utilizar el identificador esperado, tal y como se indica en las apariciones de CALL que se encuentran en los programas antiguos. La mayoría de los identificadores del programa tienen una longitud de 8 caracteres en los sistemas antiguos.

Crear un conjunto de identificadores no modificables en el programa es una forma de hacerlo. En el siguiente ejemplo, se muestra la elección de "MYUTILPG" como identificador único:

```
@Component
public class MyUtilityProgram implements Program {

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));


    public ConfigurableApplicationContext getSpringApplication() {

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }
```

Asocie el programa a un contexto

El programa necesita una instancia de `RuntimeContext` complementaria. En el caso de los programas modernizados, AWS Blu Age genera automáticamente el contexto complementario, utilizando las estructuras de datos que formaban parte del programa anterior.

Si está escribiendo su propio programa, también debe escribir el contexto complementario.

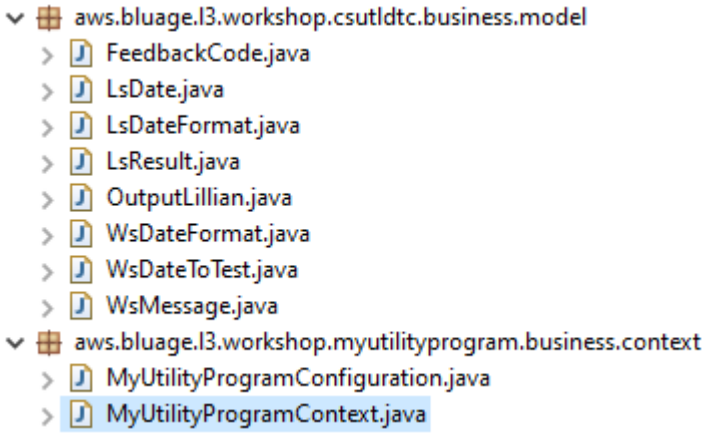
Consulte Clases relacionadas con el programa para ver que un programa requiere al menos dos clases complementarias:

- una clase de configuración
- una clase de contexto que usa la configuración

Si el programa de utilidades usa alguna estructura de datos adicional, también debería escribirse y usarse según el contexto.

Esas clases deben estar en un paquete que forme parte de una jerarquía de paquetes que se escaneará al iniciar la aplicación, para garantizar que el marco de Spring gestione el componente de contexto y la configuración.

A continuación, escribamos una configuración y un contexto mínimos en el paquete de *`base package`*`.myutilityprogram.business.context` recién creado en el proyecto de entidades:



Este es el contenido de la configuración: Utiliza una compilación de configuración similar a la de otros programas (modernizados) cercanos. Probablemente tenga que personalizarlo para sus necesidades específicas.

MyUtilityProgramConfiguration.java

```java
package aws.bluage.l3.workshop.myutilityprogram.business.context;

import java.nio.charset.Charset;

import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Lazy;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder;

/**
 * Creates Datasimplifier configuration for the MyUtilityProgram context.
 */
@org.springframework.context.annotation.Configuration
@Lazy
public class MyUtilityProgramConfiguration {

    @Bean(name = "MyUtilityProgramContextConfiguration")
    public Configuration configuration() {
        return new ConfigurationBuilder()
                .encoding(Charset.forName("CP1047"))
                .humanReadableEncoding(Charset.forName("ISO-8859-15"))
                .initDefaultByte(0)
                .build();
    }
}

```

Notas:

- La convención de nomenclatura general es ProgramNameConfiguración.
- Debe usar las anotaciones @org.springframework.context.annotation.Configuration y @Lazy
- El nombre del bean suele seguir la ProgramNameContextConfiguration convención, pero no es obligatorio. Asegúrese de evitar conflictos de nombres de beans en todo el proyecto.
- El único método a implementar debe devolver un objeto `Configuration`. Use la API fluida de `ConfigurationBuilder` como ayuda para crear una.

Y el contexto asociado:

MyUtilityProgramContext.java

```java

import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.context.annotation.Lazy;
import org.springframework.context.annotation.Scope;
import org.springframework.stereotype.Component;

import com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration;
import com.netfective.bluage.gapwalk.rt.context.RuntimeContext;

@Component("aws.bluage.l3.workshop.mytuilityprogram.business.context.MyUtilityProgramContext")
@Lazy
@Scope("prototype")
public class MyUtilityProgramContext extends RuntimeContext{

    protected MyUtilityProgramContext(@Qualifier("MyUtilityProgramContextConfiguration") Configuration configuration) {
        super(configuration);
    }

    @Override
    public void cleanUp() {
        // TODO implement clean-up of associated data structures if any
    }

    @Override
    protected void doReset() {
        // TODO implement reset of associated data structures if any
    }

}

```

Notas

- La clase de contexto debe ampliar una implementación de interfaz de `Context` existente (`RuntimeContext` o `JicsRuntimeContext`, que es una `RuntimeContext` mejorada con elementos específicos de JICS).
- La convención general de nomenclatura es ProgramNameel contexto.
- Debe declararlo como un componente de Prototype y utilizar la anotación @Lazy.
- El constructor hace referencia a la configuración asociada y utiliza la anotación @Qualifier para dirigirse a la clase de configuración adecuada.
- Si el programa de utilidades utiliza algunas estructuras de datos adicionales, deberían ser:
  - escritas y añadidas al paquete *`base package`*`.business.model`
  - referenciadas en el contexto. Eche un vistazo a otras clases de contexto existentes para ver cómo hacer referencia a las clases de estructuras de datos y adaptar los métodos de contexto (constructor, limpieza o restablecimiento) según sea necesario.

Ahora que hay un contexto específico disponible, deje que el nuevo programa lo use:

MyUtilityProgram.java

```java
import aws.bluage.l3.workshop.SpringBootLauncher;

@Component
@Import({
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramConfiguration.class,
    aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class
})
public class MyUtilityProgram implements Program {

    @Autowired
    BeanFactory beanFactory;

    /**
     * Unique identifiers for the contained program.
     */
    private static final Set<String> programIdentifiers = Collections.unmodifiableSet(Stream.of("MYUTILPG").collect(Collectors.toSet()));

    private static final String programIdentifier = "MYUTILPG";

    @Override
    public ConfigurableApplicationContext getSpringApplication() {
        return SpringBootLauncher.getCac();
    }

    @Override
    public Set<String> getProgramIdentifiers() {
        return programIdentifiers;
    }

    /**
     * {@inheritDoc}
     */
    @Override
    public String getProgramMainIdentifier() {
        return programIdentifier;
    }


    @Override
    public Context getContext() {
        return ProgramContextStore.getOrCreate(
        getProgramMainIdentifier(),
        aws.bluage.l3.workshop.myutilityprogram.business.context.MyUtilityProgramContext.class,
        beanFactory);
    }

```

Notas:

- El método getContext debe implementarse estrictamente como se muestra, utilizando una delegación al getOrCreate método de la `ProgramContextStore` clase y al Spring `BeanFactory` cableado automáticamente. Se utiliza un único identificador de programa para almacenar el contexto del programa en el `ProgramContextStore`; este identificador se denomina “identificador principal del programa”.
- Se debe hacer referencia a las clases complementarias de configuración y contexto mediante la anotación `@Import` de Spring.

Implementación de la lógica empresarial

Cuando el esquema del programa esté completo, implemente la lógica empresarial para el nuevo programa de utilidades.

Hágalo según el método `run` del programa. Este método se ejecutará cada vez que se llame al programa, ya sea a través de otro programa o mediante un script.

¡Feliz programación!

Gestionar el registro del programa

Por último, asegúrese de que el nuevo programa esté registrado correctamente en el `ProgramRegistry`. Si ha agregado el nuevo programa al paquete que ya contiene otros programas, no hay nada más que hacer. El nuevo programa se selecciona y se registra en todos sus programas vecinos al iniciar la aplicación.

Si elige otra ubicación para el programa, debe asegurarse de que el programa esté registrado correctamente al iniciar Tomcat. Para inspirarse sobre cómo hacerlo, consulte el método de inicialización de las SpringbootLauncher clases generadas en los proyectos de servicio (consulteContenido del proyecto de servicio).

Compruebe los registros de inicio de Tomcat. Se registran todos los registros de programas. Si el programa se ha registrado correctamente, encontrará la entrada de registro correspondiente.

Cuando esté seguro de que su programa está registrado correctamente, puede empezar a iterar con la codificación de la lógica empresarial.

Asignaciones de nombres totalmente cualificadas

Esta sección contiene listas de la era AWS azul y de mapeos de nombres totalmente cualificados de terceros para utilizarlos en sus aplicaciones modernizadas.

AWS Asignaciones de nombres totalmente calificadas de Blu Age

Nombre corto	Nombre completo cualificado
`CallBuilder`	`com.netfective.bluage.gapwalk.runtime.statements.CallBuilder`
`Configuration`	`com.netfective.bluage.gapwalk.datasimplifier.configuration.Configuration`

Nombre corto	Nombre completo cualificado
ConfigurationBuilder	com.netfective.bluage.gapwalk.datasimplifier.configuration.ConfigurationBuilder
ExecutionController	com.netfective.bluage.gapwalk.rt.call.ExecutionController
ExecutionControllerImpl	com.netfective.bluage.gapwalk.rt.call.internal.ExecutionControllerImpl
File	com.netfective.bluage.gapwalk.rt.io.File
MainProgramRunner	com.netfective.bluage.gapwalk.rt.call.MainProgramRunner
Program	com.netfective.bluage.gapwalk.rt.provider.Program
ProgramContextStore	com.netfective.bluage.gapwalk.rt.context.ProgramContextStore
ProgramRegistry	com.netfective.bluage.gapwalk.rt.provider.ProgramRegistry
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordEntity	com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity
RuntimeContext	com.netfective.bluage.gapwalk.rt.context.RuntimeContext

Nombre corto	Nombre completo cualificado
SimpleStateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.SimpleStateMachineController
StateMachineController	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineController
StateMachineRunner	com.netfective.bluage.gapwalk.rt.statemachine.StateMachineRunner

Asignaciones de nombres totalmente cualificadas de terceros

Nombre corto	Nombre completo cualificado
@Autowired	org.springframework.beans.factory.annotation.Autowired
@Bean	org.springframework.context.annotation.Bean
BeanFactory	org.springframework.beans.factory.BeanFactory
@Component	org.springframework.stereotype.Component
ConfigurableApplicationContext	org.springframework.context.ConfigurableApplicationContext
@Import	org.springframework.context.annotation.Import
@Lazy	org.springframework.context.annotation.Lazy

¿Qué son los simplificadores de datos en AWS Blu Age?

En los sistemas de unidad central y de gama media (denominados en el siguiente tema sistemas “heredados”), los lenguajes de programación de uso frecuente, como COBOL, PL/I o RPG, proporcionan un acceso de bajo nivel a la memoria. Este acceso se centra en el diseño de la memoria, al que se accede mediante tipos nativos, como zonificados, empaquetados o alfanuméricos, y es posible que también se agreguen mediante grupos o matrices.

En un programa determinado coexiste una combinación de accesos a un determinado fragmento de memoria, a través de campos de tipos definidos y acceso directo a bytes (memoria sin procesar). Por ejemplo, los programas COBOL pasarán los argumentos a los iniciadores como conjuntos contiguos de bytes (LINKAGE), o leerán y escribirán datos de los archivos de la misma manera (registros), al tiempo que interpretarán dichos rangos de memoria con campos de tipos definidos organizados en cuadernos.

Estas combinaciones de acceso sin proceso y estructurado a la memoria, la dependencia de un diseño de memoria preciso a nivel de bytes y los tipos heredados, como zonificados o empaquetados, son características que no están disponibles de forma nativa ni son de fácil acceso en el entorno de programación Java.

Como parte de la solución de AWS Blu Age para modernizar los programas antiguos a Java, la biblioteca Data Simplifier proporciona estas construcciones a los programas Java modernizados y las expone de una manera que resulta lo más familiar posible para los desarrolladores de Java (captres/configuradores, matrices de bytes, basadas en clases). Se trata de una dependencia fundamental para el código Java modernizado generado a partir de esos programas.

Para simplificar, la mayoría de las siguientes explicaciones se basan en construcciones de COBOL, pero puede usar la misma API para la modernización del PL1 diseño de datos de RPG, ya que la mayoría de los conceptos son similares.

Temas

- Clases principales
- Enlace y acceso a los datos
- Nombres completos calificados de los tipos de Java descritos

Clases principales

Para facilitar la lectura, en este documento se utilizan los nombres abreviados en Java de las interfaces y clases de la API de AWS Blu Age. Para obtener más información, consulte Nombres completos calificados de los tipos de Java descritos.

Representación de la memoria de bajo nivel

En el nivel más bajo, la memoria (un rango contiguo de bytes al que se puede acceder de forma rápida y aleatoria) está representada por la interfaz de `Record`. Esta interfaz es esencialmente una abstracción de una matriz de bytes de un tamaño fijo. Como tal, proporciona setters y getters capaces de acceder o modificar los bytes subyacentes.

Representación de datos estructurados

Para representar datos estructurados, como “01 elementos de datos” o “01 cuadernos”, como se encuentran en COBOL DATA DIVISION, se utilizan subclases de la clase `RecordEntity`. Por lo general, no se escriben a mano, sino que se generan con las herramientas de modernización de AWS Blu Age a partir de las correspondientes construcciones heredadas. Sigue siendo útil conocer su estructura principal y su API, para poder entender cómo los usa el código de un programa modernizado. En el caso de COBOL, ese código es Java generado a partir de su PROCEDURE DIVISION.

El código generado representa cada “01 elemento de datos” con una subclase `RecordEntity`; cada campo elemental o agregado que lo compone se representa como un campo Java privado, organizado en forma de árbol (cada elemento tiene un elemento principal, excepto el raíz).

A modo ilustrativo, he aquí un ejemplo de elemento de datos de COBOL, seguido del correspondiente código generado por AWS Blu Age que lo moderniza:

```
01 TST2.
 02 FILLER PIC X(4).
 02 F1     PIC 9(2) VALUE 42.
 02 FILLER PIC X.
 02        PIC 9(3) VALUE 123.
 02 F2     PIC X VALUE 'A'.
```

```
public class Tst2 extends RecordEntity {

    private final Group root = new Group(getData()).named("TST2");
    private final Filler filler = new Filler(root,new AlphanumericType(4));
```

```
   private final Elementary f1 = new Elementary(root,new ZonedType(2, 0, false),new
BigDecimal("42")).named("F1");
   private final Filler filler1 = new Filler(root,new AlphanumericType(1));
   private final Filler filler2 = new Filler(root,new ZonedType(3, 0, false),new
BigDecimal("123"));
   private final Elementary f2 = new Elementary(root,new
AlphanumericType(1),"A").named("F2");


   /**
    * Instantiate a new Tst2 with a default record.
    * @param configuration the configuration
    */
   public Tst2(Configuration configuration) {
       super(configuration);
       setupRoot(root);
   }
   /**
    * Instantiate a new Tst2 bound to the provided record.
    * @param configuration the configuration
    * @param record the existing record to bind
    */
   public Tst2(Configuration configuration, RecordAdaptable record) {
       super(configuration);
       setupRoot(root, record);
   }

   /**
    * Gets the reference for attribute f1.
    * @return the f1 attribute reference
    */
   public ElementaryRangeReference getF1Reference() {
       return f1.getReference();
   }

   /* *
    * Getter for f1 attribute.
    * @return f1 attribute
    */
   public int getF1() {
       return f1.getValue();
   }

   /**
```

```
     * Setter for f1 attribute.
     * @param f1 the new value of f1
     */
    public void setF1(int f1) {
        this.f1.setValue(f1);
    }
    /**
     * Gets the reference for attribute f2.
     * @return the f2 attribute reference
     */
    public ElementaryRangeReference getF2Reference() {
        return f2.getReference();
    }

    /**
     * Getter for f2 attribute.
     * @return f2 attribute
     */
    public String getF2() {
        return f2.getValue();
    }

    /**
     * Setter for f2 attribute.
     * @param f2 the new value of f2
     */
    public void setF2(String f2) {
        this.f2.setValue(f2);
    }
}
```

Campos elementales

Los campos de clase `Elementary` (o `Filler`, cuando no tienen nombre) representan una “hoja” de la estructura de datos heredada. Están asociados a un espacio contiguo de bytes subyacentes (“rango”) y, por lo general, tienen un tipo (posiblemente parametrizado) que expresa cómo interpretar y modificar esos bytes (mediante la “decodificación” y la “codificación”, respectivamente, de un valor desde/hacia una matriz de bytes).

Todos los tipos elementales son subclases de `RangeType`. Los tipos frecuentes son:

Tipo COBOL	Tipo de Data Simplifier
`PIC X(n)`	`AlphanumericType`
`PIC 9(n)`	`ZonedType`
`PIC 9(n) COMP-3`	`PackedType`
`PIC 9(n) COMP-5`	`BinaryType`

Campos agregados

Los campos agregados organizan el diseño de memoria de sus contenidos (otros agregados o campos elementales). No tienen un tipo elemental en sí mismos.

Los campos `Group` representan campos contiguos en la memoria. Cada uno de sus campos contenidos está dispuesto en el mismo orden en la memoria, el primer campo tiene un desplazamiento `0` con respecto a la posición del campo de grupo en la memoria, el segundo campo tiene un desplazamiento `0 + (size in bytes of first field)`, etc. Se utilizan para representar secuencias de campos COBOL en el mismo campo contenedor.

Los campos `Union` representan varios campos que acceden a la misma memoria. Cada uno de sus campos contenidos está dispuesto con un desplazamiento `0` con respecto a la posición del campo de unión en la memoria. Por ejemplo, se utilizan para representar el constructo COBOL “REDEFINE” (el primer elemento secundario de Union es el elemento de datos redefinido, el segundo elemento secundario es su primera redefinición, etc.).

Los campos matriciales (subclases de `Repetition`) representan la repetición, en la memoria, del diseño de su campo secundario (ya sea un agregado en sí mismo o un elemento elemental). Diseñan un número determinado de estos diseños secundarios en la memoria, cada uno de los cuales tiene un desplazamiento `index * (size in bytes of child)`. Se utilizan para representar los constructos “OCCURS” de COBOL.

Elementos primitivos

En algunos casos de modernización, los elementos primitivos también se pueden utilizar para presentar elementos de datos raíz independientes. Su uso es muy similar al de `RecordEntity`, pero no provienen de él ni se basan en el código generado. En su lugar, el motor de ejecución de

AWS Blu Age los proporciona directamente como subclases de la interfaz. `Primitive` Algunos ejemplos de estas clases proporcionadas son `Alphanumeric` o`ZonedDecimal`.

Enlace y acceso a los datos

La asociación entre los datos estructurados y los datos subyacentes se puede realizar de varias maneras.

Una interfaz importante para este propósito es `RecordAdaptable`, la cual se utiliza para obtener un `Record` que ofrece una “vista modificable” de los datos subyacentes de `RecordAdaptable`. Como veremos a continuación, varias clases implementan `RecordAdaptable`. Recíprocamente, AWS Blu Age APIs y la manipulación de código de memoria de bajo nivel (como argumentos de programas, registros de E/S de archivos, área de comunicación CICS, memoria asignada...) suelen esperar a `RecordAdaptable` como identificador para esa memoria.

En el caso de la modernización de COBOL, la mayoría de los elementos de datos están asociados a una memoria que permanecerá fija durante el tiempo de ejecución del programa correspondiente. Para ello, las subclases `RecordEntity` se instancian una vez en un objeto principal generado (el contexto del programa) y se encargan de instanciar su `Record` subyacente, en función del tamaño de bytes de `RecordEntity`.

En otros casos de COBOL, como cuando se asocian elementos LINKAGE a los argumentos del programa o se moderniza el constructo SET ADDRESS OF, una instancia `RecordEntity` debe estar asociada a un `RecordAdaptable` proporcionado. Para ello, existen dos mecanismos:

- Si la instancia de `RecordEntity` ya existe, se puede usar el método `RecordEntity.bind(RecordAdaptable)` (heredado de `Bindable`) para hacer que esta instancia “apunte” a este `RecordAdaptable`. Cualquier getter o setter al que se invoque en `RecordEntity` estará respaldado (lectura o escritura de bytes) por los bytes del `RecordAdaptable` subyacente.
- Si se va a crear una instancia de `RecordEntity`, hay disponible un constructor generado que acepte a `RecordAdaptable`.

Por el contrario, se puede acceder al `Record` actualmente vinculado a los datos estructurados. Para ello, `RecordEntity` implementa `RecordAdaptable`, por lo que se `getRecord()` puede invocar en cualquier instancia de este tipo.

Por último, muchos verbos de COBOL o CICS requieren acceso a un único campo para poder leerlos o escribirlos. La clase `RangeReference` se usa para representar dicho acceso. Sus

instancias se pueden obtener a partir de los métodos `getXXXReference()` generados para `RecordEntity` (XXX es el campo al que se accede) y pasarlas a los métodos de tiempo de ejecución. `RangeReference` se utiliza normalmente para acceder a campos enteros `RecordEntity` o `Group`, mientras que su subclase `ElementaryRangeReference` representa los accesos a los campos `Elementary`.

Tenga en cuenta que la mayoría de las observaciones anteriores se aplican a `Primitive` las subclases, ya que se esfuerzan por implementar un comportamiento similar al que proporciona el `RecordEntity` tiempo de ejecución de AWS Blu Age (en lugar del código generado). Con este propósito, todas las subclases de `Primitive` implementan las interfaces `RecordAdaptable` `ElementaryRangeReference` y `Bindable` de modo que se puedan utilizar en lugar de las subclases `RecordEntity` y los campos elementales.

Nombres completos calificados de los tipos de Java descritos

La siguiente tabla muestra los nombres completos calificados de los tipos de Java analizados en esta sección.

Nombre corto	Nombre completo calificado
`Alphanumeric`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.Alphanumeric`
`AlphanumericType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.AlphanumericType`
`BinaryType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.BinaryType`
`Bindable`	`com.netfective.bluage.gapwalk.datasimplifier.data.Bindable`
`Elementary`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Elementary`

Nombre corto	Nombre completo calificado
ElementaryRangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.ElementaryRangeReference
Filler	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Filler
Group	com.netfective.bluage.gapwalk.datasimplifier.data.structure.Group
PackedType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.PackedType
Primitive	com.netfective.bluage.gapwalk.datasimplifier.elementary.Primitive
RangeReference	com.netfective.bluage.gapwalk.datasimplifier.entity.RangeReference
RangeType	com.netfective.bluage.gapwalk.datasimplifier.metadata.type.RangeType
Record	com.netfective.bluage.gapwalk.datasimplifier.data.Record
RecordAdaptable	com.netfective.bluage.gapwalk.datasimplifier.data.RecordAdaptable

Nombre corto	Nombre completo calificado
`RecordEntity`	`com.netfective.bluage.gapwalk.datasimplifier.entity.RecordEntity`
`Repetition`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Repetition`
`Union`	`com.netfective.bluage.gapwalk.datasimplifier.data.structure.Union`
`ZonedDecimal`	`com.netfective.bluage.gapwalk.datasimplifier.elementary.ZonedDecimal`
`ZonedType`	`com.netfective.bluage.gapwalk.datasimplifier.metadata.type.ZonedType`

AWS Blu Age Blusam

En los sistemas de mainframe (denominados en el siguiente tema “heredados”), los datos empresariales suelen almacenarse mediante VSAM (método de acceso a almacenamiento virtual). Los datos se almacenan en registros (matrices de bytes), que pertenecen a un conjunto de datos.

Hay cuatro organizaciones de conjuntos de datos:

- KSDS: conjuntos de datos secuenciados por claves: los registros se indexan mediante una clave principal (no se permiten claves duplicadas) y, opcionalmente, claves alternativas adicionales. Todos los valores clave son subconjuntos de la matriz de bytes del registro y cada clave se define por:
  - un desplazamiento (basado en 0, siendo 0 el inicio del contenido de la matriz de bytes del registro, medido en bytes)
  - una longitud (expresada en bytes)

  - si tolera valores duplicados o no
- ESDS: conjuntos de datos secuenciados por entradas: se accede a los registros principalmente de forma secuencial (según su orden de inserción en el conjunto de datos), pero se puede acceder a ellos mediante claves alternativas adicionales;
- RRDS: conjuntos de datos de registros relativos: se accede a los registros mediante saltos, utilizando números de registro relativos. Los saltos se pueden realizar hacia adelante o hacia atrás.
- LDS: conjuntos de datos lineales: no hay registros allí, simplemente un flujo de bytes, organizados en páginas. Se utiliza principalmente con fines internos en plataformas antiguas.

Al modernizar las aplicaciones heredadas con el enfoque de refactorización de AWS Blu Age, las aplicaciones modernizadas ya no están diseñadas para acceder a los datos almacenados en el VSAM, sino que, al mismo tiempo, conservan la lógica de acceso a los datos. El componente Blusam es la respuesta: permite importar datos de las exportaciones de conjuntos de datos VSAM heredadas, proporciona una API para que la aplicación modernizada pueda manipularlos, además de una aplicación web dedicada a la administración. Consulte the section called “Consola de administración de Blusam”.

> **Note**
>
> Blusam solo admite KSDS, ESDS y RRDS.

La API de Blusam permite conservar la lógica de acceso a los datos (lecturas secuenciales, aleatorias y relativas; insertar, actualizar y eliminar registros), mientras que la arquitectura de componentes, que se basa en una combinación de estrategias de almacenamiento en caché y almacenamiento basado en RDBMS, permite operaciones de E/S de alto rendimiento con recursos limitados.

Infraestructura de Blusam

Blusam se basa en el RDBMS de PostgreSQL para el almacenamiento de conjuntos de datos, tanto para los datos de registros sin procesar como para los índices de claves (cuando proceda). La opción preferida es utilizar el motor compatible con PostgreSQL de Amazon Aurora. Los ejemplos e ilustraciones de este tema se basan en este motor.

> **Note**
>
> Al iniciar el servidor, el tiempo de ejecución de Blusam comprueba la presencia de algunas tablas técnicas obligatorias y las crea si no se encuentran. En consecuencia, el rol utilizado en la configuración para acceder a la base de datos de Blusam debe tener concedidos los derechos para crear, actualizar y eliminar las tablas de base de datos (tanto las filas como las propias definiciones de las tablas). Para obtener más información sobre cómo deshabilitar Blusam, consulte the section called “Configuración de Blusam”.

Almacenamiento en caché

Además del almacenamiento en sí, Blusam funciona más rápido cuando se combina con una implementación de caché.

Actualmente se admiten dos motores de caché EhCache y Redis, cada uno con su propio caso de uso:

- EhCache : caché local volátil integrada e independiente
  - NO es apta para la implementación de un entorno gestionado de modernización de AWS mainframe.
  - Suele utilizarse cuando se utiliza un nodo único, como un único servidor de Apache Tomcat, para ejecutar las aplicaciones modernizadas. Por ejemplo, el nodo puede estar dedicado a alojar tareas de trabajos por lotes.
  - Volátil: la instancia de EhCache caché es volátil; su contenido se perderá al cerrar el servidor.
  - Embebido: el servidor EhCache y el servidor comparten el mismo espacio de memoria de la JVM (esto debe tenerse en cuenta a la hora de definir las especificaciones de la máquina de alojamiento).
- Redis: caché persistente compartida
  - Apto para la implementación de un entorno gestionado por la modernización del AWS mainframe.
  - Normalmente se usa en situaciones con varios nodos, en concreto, cuando varios servidores están detrás de un equilibrador de carga. El contenido de la caché se comparte entre todos los nodos.

- Redis es persistente y no se ha enlazado a los ciclos de vida de los nodos. Se ejecuta en su propia máquina o servicio dedicado (por ejemplo, Amazon ElastiCache). La memoria caché es remota para todos los nodos.

Bloqueo

Para gestionar el acceso simultáneo a los registros y conjuntos de datos, Blusam se basa en un sistema de bloqueo configurable. El bloqueo se puede aplicar a ambos niveles: conjuntos de datos y registros:

- El bloqueo de un conjunto de datos con fines de escritura impedirá que todos los demás clientes realicen operaciones de escritura en él, en cualquier nivel (registro o conjunto de datos).
- Si se bloquea en el nivel de registro para escritura, se impedirá que otros clientes realicen operaciones de escritura únicamente en el registro determinado.

La configuración del sistema de bloqueo de Blusam debe realizarse de acuerdo con la configuración de la memoria caché:

- Si EhCache se elige como implementación de caché, no se requiere ninguna configuración de bloqueo adicional, ya que se debe usar el sistema de bloqueo en memoria predeterminado.
- Si se elige Redis como implementación de caché, se requiere una configuración de bloqueo basada en Redis para permitir el acceso simultáneo desde varios nodos. La caché de Redis utilizada para bloqueos no tiene por qué ser la misma que la que se utiliza para los conjuntos de datos. Para obtener información sobre la configuración de un sistema de bloqueo basado en Redis, consulte the section called “Configuración de Blusam”.

Tipos intrínsecos de Blusam y migración de datos desde versiones heredadas

Almacenamiento de conjuntos de datos: registros e índices

Cada conjunto de datos heredado, cuando se importe en Blusam, se almacenará en una tabla dedicada; cada fila de la tabla representa un registro que consta de dos columnas:

- La columna de ID numérica, de tipo entero grande, es la clave principal de la tabla y se utiliza para almacenar la dirección de bytes relativa (RBA) del registro. La RBA representa el desplazamiento en bytes desde el inicio del conjunto de datos y comienza en 0.

- La columna de registro de matriz de bytes, que se utiliza para almacenar el contenido del registro sin procesar.

Consulte, por ejemplo, el contenido de un conjunto de datos del KSDS utilizado en la CardDemo aplicación:

```
1  SELECT * FROM public.aws_m2_carddemo_acctdata_vsam_ksds
2  ORDER BY id ASC
```

Data output | Messages | Notifications

	id [PK] bigint	record bytea
1	0	[binary data]
2	300	[binary data]
3	600	[binary data]
4	900	[binary data]
5	1200	[binary data]
6	1500	[binary data]
7	1800	[binary data]
8	2100	[binary data]
9	2400	[binary data]
10	2700	[binary data]
11	3000	[binary data]
12	3300	[binary data]
13	3600	[binary data]

- Este conjunto de datos concreto tiene registros de una longitud fija de 300 bytes (de ahí que el conjunto de identificadores sea múltiplo de 300).
- De forma predeterminada, la herramienta pgAdmin utilizada para consultar las bases de datos de PostgreSQL no muestra el contenido de las columnas del conjunto de bytes, sino que imprime una etiqueta [datos binarios].
- El contenido del registro sin procesar coincide con el conjunto de datos sin procesar exportado del sistema anterior, sin ninguna conversión. En concreto, no se realiza ninguna conversión del conjunto de caracteres, lo cual implica que las partes alfanuméricas del registro deberán

descodificarse mediante aplicaciones modernizadas que utilicen el conjunto de caracteres heredado, muy probablemente una variante de EBCDIC.

En cuanto a los índices de claves y metadatos del conjunto de datos: cada conjunto de datos está asociado a dos filas de la tabla llamada `metadata`. Esta es la convención de nomenclatura predeterminada. Para obtener más información sobre cómo personalizarla, consulte the section called “Configuración de Blusam”.

	name [PK] text	metadata oid
1	AWS_M2_CARDDEMO_ACCTDATA_VSAM_KSDS	66320
2	AWS_M2_CARDDEMO_ACCTDATA_VSAM_KSDS___internal	66321

- La primera fila tiene el nombre del conjunto de datos como valor de la columna nombre. La columna metadatos es una columna binaria que contiene una serialización binaria de los metadatos generales del conjunto de datos determinado. Para obtener más información, consulte the section called “Atributos generales de metadatos de conjuntos de datos”.
- La segunda fila tiene el nombre del conjunto de datos con el sufijo `___internal'` como valor de la columna nombre. El contenido binario de la columna metadatos depende del peso del conjunto de datos.
  - Para conjuntos de datos pequeños y medianos, el contenido es una serialización comprimida de:
    - definición de las claves utilizadas por el conjunto de datos; la definición de clave principal (para KSDS) y definiciones de claves alternativas, si corresponde (para KSDS o ESDS)
    - los índices clave, si procede (KSDS o ESDS con definiciones de clave alternativas): se utilizan para la navegación indexada de registros; el índice clave asigna un valor clave a la RBA de un registro;
    - mapa de longitud de registros: se utiliza para la navegación secuencial o relativa de los registros;
  - En el caso de conjuntos de datos grandes o muy grandes, el contenido es una serialización comprimida de:
    - definición de las claves utilizadas por el conjunto de datos; la definición de clave principal (para KSDS) y definiciones de claves alternativas, si corresponde (para KSDS o ESDS)

Además, los índices de conjuntos de datos grandes o muy grandes (si corresponde) se almacenan mediante un mecanismo de paginación; las serializaciones binarias de las páginas de índice se

almacenan como filas de una tabla dedicada (una tabla por clave de conjunto de datos). Cada página de índices se almacena en una fila que tiene las siguientes columnas:

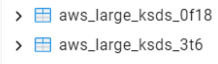
- id: identificador técnico de la página de índices (clave principal numérica);
- firstkey: valor binario del primer valor clave (el más bajo) almacenado en la página de índices;
- lastkey: valor binario del último valor clave (el más alto) almacenado en la página de índices;
- metadatos: serialización comprimida binaria de la página de índices (mapeo de valores clave a registros). RBAs

	id [PK] bigint	firstkey bytea	lastkey bytea	metadata oid
1	1	[binary data]	[binary da...	6458928
2	2	[binary data]	[binary da...	6458929
3	3	[binary data]	[binary da...	6458930
4	4	[binary data]	[binary da...	6458931
5	5	[binary data]	[binary da...	6458932
6	6	[binary data]	[binary da...	6458933
7	7	[binary data]	[binary da...	6458934
8	8	[binary data]	[binary da...	6458935
9	9	[binary data]	[binary da...	6458936

El nombre de la tabla es una concatenación del nombre del conjunto de datos y el nombre interno de la clave, que contiene información sobre la clave, como el desplazamiento de la clave, si la clave acepta duplicados (si se establece en verdadero para permitir duplicados) y la longitud de la clave. Por ejemplo, considere un conjunto de datos denominado «AWS_LARGE_KSDS» que tiene las dos claves definidas siguientes:

- clave principal [desplazamiento: 0, duplicados: false, longitud: 18]
- clave alternativa [desplazamiento: 3, duplicados: verdadero, longitud: 6]

En este caso, las tablas siguientes almacenan los índices relacionados con las dos claves.



Optimización del rendimiento de E/S mediante un mecanismo de escritura diferida

Para optimizar el rendimiento de las operaciones de inserción, actualización y eliminación, el motor Blusam se basa en un mecanismo configurable de escritura trasera. El mecanismo se basa en un grupo de subprocesos dedicados que se ocupan de las operaciones de persistencia mediante consultas de actualización masiva, con el fin de maximizar el rendimiento de E/S hacia el almacenamiento de Blusam.

El motor de Blusam recopila todas las operaciones de actualización realizadas en los registros por las aplicaciones y crea lotes de registros que se envían para su tratamiento en los subprocesos dedicados. A continuación, los lotes se conservan en el almacenamiento de Blusam, mediante consultas de actualización masivas, lo cual evita el uso de operaciones de persistencia atómica y garantiza el mejor uso posible del ancho de banda de la red.

El mecanismo utiliza un retraso configurable (el valor predeterminado es de un segundo) y un tamaño de lote configurable (el valor predeterminado es de 10 000 registros). Las consultas de persistencia de compilación se ejecutan en cuanto se cumple la primera de las dos condiciones siguientes:

- El retraso configurado ha transcurrido y el lote no está vacío.
- El número de registros del lote que se tratará alcanza el límite configurado.

Para obtener información sobre cómo configurar el mecanismo de escritura diferida, consulte the section called “Propiedades opcionales”.

Elección del esquema de almacenamiento adecuado

Como se muestra en la sección anterior, la manera en que se almacenan los conjuntos de datos depende de su peso. Pero, ¿qué se considera pequeño, mediano o grande para un conjunto de datos? ¿Cuándo elegir la estrategia de almacenamiento paginado en lugar de la normal?

La respuesta a esa pregunta depende de lo siguiente.

- La cantidad de memoria disponible en cada uno de los servidores que alojan las aplicaciones modernizadas que utilizarán esos conjuntos de datos.
- La cantidad de memoria disponible en la infraestructura de caché (si existe).

Cuando se utiliza un esquema de almacenamiento de índices no paginados, las colecciones completas de tamaños de registros e índices de claves se cargarán en la memoria del servidor

en el momento de abrir los conjuntos de datos, para cada conjunto de datos. Además, si se utiliza almacenamiento en caché, es posible que todos los registros del conjunto de datos se precarguen en la memoria caché con el método habitual, lo que podría provocar el agotamiento de los recursos de memoria en la infraestructura de la caché.

Según el número de claves definidas, la longitud de los valores clave, el número de registros y el número de conjuntos de datos abiertos al mismo tiempo, la cantidad de memoria consumida puede evaluarse aproximadamente para los casos de uso conocidos determinados.

Para obtener más información, consulte the section called “Estimación del espacio de memoria para un conjunto de datos determinado”.

Migración de Blusam

Una vez que se ha seleccionado el esquema de almacenamiento adecuado para un conjunto de datos determinado, el almacenamiento de Blusam debe rellenarse mediante la migración de los conjuntos de datos heredados.

Para ello, hay que utilizar exportaciones binarias sin procesar de los conjuntos de datos heredados, sin que se utilice ninguna conversión de conjuntos de caracteres durante el proceso de exportación. Al transferir las exportaciones de conjuntos de datos desde el sistema heredado, asegúrese de no dañar el formato binario. Por ejemplo, aplique el modo binario al usar FTP.

Las exportaciones binarias sin procesar solo contienen los registros. No es necesario que el keys/indexes exports as all keys/indexes mecanismo de importación lo vuelva a calcular sobre la marcha.

Una vez disponible la exportación binaria de un conjunto de datos, existen varias opciones para migrarlo a Blusam:

En el entorno gestionado por la modernización AWS del mainframe:

- Importe conjuntos de datos mediante la característica específica. Consulte the section called “Importación de conjuntos de datos para aplicaciones de ”.

o

- Utilice la función de importación masiva de conjuntos de datos. Consulte the section called “Referencia de definición de conjuntos de datos” y the section called “Ejemplo de formato de solicitud de conjunto de datos para VSAM”.

o

- Utilice un script de Groovy para importar conjuntos de datos, utilizando servicios de carga dedicados.

> **Note**
>
> Por ahora, solo es posible importar LargeKSDS y LargeESDS en entornos administrados por Mainframe Modernization mediante scripts de groovy.

En AWS Blu Age Runtime en Amazon EC2:

- Importe el conjunto de datos mediantethe section called “Consola de administración de Blusam”.

o

- Utilice un script de Groovy para importar conjuntos de datos, utilizando servicios de carga dedicados.

Importación de conjuntos de datos con scripts de Groovy

Esta sección le ayudará a escribir scripts de Groovy para importar conjuntos de datos heredados en Blusam.

Empieza con algunas importaciones obligatorias:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList; //used for alternate keys if any
```

Después de eso, para cada conjunto de datos que se importará, el código se basa en el patrón determinado:

1. crear o borrar un objeto de mapa;
2. rellenar el mapa con las propiedades requeridas (esto varía según los tipos de conjuntos de datos; consulte a continuación para ver una información más detallada);

3. recuperar el servicio de carga adecuado para usarlo como conjunto de datos en el registro de servicios;
4. ejecutar el servicio usando el mapa como argumento.

Hay 5 implementaciones de servicios que se pueden recuperar del registro de servicios mediante los siguientes identificadores:

- `"BluesamKSDSFileLoader"`: para KSDS de tamaño pequeño/mediano
- `"BluesamESDSFileLoader"`: para ESDS de tamaño pequeño/mediano
- `"BluesamRRDSFileLoader"`: para RRDS
- `"BluesamLargeKSDSFileLoader"`: para KSDS grandes
- `"BluesamLargeESDSFileLoader"`: para ESDS grandes

Elegir la versión normal o la versión grande del servicio para KSDS/ESDS depende del tamaño de los conjuntos de datos y de la estrategia de almacenamiento que desee aplicarle. Para obtener información sobre cómo elegir la estrategia de almacenamiento adecuada, consulte the section called “Elección del esquema de almacenamiento adecuado”.

Para poder importar correctamente el conjunto de datos en Blusam, se deben proporcionar las propiedades adecuadas al servicio de carga.

Propiedades comunes:

- Obligatorio (para todos los tipos de conjuntos de datos)
  - bluesamManager: el valor esperado es `applicationContext.getBean(BluesamManager.class)`
  - datasetName: nombre del conjunto de datos, como cadena
  - "inFilePath": ruta a la exportación del conjunto de datos heredado, en forma de cadena
  - recordLength: longitud de registro fija o 0 para un conjunto de datos de longitud de registro variable, como número entero
- Opcional
  - No se admite para conjuntos de datos de gran tamaño:
    - isAppend: un indicador booleano que indica que la importación se está realizando en modo de anexión (anexando registros a un conjunto de datos de Blusam existente).

    - useCompression: un indicador booleano que indica que la compresión se utilizará para almacenar metadatos.
  - Solo para conjuntos de datos grandes:
    - "indexingPageSizeInMb": el tamaño en megabytes de cada página de índice, para cada una de las claves del conjunto de datos, expresado como un entero estrictamente positivo

Propiedades dependientes del tipo de conjunto de datos:

- KSDS/KSDS grande:
  - mandatory
    - primaryKey: la definición de clave principal, mediante una llamada al constructor `com.netfective.bluage.gapwalk.bluesam.metadata.Key`.
  - opcional:
    - alternateKeys: una lista (`java.util.List`) de definiciones de claves alternativas, creada mediante llamadas al constructor `com.netfective.bluage.gapwalk.bluesam.metadata.Key`.
- ESDS/ESDS grande:
  - opcional:
    - alternateKeys: una lista (`java.util.List`) de definiciones de claves alternativas, creada mediante llamadas al constructor `com.netfective.bluage.gapwalk.bluesam.metadata.Key`.
- RRDS:
  - ninguno.

Llamadas al constructor de claves:

- `new Key(int offset, int length)`: crea un objeto de clave, con los atributos de claves determinados (desplazamiento y longitud) y no se permiten duplicados. Esta variante debe usarse para definir una clave principal.
- `new Key(boolean allowDuplicates, int offset, int length)`: crea un objeto de clave, con los atributos de claves determinados (desplazamiento y longitud) e indicador que permiten duplicados.

Los siguientes ejemplos de Groovy muestran diversos escenarios de carga.

Carga de un KSDS grande, con dos claves alternativas:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;

// Loading a large KSDS into Blusam
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "largeKsdsSample");
map.put("inFilePath", "/work/samples/largeKsdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
ArrayList altKeys = [new Key(true, 10, 8), new Key(false, 0, 9)]
map.put("alternateKeys", altKeys);
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamLargeKSDSFileLoader");
service.runService(map);
```

Carga de un ESDS de longitud de registro variable, sin claves alternativas:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry

// Loading an ESDS into Blusam
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "esdsSample");
map.put("inFilePath", "/work/samples/esdsSampleExport");
map.put("recordLength", 0);
def service = ServiceRegistry.getService("BluesamESDSFileLoader");
service.runService(map);
```

Las exportaciones de conjuntos de datos de longitud de registro variable contendrán la información obligatoria sobre la palabra de descripción de registro (RDW) para permitir la división de los registros en el momento de la lectura.

Carga de un RRDS de longitud de registro fija:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry

// Loading a RRDS into Blusam
def map = [:]
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", "rrdsSample");
map.put("inFilePath", "/work/samples/rrdsSampleExport");
map.put("recordLength", 180);
def service = ServiceRegistry.getService("BluesamRRDSFileLoader");
service.runService(map);
```

Carga de conjuntos de datos en modo multiesquema:

Modo multiesquema: en algunos sistemas heredados, los archivos VSAM se organizan en conjuntos de archivos, lo que permite a los programas acceder a los datos y modificarlos dentro de particiones específicas. Los sistemas modernos tratan cada conjunto de archivos como un esquema, lo que permite una partición de datos y un control de acceso similares.

Para activar el modo multiesquema en el `application-main.yml` archivo, consulte. the section called “Configuración de Blusam” En este modo, los conjuntos de datos se pueden cargar en un esquema específico mediante un contexto compartido, que es un registro en memoria para la información de tiempo de ejecución. Para cargar un conjunto de datos en un esquema específico, coloque el nombre del esquema correspondiente como prefijo al nombre del conjunto de datos.

Cargar un archivo KSDS en un esquema específico para el modo multiesquema:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;
import com.netfective.bluage.gapwalk.rt.shared.SharedContext;

// Loading a KSDS into Blusam
def map = [:]
String schema = "schema1";
String datasetName = schema+"|"+"ksdsSample";
SharedContext.get().setCurrentBlusamSchema(schema);
schema = SharedContext.get().getCurrentBlusamSchema();
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
```

```
map.put("datasetName", datasetName);
map.put("inFilePath", "/work/samples/ksdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamKSDSFileLoader");
service.runService(map);
```

Cargar un archivo KSDS grande en un esquema específico para el modo multiesquema:

```
import com.netfective.bluage.gapwalk.bluesam.BluesamManager
import com.netfective.bluage.gapwalk.bluesam.metadata.Key;
import com.netfective.bluage.gapwalk.rt.provider.ServiceRegistry
import java.util.ArrayList;
import com.netfective.bluage.gapwalk.rt.shared.SharedContext;

// Loading a Large KSDS into Blusam
def map = [:]
String schema = "schema1";
String datasetName = schema+"|"+"largeKsdsSample";
SharedContext.get().setCurrentBlusamSchema(schema);
schema = SharedContext.get().getCurrentBlusamSchema();
map.put("bluesamManager", applicationContext.getBean(BluesamManager.class));
map.put("datasetName", datasetName);
map.put("inFilePath", "/work/samples/LargeKsdsSampleExport");
map.put("recordLength", 49);
map.put("primaryKey", new Key(0, 18));
map.put("indexingPageSizeInMb", 25);
def service = ServiceRegistry.getService("BluesamLargeKSDSFileLoader");
service.runService(map);
```

Además, se puede utilizar una entrada de configuración (que se establecerá en el archivo de `application-main.yml` configuración) para ajustar el proceso de importación:

- `bluesam.fileLoading.commitInterval`: un entero estrictamente positivo que define el intervalo de confirmación para un mecanismo de ESDS/KSDS/RRDS importación normal. No se aplica a las importaciones de conjuntos de datos de gran tamaño. El valor predeterminado es 100 000.

Configuración de Blusam

La configuración de Blusam se realiza en el archivo de `application-main.yml` configuración (o en el archivo de `application-bac.yml` configuración para el despliegue independiente de la aplicación Blusam Administration Console (BAC)).

Blusam debe configurarse en dos aspectos:

- Configuración de acceso a las cachés y al almacenamiento de Blusam
- Configuración del motor de Blusam

Configuración de acceso a las cachés y al almacenamiento de Blusam

Para obtener información sobre cómo configurar el acceso a las cachés y al almacenamiento de Blusam mediante administradores de secretos u orígenes de datos, consulte the section called “AWS Configuración de Blu Age Runtime”.

> **Note**
>
> En cuanto al acceso al almacenamiento de Blusam, las credenciales utilizadas apuntarán a un rol de conexión, con los privilegios correspondientes. Para que el motor de Blusam pueda funcionar de la manera prevista, el rol de conexión debe tener los siguientes privilegios:

- conectarse a la base de datos
- crear/eliminar/alterar/truncar tablas y vistas
- seleccionar/insertar/ eliminar/actualizar filas en tablas y vistas
- ejecutar funciones o procedimientos

Configuración del motor de Blusam

Deshabilitación de la compatibilidad con Blusam

En primer lugar, mencionemos que es posible deshabilitar por completo la compatibilidad con Blusam al establecer la propiedad `bluesam.disabled` en `true`. Al iniciar la aplicación, aparecerá un mensaje informativo en los registros del servidor para recordar que se ha deshabilitado Blusam:

```
BLUESAM is disabled. No operations allowed.
```

En ese caso, no es necesario realizar ninguna otra configuración sobre Blusam y cualquier intento de utilizar las funciones relacionadas con Blusam (ya sea mediante programación o mediante llamadas REST) provocará una interrupción `UnsupportedOperationException` en la ejecución del código Java y aparecerá un mensaje explicativo sobre la desactivación de Blusam.

Propiedades del motor de Blusam

Las propiedades de configuración del motor de Blusam se reagrupan bajo el prefijo de clave de Blusam:

Propiedades obligatorias

- `cache`: se valorará con la implementación de caché elegida. Los valores válidos son:
  - `ehcache`: para el uso de ehcache incrustado local. Consulte las restricciones de casos de uso relacionadas más arriba.
  - `redis`: para el uso de la caché de Redis remota compartida. Esta es la opción preferida para el caso de uso gestionado de la modernización de mainframes. AWS
  - `none`: para deshabilitar el almacenamiento en caché.
- `persistence`: se valorará con pgsql (motor PostgreSQL: versión mínima 10.0; versión recomendada >=14.0
- referencia de origen de datos: `<persistence engine>.dataSource` apuntará a la definición de dataSource para la conexión con el almacenamiento de Blusam, definida en otra parte del archivo de configuración. Por lo general, se le llama `bluesamDs`.

> **Note**
>
> Siempre que se utilice Redis como mecanismo de caché, ya sea para datos o bloqueos (consulte a continuación), se debe configurar el acceso a las instancias de Redis. Para obtener más información, consulte the section called “Propiedades de caché de Redis disponibles”.

Propiedades opcionales

Bloqueos de Blusam: las propiedades llevan el prefijo `locks`.

- `cache`: el único valor utilizable es `redis`, para especificar que se utilizará el mecanismo de bloqueo basado en Redis (que también se utilizará cuando la caché de almacenamiento de Blusam esté basada en Redis). Si falta la propiedad o no está establecida en `redis`, se utilizará en su lugar el mecanismo de bloqueos en memoria predeterminado.
- `lockTimeOut`: un valor entero largo positivo, que indica el tiempo de espera expresado en milisegundos antes de que un intento de bloquear un elemento ya bloqueado se marque como con error. El valor predeterminado es `500`.
- `locksDeadTime`: un valor entero largo positivo, que representa el tiempo máximo, expresado en milisegundos, en que una aplicación puede mantener un bloqueo. Los bloqueos se marcan automáticamente como vencidos y se liberan una vez transcurrido ese tiempo. El valor predeterminado es `1000`.
- `locksCheck`: una cadena, utilizada para definir la estrategia de control de bloqueos utilizada por el administrador de bloqueos de Blusam actual, sobre la eliminación de bloqueos vencidos. Debe seleccionarse entre los siguientes valores:
  - `off`: no se realiza ninguna comprobación. Se desaconseja, ya que podrían producirse bloqueos.
  - `reboot`: las comprobaciones se realizan en el reinicio o al iniciar la aplicación. Todos los bloqueos vencidos se liberan en ese momento. Esta es la opción predeterminada.
  - `timeout`: las comprobaciones se realizan en el reinicio o al iniciar la aplicación, o cuando vence un tiempo de espera durante un intento de bloquear un conjunto de datos. Los bloqueos caducados se liberan inmediatamente.

Mecanismo de escritura diferida: las propiedades llevan el prefijo de la clave `write-behind`:

- `enabled`: `true` (valor predeterminado y recomendado) o `false`, para habilitar o deshabilitar el mecanismo de escritura diferida. La deshabilitación del mecanismo afectará en gran medida al rendimiento de la escritura, por lo que no se recomienda.
- `maxDelay`: una duración máxima para que se desencadenen los subprocesos. El valor predeterminado es `"1s"` (un segundo). El mantenimiento del valor predeterminado suele ser buena idea, a menos que condiciones específicas requieran ajustar este valor. En cualquier caso, el valor debe mantenerse bajo (menos de 3 segundos). El formato de la cadena de retraso es: `<time unit>` donde `<integer value><optional whitespace><time unit>` debe seleccionarse entre los siguientes valores:
  - `"ns"`: nanosegundos
  - `"µs"`: microsegundos

  - `"ms"`: milisegundos
  - `"s"`: segundos
- `threads`: el número de subprocesos de escritura diferida dedicada. El valor predeterminado es `5`. Debe ajustar este valor de acuerdo con la potencia de computación del host que ejecuta el motor de Blusam. No es relevante utilizar un valor mucho más alto, con la esperanza de aumentar el rendimiento, ya que el factor limitante será la capacidad de RDBMS de almacenamiento para gestionar numerosas consultas por lotes simultáneas. Los valores recomendados suelen oscilar entre 4 y 8.
- `batchSize`: un entero positivo que representa el número máximo de registros de un lote que se enviarán a un subproceso para su tratamiento masivo. Su valor debe estar comprendido entre 1 y 32767. El valor predeterminado es `10000`. Usar 1 como valor va en contra de la finalidad del mecanismo, que consiste en evitar el uso de consultas de actualización atómica; el valor mínimo adecuado que se utilizará es `1000` aproximadamente.

Ajuste de precisión integrado: las propiedades EhCache llevan el prefijo de la clave: `ehcache`

- `resource-pool`:
  - `size`: tamaño de memoria asignado a la memoria caché incrustada, expresada como cadena. El valor predeterminado es `"1024MB"` (1 gigabyte). Debe ajustarse en función de la memoria disponible de la máquina que aloja el motor de Blusam y del tamaño de los conjuntos de datos que utiliza la aplicación. El formato de la cadena de tamaño es: `<integer value><optional whitespace><memory unit>` donde `<memory-unit>` debe seleccionarse entre los siguientes valores:
    - `B`: bytes
    - `KB`: kilobytes
    - `MB`: megabytes
    - `GB`: gigabytes
    - `TB`: terabytes
  - `heap`: `true` o `false`, para indicar si la memoria caché consumirá memoria dinámica de JVM o no. El valor predeterminado es `true` (la opción más rápida para el rendimiento de la caché, pero el almacenamiento en caché consume memoria de la memoria RAM dinámica de JVM). Si esta propiedad se establece en `false`, se cambiará a la memoria no dinámica, que será más lenta debido a los intercambios requeridos con la pila de JVM.

- `timeToLiveMillis`: el tiempo (en milisegundos) durante el que una entrada de la caché permanece en la memoria caché antes de que se considere caducada y eliminada. Si no se especifica esta propiedad, las entradas de la caché no caducarán automáticamente de forma predeterminada.

Propiedades de configuración de varios esquemas

- `multiSchema`: false (valor predeterminado) o true, para deshabilitar o habilitar el modo multiesquema para Blusam; disponible a partir de la versión 4.4.0.
- `pgsql`:
  - `schemas`: una lista de nombres de esquemas que la aplicación utilizará en el modo multiesquema para Blusam.
  - `fallbackSchema`: El nombre del esquema alternativo para su uso en el modo multiesquema. Si no se encuentra un conjunto de datos en el contexto del esquema actual, este esquema se utilizará para las operaciones relacionadas con Blusam en ese conjunto de datos.

Fragmento de código de configuración de muestra:

```
dataSource:
  bluesamDs:
    driver-class-name: org.postgresql.Driver
    ...
    ...
bluesam:
  locks:
    lockTimeOut: 700
  cache: ehcache
  persistence: pgsql
  ehcache:
    resource-pool:
      size: 8GB
  write-behind:
    enabled: true
    threads: 8
    batchsize: 5000
  pgsql:
    dataSource : bluesamDs
```

Ejemplo de fragmento de configuración (con el modo multiesquema activado para Blusam):

```
dataSource:
  bluesamDs:
    driver-class-name: org.postgresql.Driver
    ...
    ...
bluesam:
  locks:
    lockTimeOut: 700
  cache: ehcache
  persistence: pgsql
  ehcache:
    resource-pool:
      size: 8GB
  write-behind:
    enabled: true
    threads: 8
    batchsize: 5000
  multiSchema: true
  pgsql:
    dataSource : bluesamDs
    schemas:
      - "schema1"
      - "schema2"
      - "schema3"
    fallbackSchema: schema3
```

> **Note**
>
> Los esquemas de metadatos de Blusam, incluidos los esquemas enumerados en el `application-main.yml` archivo para el modo multiesquema, se crean en la base de datos de blusam si no existen y el usuario tiene los privilegios suficientes.

Consola de administración de Blusam

La consola de administración de Blusam (BAC) es una aplicación web, que se utiliza para administrar el almacenamiento de Blusam. Para obtener más información sobre la BAC, consulte the section called “Consola de administración de Blusam”.

Apéndice

Atributos generales de metadatos de conjuntos de datos

Lista de atributos de serialización de metadatos de conjuntos de datos generales:

- nombre (del conjunto de datos)
- tipo (KSDS, LargeKSDS, ESDS, LargeESDS o RRDS)
- indicador de preparación de la memoria caché (si el conjunto de datos debe estar precargado en la memoria caché al iniciar el servidor o no)
- indicador de uso de la compresión (si se almacenarán registros en formato comprimido o sin procesar)
- fecha de creación
- fecha de la última modificación
- indicador de registro de longitud fija (independientemente de si los registros del conjunto de datos tienen todos la misma longitud o no)
- longitud de registro: solo es significativa para longitud de registro fija
- tamaño de página (se utiliza para personalizar las consultas SQL paginadas que se utilizan para precargar la caché cuando es necesario)
- tamaño (tamaño del conjunto de datos, longitud acumulada de los registros)
- último desplazamiento (desplazamiento, es decir, RBA del último registro agregado al conjunto de datos)
- siguiente desplazamiento (siguiente desplazamiento disponible para agregar un nuevo registro al conjunto de datos)
- si es significativa, definición de las claves utilizadas por el conjunto de datos; cada clave se define por su tipo (principal o parte de la colección de claves alternativas) y tres atributos:
  - desplazamiento: posición en el registro del byte inicial del valor de clave;
  - longitud: longitud en bytes del valor de clave. Por lo tanto, el valor de clave es la matriz de bytes, que es el subconjunto del registro que comienza en `key offset` y termina en la posición `key offset + length - 1`;
  - indicador de duplicados permitidos: si la clave acepta duplicados o no (establecer en verdadero para permitir duplicados).

Estimación del espacio de memoria para un conjunto de datos determinado

En el caso de conjuntos de datos pequeños o medianos, los metadatos (tamaños e índices para varias claves) se cargarán por completo en la memoria. La asignación de los recursos adecuados a la máquina que aloja el servidor utilizado para ejecutar las aplicaciones modernizadas requiere determinar el consumo de memoria que provocan los conjuntos de datos de Blusam, en concreto, en lo que respecta a los metadatos. En esta sección se ofrecen respuestas prácticas a los operadores interesados.

Las fórmulas especificadas solo se aplican a los conjuntos de datos pequeños y medianos de Blusam y no utilizan la estrategia de almacenamiento Grande.

Metadatos de conjuntos de datos de Blusam

En el caso de un conjunto de datos de Blusam, los metadatos se dividen en dos partes:

- metadatos básicos: contienen información global sobre el conjunto de datos. El espacio de memoria de esto se puede considerar insignificante en comparación con los metadatos internos.
- metadatos internos: contienen información sobre el tamaño de los registros y los índices de claves; cuando un conjunto de datos no está vacío, esto es lo que consume memoria cuando se carga en el servidor de aplicaciones que aloja las aplicaciones modernizadas. En las siguientes secciones se detalla cómo crece la memoria consumida con el número de registros.

Cálculo del espacio de metadatos internos

Mapa de tamaños de registros

En primer lugar, los metadatos internos almacenan un mapa para que contenga el tamaño de cada registro (como un número entero) dada su RBA (dirección de bytes relativa, almacenada como un número largo).

El espacio de memoria de esa estructura de datos es, en bytes: `80 * number of records`.

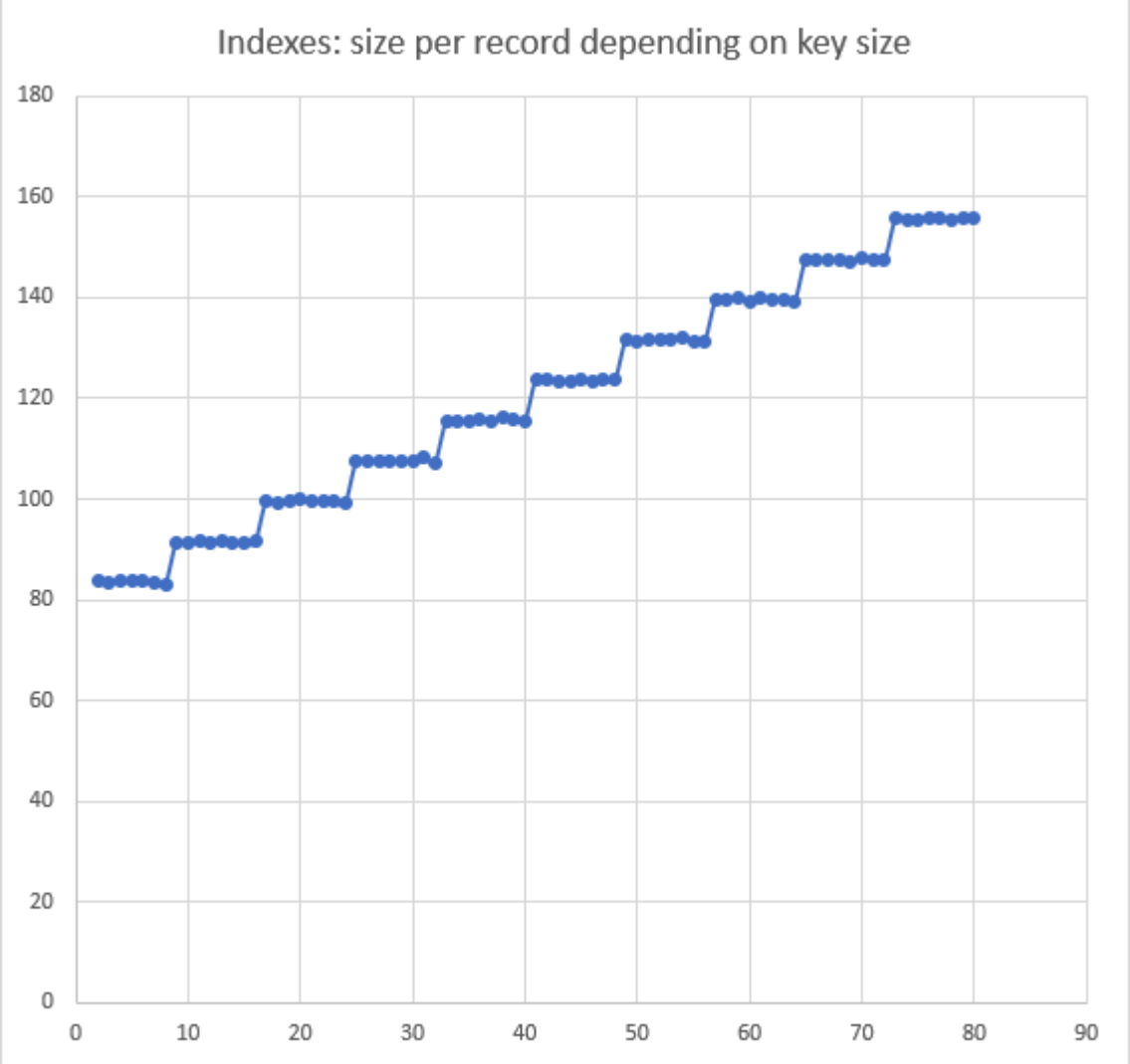
Esto se aplica a todos los tipos de conjuntos de datos.

Índices

En cuanto a los índices de la clave principal de KSDS o de las claves alternativas tanto de ESDS como de KSDS, el cálculo del tamaño depende de los dos factores siguientes:

- el número de registros en el conjunto de datos;
- el tamaño de la clave, en bytes.

El siguiente gráfico muestra el tamaño del índice de claves por registro (eje y) en función del tamaño de la clave (eje x).



La fórmula correspondiente para evaluar la huella de un índice de claves determinado de un conjunto de datos es:

```
index footprint = number of records * ( 83 + 8 (key length / 8))
```

donde / representa la división de enteros.

Ejemplos:

- conjunto de datos 1:
  - número de registros = 459 996
  - longitud de clave = 15, por lo tanto (longitud de clave/8) = 1
  - huella de índice = 459 996 * (83 + (8*1)) = 41 859 636 bytes (= 39 MB aproximadamente)
- conjunto de datos 2:
  - número de registros = 13 095 783
  - longitud de clave = 18, por lo tanto (longitud de clave/8) = 2
  - huella de índice = 13 095 783 * (83 + (8*2)) = 1 296 482 517 bytes (= 1,2 GB aproximadamente)

La huella total de un conjunto de datos determinado es la suma de todas las huellas de todos los índices clave y la huella del mapa de tamaños de registros.

Por ejemplo, tomando el ejemplo del conjunto de datos 2, que tiene una sola clave, la huella global es:

- Mapa de tamaños de registros: 13 095 783 * 80 = 1 047 662 640 bytes
- Índices clave: 1 296 482 517 bytes (consulte más arriba)
- Huella total = 2 344 145 157 bytes (= 2,18 GB aproximadamente)

Programas disponibles en la aplicación web de utilidad

La aplicación web de utilidades proporciona soporte para varios programas de utilidad de plataformas antiguas, como IDCAMS, INFUTILB, SORT, etc. Para configurar el acceso a la aplicación, consulte. Configuración del acceso a las utilidades para las aplicaciones administradas

Lista de programas

- Utilidad JCLBCICS- Se utiliza por lotes para establecer el estado del conjunto de datos de bluesam en. open/enabled or closed/disabled

Utilidad JCLBCICS

JCLBCICS es un programa de utilidad de JCL diseñado para configurar el conjunto de datos bluesam en un open/enabled or closed/disabled. An open/enabled status will block access to the dataset from batch programs while a closed/disabled estado que hace que el conjunto de datos no esté disponible para acceder a los servicios en línea del JICS.

Uso

- JCLBCICS cambia la columna STATUS de la tabla FILE_TABLE de Jics y la columna OPEN_STATUS de la tabla BLUESAM_STATUS de Bluesam en función de la increíble configuración del nombre DD.

```
.open(ddName) -> ENABLED in Jics FILE_TABLE table, OPEN in Bluesam BLUESAM_STATUS
 table
.close(ddName) -> DISABLED in Jics FILE_TABLE table, CLOSED in Bluesam BLUESAM_STATUS
 table
```

- El tamaño del nombre DD `application-utility-pgm.yml` se puede configurar globalmente en el archivo de configuración.

```
jclbcics.ddname.size: 7
```

- El tamaño del nombre DD global se puede anular en un paso individual proporcionando el tamaño anulado con las siguientes líneas en groovy y, a continuación, utilizar StepParams como parámetro para ese paso.

```
TreeMap stepMapTransfo = [:]
Map stepParams = ["MapTransfo":stepMapTransfo]
stepParams["MapTransfo"]["JCLBCICS_OVERRIDDEN_SIZE"] = '7'
...
.withParameters(stepParams)
.runProgram("JCLBCICS")
```

- Al configurar el tamaño del nombre DD, el tamaño máximo efectivo del nombre DD es 8.
- Si la longitud del DDName es mayor que el tamaño del nombre DD proporcionado, se truncará desde el final para que coincida con el tamaño del nombre DD.
- Se admiten caracteres comodín en ddName si se añade un asterisco (*) al final de ddName o si la longitud del ddName es inferior a 8.

```
.open("DTSNAME*")
```

Código de ejemplo

```
// DD name with overridden size of 7 bytes
 def stepSTEP007(Object shell, Map params, Map programResults) {
```

```
    shell.with {
      if (checkValidProgramResults(programResults)) {
        TreeMap stepMapTransfo = [:]
        Map stepParams = ["MapTransfo":stepMapTransfo]
        stepParams["MapTransfo"]["JCLBCICS_OVERRIDDEN_SIZE"] = '7'
        return execStep("STEP007", "JCLBCICS", programResults, {
          mpr
            .withDatasetsConfiguration(new DatasetsConfiguration()
            .close("DTSNAME"))
            .withParameters(stepParams)
            .runProgram("JCLBCICS")
         })
      }
    }
}
```

AWS Consola de administración Blu Age Blusam

La Consola de administración de Blusam (BAC) es una aplicación web segura para gestionar los conjuntos de datos de Blusam. En esta guía se trata la interfaz de usuario de BAC. Para obtener más información sobre la administración remota a través de los puntos de conexión REST, consulte the section called “Puntos de conexión REST de la consola de aplicaciones de Blusam ”.

Temas

- Implementación de la BAC
- Uso de la BAC
- Formato JSON de LISTCAT

Implementación de la BAC

La BAC está disponible como una aplicación web única y protegida, que utiliza el formato de archivo web (.war). Está diseñada para implementarse junto con la aplicación BluAge Gapwalk, en un servidor de aplicaciones Apache Tomcat, pero también se puede implementar como una aplicación independiente. La BAC hereda el acceso al almacenamiento de Blusam de la configuración de la aplicación Gapwalk, si está presente.

La BAC tiene su propio archivo de configuración dedicado, llamada `application-bac.yml`. Para obtener más información sobre la configuración, consulte the section called “Archivo de configuración dedicado de BAC”.

La BAC está protegida. Para obtener más información detallada sobre la configuración de seguridad, consulte the section called “Configuración de la seguridad para la BAC”.

Archivo de configuración dedicado de BAC

Implementación independiente: si la BAC se implementa solo en la aplicación Gapwalk, la conexión al almacenamiento de Blusam debe configurarse en el archivo de configuración application-bac.yml.

Los valores predeterminados para la configuración de los conjuntos de datos que se utilizan para examinar los registros de los conjuntos de datos deben establecerse en el archivo de configuración. Consulte the section called “Búsqueda de registros en un conjunto de datos”. La página de navegación de registros puede utilizar un mecanismo de máscara opcional que permite mostrar una vista estructurada del contenido de un registro. Algunas propiedades afectan a la vista de los registros cuando se utilizan máscaras.

Las siguientes propiedades configurables se deben establecer en el archivo de configuración. La aplicación BAC no asume ningún valor predeterminado para estas propiedades.

Clave	Tipo	Descripción
`bac.crud.limit`	entero	Un valor entero positivo que indica el número máximo de registros devueltos al examinar los registros. El uso de `0` indica ilimitado . Valor recomendado: `10` (a continuación, ajuste el conjunto de datos de valores por conjunto de datos en la página de navegación, según sus necesidades).
`bac.crud.encoding`	cadena	El nombre del conjunto de caracteres predeterminado, que se utiliza para decodific ar los bytes de los registros como contenido alfanumér ico. El nombre del conjunto de caracteres proporcionado

Clave	Tipo	Descripción
		debe ser compatible con Java (consulte la documentación de Java para ver los conjuntos de caracteres admitidos). Valor recomendado: el conjunto de caracteres heredado utilizado en la plataforma heredada de donde provienen los conjuntos de datos; la mayoría de las veces será una variante de EBCDIC.
`bac.crud.initCharacter`	cadena	El carácter predeterminado (byte) que se utiliza para iniciar los elementos de datos. Se pueden usar dos valores especiales: `"LOW-VALUE"` el byte 0x00 (valor recomendado) y `"HI-VALUE"` , el byte 0xFF. Se usa cuando se aplican máscaras.
`bac.crud.defaultCharacter`	cadena	El carácter predeterminado (byte), como cadena de un carácter, se utiliza para rellenar registros (a la derecha). Valor recomendado: `" "` (espacio). Se usa cuando se aplican máscaras.

Clave	Tipo	Descripción
`bac.crud.blankCharacter`	cadena	El carácter predeterminado (byte), como cadena de un carácter, se utiliza para representar los espacios en blanco en los registros. Valor recomendado: `" "` (espacio). Se usa cuando se aplican máscaras.
`bac.crud.strictZoned`	booleano	Un indicador que indica qué modo zonificado se utiliza para el registro. Si `true`, se utilizará el modo de zona estricta; si `false`, se utilizará el modo zonificado modificado. Valores recomendados: `true`. Se usa cuando se aplican máscaras.
`bac.crud.decimalSeparator`	cadena	El carácter utilizado como separador decimal en campos editados numéricos (se utiliza cuando se aplican máscaras).
`bac.crud.currencySign`	cadena	El carácter predeterminado, como cadena de un carácter, que se utiliza para representar la moneda en los campos editados numéricos, cuando se aplica el formato (se utiliza cuando se aplican máscaras).

Clave	Tipo	Descripción
`bac.crud.pictureCurrencySign`	cadena	El carácter predeterminado, como cadena de un carácter, que se utiliza para representar la moneda en los campos editados numéricos (se utiliza cuando se aplican máscaras).

El siguiente ejemplo es un código de fragmento de archivo de configuración.

```
bac.crud.limit: 10
bac.crud.encoding: ascii
bac.crud.initCharacter: "LOW-VALUE"
bac.crud.defaultCharacter: " "
bac.crud.blankCharacter: " "
bac.crud.strictZoned: true
bac.crud.decimalSeparator: "."
bac.crud.currencySign: "$"
bac.crud.pictureCurrencySign: "$"
```

Configuración de la seguridad para la BAC

La configuración de la seguridad de la BAC se basa en los mecanismos que se detallan en esta página de documentación. El esquema de autenticación es OAuth2 y se proporcionan los detalles de configuración de Amazon Cognito o Keycloak.

Si bien se puede aplicar una configuración general, aquí es necesario especificar algunos detalles sobre la BAC. El acceso a las características de la BAC está protegido mediante una política basada en roles y se basa en los siguientes roles.

- ROLE_USER:
  - Rol de usuario básico
  - No se permiten la importación, exportación, creación o eliminación de conjuntos de datos.
  - No hay control sobre las políticas de almacenamiento en caché.
  - No se permiten características de administración.
- ROLE_ADMIN:

- Hereda permisos ROLE_USER.
- Se permiten todas las operaciones de conjuntos de datos.
- Se permite la administración de políticas de almacenamiento en caché.

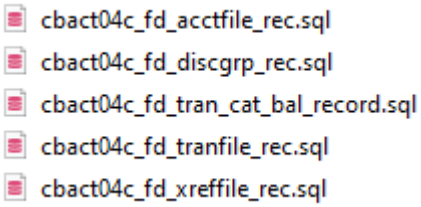
Instalación de las máscaras

En el almacenamiento de Blusam, los registros de los conjuntos de datos se almacenan en una columna de matriz de bytes en la base de datos, por motivos de versatilidad y rendimiento. Tener acceso a una vista estructurada, mediante campos, de los registros comerciales, basada en el punto de vista de la aplicación, es una característica práctica de la BAC. Esto se basa en las máscaras SQL generadas durante el proceso de modernización BluAge impulsado.

Para generar las máscaras SQL, asegúrese de establecer la opción correspondiente (`export.SQL.masks`) en la configuración del Centro de BluInsights Transformación en true:

Las máscaras forman parte de los artefactos de modernización desde los que se pueden descargar BluInsights para un proyecto determinado. Son scripts SQL, organizados por programas modernizados, que ofrecen el punto de vista aplicativo sobre los registros de conjuntos de datos.

Por ejemplo, con la aplicación de CardDemo ejemplo de AWS, puede encontrar en los artefactos descargados del resultado de la modernización de esta aplicación las siguientes máscaras SQL para el programa CBAct04c.cbl:



El nombre de cada máscara SQL es la concatenación del nombre del programa y el nombre de la estructura de registros de un conjunto de datos determinado dentro del programa.

Por ejemplo, si observamos el programa [CBACT04C.cbl], la entrada de control de archivos dada:

```
    FILE-CONTROL.
        SELECT TCATBAL-FILE ASSIGN TO TCATBALF
               ORGANIZATION IS INDEXED
               ACCESS MODE  IS SEQUENTIAL
               RECORD KEY   IS FD-TRAN-CAT-KEY
               FILE STATUS  IS TCATBALF-STATUS.
```

está asociada a la definición de registro FD determinada

```
       FILE SECTION.
       FD  TCATBAL-FILE.
       01  FD-TRAN-CAT-BAL-RECORD.
           05 FD-TRAN-CAT-KEY.
              10 FD-TRANCAT-ACCT-ID             PIC 9(11).
              10 FD-TRANCAT-TYPE-CD             PIC X(02).
              10 FD-TRANCAT-CD                  PIC 9(04).
           05 FD-FD-TRAN-CAT-DATA               PIC X(33).
```

La máscara SQL coincidente denominada `cbact04c_fd_tran_cat_bal_record.SQL` es la máscara que proporciona el punto de vista del programa CBACT04C.cbl en el registro FD denominado `FD-TRAN-CAT-BAL-RECORD`.

Su contenido es:

```
-- Generated by Blu Age Velocity
-- Mask : cbact04c_fd_tran_cat_bal_record

INSERT INTO mask (name, length) VALUES ('cbact04c_fd_tran_cat_bal_record', 50);
```

```
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_trancat_acct_id', 1, 11, false, 'zoned', 'integerSize=11!fractionalSize=0!
signed=false', (SELECT MAX(id) FROM mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_trancat_type_cd', 12, 2, false, 'alphanumeric', 'length=2', (SELECT MAX(id) FROM
 mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk)
 VALUES ('fd_trancat_cd', 14, 4, false, 'zoned', 'integerSize=4!fractionalSize=0!
signed=false', (SELECT MAX(id) FROM mask));
  INSERT INTO mask_item (name, c_offset, length, skip, type, options, mask_fk) VALUES
 ('fd_fd_tran_cat_data', 18, 33, false, 'alphanumeric', 'length=33', (SELECT MAX(id)
 FROM mask));
```

Las mascarillas se almacenan en el almacén de Blusam mediante dos tablas:

- máscara: se usa para identificar las máscaras. Las columnas de la tabla de máscaras son:
  - nombre: se usa para almacenar la identificación de la máscara (se usa como clave principal, por lo que debe ser único)
  - longitud: tamaño en bytes de la máscara de registro
- mask_item: se utiliza para almacenar los detalles de la máscara. Cada campo elemental de una definición de registro FD generará una fila en la tabla mask_item, con detalles sobre cómo interpretar la parte del registro determinada. Las columnas de la tabla mask_item son:
  - name: nombre del campo de registro, basado en el nombre elemental, utilizando minúsculas y sustituyendo el guion por el carácter de subrayado
  - c_offset: desplazamiento basado en 1 de la subparte de registro, utilizado para el contenido del campo
  - length: longitud en bytes de la subparte de registro, utilizada para el contenido del campo
  - skip: marca para indicar si la parte del registro determinada debe omitirse o no en la presentación de la vista
  - type: el tipo de campo (basado en su cláusula de imagen heredada)
  - options: opciones de tipo adicionales, dependientes de tipo
  - mask_fk: referencia al identificador de máscara al que asociar este elemento

Tenga en cuenta lo siguiente:

- Las máscaras SQL representan el punto de vista de un programa sobre los registros de un conjunto de datos: varios programas pueden tener un punto de vista diferente sobre un conjunto

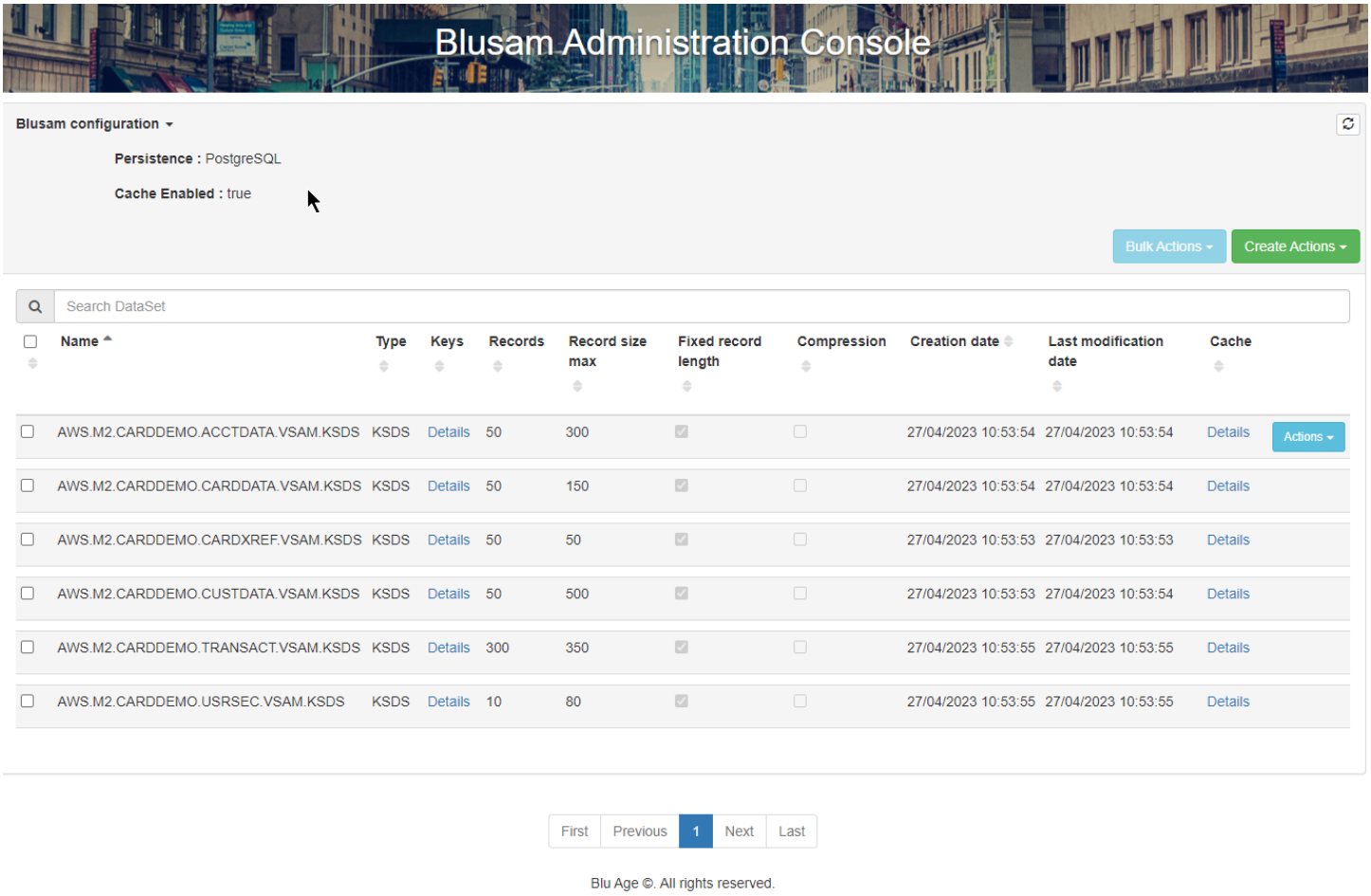
de datos determinado; instale únicamente las máscaras que considere pertinentes para su propósito.

- Una máscara SQL también puede representar el punto de vista de un programa basándose en una estructura de datos de 01 de la sección WORKING STORAGE, no solo de un registro FD. Las máscaras SQL se organizan en subcarpetas en función de su naturaleza:
  - Las máscaras basadas en registros FD se encontrarán en la subcarpeta denominada `file`.
  - Las máscaras basadas en la estructura de datos 01 se encontrarán en la subcarpeta denominada `working`.

  Si bien las definiciones de los registros FD siempre coinciden con el contenido de registro de un conjunto de datos, es posible que las estructuras de datos 01 no estén alineadas o que solo representen un subconjunto de un registro de conjunto de datos. Antes de usarlas, inspeccione el código y comprenda las posibles deficiencias.

Uso de la BAC

Como la BAC está protegida y proporciona permisos para usar las características en función del rol del usuario, el primer paso para acceder a la aplicación es autenticarse. Tras el paso de autenticación, se le redirigirá a la página de inicio. La página de inicio presenta la lista paginada de conjuntos de datos que se encuentra en el almacenamiento de Blusam:



Para volver a la página de inicio con la lista de conjuntos de datos, elija el logotipo de Blu Age en la esquina superior izquierda de cualquier página de la aplicación. En la siguiente imagen se muestra el logotipo.

El encabezado plegable, denominado «BluSam configuración», contiene información sobre la configuración de BluSam almacenamiento utilizada:

- `Persistence`: el motor de almacenamiento persistente (PostgreSQL)
- `Cache Enabled`: si la caché de almacenamiento está habilitada

En la parte derecha del encabezado, hay dos listas desplegables; cada una de las ellas muestra las operaciones relacionadas con los conjuntos de datos:

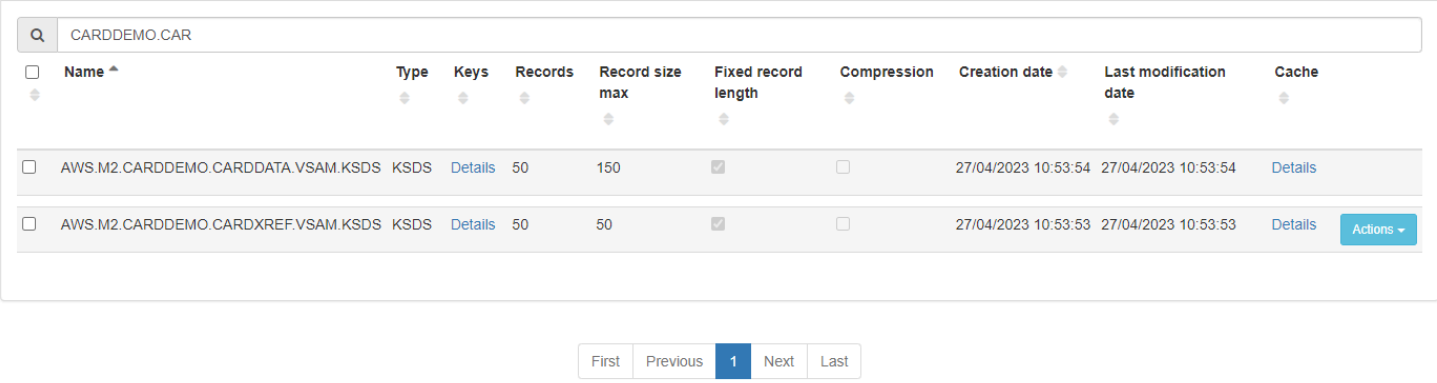
- Acciones masivas

- Crear acciones

Para obtener más información sobre el contenido detallado de estas listas, consulte the section called “Operaciones de conjuntos de datos existentes”.

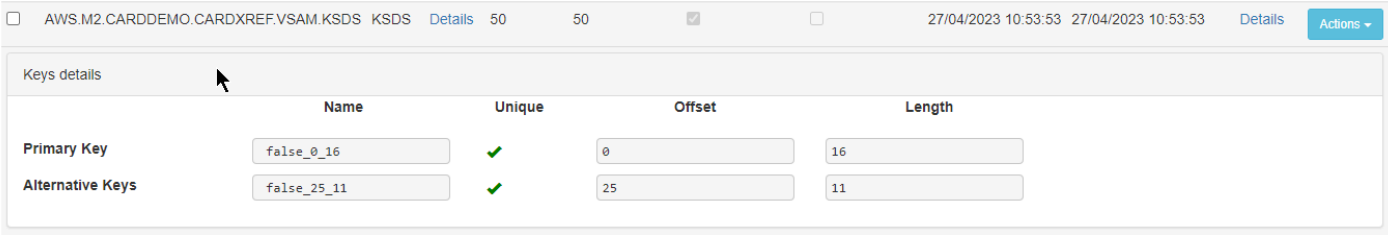
El botón Acciones masivas está deshabilitado cuando no se ha seleccionado ningún conjunto de datos.

Puede utilizar el campo de búsqueda para filtrar la lista en función de los nombres de los conjuntos de datos:



La siguiente lista paginada muestra un conjunto de datos por fila de tabla, con las siguientes columnas:

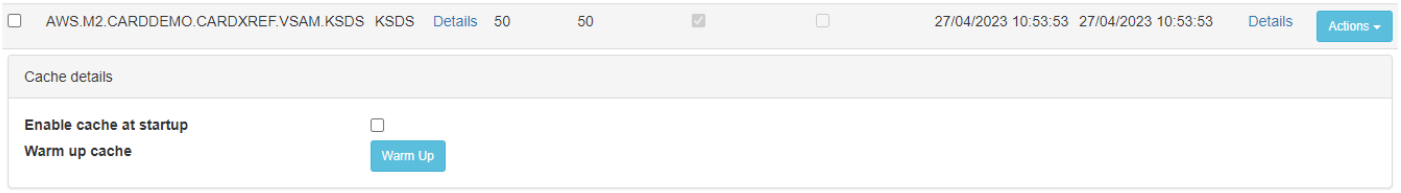
- Casilla de selección: casilla para seleccionar el conjunto de datos actual.
- Nombre: el nombre del conjunto de datos.
- Tipo: el tipo del conjunto de datos, uno de los siguientes:
  - KSDS
  - ESDS
  - RRDS
- Teclas: enlace para mostrar u ocultar detalles sobre las teclas (si las hay). Por ejemplo, el KSDS indicado tiene la clave principal obligatoria y una clave alternativa.



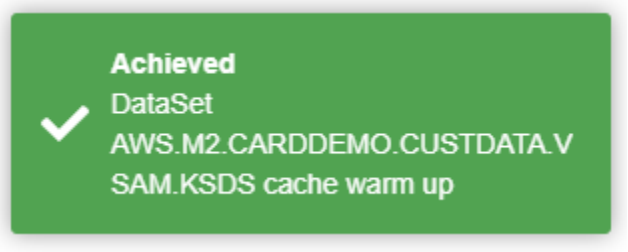
Hay una fila por clave, con las siguientes columnas. Ninguno de los campos se puede editar.

- Naturaleza de la clave: clave principal o clave alternativa
- Nombre: el nombre de la clave
- Única: indica si la clave acepta entradas duplicadas
- Desplazamiento: desplazamiento del inicio de la clave dentro del registro
- Longitud: longitud en bytes de la parte clave del registro

- Registros: el número total de registros del conjunto de datos.
- Tamaño máximo de registro: el tamaño máximo de los registros, expresado en bytes.
- Longitud de registro fija: casilla de verificación que indica si los registros son de longitud fija (seleccionados) o variables (no seleccionados).
- Compresión: casilla de verificación que indica si se aplica la compresión (está seleccionada) o no (no está seleccionada) a los índices almacenados.
- Fecha de creación: la fecha en que se creó el conjunto de datos en el almacenamiento de Blusam.
- Fecha de la última modificación: la fecha en que el conjunto de datos se actualizó por última vez en el almacenamiento de Blusam.
- Caché: un enlace para mostrar u ocultar detalles sobre la estrategia de almacenamiento en caché aplicada a este conjunto de datos.



  - Habilitar la memoria caché al inicio: una casilla de verificación para especificar la estrategia de almacenamiento en caché de inicio para este conjunto de datos. Si se selecciona, el conjunto de datos se cargará en la memoria caché al inicio.
  - Memoria caché de preparación: un botón para cargar el conjunto de datos determinado en la memoria caché y empezar inmediatamente (pero hidratar la memoria caché lleva algún tiempo, según el tamaño del conjunto de datos y el número de claves). Una vez que el conjunto de datos se carga en la memoria caché, aparece una notificación como la siguiente.

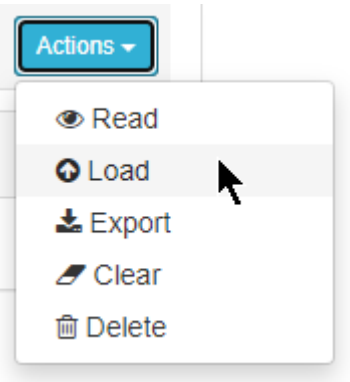


- Acciones: una lista desplegable de posibles operaciones de conjuntos de datos. Para obtener más información, consulte the section called “Operaciones de conjuntos de datos existentes”.

En la parte inferior de la página, hay un widget de navegación paginado normal para navegar por las páginas de la lista de conjuntos de datos.

Operaciones de conjuntos de datos existentes

Para cada conjunto de datos de la lista paginada, hay una lista desplegable Acciones con el siguiente contenido:



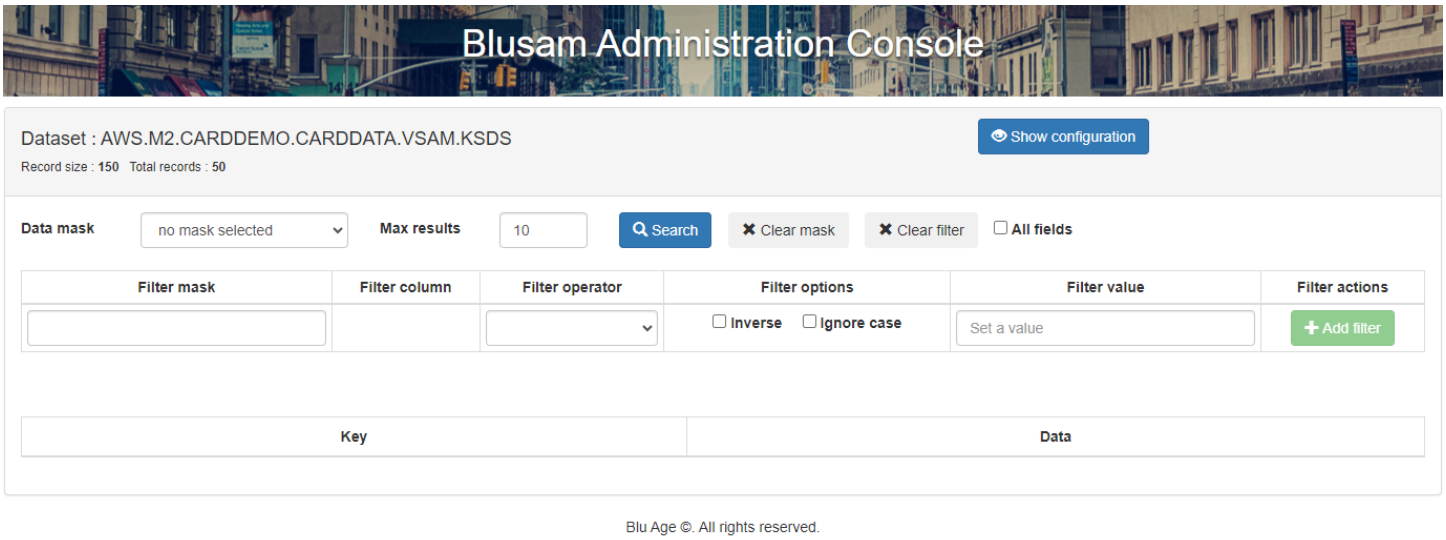
Cada elemento de la lista es un enlace activo que permite realizar la acción especificada en el conjunto de datos:

- Lectura: busque registros en los conjuntos de datos.
- Carga: importe registros de un archivo de conjunto de datos heredado.
- Exportar: exporte los registros e un archivo sin formato (compatible con los sistemas heredados).
- Borrar: elimine todos los registros del conjunto de datos.
- Eliminar: elimine el conjunto de datos del almacenamiento.

Los detalles de cada acción se encuentran en las siguientes secciones.

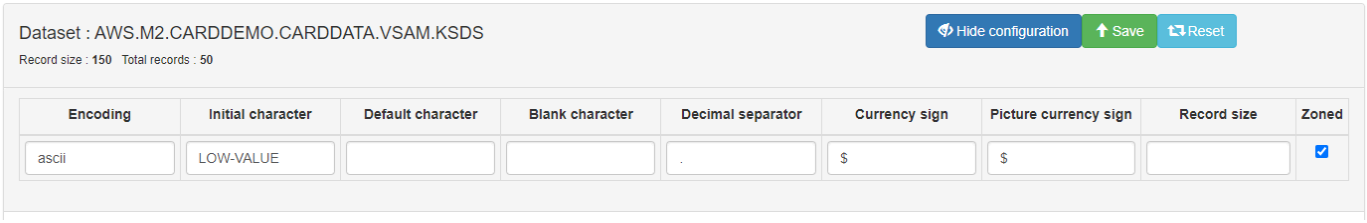
Búsqueda de registros en un conjunto de datos

Al elegir la acción Leer para un conjunto de datos determinado, se obtiene la página siguiente.



La página está compuesta por:

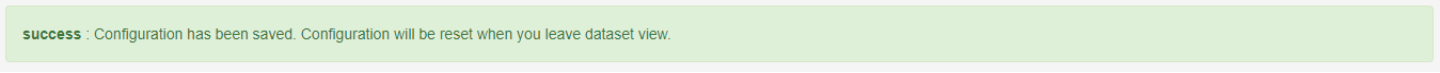
- un encabezado, con:
  - Conjunto de datos: el nombre del conjunto de datos
  - Tamaño del registro: longitud fija del registro, expresada en bytes
  - Total de registros: el número total de registros almacenados para este conjunto de datos
  - Botón Mostrar configuración (en el lado derecho): un botón de alternancia para mostrar/ocultar la configuración del conjunto de datos. Al principio, la configuración está oculta. Al utilizar el botón, verá la configuración, tal y como se muestra en la siguiente imagen.



    Cuando se muestra la configuración, hay dos botones nuevos: Guardar y Restablecer, que se usan respectivamente para:

    - guardar la configuración de este conjunto de datos y de la sesión de trabajo actual;
    - restablecer la configuración a los valores predeterminados en todos los campos.
  - Una lista de propiedades configurables para personalizar la experiencia de navegación para el conjunto de datos determinado.

Las propiedades configurables coinciden con las propiedades de configuración descritas en the section called “Archivo de configuración dedicado de BAC”. Consulte esa sección para comprender el significado de cada columna y los valores aplicables. Aquí se puede redefinir cada valor para el conjunto de datos y guardarse para la sesión de trabajo (mediante el botón Guardar). Después de guardar la configuración, aparecerá un banner similar al que se muestra en la siguiente imagen.



El banner indica que la sesión de trabajo finaliza al salir de la página actual.

Hay una propiedad configurable adicional que no está documentada en la sección de configuración: el tamaño del registro. Se usa para especificar un tamaño de registro determinado, expresado en bytes, que filtrará las máscaras aplicables a este conjunto de datos: solo las máscaras cuya longitud total coincida con el tamaño de registro dado aparecerán en la lista desplegable Enmascaramiento de datos.

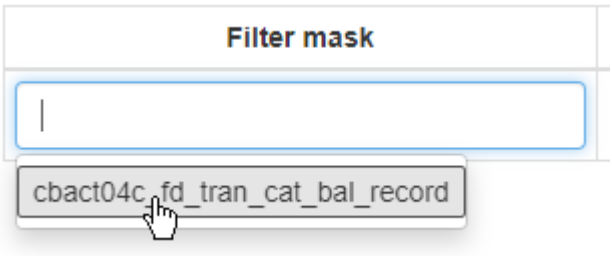
La recuperación de registros del conjunto de datos se desencadena mediante el botón Buscar, utilizando todas las opciones y filtros cercanos.

Primera línea de opciones:

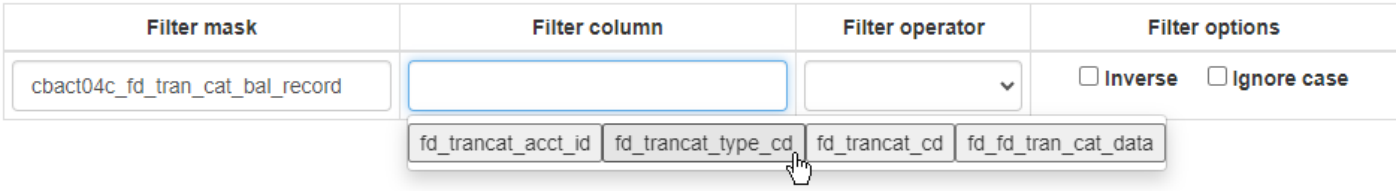
- la lista desplegable Enmascaramiento de datos muestra las máscaras aplicables (respetando el tamaño del registro). Tenga en cuenta que hacer coincidir el tamaño del registro no es suficiente para que sea una máscara útil y eficaz. La definición de máscara también debe ser compatible con el contenido de los registros. El enmascaramiento de datos elegido aquí tiene
- Resultados máximos: limita el número de registros recuperados por la búsqueda. Establézcalo en 0 para obtener resultados ilimitados (paginados) del conjunto de datos.
- Botón de búsqueda: inicie la recuperación de registros mediante filtros y opciones.
- Botón para borrar la máscara: borrará la máscara utilizada si la hubiera y devolverá a la página de resultados a una presentación de claves o datos sin procesar.
- Botón Borrar filtro: borrará los filtros utilizados si los hubiera y actualizará la página de resultados en consecuencia.
- Todos los campos se alternan: cuando se seleccionan, los elementos de máscara definidos con `skip = true` se muestran de todos modos; de lo contrario, los elementos de máscara con `skip = true` se ocultan.

Próximas líneas de filtros: es posible definir una lista de filtros en función del uso de las condiciones de filtrado aplicadas a los campos (columnas) de una máscara determinada, como se muestra en la siguiente imagen.

- Máscara de filtro: el nombre de la máscara de la que se elegirá la columna de filtrado. Al elegir el campo, aparece la lista de máscaras aplicables. Puede elegir la máscara que desee en esa lista.

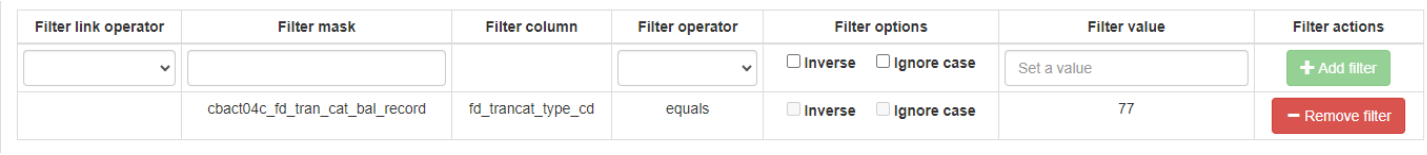


- Columna de filtro: el nombre del campo (columna) de la máscara, que se utiliza para filtrar registros. Al elegir el campo, aparece la lista de columnas de la máscara. Para rellenar el campo Filtrar columna, elija la celda que desee.



- Operador de filtro: operador que se aplicará a la columna seleccionada. Los siguientes operadores están disponibles:
  - es igual a: el valor de la columna del registro debe ser igual al valor del filtro.
  - empieza por: el valor de la columna del registro debe empezar por el valor del filtro.
  - termina con: el valor de la columna del registro debe terminar con el valor del filtro.
  - contiene: el valor de la columna del registro debe contener el valor del filtro.
- Opciones de filtro:
  - Inversa: aplique la condición inversa para el operador de filtro; por ejemplo, igual a se sustituye por no igual a.
  - Ignorar mayúsculas y minúsculas: ignorar mayúsculas y minúsculas en las comparaciones alfanuméricas para el operador del filtro.
- Valor de filtro: el valor utilizado para la comparación por el operador del filtro con la columna del filtro.

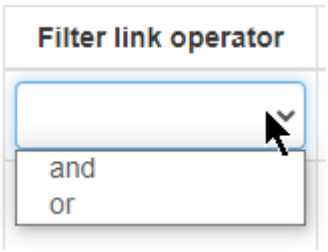
Una vez establecido el número mínimo de elementos de filtro (se debe establecer como mínimo la máscara del filtro, la columna del filtro, el operador del filtro y el valor del filtro), se habilita el botón Agregar filtro y, al hacer clic en él, se crea una nueva condición de filtro en los registros recuperados. Se agrega otra fila de condiciones de filtro vacía en la parte superior y la condición de filtro agregada tiene un botón Eliminar filtro que se puede utilizar para suprimir la condición de filtro determinada:



Al iniciar la búsqueda, los resultados filtrados aparecen en una tabla paginada.

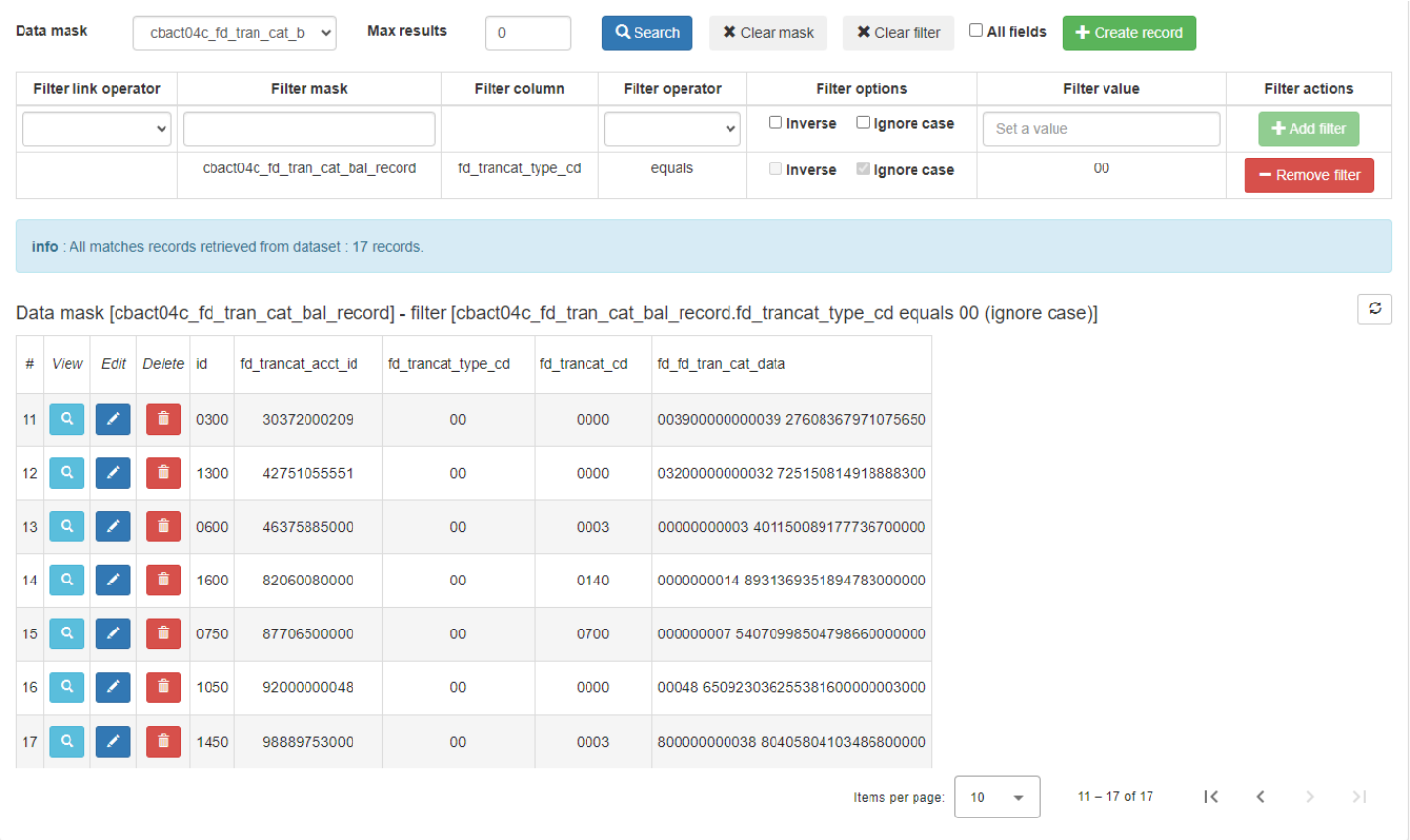
Nota

- Los filtros sucesivos se enlazan mediante y u o. Cada nueva definición de filtro comienza con la configuración del operador de enlace, como se muestra en la siguiente imagen.



- Es posible que no haya ningún registro que coincida con las condiciones de filtro especificadas.

De lo contrario, la tabla de resultados se parece a la de la imagen siguiente.



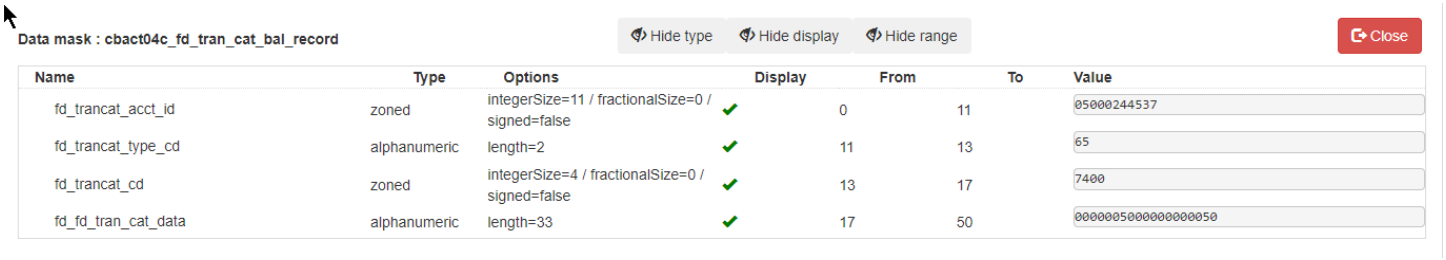
Un encabezado indica el número total de registros que cumplen las condiciones del filtro. Después del encabezado, verá lo siguiente.

- Recordatorio de la máscara de datos utilizada (si la hubiera) y de las condiciones de filtro.
- Un botón de actualización que puede utilizar para activar la actualización de toda la tabla de resultados con los valores más reciente del almacenamiento de Blusam (ya que podría haberla actualizado otro usuario, por ejemplo).

Para cada registro recuperado, la tabla tiene una fila que muestra el resultado de la aplicación de la máscara de datos al contenido de los registros. Cada columna es la interpretación de la subparte del registro según el tipo de columna (y utilizando la codificación seleccionada). A la izquierda de cada fila, hay tres botones:

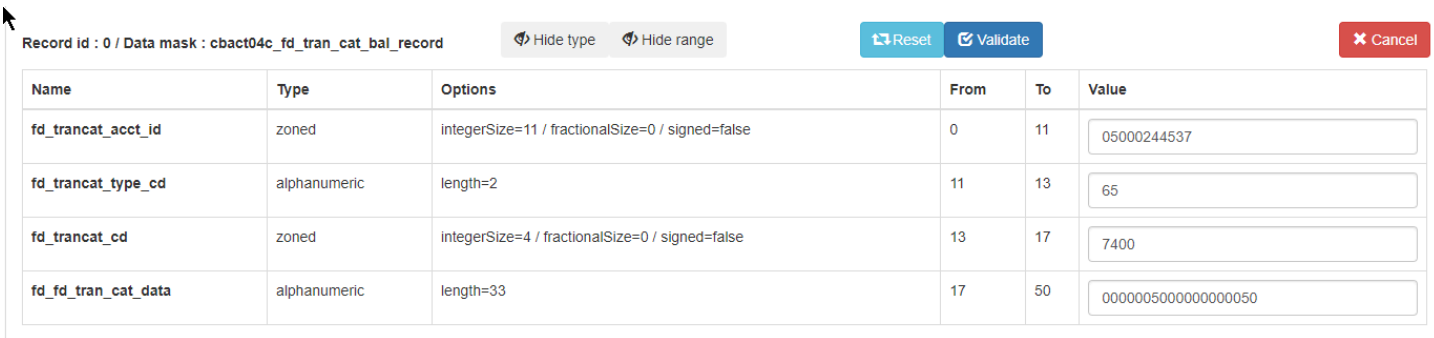
- un botón de lupa: lleva a una página específica que muestra el contenido detallado del registro;
- un botón de lápiz: lleva a una página de edición dedicada al contenido del registro;
- un botón de papelera: se usa para borrar el registro determinado del almacenamiento de Blusam.

Visualización del contenido del registro en detalle:



- Tres botones de alternancia para ocultar o mostrar algunas columnas:
  - Ocultar/mostrar el tipo
  - Ocultar/mostrar el indicador de visualización
  - Ocultar/mostrar el rango
- Para salir de esta página dedicada y volver a la tabla de resultados, elija Cerrar.
- Cada fila representa una columna de la máscara de datos, con las siguientes columnas:
  - Nombre: el nombre de la columna
  - Tipo: el tipo de la columna
  - Visualización: el indicador de pantalla; se mostrará una marca de verificación verde si el elemento de máscara correspondiente está definido con `skip = false`; de lo contrario, aparecerá una cruz roja.
  - Desde y hasta: el rango basado en 0 para la subparte del registro.
  - Valor: el valor interpretado de la subparte del registro, utilizando el tipo y la codificación.

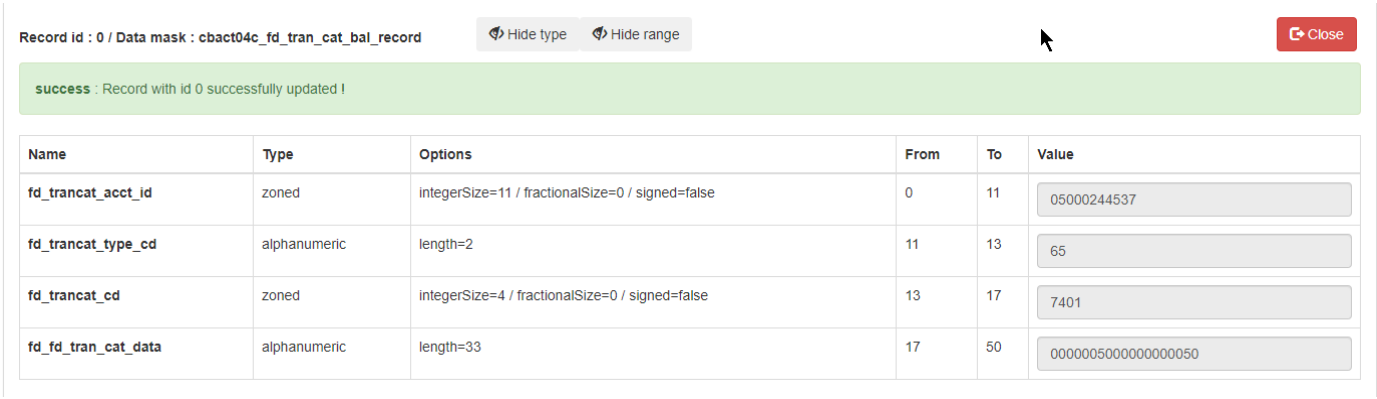
Edición del contenido del registro:



La página de edición es similar a la página de visualización descrita anteriormente, excepto que los valores de los elementos de la máscara son editables. Tres botones controlan el proceso de actualización:

- Restablecer: restablece los valores editables a los valores de registro iniciales (antes de cualquier edición).
- Validar: valida la entrada con respecto al tipo de elemento de máscara. Para cada elemento de máscara, el resultado de la validación se imprimirá con etiquetas visuales (0K y una casilla de verificación si la validación se realizó correctamente, ERR0R y una cruz roja si se produjo un error en la validación, junto con un mensaje de error con pistas acerca del error de validación). Si la validación se ha realizado correctamente, aparecerán dos botones nuevos:
  - Guardar: intente actualizar el registro existente en el almacenamiento de Blusam.
  - Guardar una copia: intente crear un registro nuevo en el almacenamiento de Blusam.

  - Si se ha guardado correctamente el registro en el almacenamiento, aparecerá un mensaje y la página pasará al modo de solo lectura (ya no se pueden editar los valores de los elementos de máscara):

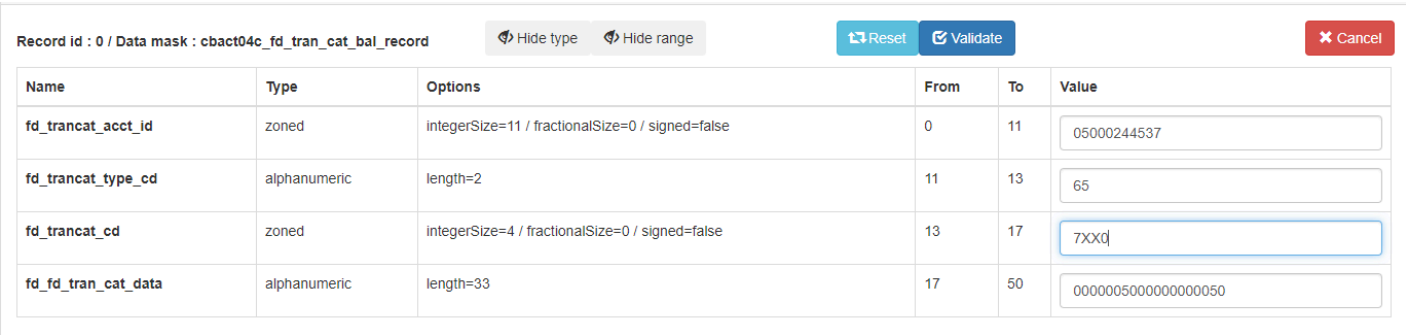


  - Si por alguna razón se produce un error en la persistencia del registro en el almacenamiento, aparecerá un mensaje de error en rojo que indica el motivo del error. El caso más habitual de los errores es que al almacenar el registro se dañe la clave (clave no válida o duplicada). Para ver una ilustración, consulte la nota siguiente.
  - Para salir, elija el botón Cerrar.
- Cancelar: finaliza la sesión de edición, cierra la página y le devuelve a la página de la lista de registros.

Nota:

- El mecanismo de validación solo comprueba que el valor del elemento de máscara sea formalmente compatible con el tipo de elemento de máscara. Por ejemplo, vea esta validación con errores en un elemento de máscara numérica:

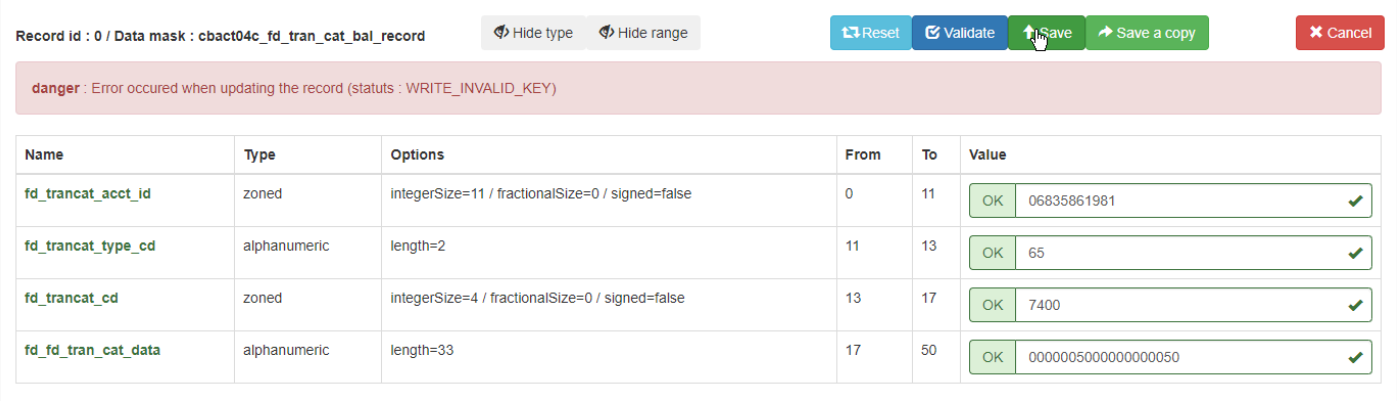
- El mecanismo de validación podría intentar corregir automáticamente la entrada no válida y mostrar un mensaje informativo en azul para indicar que el valor se ha corregido automáticamente, según su tipo. Por ejemplo, si se introduce 7XX0 como el valor numérico en el elemento de máscara numérica `fd_trncat_cd`:



  La validación de llamadas genera lo siguiente:

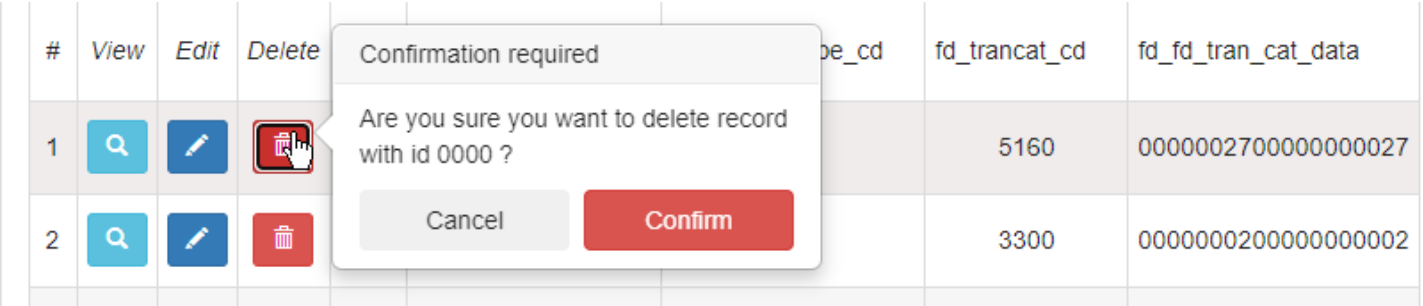
- El mecanismo de validación no comprueba si el valor determinado es válido en cuanto a la integridad de la clave (si hay alguna clave única incluida en el conjunto de datos determinado). Por ejemplo, a pesar de que la validación se haya realizado correctamente, si los valores

proporcionados llevan a una situación de clave no válida o duplicada, se generará un error en la persistencia y aparecerá un mensaje de error:



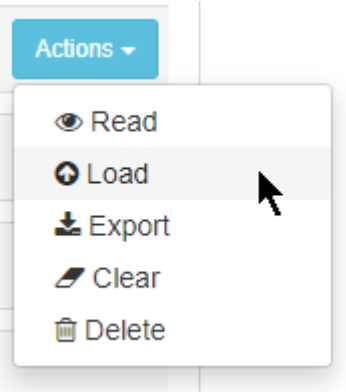
Eliminación de un registro:

Para eliminar un registro, elija el botón de papelera:

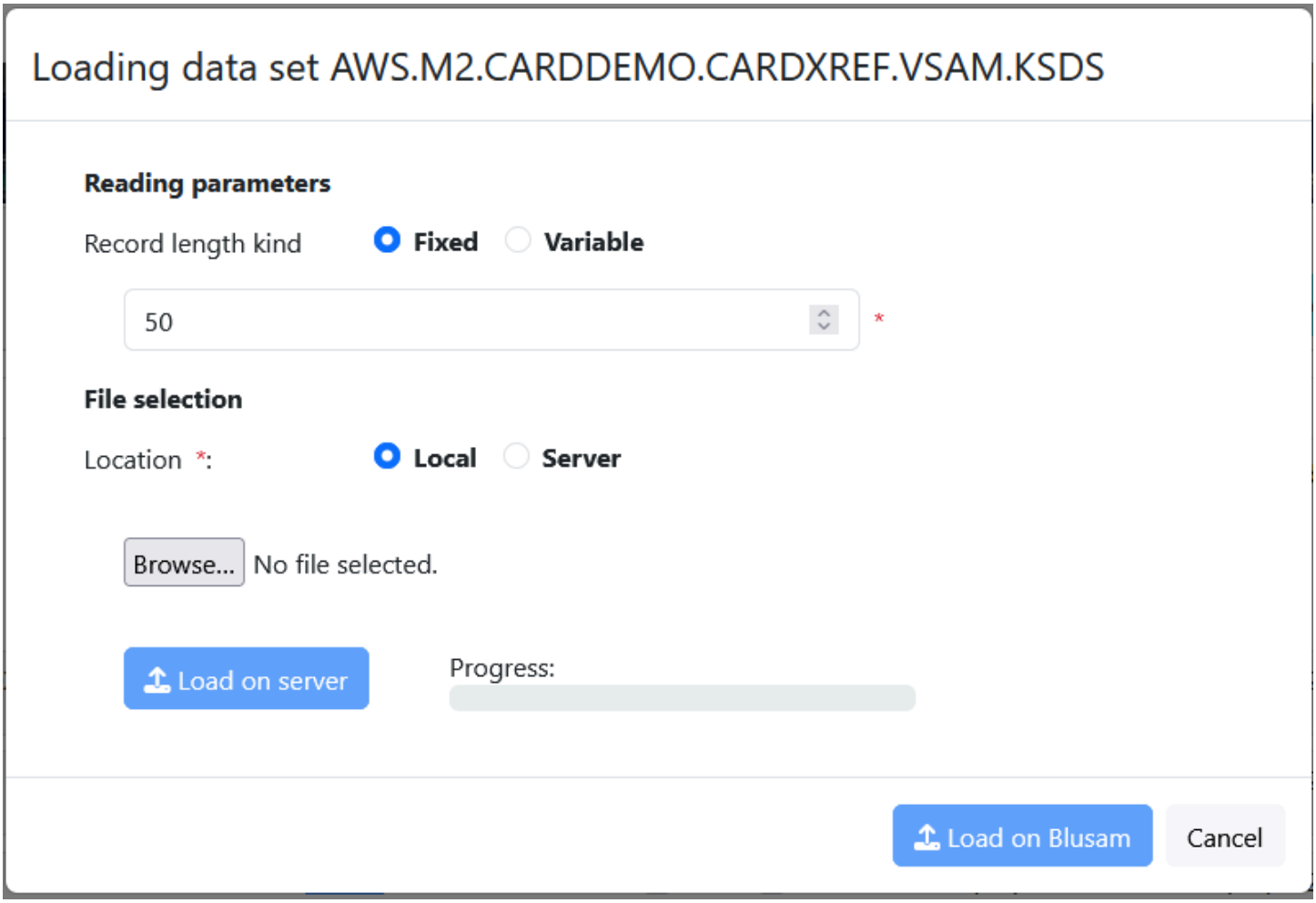


Carga de registros en un conjunto de datos

Para cargar registros en un conjunto de datos, elija Acciones y, a continuación, Cargar.



Aparecerá una ventana con opciones de carga.

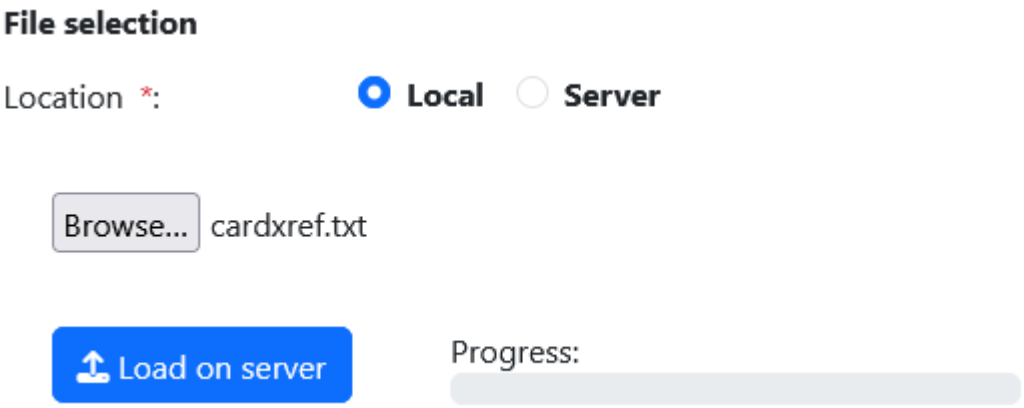


Al principio, los botones Cargar en el servidor y Cargar en Blusam están deshabilitados.

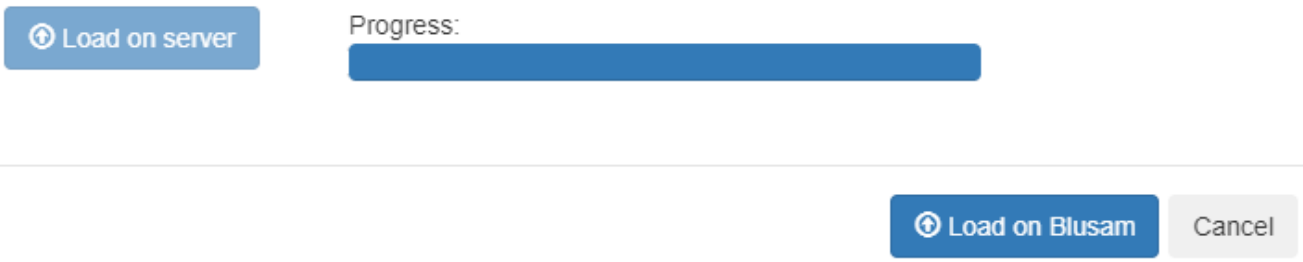
Parámetros de lectura:

- Tipo de longitud de registro:
  - Longitud de registro fija o variable: utilice el botón de opción para especificar si la exportación del conjunto de datos heredado utiliza registros de longitud fija o registros de longitud variable (se espera que los registros comiencen con bytes de RDW). Si elige Fijo, la longitud del registro debe especificarse (en bytes) como valor entero positivo en el campo de entrada. El valor debe rellenarse previamente con la información procedente del conjunto de datos. Si elige Variable, el campo de entrada determinado desaparecerá.
  - Selección de archivos:
    - Local: elija el archivo del conjunto de datos de su equipo local, utilizando el selector de archivos que aparece a continuación (Nota: El selector de archivos utiliza la configuración regional del navegador para imprimir sus mensajes; aquí está en francés, pero es posible

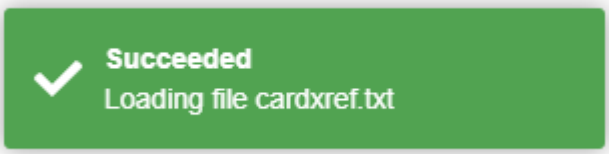
que tenga un aspecto diferente en su equipo, lo cual se espera). Tras realizar la selección, la ventana se actualiza con el nombre del archivo de datos y se habilita el botón Cargar en el servidor:



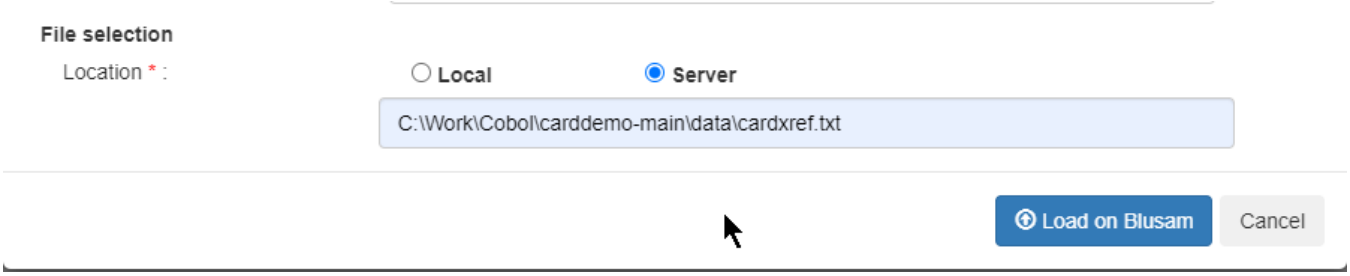
Elija Cargar en el servidor. Cuando la barra de progreso llegue a su fin, se habilitará el botón Cargar en Blusam:



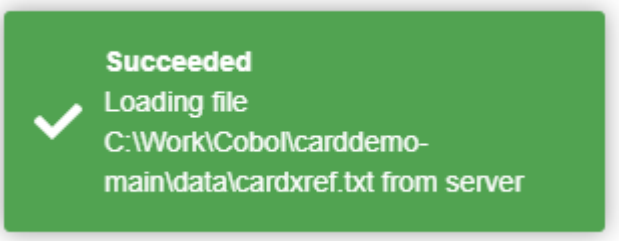
Para completar el proceso de carga en el almacén de Blusam, elija Cargar en Blusam. De lo contrario, seleccione Cancelar. Si decide continuar con el proceso de carga, aparecerá una notificación en la esquina inferior derecha cuando se complete el proceso de carga:



- Servidor: al elegir esta opción, aparecerá un campo de entrada y desaparecerá el botón Cargar en el servidor. El campo de entrada es donde debe especificar la ruta al archivo del conjunto de datos en el servidor de Blusam (esto supone que ha transferido primero el archivo determinado al servidor de Blusam). Después de especificar la ruta, se habilita la opción Cargar en Blusam:

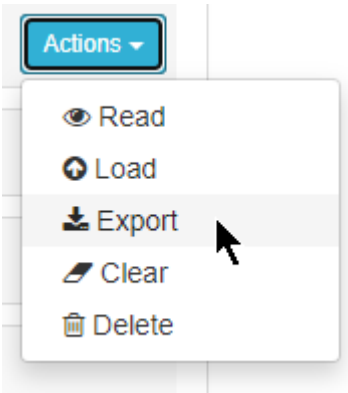


Para completar el proceso de carga, elija Cargar en Blusam. De lo contrario, seleccione Cancelar. Si decide continuar con la carga, aparecerá una notificación una vez finalizado el proceso de carga. La notificación es diferente de la que se carga desde el navegador, ya que muestra la ruta del servidor de archivos de datos seguida del texto desde el servidor:

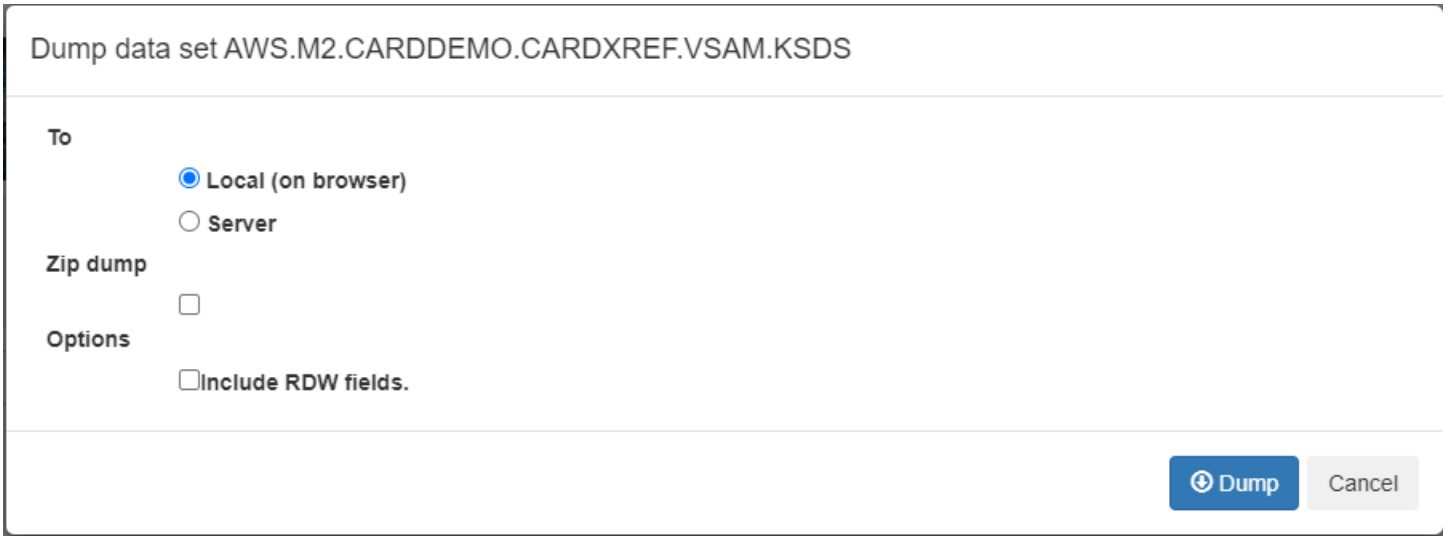


Exportación de registros desde un conjunto de datos

Para exportar los registros del conjunto de datos, elija Acciones en la fila del conjunto de datos actual y, a continuación, elija Exportar:

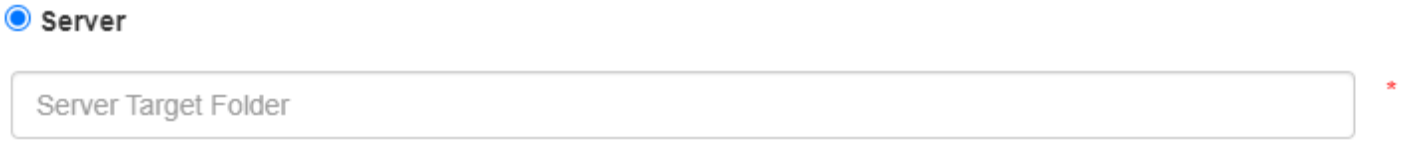


Aparecerá la siguiente ventana emergente.



Opciones:

A: seleccione un botón de opción para seleccionar el destino de la exportación, ya sea como descarga en el navegador (Local (en el navegador)) o a una carpeta determinada del Servidor que aloja la aplicación de BAC. Si opta por exportar con la opción Servidor, aparecerá un nuevo campo de entrada:



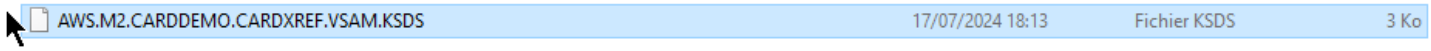
Como indica el asterisco rojo a la derecha del campo de entrada, es obligatorio proporcionar una ubicación de carpeta válida en el servidor (el botón de volcado estará inactivo mientras no se indique ninguna ubicación de carpeta).

Para exportar al servidor, debe disponer de suficientes derechos de acceso al sistema de archivos del servidor si piensa manipular el archivo del conjunto de datos exportado después de la exportación.

Volcado zip: casilla de verificación que genera un archivo comprimido en lugar de un archivo sin procesar.

Opciones: para incluir una palabra de descripción de registro (RDW) al principio de cada registro del conjunto de datos exportado, en el caso de un conjunto de datos de registros de longitud variable, seleccione Incluir campos RDW.

Para iniciar el proceso de exportación del conjunto de datos, elija Volcar. Si elige exportar al navegador, busque en la carpeta de descargas el archivo del conjunto de datos de exportación. El archivo tendrá el mismo nombre que el conjunto de datos:

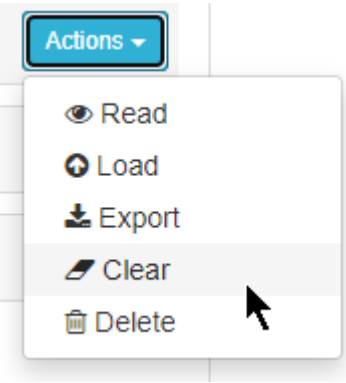


Nota:

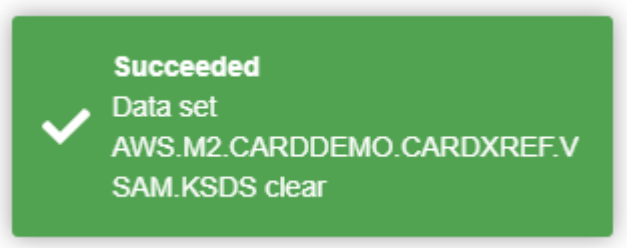
- En el caso de KSDS, los registros se exportarán siguiendo el orden de las claves principales.
- En el caso de ESDS y RRDS, los registros se exportarán siguiendo el orden RBA (dirección de bytes relativa).
- Para todos los tipos de conjuntos de datos, los registros se exportarán como matrices binarias sin procesar (no se realizará ningún tipo de conversión), lo que garantiza la compatibilidad directa con las plataformas heredadas.

Borrado de registros de un conjunto de datos

Para borrar todos los registros de un conjunto de datos, elija Acciones y, a continuación, Borrar:

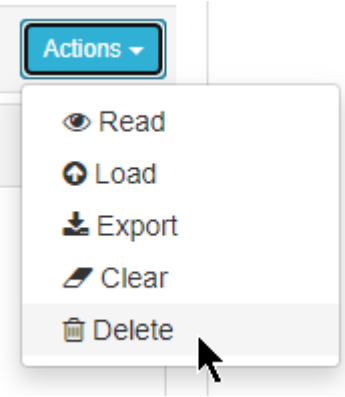


Después de eliminar todos los registros de un conjunto de datos, aparece la siguiente notificación.

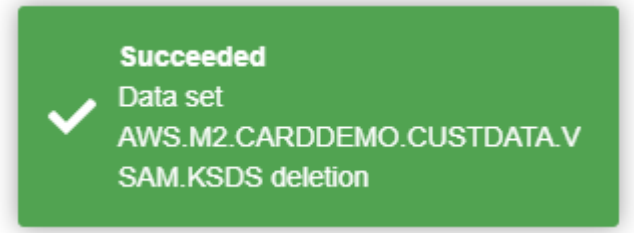


Eliminación de un conjunto de datos

Para eliminar un conjunto de datos, elija Acciones y, a continuación, Eliminar:



Después de eliminar un conjunto de datos, aparece la siguiente notificación:

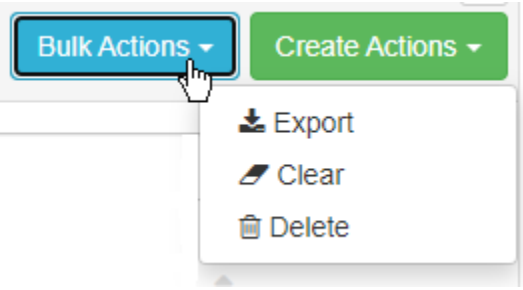


Operaciones masivas

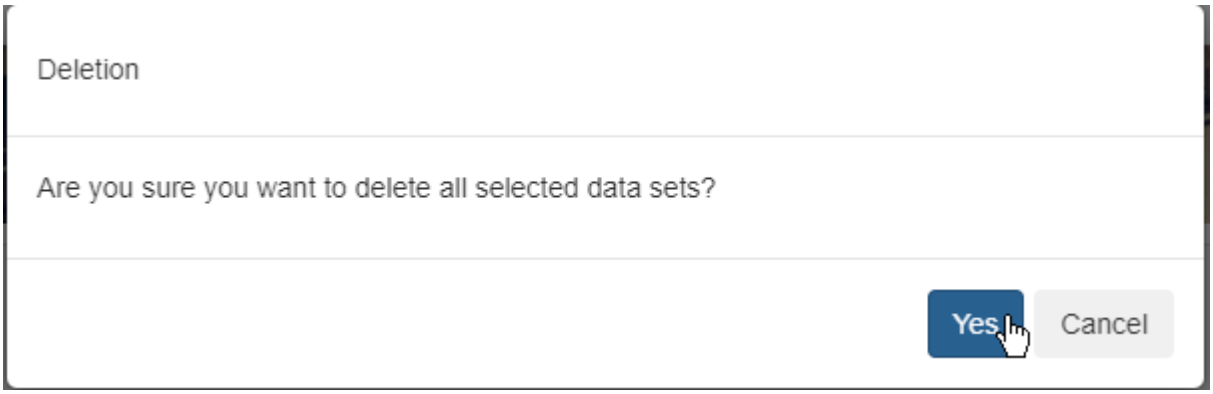
Hay tres operaciones masivas disponibles en los conjuntos de datos:

- Exportación
- Clear
- Eliminar

Las operaciones masivas solo se pueden aplicar a una selección de conjuntos de datos (es necesario seleccionar al menos un conjunto de datos); la selección de los conjuntos de datos se realiza marcando las casillas de selección situadas a la izquierda de las filas de los conjuntos de datos, en la tabla de listas de conjuntos de datos. Al seleccionar al menos un conjunto de datos, se habilitará la lista desplegable Acciones masivas:



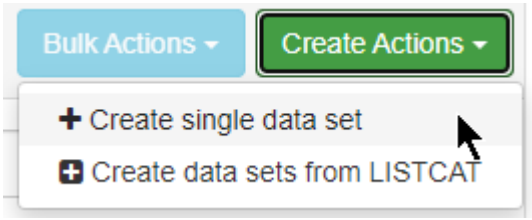
Además del hecho de que las acciones determinadas se aplican a una selección de conjuntos de datos y no a uno solo, las acciones son similares a las descritas anteriormente. Así pues, consulte la documentación específica sobre las acciones para obtener más información. El contenido del texto de las ventanas emergentes será algo diferente para reflejar su naturaleza masiva. Por ejemplo, al intentar eliminar varios conjuntos de datos, la ventana emergente tendrá el siguiente aspecto:



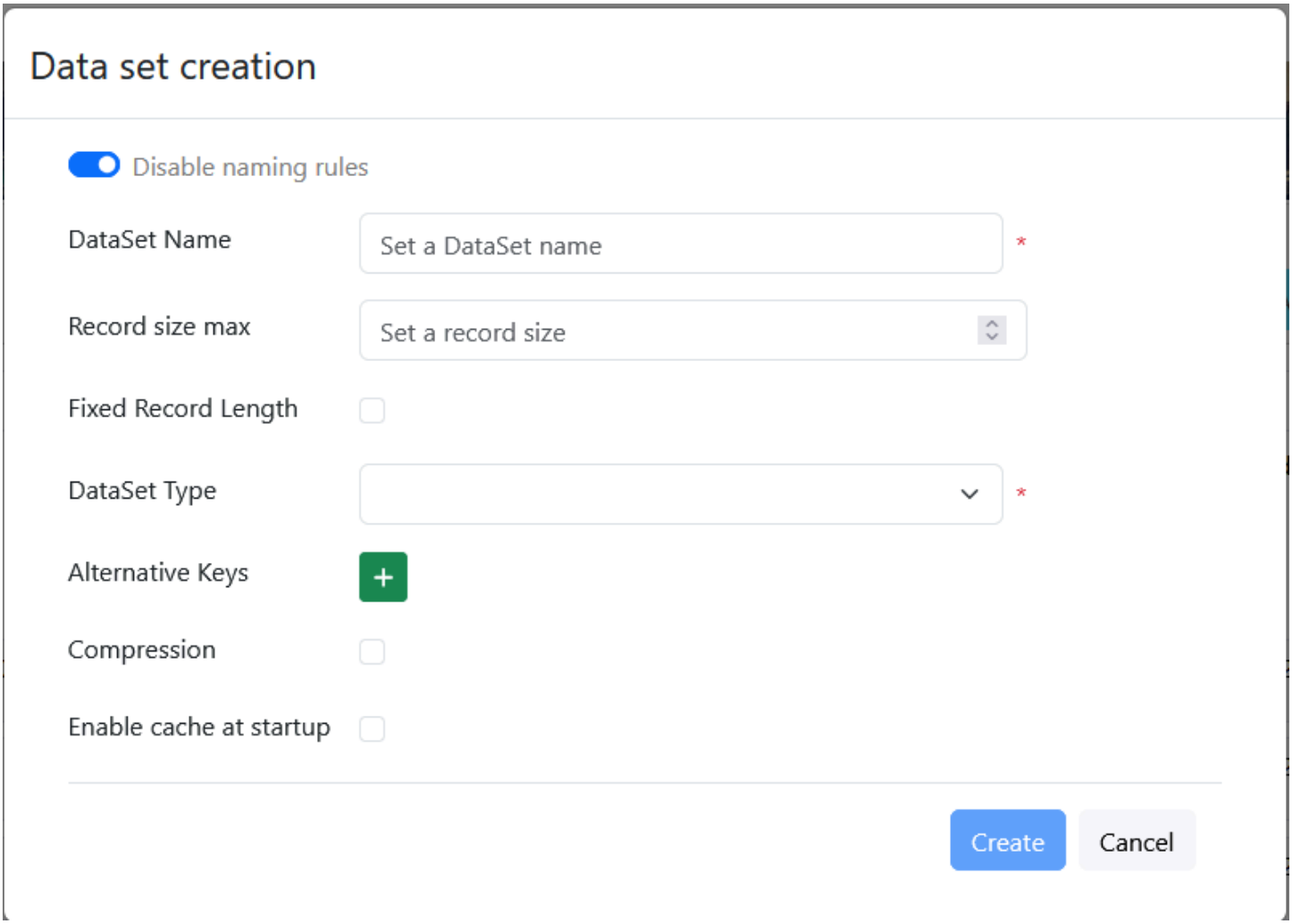
Creación de operaciones

Creación de un conjunto de datos único

Elija Acciones y, a continuación, Crear conjunto de datos único:

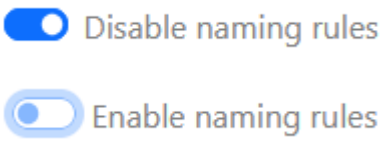


El formulario de creación del conjunto de datos se mostrará entonces como ventana emergente:

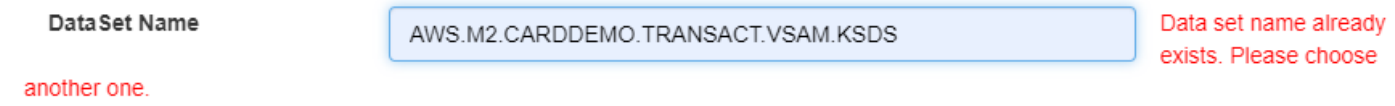


Puede especificar los siguientes atributos para la definición del conjunto de datos:

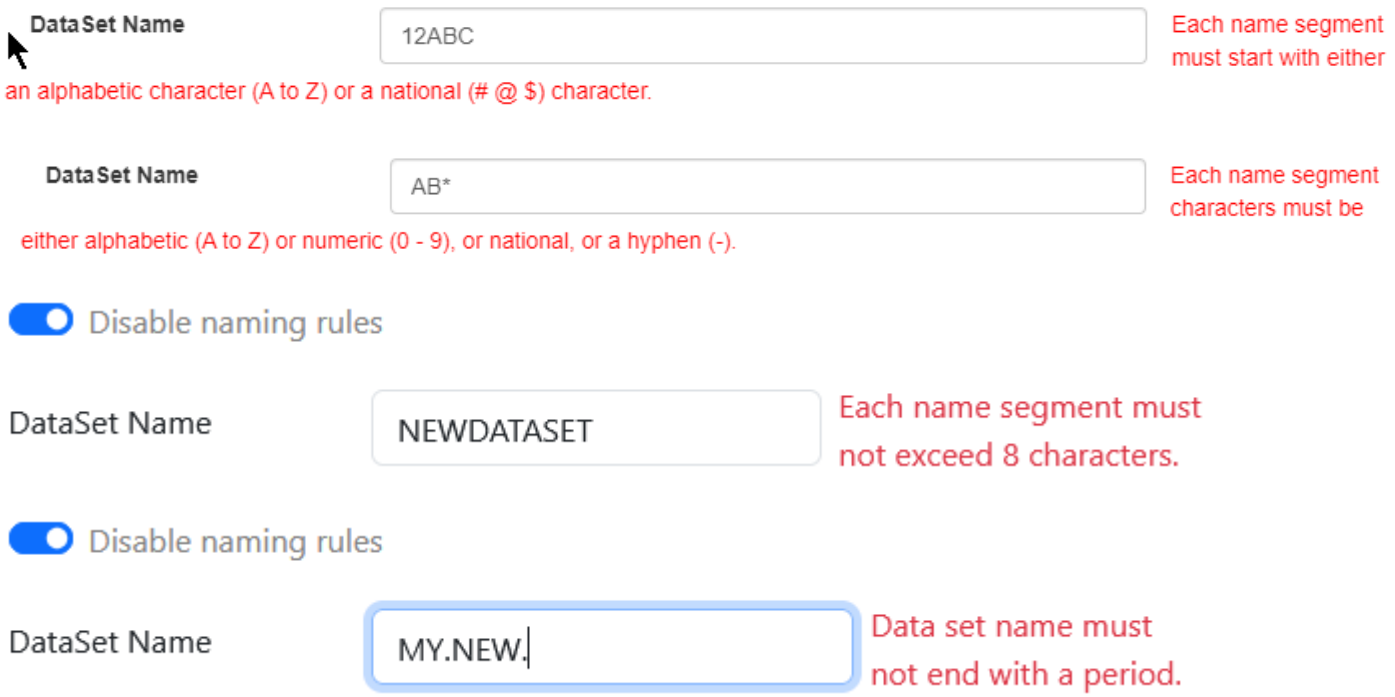
- Activación y desactivación de reglas de nomenclatura: utilice el widget de alternancia Deshabilitar reglas de nomenclatura/Habilitar reglas de nomenclatura para deshabilitar o activar las convenciones de nomenclatura de los conjuntos de datos. Le recomendamos que deje la opción activada en el valor predeterminado, con las reglas de nomenclatura de los conjuntos de datos habilitadas (el widget de alternancia debería mostrar la opción Deshabilitar reglas de nomenclatura):

- Nombre del conjunto de datos: el nombre del conjunto de datos. Si especifica un nombre que ya está en uso, aparecerá el siguiente mensaje de error.



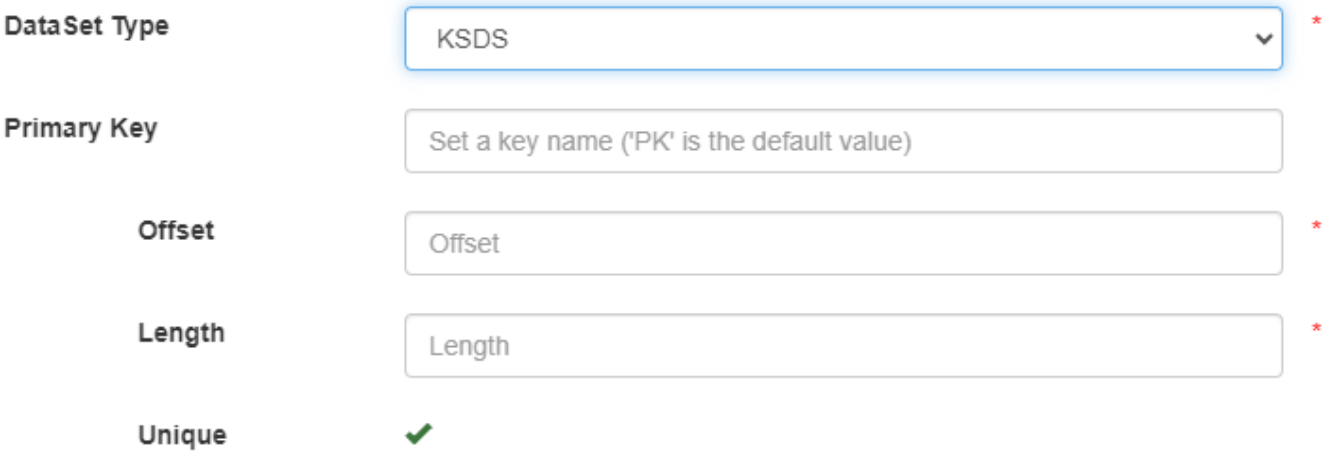
El nombre también debe respetar la convención de nomenclatura si está habilitado:



- Tamaño máximo del registro: debe ser un entero positivo que represente el tamaño del registro de un conjunto de datos con registros de longitud fija. Puede dejarlo en blanco para los conjuntos de datos con registros de longitud variable.
- Registro de longitud fija: casilla de verificación para especificar si la longitud del registro es fija o variable. Si se selecciona, el conjunto de datos tendrá registros de longitud fija; de lo contrario, la longitud del registro será variable.

  Al importar datos heredados a un conjunto de datos de registros de longitud variable, los registros heredados proporcionados deben contener la palabra de descripción de registro (RDW) que indica la longitud de cada registro.
- Tipo de conjunto de datos: lista desplegable para especificar el tipo de conjunto de datos actual. Se admiten los siguientes tipos:
  - ESDS
  - LargeESDS
  - KSDS

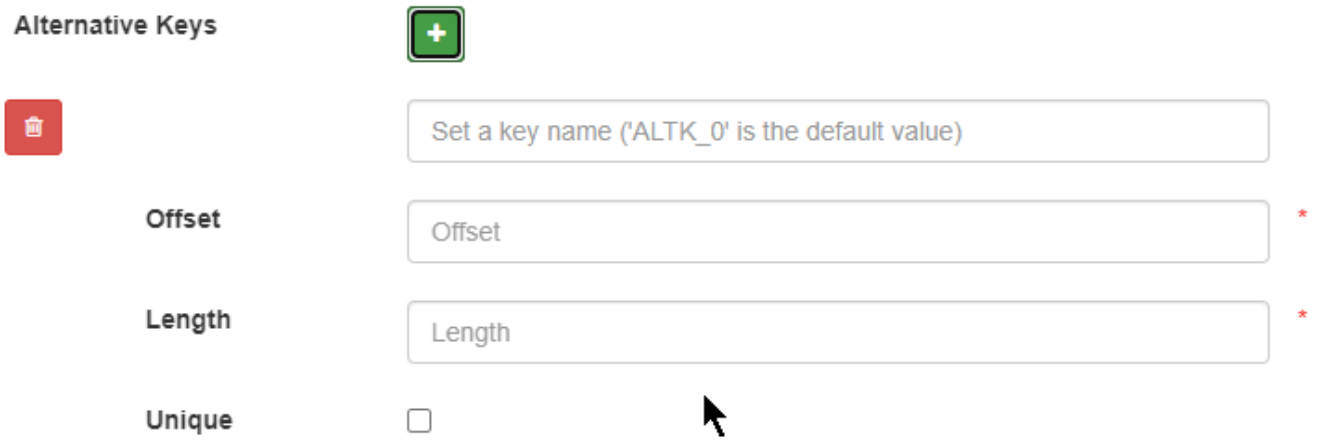
Para KSDS, debe especificar la clave principal:



Para la clave principal, haga lo siguiente:

- Nombre: este campo es opcional. El valor predeterminado es **PK**.
- Desplazamiento: el desplazamiento basado en 0 de la clave principal dentro del registro. El valor debe ser un número entero positivo. Este campo es obligatorio.
- Longitud: la longitud de la clave principal. Esta longitud debe ser un número entero positivo. Este campo es obligatorio.

Para KSDS y ESDS, si lo desea, puede definir opcionalmente un conjunto de claves alternativas. Para ello, pulse el botón Más situado delante de la etiqueta Teclas alternativas. Cada vez que pulse ese botón, aparecerá una nueva sección de definición de claves alternativas en el formulario de creación de conjunto de datos:



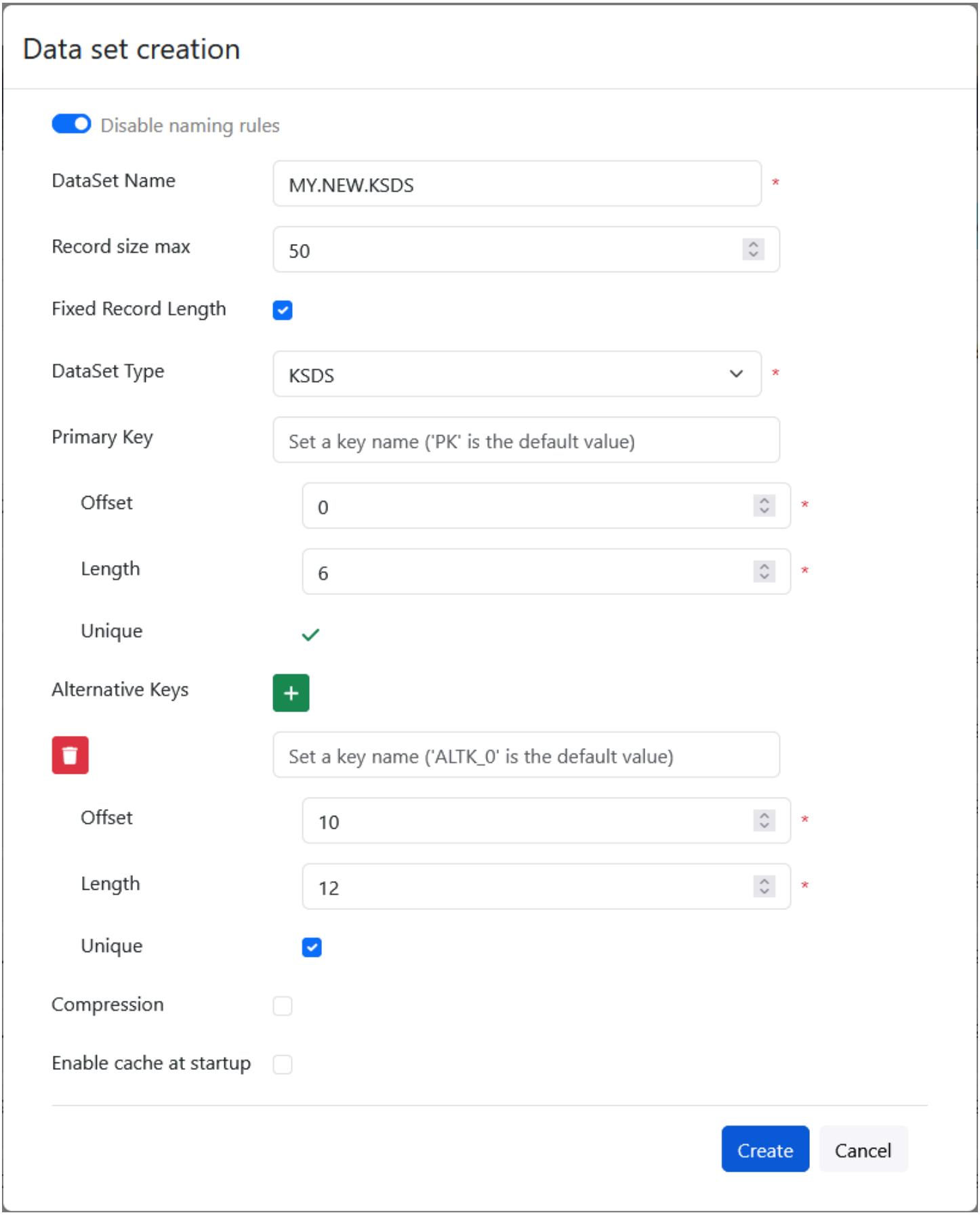
Para cada clave alternativa, debe proporcionar lo siguiente:

- Nombre: este campo es opcional. El valor predeterminado es **ALTK_#**, donde **#** representa un contador que se incrementa automáticamente y empieza en 0.
- Desplazamiento: el desplazamiento basado en 0 de la clave alternativa dentro del registro. Debe ser un entero positivo. Este campo es obligatorio.
- Longitud: la longitud de la clave alternativa. Esta longitud debe ser un número entero positivo. Este campo es obligatorio.
- Única: casilla de verificación para indicar si la clave alternativa aceptará entradas duplicadas. Si se selecciona, la clave alternativa se definirá como única (NO se aceptarán entradas de clave duplicadas). Este campo es obligatorio.

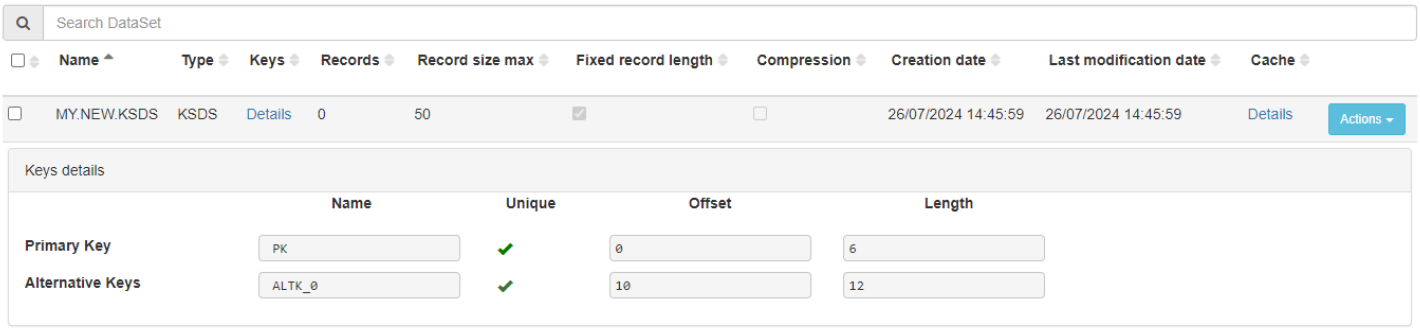
Para eliminar la definición de clave alternativa, utilice el botón de papelera situado a la izquierda.

- Compresión: una casilla de verificación para especificar si la compresión se utilizará para almacenar el conjunto de datos.
- Habilitar la memoria caché al inicio: casilla de verificación para especificar si el conjunto de datos debe cargarse en la memoria caché al iniciar la aplicación.

Tras especificar las definiciones de atributos, elija Crear para continuar:


Data set creation
Disable naming rules
DataSet Name
MY.NEW.KSDS
Record size max
50
Fixed Record Length
DataSet Type
KSDS
Primary Key
Set a key name ('PK' is the default value)
Offset
0
Length
6
Unique
Alternative Keys
Set a key name ('ALTK_0' is the default value)
Offset
10
Length
12
Unique
Compression
Enable cache at startup
Create
Cancel

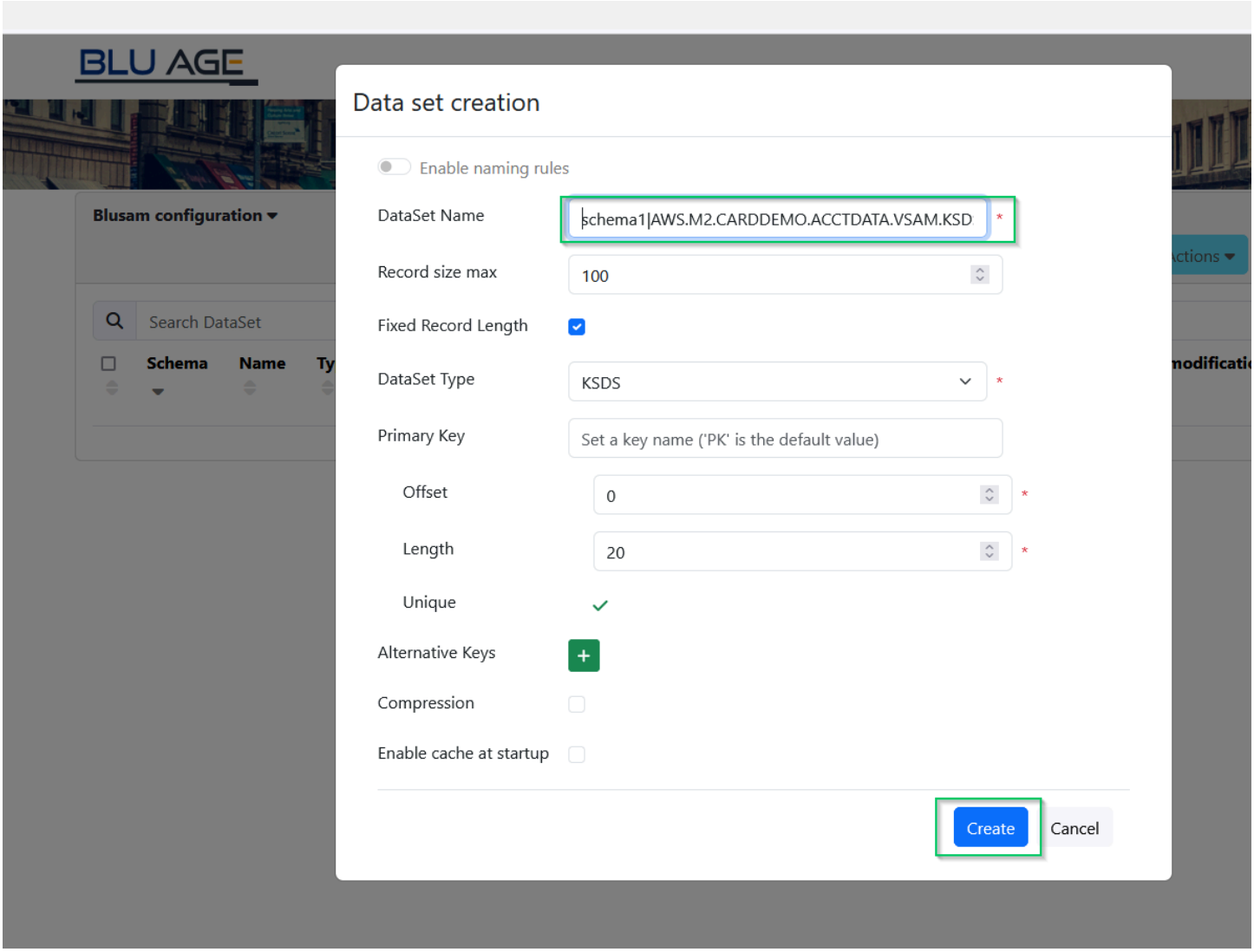
Se cerrará la ventana de creación y se mostrará la página de inicio que muestra la lista de conjuntos de datos. Puede ver los detalles del conjunto de datos recién creado.



Cree un conjunto de datos único en modo multiesquema

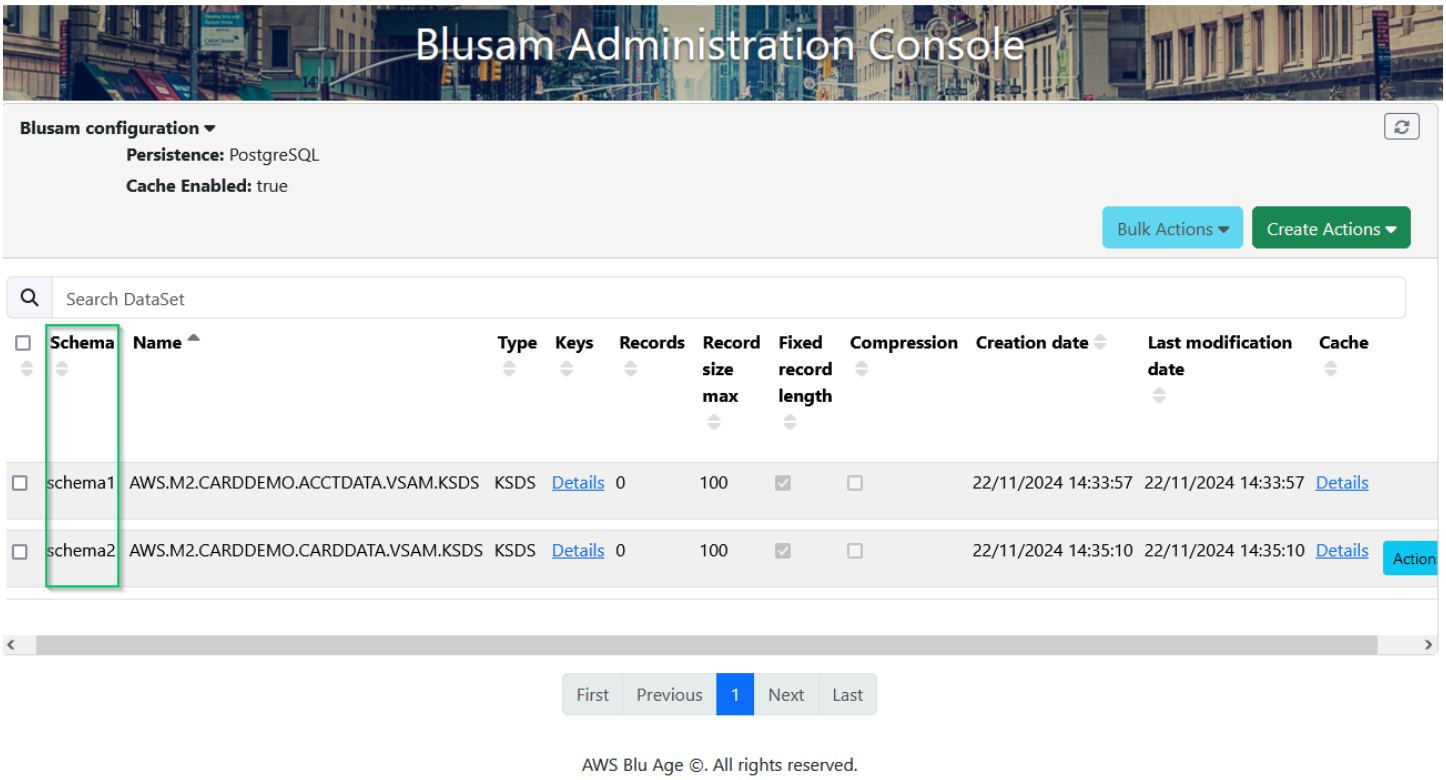
Se puede crear un conjunto de datos en un modo de varios esquemas anteponiendo el nombre del conjunto de datos con el nombre del esquema seguido de un símbolo vertical (|) (por ejemplo,). `schema1|AWS.M2.CARDDEMO.ACCTDATA.VSAM.KSDS`

> **Note**
>
> El esquema utilizado para crear el conjunto de datos debe especificarse en la `application-main.yml` configuración. Para obtener más información, consulte the section called “Propiedades de configuración de varios esquemas ”.



Si no se proporciona un prefijo de esquema, el conjunto de datos se creará en el esquema predeterminado especificado en la URL de la fuente de datos de Blusam en la configuración de Blusam Datasource. Si no se especifica ningún esquema en la URL de la fuente de datos de Blusam, se usa el esquema «público» de forma predeterminada.

> **Note**
>
> En el modo multiesquema, la consola BAC muestra la información del esquema del conjunto de datos en la primera columna.



Creación de conjuntos de datos a partir de LISTCAT

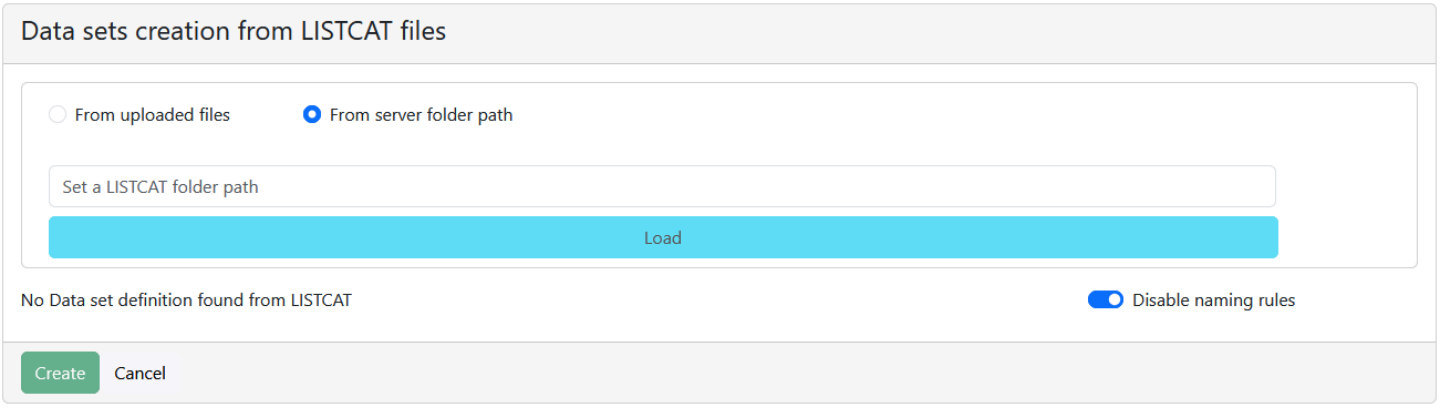
Esta función permite aprovechar los archivos JSON de LISTCAT creados durante el proceso de BluAge BluInsights transformación mediante Transformation Center como resultado del análisis de la exportación de LISTCAT desde las plataformas heredadas: las exportaciones de LISTCAT se analizan y se transforman en archivos JSON que contienen las definiciones de los conjuntos de datos (nombres, tipos de conjuntos de datos, definiciones de claves y si la longitud del registro es fija o variable).

Tener los archivos JSON de LISTCAT permite crear conjuntos de datos directamente sin tener que introducir manualmente toda la información requerida para los conjuntos de datos. También puede crear una colección de conjuntos de datos directamente en lugar de tener que crearlos uno a uno.

Si no hay ningún archivo JSON de LISTCAT disponible para su proyecto (por ejemplo, porque no había ningún archivo de exportación LISTCAT disponible en el momento de la transformación), siempre puede crear uno manualmente, siempre que utilice el formato JSON de LISTCAT que se detalla en el apéndice.

En la lista desplegable Crear acciones, elija Crear conjuntos de datos desde LISTCAT.

Aparecerá la siguiente página dedicada:



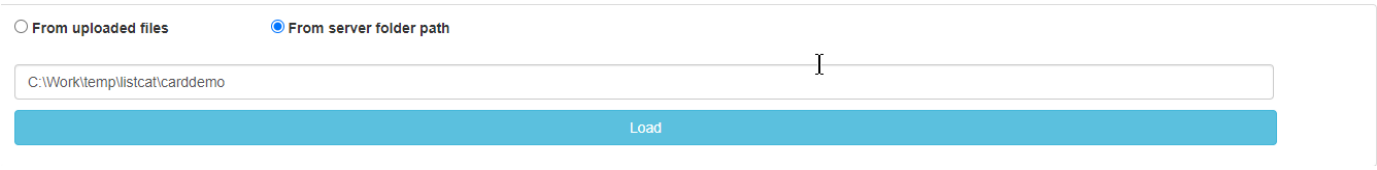
En este momento, el botón Cargar está deshabilitado, lo cual se espera.

Utilice los botones de opción para especificar cómo desea proporcionar los archivos JSON de LISTCAT. Dispone de dos opciones:

- Puede usar su navegador para cargar los archivos JSON.
- Puede seleccionar los archivos JSON desde una ubicación de carpeta en el servidor. Para elegir esta opción, primero debe copiar los archivos JSON en la ruta de carpeta determinada del servidor con los derechos de acceso adecuados.

Uso de archivos JSON en el servidor

1. Establezca la ruta de la carpeta en el servidor, apuntando a la carpeta que contiene los archivos JSON de LISTCAT:

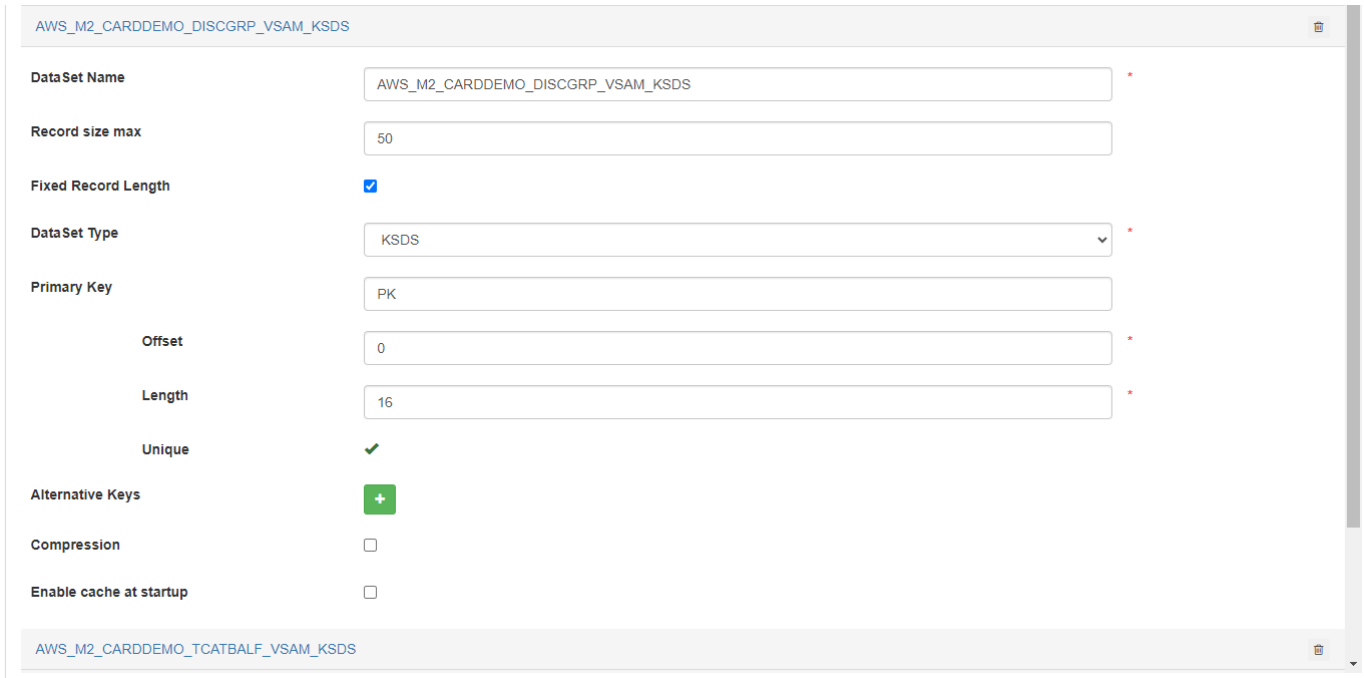


2. Elija el botón Cargar. Todas las definiciones de conjuntos de datos reconocidas se enumerarán en una tabla:

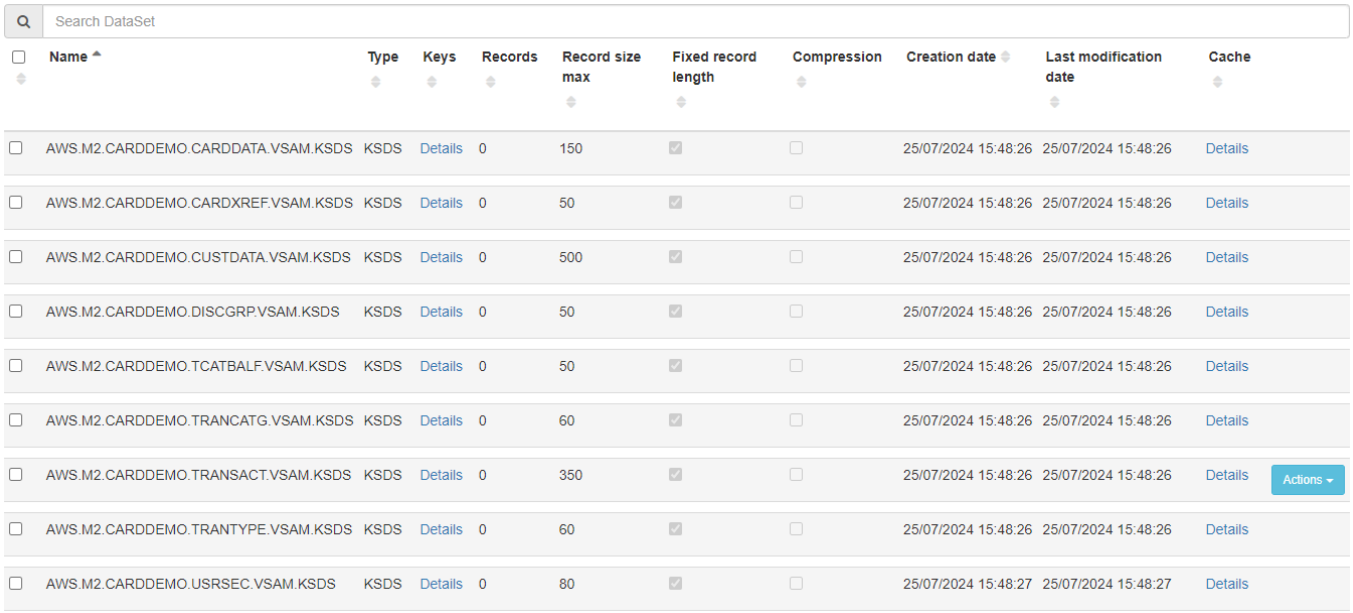
Cada fila representa una definición de conjunto de datos. Puede utilizar el botón de papelera para eliminar una definición de conjunto de datos de la lista.

> **Important**
>
> La eliminación de la lista es inmediata, sin ningún mensaje de advertencia.

3. El nombre de la izquierda es un enlace. Puede elegirlo para mostrar u ocultar los detalles de la definición del conjunto de datos, que es editable. Puede modificar la definición libremente, empezando en función del archivo JSON analizado.



4. Para crear todos los conjuntos de datos, elija Crear. Se crearán todos los conjuntos de datos y se mostrarán en la página de resultados de los conjuntos de datos. Todos los conjuntos de datos recién creados tendrán 0 registros.



Carga de archivos en el servidor

1. Esta opción es similar a usar los archivos de la ruta de la carpeta del servidor, pero en este caso primero debe cargar los archivos mediante el selector de archivos. Seleccione todos los archivos que desee cargar desde su máquina local y, a continuación, elija Cargar en el servidor.

2. Cuando la barra de progreso llegue al final, todos los archivos se han cargado correctamente en el servidor y el botón Cargar está habilitado. Pulse el botón Cargar y utilice las definiciones de conjuntos de datos descubiertas tal y como se ha explicado anteriormente.

Formato JSON de LISTCAT

El formato JSON de LISTCAT se define mediante los siguientes atributos:

- catalogId opcional: identificador del catálogo heredado como cadena o predeterminado para el catálogo predeterminado.
- identificador: nombre del conjunto de datos, como cadena.
- isIndexed: indicador booleano para indicar KSDS: verdadero para KSDS, falso en caso contrario.
- isLinear: indicador booleano para indicar ESDS: verdadero para ESDS, falso en caso contrario.
- isRelative: indicador booleano para indicar RRDS: verdadero para RRDS, falso en caso contrario
- Nota: isIndexed, isLinear e isRelative son mutuamente excluyentes.
- «isFixedLengthRegistro»: un indicador booleano: se establece en verdadero si una longitud fija registra el conjunto de datos, en falso en caso contrario.
- "avgRecordSize«: Tamaño medio del registro en bytes, expresado como un entero positivo.
- "maxRecordSize«: Tamaño máximo del registro en bytes, expresado como un número entero. Debe ser igual al tamaño avgRecordSize de registro de longitud fija.
- solo para KSDS: definición de clave principal obligatoria (como objeto anidado)
  - etiquetada primaryKey
  - offset: desplazamiento de bytes basado en 0 para la clave principal del registro.
  - length: longitud en bytes de la clave principal.
  - unique: se debe establecer en verdadero para la clave principal.
- para KSDS/ESDS, colección de claves alternativas (como colección de objetos anidados):
  - etiquetadas alternateKeys
  - Para cada clave alternativa:
    - offset: desplazamiento de bytes basado en 0 para la clave alternativa del registro.
    - length: longitud en bytes de la clave alternativa.
    - unique: debe establecerse en verdadero para la clave alternativa, si la clave no acepta entradas duplicadas; falso en caso contrario.
- si no hay claves alternativas presentes, proporcione una colección vacía:

```
alternateKeys: []
```

El siguiente es un archivo JSON KSDS LISTCAT de ejemplo.

```
{
  "catalogId": "default",
```

```
  "identifier": "AWS_M2_CARDDEMO_CARDXREF_VSAM_KSDS",
  "isIndexed": true,
  "isLinear": false,
  "isRelative": false,
  "isFixedLengthRecord": true,
  "avgRecordSize": 50,
  "maxRecordSize": 50,
  "primaryKey": {
    "offset": 0,
    "length": 16,
    "unique": true
  },
  "alternateKeys": [
    {
      "offset": 25,
      "length": 11,
      "unique": false
    }
  ]
}
```

Configurar la configuración de AWS Blu Age Runtime

El tiempo de ejecución de AWS Blu Age y el código del cliente son aplicaciones web que utilizan el marco Spring Boot. Aproveche las capacidades de Spring para proporcionar la configuración, con varias ubicaciones posibles y reglas de prioridad. También existen reglas de prioridad similares para el suministro de muchos otros archivos, como scripts groovy, sql, etc.

El AWS Blu Age Runtime también contiene aplicaciones web opcionales adicionales, que se pueden activar si es necesario.

Temas

- Principios básicos de la configuración de aplicaciones
- Prioridad de la aplicación
- JNDI para bases de datos
- AWS Secretos de Blu Age Runtime
- Otros archivos (groovy, sql, etc.)
- Aplicación web adicional

- Habilitar propiedades para AWS Blu Age Runtime
- Propiedades de caché de Redis disponibles en AWS Blu Age Runtime
- Configuración de la seguridad para las aplicaciones de Gapwalk

Principios básicos de la configuración de aplicaciones

La forma predeterminada de gestionar la configuración de la aplicación es mediante el uso de archivos YAML dedicados que se proporcionan en la carpeta `config` del servidor de aplicaciones. Hay dos archivos de configuración YAML principales:

- `application-main.yaml`
- `application-`*`profile`*`.yaml` (donde el valor *`profile`* se configura durante la generación de la aplicación).

El primer archivo configura el marco, es decir `Gapwalk-application.war`, mientras que el segundo es para opciones adicionales específicas para la aplicación cliente. Esto funciona con el uso de perfiles de Spring: la aplicación de Gapwalk usa el perfil `main`, mientras que la aplicación cliente usa el perfil *`profile`*.

A continuación se muestra un ejemplo de archivo YAML principal típico.

```
#########################################
##### JICS datasource configuration #####
#########################################
datasource:
  jicsDs:
    driver-class-name : org.postgresql.Driver
    url: jdbc:postgresql://localhost/jics
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource

#####################################################
##### Embedded Bluesam datasource configuration #####
#####################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

#########################################
##### Embedded Bluesam configuration #####
#########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql #pgsql, mssql, xodus...
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
```

A continuación se muestra un ejemplo de archivo YAML de cliente típico.

```
# Logback context logger integration.
logging.config : classpath:logback-[redacted].xml
# Limits Spring logger output.
logging.level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN
logging.level.org.springframework.statemachine : WARN
# If the datasource support mode is not static-xa, spring JTA transactions autoconfiguration must me disabled
spring.jta.enabled : false

spring:
  aws:
    client:
      datasources:
        names: primary
        primary:
          secret: arn:aws:secretsmanager:[redacted]

spring.jta.atomikos.datasource.primary.unique-resource-name: primary
spring.jta.atomikos.datasource.primary.xa-data-source-class-name: org.postgresql.xa.PGXADataSource
spring.jta.atomikos.datasource.primary.maxPoolSize: 20
spring.jta.atomikos.datasource.primary.autoCommit: false
```

Para obtener más información sobre el contenido de los archivos YAML, consulte Habilitar propiedades para AWS Blu Age Runtime.

Prioridad de la aplicación

Para estos archivos de configuración, se aplican las reglas de prioridad de Spring. En particular:

- El archivo YAML `application-main` aparece en el archivo war principal de Gapwalk con los valores predeterminados y el archivo de la carpeta `config` lo sustituye.
- Lo mismo debería hacerse con la configuración de la aplicación cliente
- Es posible que se transfieran parámetros adicionales a través de la línea de comandos en el momento del lanzamiento del servidor. Anularían los de YAML.

Para obtener más información, consulte la documentación oficial de Spring Boot.

JNDI para bases de datos

La configuración de la base de datos puede proporcionarse con JNDI en el archivo context.xml de Tomcat. Cualquier configuración de este tipo anularía la de YAML. Pero tenga en cuenta que, si usa eso, no podrá guardar sus credenciales en un administrador secreto (consultar a continuación).

El siguiente ejemplo muestra ejemplos de configuraciones para JICS y bases de datos. BluSam

```
<Resource auth="Container" driverClassName="org.postgresql.Driver" initialSize="0"
 maxIdle="5"
   maxOpenPreparedStatements="-1" maxTotal="10" maxWaitMillis="-1" name="jdbc/jics"
   poolPreparedStatements="true" testOnBorrow="false" type="javax.sql.DataSource"
   url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX"
 password="XXXX" />
```

jdbc/jics

Sería `jdbc/jics` para la base de datos de JICS y `jdbc/bluesam` (preste atención a la e) para la base de datos de Blusam.

url="jdbc:postgresql://XXXX.rds.amazonaws.com:5432/XXXX" username="XXXX" password="XXXX"

URL, nombre de usuario y contraseña de la base de datos.

AWS Secretos de Blu Age Runtime

Algunas de las configuraciones de recursos que contienen credenciales se pueden proteger aún más mediante el uso de secretos de AWS . La idea es almacenar los datos críticos en un lugar AWS

secreto y tener una referencia al secreto en la configuración de YAML para que el contenido secreto se recoja sobre la marcha al iniciar Apache Tomcat.

Secretos de Aurora

La configuración de la base de datos Aurora (para JICS, Blusam, base de datos de cliente, etc.) utilizará el secreto de base de datos integrado, que rellenará automáticamente todos los campos relevantes de la base de datos correspondiente.

> **Note**
>
> La clave `dbname` es opcional y, según la configuración de la base de datos, se incluirá en el secreto o no. Puede agregarla allí manualmente o proporcionando el nombre del archivo YAML.

Otros secretos

Hay otros secretos para los recursos que tienen una sola contraseña (en particular, las cachés de Redis protegidas con contraseña). En este caso, se debe usar el otro tipo de secreto con una sola clave.

Referencias en YAML a secretos

El archivo `application-main.yml` puede hacer referencia al ARN secreto para varios recursos.

Base de datos de JICS

Credenciales de la base de datos de JICS con `spring.aws.jics.db.secret`

```
spring:
   aws:
     jics:
       db:
         dbname: jics
         secret: arn:aws:secretsmanager:XXXX
```

Claves secretas de bases de datos de JICS compatibles:

Clave secreta	Descripción de la clave secreta
host	El nombre del host
puerto	El puerto
dbname	El nombre de la base de datos
nombre de usuario	El nombre de usuario
contraseña	La contraseña
engine	Motor de base de datos: Postgres, Oracle, Db2, Microsoft SQL Server
currentSchema	Esquema específico que se utilizará (solo compatible con Db2)
sslConnection	Si se usará la conexión SSL (solo compatible con Db2)
sslTrustStoreLocation	La ubicación del almacén de confianza en el cliente (solo compatible con Db2)
sslTrustStoreContraseña	La contraseña del almacén de confianza del cliente (solo compatible con Db2)

> **Note**
>
> El nombre de la base de datos se proporciona en el secreto o en la referencia de YAML `spring.aws.jics.db.dbname`.

Base de datos de Blusam

Las credenciales de la base de datos de Blusam con `spring.aws.client.bluesam.db.secret`

```
spring:
   aws:
```

```
    client:
      bluesam:
        db:
          dbname: bluesam
          secret: arn:aws:secretsmanager:XXXX
```

Claves secretas de la base de datos Blusam admitidas:

Clave secreta	Descripción de la clave secreta
host	El nombre del host
puerto	El puerto
dbname	El nombre de la base de datos
nombre de usuario	El nombre de usuario
contraseña	La contraseña
engine	Motor de base de datos: Postgres

> **Note**
>
> El nombre de la base de datos se proporciona en el secreto o en la referencia de YAML `spring.aws.client.bluesam.db.dbname`.

Base de datos de clientes

El `application-profile.yml` del cliente puede hacer referencia al ARN secreto de la base de datos del cliente. Esto requiere una propiedad adicional para enumerar los nombres de orígenes de datos `spring.aws.client.datasources.names`. Para cada nombre de origen de datos `ds_name`, especifique el ARN secreto en la siguiente propiedad: `spring.aws.client.datasources.ds_name.secret`. Ejemplo:

```
spring:
   aws:
     client:
       datasources:
```

```
        names: primary,host
        primary:
          secret: arn:aws:secretsmanager:XXXX
        host:
          dbname: hostdb
          secret: arn:aws:secretsmanager:XXXX
```

names: primary,host:

Un ejemplo con dos orígenes de datos de clientes denominados principal y host, cada uno con su base de datos y sus credenciales.

dbname: hostdb:

En este ejemplo, el nombre de la base de datos “host” no está en el secreto, sino que se proporciona aquí, mientras que la base de datos “primary” sí está en el secreto.

Claves secretas de bases de datos de cliente admitidos:

Clave secreta	Descripción de la clave secreta
host	El nombre del host
puerto	El puerto
dbname	El nombre de la base de datos
nombre de usuario	El nombre de usuario
contraseña	La contraseña
engine	Motor de base de datos: Postgres, Oracle, Db2, Microsoft SQL Server
currentSchema	Esquema específico que se utilizará (solo compatible con Db2)
sslConnection	Si se usará la conexión SSL (solo compatible con Db2)
sslTrustStoreLocation	La ubicación del almacén de confianza en el cliente (solo compatible con Db2)

Clave secreta	Descripción de la clave secreta
sslTrustStoreContraseña	La contraseña del almacén de confianza del cliente (solo compatible con Db2)

Base de datos de utilidades PGM

El archivo `application-utility-pgm.yml` puede hacer referencia al ARN secreto para obtener varios recursos.

- `spring.aws.client.datasources.primary`
  - `secret`

    ARN secreto para la base de datos de la aplicación.

Tipo: cadena

- `type`

Nombre completo de la implementación del grupo de conexiones que se usará.

Tipo: cadena

Valor predeterminado: `com.zaxxer.hikari.HikariDataSource`

- `spring.aws.client.utility.pgm.datasources`
  - `names`

Lista de nombres de orígenes de datos

Tipo: cadena

- `dsname`
  - `dbname`

Nombre del host

Tipo: cadena

- `secret`

ARN secreto de la base de datos de host

Tipo: cadena

- `type`

Nombre completo de la implementación del grupo de conexiones que se usará.

Tipo: cadena

Valor predeterminado: `com.zaxxer.hikari.HikariDataSource`

Para un secreto de varios orígenes de datos:

```
spring:
   aws:
     client:
       primary:
         secret: arn:aws:secretsmanager:XXXX
         type: dataSourceType
       utility:
         pgm:
           datasources:
             names: dsname1,dsname2,dsname3
               dsname1:
                 dbname: dbname1
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
               dsname2:
                 dbname: dbname2
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
               dsname3:
                 dbname: dbname3
                 secret: arn:aws:secretsmanager:XXXX
                 type: dataSourceType
```

No hay claves secretas compatibles con XA

- motor (postgres/oracle/db2/mssql)

- puerto
- dbname
- currentSchema
- nombre de usuario
- contraseña
- url
- sslConnection
- sslTrustStoreLocation
- sslTrustStoreContraseña

Para `postgres`, solo el valor de la clave secreta `sslMode` (`disable/allow/prefer/require/verify-ca/verify-full`) y la propiedad YAML `spring.aws.rds.ssl.cert-path`, es posible conectarse con SSL.

Claves secretas compatibles con XA

Si la base de datos del cliente utiliza XA, las subpropiedades xa se admiten mediante valores secretos.

- host
- puerto
- dbname
- currentSchema
- nombre de usuario
- contraseña
- url
- sslConnection (verdadero/falso)
- sslTrustStoreLocation
- sslTrustStoreContraseña

Sin embargo, para otras propiedades xa (por ejemplo, `maxPoolSize` o`driverType`), se debe seguir proporcionando la clave YAML normal `spring.jta.atomikos.datasource.XXXX.unique-resource-name`.

El valor secreto anula las propiedades YAML.

BAC y JAC de superadministrador predeterminado

También puede configurar application-main.yml para recuperar el nombre de usuario y la contraseña del usuario superadministrador predeterminado en el secreto de AWS Secrets Manager especificando el ARN. En el siguiente ejemplo, se muestra cómo declarar este secreto en un archivo YAML.

```
spring:
   aws:
     client:
       defaultSuperAdmin:
         secret: arn:aws:secretsmanager:XXXX
```

Claves secretas de bases de datos de superadministrador predeterminadas admitidas:

Clave secreta	Descripción de la clave secreta
nombre de usuario	El nombre de usuario.
contraseña	La contraseña.

OAuth2

También puedes configurar `application-main.yml` para recuperar el secreto del OAuth2 cliente especificando el proveedor y el ARN. AWS Secrets Manager El valor predeterminado de la propiedad del proveedor es Amazon Cognito. El siguiente es un ejemplo de configuración para el proveedor Keycloak: OAuth2

```
spring:
   aws:
     client:
       provider: keycloak
       keycloak:
         secret: arn:aws:secretsmanager:XXXX
```

En este ejemplo, el secreto del cliente del OAuth2 proveedor Keycloak se recupera del ARN especificado en AWS Secrets Manager. Esta configuración admite varios proveedores al resolver dinámicamente el nombre del proveedor y el ARN secreto correspondiente.

Claves secretas compatibles: OAuth2

Clave secreta	Descripción de la clave secreta
client-secret	El secreto generado por el servidor de autorización durante el proceso de registro de la aplicación.

Administrador de secretos de las cachés de Redis

El archivo `application-main.yml` puede hacer referencia al ARN secreto de las cachés de Redis. Las admitidas son:

- Credenciales de Redis de Gapwalk con `spring.aws.client.gapwalk.redis.secret`
- Credenciales de Redis de Blusam con `spring.aws.client.bluesam.redis.secret`
- Credenciales de Redis de bloqueos de Blusam con `spring.aws.client.bluesam.locks.redis.secret`
- Credenciales de Redis de catálogo de conjuntos de datos con `spring.aws.client.dataset.catalog.redis.secret`
- Credenciales Redis de JICS con `spring.aws.client.jics.redis.secret`
- Credenciales de Redis de sesiones con `spring.aws.client.jics.redis.secret`
- Credenciales de Redis del rastreador de sesiones con `spring.aws.client.session.tracker.redis.secret`
- Credenciales de Redis de colas TS de JICS con `spring.aws.client.jics.queues.ts.redis.secret`
- Credenciales de Redis de puntos de comprobación de JCL con `spring.aws.client.jcl.checkpoint.redis.secret`
- Credenciales de Redis de bloqueos de archivos con `spring.aws.client.gapwalk.files.locks.redis.secret`
- Credenciales de Redis de bloqueos de Blu4IV con `spring.aws.client.blu4iv.locks.redis.secret`

En el siguiente ejemplo, se muestra cómo declarar estos secretos en un archivo YAML.

```
spring:
   aws:
     client:
       gapwalk:
         redis:
           secret: arn:aws:secretsmanager:XXXX
       bluesam:
         locks:
           redis:
             secret: arn:aws:secretsmanager:XXXX
         redis:
           secret: arn:aws:secretsmanager:XXXX
       dataset:
         catalog:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       jics:
         redis:
           secret: arn:aws:secretsmanager:XXXX
       session:
         tracker:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       jics:
         queues:
           ts:
             redis:
               secret: arn:aws:secretsmanager:XXXX
       jcl:
         checkpoint:
           redis:
             secret: arn:aws:secretsmanager:XXXX
       gapwalk:
         files:
           locks:
             redis:
               secret: arn:aws:secretsmanager:XXXX
       blu4iv:
         locks:
           redis:
             secret: arn:aws:secretsmanager:XXXX
```

Claves secretas de Redis admitidas:

Clave secreta	Descripción de la clave secreta
hostname	El nombre de host del servidor de Redis.
puerto	El puerto del servidor de Redis.
nombre de usuario	El nombre de usuario.
contraseña	La contraseña.

Administrador de secretos para la configuración de contraseñas SSL

El archivo `application-main.yml` puede hacer referencia al ARN del secreto para la configuración de la contraseña SSL. Se admite lo siguiente:

- Credenciales de SSL de Gapwalk con `spring.aws.client.ssl.secret`

En el siguiente ejemplo, se muestra cómo declarar estos secretos en un archivo YAML.

```
spring:
   aws:
     client:
       ssl:
         secret: arn:aws:secretsmanager:XXXX
```

Clave secreta	Descripción de la clave secreta
trustStorePassword	La contraseña del almacén de confianza
keyStorePassword	La contraseña del almacén de claves

Administrador de secretos para la configuración de contraseñas de IBM MQ

El `application-main.yml` archivo puede hacer referencia al ARN secreto de la configuración de IBM MQ. Se admite lo siguiente:

- Las conexiones de IBM MQ se definen como una lista, al igual que las credenciales:

`mq.queues.jmsMQQueueManagers[N].secret:`

N empieza en 0 para la primera conexión.

En el siguiente ejemplo, se muestra cómo declarar estos secretos en un archivo YAML.

```
mq.queues.jmsMQQueueManagers[0].secret: Secret-0-ARN
mq.queues.jmsMQQueueManagers[1].secret: Secret-1-ARN
```

Para obtener información sobre el secreto ARNs, consulta ¿Qué hay en un secreto de Secrets Manager?

Las propiedades definidas en el secreto anularán sus valores correspondientes en la configuración de `jmsMQ` YAML.

Si `queueManager` se establece en el secreto, anulará el `mq.queues.jmsMQQueueManagers[N].jmsMQQueueManager` valor del archivo YAML.

Clave secreta	Descripción de la clave secreta
Gestor de colas	El nombre del administrador de colas MQ de IBM.
appName	El nombre de la aplicación IBM MQ.
channel	El nombre del canal MQ de IBM.
host	El nombre de host de IBM MQ.
puerto	El puerto MQ de IBM.
userId	El nombre de usuario de IBM MQ.
contraseña	La contraseña de usuario de IBM MQ.
maxPoolSize	El tamaño máximo del pool MQ de IBM.
sslCipherKey	La suite de cifrado SSL MQ de IBM.

Otros archivos (groovy, sql, etc.)

Los demás archivos utilizados por el proyecto del cliente utilizan reglas de prioridad similares a las de la configuración de Spring. Ejemplos:

- Los scripts groovy son los archivos `.groovy` de la carpeta o las subcarpetas `scripts`.
- Los scripts SQL son los archivos `.sql` de la carpeta o las subcarpetas `sql`.
- Los scripts daemon son los archivos `.groovy` de la carpeta o las subcarpetas `daemons`.
- Los archivos de asignación de base de datos de consultas se llaman `queries-database.mapping` y se encuentran en las subcarpetas de la carpeta `sql`.
- Las plantillas de Jasper son los archivos `.jrxml` de la carpeta o las subcarpetas `templates`.
- Los catálogos de conjuntos de datos son los archivos `.json` de la carpeta `catalog`.
- Los archivos Lnk son los archivos `.json` de la carpeta `lnk`.

Todas estas ubicaciones se pueden sobrescribir mediante una propiedad del sistema o una propiedad YAML de cliente.

- Para los scripts de Groovy: `configuration.scripts`
- Para los scripts de SQL: `configuration.sql`
- Para los scripts de Daemon: `configuration.daemons`
- Para el archivo de asignación de bases de datos de consultas: `configuration.databaseMapping`
- Para las plantillas de Jasper: `configuration.templates`
- Para los catálogos de conjuntos de datos: `configuration.catalog`
- Para los archivos Lnk: `configuration.lnk`

Si no se encuentra la propiedad, los archivos se tomarán de la ubicación predeterminada mencionada anteriormente. La búsqueda se realizará primero con la carpeta de trabajo de Tomcat como raíz y, por último, en el archivo war de la aplicación.

Aplicación web adicional

El motor de ejecución de AWS Blu Age contiene aplicaciones web adicionales en su `webapps-extra` carpeta. El servidor Tomcat no sirve estas aplicaciones de forma predeterminada.

La aceptación de estas aplicaciones web depende del proyecto de modernización y se realiza moviendo el archivo war deseado de la carpeta `webapps-extra` a la carpeta `webapps`. Después de eso, el servidor Tomcat servirá el archivo war la próxima vez que se inicie.

También se puede añadir alguna configuración adicional específica del proyecto en un archivo de configuración YAML para cada archivo war adicional, tal y como se ha hecho en el archivo `application-main.yml` y se ha explicado anteriormente. Los archivos war adicionales son:

- `gapwalk-utility-pgm.war`: contiene soporte para los programas de utilidades de ZOS y utiliza `application-utility-pgm.yaml` como configuración.
- `gapwalk-cl-command.war`: contiene soporte para los programas de utilidades de AS/400 y utiliza `application-cl-command.yaml` como configuración.
- `gapwalk-hierarchical-support.war`: contiene soporte para transacciones IMS/MFS y utiliza `application-jhdb.yaml` como configuración.

Habilitar propiedades para AWS Blu Age Runtime

En las aplicaciones Spring Boot, `application-main.yml` es el archivo de configuración en el que definimos diferentes tipos de propiedades, como el puerto de agente de escucha, la conectividad de la base de datos y muchos más. Puede usar esta página para obtener información sobre las propiedades disponibles para el tiempo de ejecución de AWS Blu Age y cómo habilitarlas.

Temas

- Notación YML
- Inicio rápido y casos de uso
- Propiedades disponibles para la aplicación principal
- Propiedades disponibles para las aplicaciones web opcionales
- Propiedades disponibles para la aplicación cliente

Notación YML

En la siguiente documentación, una propiedad como `parent.child1.child2=true` se escribe de la siguiente manera en formato YAML.

```
parent:
 child1:
```

```
    child2: true
```

Inicio rápido y casos de uso

Los siguientes casos de uso muestran ejemplos de las claves y valores aplicables.

- Archivo application-main.yml predeterminado

```
----
#### DEFAULT APPLICATION-MAIN.YML FILE      #####
#### SHOWING USEFUL CONFIGURATION ELEMENTS #####
#### SHOULD BE OVERRIDDEN AND EXTERNALIZED   #####

###################################
##### Logging configuration #####
###################################
logging:
  config: classpath:logback-main.xml
  level.org.springframework.beans.factory.support.DefaultListableBeanFactory : WARN

###################################
##### Spring configuration #####
###################################
spring:
  quartz:
    auto-startup: false
    scheduler-name: Default
    properties:
      org.quartz.threadPool.threadCount: 1
  jta:
    enabled: false
    atomikos.properties.maxTimeout : 600000
    atomikos.properties.default-jta-timeout : 100000
  jpa:
# DISABLE OpenEntityManagerInViewInterceptor
    open-in-view: false
    # Fix Postgres JPA Error:
    # Method org.postgresql.jdbc.PgConnection.createClob() is not yet implemented.
    properties.hibernate.temp.use_jdbc_metadata_defaults : false
#####################################
##### Jics tables configuration #####
#####################################
```

```
    # The dialect should match the jics datasource choice
    database-platform : org.hibernate.dialect.PostgreSQLDialect #
 org.hibernate.dialect.PostgreSQLDialect, org.hibernate.dialect.SQLServerDialect

    # those properties can be used to create and initialize jics tables
 automatically.
#    properties:
#      hibernate:
#      globally_quoted_identifiers: true
#        hbm2ddl:
#           import_files_sql_extractor :
 org.hibernate.tool.hbm2ddl.MultipleLinesSqlCommandExtractor
#           import_files : file:./setup/initJics.sql
#           auto : create

##########################
###### Level 2 cache #####
##########################
#        cache:
#          use_second_level_cache: true
#          use_query_cache: true
#          region:
#            factory_class: org.hibernate.cache.ehcache.EhCacheRegionFactory
#      javax:
#        persistence:
#          sharedCache:
#            mode: ENABLE_SELECTIVE
##########################
###### Redis settings #####
##########################
  session:
    store-type: none #redis

# Secret manager configuration for global Redis cache
    aws:
      client:
        gapwalk:
          redis:
            secret: arn:aws:secretsmanager:XXXX

########################################
##### JICS datasource configuration #####
########################################
datasource:
```

```
  jicsDs:
    driver-class-name : org.postgresql.Driver # org.postgresql.Driver,
 com.microsoft.sqlserver.jdbc.SQLServerDriver
    url: jdbc:postgresql://localhost/jics # jdbc:postgresql://localhost:5433/jics,
 jdbc:sqlserver://localhost\SQLEXPRESS:1434;databasename=jics;
    username: jics
    password: jics
    type : org.postgresql.ds.PGSimpleDataSource #
 org.postgresql.ds.PGSimpleDataSource,
 com.microsoft.sqlserver.jdbc.SQLServerDataSource

########################################################
##### Embedded Bluesam datasource configuration #####
########################################################
  bluesamDs :
    driver-class-name : org.postgresql.Driver
    url : jdbc:postgresql://localhost/bluesam
    username : bluesam
    password : bluesam
    type : org.postgresql.ds.PGSimpleDataSource

#########################################
##### Embedded Bluesam configuration #####
#########################################
bluesam :
  remote : false
  cache : ehcache
  persistence : pgsql
  ehcache:
    resource-pool:
      size: 4GB
  write-behind:
    enabled: true
  pgsql :
    dataSource : bluesamDs

########################
##### Jics settings #####
########################
rabbitmq.host: localhost
jics:
  cache: false #redis
  resource-definitions.store-type: jpa # default value: jpa, other possible value:
 redis
```

```
jics.disableSyncpoint : false
#jics.initList:
#jics.parameters.datform: DDMMYY
#jics.parameters.applid: VELOCITY
#jics.parameters.sysid: CICS
#jics.parameters.eibtrmid: TERM
#jics.parameters.userid: MYUSERID
#jics.parameters.username: MYUSERNAME
#jics.parameters.opid: XXX
#jics.parameters.cwa.length: 0
#jics.parameters.netname: MYNETNAME
#jics.parameters.jobname: MJOBNAME
#jics.parameters.sysname: SYSNAME

################################################
##### Jics RunUnitLauncher pool settings #####
################################################
#jics.runUnitLauncherPool.enable: false
#jics.runUnitLauncherPool.size: 20
#jics.runUnitLauncherPool.validationInterval: 1000

#########################
##### Jhdb settings #####
#########################
#jhdb.lterm: LTERMVAL
#jhdb.identificationCardData: SomeIDData

###################################
##### DateHelper configuration ####
###################################
#forcedDate: "2013-08-26T12:59:58+01:57"

#############################
##### Sort configuration ####
#############################
#externalSort.threshold: 256MB

###################################
##### Server timeout (10 min)  ####
###################################
spring.mvc.async.request-timeout: 600000

#############################
```

```
##### DATABASE STATISTICS  ####
################################
databaseStatistics : false

######################
##### CALLS GRAPH ####
######################
callGraph : false

####################################
#####     SSL configuration    #####
####################################
gapwalk.ssl.enabled : true
gapwalk.ssl.trustStore : "./config/clientkey.jks"
gapwalk.ssl.trustStorePassword : mysslcertifpassword

##################################
##### MQ settings #####
##################################
mq.queues: jmsmq
mq.queues.jmsMQQueueManagers[0].jmsMQQueueManager: QM1
mq.queues.jmsMQQueueManagers[0].jmsMQAppName: Gapwalk
mq.queues.jmsMQQueueManagers[0].jmsMQChannel: DEV.APP.SVRCONN
mq.queues.jmsMQQueueManagers[0].jmsMQHost: localhost
mq.queues.jmsMQQueueManagers[0].jmsMQPort: 1415
mq.queues.jmsMQQueueManagers[0].jmsMQUserid: app
mq.queues.jmsMQQueueManagers[0].jmsMQSSLCipher: "*TLS12ORHIGHER"
mq.queues.jmsMQQueueManagers[1].jmsMQQueueManager: QM2
mq.queues.jmsMQQueueManagers[1].jmsMQAppName: Gapwalk
mq.queues.jmsMQQueueManagers[1].jmsMQChannel: DEV.APP.SVRCONN
mq.queues.jmsMQQueueManagers[1].jmsMQHost: localhost
mq.queues.jmsMQQueueManagers[1].jmsMQPort: 1415
mq.queues.jmsMQQueueManagers[1].jmsMQUserid: app

################################
##### SQL SHIFT CODE POINT ####
################################
# Code point 384 match unicode character \u0180
sqlCodePointShift : 384

################################
##### LOCK TIMEOUT RECORD  ####
################################
# Blu4IV record lock timeout
```

```
lockTimeout : 100

##############################
##### REPORTS OUTPUT PATH ####
##############################
reportOutputPath: reports

##############################
##### TASK EXECUTOR       ####
##############################
taskExecutor:
  corePoolSize: 5
  maxPoolSize: 10
  queueCapacity: 50
  allowCoreThreadTimeOut: false

##############################
##### PROGRAM NOT FOUND   ####
##############################
stopExecutionWhenProgNotFound: false

########################################################
#####  DISP DEFAULT VALUE (to be removed one day) ####
########################################################
defaultKeepExistingFiles: true

########################################################
#####  BLOCKSIZE DEFAULT VALUE  ####
########################################################
#blockSizeDefault: 32760

####################################
#####  JOBQUEUE CONFIGURATION    ####
####################################
jobqueue:
  api.enabled: false
  impl: none # possible values: quartz, none
  schedulers:  # list of schedulers
    -
      name: queue1
      threadCount: 5
    -
      name: queue2
      threadCount: 5
```

```
#############################################################################
##### QUERY BUILDING                                                      ##
# useConcatCondition : false by default
# if true, in the query, the where condition is build with key concatenation  ##
#############################################################################
# query.useConcatCondition: true

########################################
#####  JCL Batch Restart Mechanism  ####
########################################
jcl:
checkpoint:
enabled: false
#expireTimeout: -1
#expireTimeoutUnit: SECONDS # Supported values: java.util.concurrent.TimeUnit
#provider: redis

----
```

- Utilice archivos de longitud variable con los comandos LISTCAT

```
[**/*.*]
encoding=IBM930
reencoding=false

[global]
listcat.variablelengthpreprocessor.enabled=true
listcat.variablelengthpreprocessor.type=rdw
# use "rdw" if your .listcat file contains a set of records (RDW)
# use "bdw" if your .listcat file contains a set of blocks (bdw)
```

- Proporcione un valor indicador de bytes null en la utilidad LOAD/UNLOAD

```
# Unload properties
# For date/time: if use database configuration is enabled, formats are ignored
# For nbi; use hexadecimal syntax to specify the byte value
# - When the value is null in database : the value dumped to the file is filled by
 low value characters and the NBI is
```

```
# equal to the byte 6F (the ? character)
# - When the value is not null in database and the column is nullable: the NBI is
 equal to the byte 00 (low value) and NOT
# equal to the byte 40 (space)
unload:
  sqlCodePointShift: 0
  nbi:
    whenNull: "6F"
    whenNotNull: "00"
  useDatabaseConfiguration: false
  format:
    date: MM/dd/yyyy
    time: HH.mm.ss
    timestamp: yyyy-MM-dd-HH.mm.ss.SSSSSS
```

Propiedades disponibles para la aplicación principal

Esta tabla proporciona una vista exhaustiva de los parámetros clave/valores.

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`logging.config`	Ruta	classpath:logback-main.xml	Clave estándar para la referencia al archivo de configuración del logback. También hay disponibles otras claves de registro estándar.	
`spring.jta.enabled`	valor booleano	false	Clave estándar. Si el modo de soporte del origen de datos no es static-xa, la configura	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
			ción automática de las transacciones JTA de Spring debe estar deshabilitada.	
`datasource.jicsDs` + `-driver-class-name` + `-url` + `-username` + `-password` + `-type`	Origen de datos estándar de Spring con subclaves		Contiene la información de conexión de la base de datos de Jics. Como alternativa, es muy recomendable el uso de secretos de AWS , como se explica en the section called “Base de datos de JICS”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`datasource.bluesamDs + -driver-class-name + -url + -username + -password + -type`	Origen de datos estándar de Spring con subclaves		Contiene la información de conexión de la base de datos de Blusam. Como alternativa, es muy recomendable el uso de secretos de AWS , como se explica en the section called “Base de datos de Blusam”.	
`bluesam.disabled`	valor booleano	false	Si se debe deshabilitar completamente Blusam.	
`bluesam.cache`	cadena		Si no se establece, no se utilizará la caché de Blusam. Los valores posibles (implementaciones de caché) son cache y redis (the section called “Propiedades de caché de Redis”).	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`bluesam.maxBluesamDisablingThreadpoolSize`	número	10	Especifica el tamaño máximo del grupo de subprocesos utilizado para deshabilitar los conjuntos de datos de bluesam para el procesamiento por lotes.	4.5.0
`bluesam.bluesamStatusPollingInterval`	número	1 000	Especifica el tiempo (en milisegundos) que se debe esperar entre cada iteración al sondear el estado de bluesam para comprobar las actividades en línea.	4.5.0
`bluesam.maxBluesamStatusPollingRetry`	número	3	Especifica el número máximo de reintentos cuando se produce un error al sondear el estado de imagen azul.	4.5.0

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`bluesam.checkBluesamStatus`	valor booleano	false	Especifica si se debe comprobar o no el estado del conjunto de datos de bluesam antes de acceder a él.	4.5.0
`spring.aws.client.bluesam.redis.secret`	cadena	null	Especifica el ARN secreto de credencial para la caché de Blusam Redis; consulte. the section called “AWS Secretos de Blu Age Runtime”	
`spring.aws.client.bluesam.locks.redis.secret`	cadena	null	Especifica el ARN secreto de credencia les para caché de Redis de bloqueos de Blusam; consulte the section called “AWS Secretos de Blu Age Runtime”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`forcedDate`	cadena		Fuerza la fecha a la fecha proporcionada, si la hay.	
`frozenDate`	booleano	true	Especifica si se debe congelar la fecha. Solo se aplica si también `forcedDate` está establecida.	
`externalSort.threshold`	tamaño de datos (ejemplo: 12 MB)		El umbral de clasificación: cuándo cambiar a una ordenación externa (fusión).	
`blockSizeDefault`	número	32760	El tamaño de bloque predeterminado que se utilizará para los bytes de BDW.	
`jics.parameters.datform`	cadena	MMDDYY	El formato de fecha.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.initList`	cadena		La lista de inicialización de JICS, separada por comas. Si está presente, define los nombres de las listas separados por comas para activarlas al iniciar Apache Tomcat entre las listas de CICS. Ejemplo de valor: `$UUU,DFH$IVPL,PEZ1` Esto se extenderá en cascada a los grupos contenidos en esas listas y a sus definiciones de recursos subyacentes, que luego estarán visibles en el entorno de tiempo de ejecución. Vacío de forma predeterminada.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.parameters.applid`	cadena	VELOCITY	El valor de applied to para identificar la aplicación en JICS (4 caracteres como mínimo, sin longitud máxima).	
`jics.parameters.sysid`	cadena	CICS	La identificación del sistema (SYSID).	
`jics.parameters.eibtrmid`	cadena	TERM	El identificador del terminal (4 caracteres como máximo, 1 como mínimo).	
`jics.parameters.userid`	cadena		El identificador de usuario (8 caracteres como máximo, sin mínimo). Si no se proporciona ningún valor (en blanco de forma predeterminada), el identificador de sesión HTTP se utiliza como identificador de usuario.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.parameters.username`	cadena	MYUSERNAME	El nombre de usuario (10 caracteres como máximo, 1 como mínimo).	
`jics.parameters.netname`	cadena	MYNETNAME	El nombre de la red (8 caracteres como máximo, 1 como mínimo).	
`jics.parameters.opid`	cadena	XXX	La identificación del operador de 3 caracteres.	
`jics.parameters.jobname`	cadena	MJOBNAME	El nombre del trabajo.	
`jics.parameters.sysname`	cadena	SYSNAME	El nombre del sistema AS4 00 (sysname).	
`jics.parameters.cwa.length`	número	0	La longitud del área de trabajo común (CWA).	
`jics.parameters.charset`	cadena	CP037	Juego de caracteres utilizado globalmente por JICS.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.parameters.tsqimpl`	cadena	bluesam	Implementación de la cola de almacenamiento temporal (TSQ) de JICS (los valores permitidos son/`bluesam` /`memory` /`redis`)	
`jics.queues.ts.redis.*`	Propiedades de Redis admitidas		Especifica las propiedades de configuración del servidor de Redis de colas de TS de JICS; consulte the section called “Propiedades de Redis admitidas”.	
`spring.aws.client.jics.queues.ts.redis.secret`	cadena	null	Especifica el ARN secreto de credenciales para el servidor de Redis de colas de TS de JICS; consulte the section called “AWS Secretos de Blu Age Runtime”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`lockTimeout`	número	500	El tiempo de espera del bloqueo, en milisegundos.	
`sqlCodePointShift`	número		Opcional. El cambio de punto del código sql. Cambia el punto de código de los caracteres de control que podemos encontrar al migrar datos de RDBMS antiguos a un RDBMS moderno. Por ejemplo, puede especificar `384` para que coincida con un carácter Unicode `\u0180`.	
`sqlIntegerOverflowAllowed`	valor booleano	false	Especifica si se permite el desbordamiento de enteros de SQL, es decir, si se permite colocar valores más grandes en la variable host.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`database.cursor.overflow.allowed`	booleano	true	Especifica si se debe permitir el desbordamiento del cursor. Configúrelo en `true` para realizar una siguiente llamada en el cursor, sea cual sea su posición. Configúrelo en `false` para comprobar si el cursor está en la última posición antes de realizar una siguiente llamada con el cursor. Actívela solo si el cursor es desplazable (sensible o INSENSIBLE).	
`reportOutputPath`	cadena	`/reports`	La ruta de salida del informe.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`spring.session.store-type`	cadena	none	La memoria caché de sesiones para entornos de alta disponibilidad. Los valores posibles son `none` o `redis`. El valor predeterminado es `none`.	
`stopExecutionWhenProgNotFound`	booleano	true	Especifica si se debe detener la ejecución si no se encuentra un programa. Si se establece en `true`, interrumpe la ejecución si no se encuentra ningún programa.	
`forceHR`	valor booleano	false	Especifica si se debe utilizar SYSPRINT inteligible, ya sea en la consola o en la salida de un archivo.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`rollbackOnRTE`	valor booleano	false	Especifica si se debe revertir la transacción implícita de la unidad de ejecución en las excepciones de tiempo de ejecución.	
`sctThreadLimit`	long	5	El límite de subprocesos para activar scripts.	
`dataSimplifier.onInvalidNumericData`	cadena	reject	Cómo reaccionar al decodificar datos numéricos no válidos. Los valores permitidos son `reject` /`tolerates paces` /`tolerates paceslowvalues` /`toleratemost` . El valor predeterminado es `reject`.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`filesDirectory`	cadena		El directorio de los archivos de entrada/salida de los lotes.	
`ims.messages.extendedSize`	valor booleano	false	Especifica si se establecerá el tamaño extendido en los mensajes IMS.	
`defaultKeepExistingFiles`	valor booleano	false	Especifica si se debe establecer el valor anterior predeterminado del conjunto de datos.	
`jics.db.ddlScriptLocation`	cadena		La ubicación del script DDL de JICS. Permite iniciar el esquema de la base de datos JICS mediante un script .sql. Está en blanco por defecto. Por ejemplo,. `./jics/sql/jics.sql`	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.db.schemaTestQueryLocation`	cadena		Ubicación del archivo sql que debe contener una consulta única que devuelva el número de objetos del esquema jics (si lo hay).	
`jics.db.dataScriptLocation`	cadena		Define la ruta a los scripts SQL utilizados para inicializar la base de datos JICS. Acepta una lista de archivos y directorios separados por comas, lo que permite especificar varios scripts y carpetas.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.db.dataTestQueryLocation`	cadena		Ubicación de un script sql que contiene una única consulta sql que se espera que devuelva un recuento de objetos (por ejemplo: contar el número de registros de la tabla del programa jics). Si el recuento es igual a 0, la base de datos se cargará mediante el `jics.db.dataScriptLocation` script; de lo contrario, se omitirá la carga de la base de datos.	
`jics.data.dataJsonInitLocation`	cadena			

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.xa.agent.timeout`	número			
`query.useConcatCondition`	valor booleano	false	Especifica si la condición clave se crea mediante la concatenación de claves o no.	
`system.qdecfmt`	cadena			
`disposition.checkexistence`	valor booleano	false	Especifica si se debe publicar una comprobación de la existencia del archivo para el conjunto de datos con DISP SHR u OLD.	
`useControlMVariable`	valor booleano	false	Especifica si se debe utilizar la especificación Control-M para el reemplazo de variables.	
`card.encoding`	cadena	CP1145	Codificación de tarjetas: para usar con `useControlMVariable` .	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mapTransfo.prefixes`	cadena	&,@,%%	Lista de prefijos que se utilizarán al transformar las variables de ControlM. Cada uno separado por comas.	
`checkinputfilesize`	valor booleano	false	Especifica si se debe activar una comprobación si el tamaño del archivo es un múltiplo del tamaño del registro.	
`stepFailWhenAbend`	booleano	true	Especifica si se debe generar una curva si un paso falla o completa la ejecución.	
`bluesam.fileLoading.commitInterval`	número	100000	El intervalo de confirmación de Bluesam.	
`uppercaseUserInput`	booleano	true	Especifica si la entrada del usuario debe estar en mayúsculas.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jhdb.lterm`	cadena		Le permiten forzar un ID de terminal lógico común en el caso de una emulación de IMS. Si no se establece, se utiliza SessionID.	
`jhdb.identificationCardData`	cadena		Se utiliza para codificar algunos datos de la tarjeta de identificación del operador en el campo MID designado por el parámetro CARD. Está en blanco por defecto, sin restricciones de entrada.	
`encoding`	cadena	ASCII	La codificación utilizada en los proyectos (no en los archivos groovy). Espera una codificación válida `CP1047IBM930AS`	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`cl.configuration.context.encoding`	cadena	CP297	La codificación de los archivos CL. Espera una codificación válida`CP1047`,`IBM` El valor predeterminado es `CP297`	
`cl.zonedMode`	cadena	EBCDIC_STRICT	El modo para codificar o decodificar los comandos del lenguaje de control (CL). Los valores permitidos son`EBCDIC_STRICT` /`EBCDIC_MODIFIED` /`AS400`.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`ims.programs`	cadena		Lista de programas IMS que se van a utilizar. Separe cada parámetro con punto y coma (`;`) y cada transacción con una coma (`,`). Por ejemplo:`PCP008,PCT008;PCP054,PCT054;PCP066,PCT066;PCP068,PCT068;`	
`jhdb.configuration.context.encoding`	cadena	CP297	La codificación JHDB (base de datos jerárquica de Java). Espera una cadena de codificación válida`CP1047`,`IBM`	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jhdb.metadata.extrapath`	cadena	file:./setup/	Parámetro de configuración que especifica una carpeta raíz adicional específica para el tiempo de ejecución para las carpetas psbs y dbds.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jhdb.checkpointPersistence`	cadena	none	El modo de persistencia del punto de control. Los valores permitidos sonnone /add /end. Utilice add para conservar los puntos de control cuando se crea y se agrega uno nuevo al registro. Utilice end para conservar el punto de control al cerrar el servidor. Cualquier otro valor deshabilita la persistencia. Tenga en cuenta que cada vez que se añada un nuevo punto de control al registro, todos los puntos de comprobación existentes se serializarán y el archivo se borrará. No es	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
			un anexo a los datos existentes en el archivo. Por lo tanto, dependiendo del número de puntos de control, puede tener algunos efectos en el rendimiento.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jhdb.checkpointPath`	cadena	archivo:. / configuración/	Si `jhdb.checkpointPersistence` no es `none`, este parámetro le permite configurar la ruta de persistencia del punto de control (ubicación de almacenamiento del archivo checkpoint.dat). Todos los datos de los puntos de control contenidos en el registro se serializan y se copian en un archivo (checkpoint.dat) ubicado en la carpeta proporcionada. Tenga en cuenta que esta copia de seguridad solo afecta a los datos de los puntos de control (ScriptID, StepID, posición de la base de	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
			datos y área del punto de control).	
`jhdb.navigation.cachenexts`	número	5000	La duración de la memoria caché (en milisegundos) utilizada en la navegación jerárquica de un RDBMS.	
`jhdb.use-db-prefix`	booleano	true	Especifica si se debe habilitar un prefijo de base de datos en la navegación jerárquica de un RDBMS.	
`jhdb.query.limitJoinUsage`	booleano	true	Especifica si se debe utilizar el parámetro límite de uso de uniones en los gráficos del RDBMS.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`taskExecutor.corePoolSize`	número	5	Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño del grupo de núcleos.	
`taskExecutor.maxPoolSize`	número	10	Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño máximo del pool (número máximo de subprocesos paralelos).	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`taskExecutor.queueCapacity`	número	50	Cuando se inicia una transacción en un terminal mediante un script groovy, se crea un nuevo subproceso. Utilice este parámetro para configurar el tamaño de la cola. (= número máximo de transacciones pendientes cuando se alcanza `taskExecutor.maxPoolSize`)	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`taskExecutor.allowCoreThreadTimeOut`	valor booleano	false	Especifica si se debe permitir que se agote el tiempo de espera de los subprocesos principales en JCIS. Esto permite un crecimiento y una reducción dinámicos incluso en combinación con una cola distinta de cero (ya que el tamaño máximo del grupo solo aumentará una vez que la cola esté llena).	
`jics.runUnitLauncherPool.enable`	valor booleano	false	Especifica si se debe activar el grupo de lanzadores de unidades de ejecución en JICS.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.runUnitLauncherPool.size`	número	20	El tamaño del conjunto de lanzadores de unidades de ejecución en JICS.	
`jics.runUnitLauncherPool.validationInterval`	número	1 000	El intervalo entre cada ejecución de la tarea que ajusta el tamaño del grupo.	
`jics.runUnitLauncherPool.parallelism`	número	2	El número de subprocesos que se utilizan para generar las instancias que faltan en la cola cuando se ejecuta la tarea de ajuste.	
`context.preconstruct.enable`	valor booleano	false	Especifica si se activará la construcción previa del contexto del programa.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`context.preconstruct.frequencyInMillis`	número	100	El intervalo entre cada ejecución de la tarea que ajusta el tamaño del grupo.	
`context.preconstruct.parallelism`	número	5	El número de subprocesos que se utilizan para generar las instancias que faltan en la cola cuando se ejecuta la tarea de ajuste.	
`context.preconstruct.minInstances`	número	2	El número de instancias que se crearán la primera vez que se necesite un contexto.	
`spring.aws.application.credentials`	cadena	null	Cargue las AWS credenciales del archivo de perfiles de credenciales en JICS.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jics.queues.sqs.region`	cadena	eu-west-1	La AWS región de Amazon Simple Queue Service, utilizada en JICS.	
`jics.jcl.rt.encoding`	cadena	CP037	La codificación de los scripts JCL escritos en la cola JICS dedicada.	
`jics.jcl.rt.queue`	cadena	JICS	El nombre de la cola en la que se pueden escribir los scripts de JCL línea por línea en tiempo de ejecución.	
`mq.queues.sqs.region`	cadena	eu-west-3	La AWS región del servicio AWS SQS MQ.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`quartz.scheduler.stand-by-if-error`	valor booleano	false	Especifica si se debe activar la ejecución del trabajo si el programador de trabajos está en modo de espera. Si es verdadero, cuando está habilitada, no se activa la ejecución de tareas.	
`databaseStatistics`	valor booleano	false	Especifica si se debe permitir que los compiladores de SQL recopilen y muestren información estadística.	
`dbDateFormat`	cadena	aaaa-MM-dd	El formato de fecha objetivo de la base de datos.	
`dbTimeFormat`	cadena	HH:mm:ss	El formato de hora objetivo de db.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`dbTimestampFormat`	cadena	yyyy-MM-dd HH:MM:SS.SSSSSS	El formato de marca de tiempo objetivo de la base de datos.	
`dateTimeFormat`	cadena	ISO	dateTimeFormat Describe cómo incluir el tipo de fecha, hora y hora de la base de datos en las entidades simplificadoras de datos. Los valores permitidos son `ISO` /EUR /EUR /USA /LOCAL	
`localDateFormat`	cadena		Lista de formatos de fecha local. Separe cada formato con \.	
`localTimeFormat`	cadena		Lista de formatos de hora local. Separe cada formato con \	
`localTimestampFormat`	cadena		Lista de formatos de marcas de tiempo locales. Separe cada formato con \.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`pgmDateFormat`	cadena	aaaa-MM-dd	formato de fecha hora	
`pgmTimeFormat`	cadena	Hh.mm.ss	Formato de hora utilizado para la ejecución de los pgm (programas).	
`pgmTimestampFormat`	cadena	yyyy-MM-dd-HH.mm.ss.SSSSSS	Formato de marca temporal	
`cacheMetadata`	booleano	true	Especifica si se deben almacenar en caché los metadatos de la base de datos	
`forceDisableSQLTrimStringType`	valor booleano	false	Especifica si se debe deshabilitar el recorte de todos los parámetros de la cadena SQL.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`fetchSize`	número		El valor fetchSize de los cursores. Se usa cuando se obtienen datos mediante fragmentos a través de utilidades de carga o descarga.	
`check-groovy-file`	booleano	true	Especifica si se debe comprobar el contenido de los archivos groovy antes de registrarlos.	
`qtemp.uuid.length`	número	9	La longitud del identificador único de QTEMP.	
`qtemp.dblog`	valor booleano	false	Si se debe habilitar el registro de la base de datos QTEMP.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`qtemp.cleanup.threshold.hours`	número	0	Para especificar cuándo `qtemp.dblog` está habilitado. La vida útil de la partición de base de datos (en horas).	
`sort.function`	cadena		El nombre de la función de ordenación de la base de datos blu4iv.	
`invalidDataTolerence`	booleano	true	Especifica si se toleran datos no válidos para el tipo empaquetado.	
`program.timeout`	número	-1	Especifica un tiempo de espera en segundos para la ejecución de cualquier programa o transacción. Transcurrido este tiempo, el sistema intentará interrumpir el programa.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk.line.separator`	cadena	null	Especifica el tipo de separador de líneas en Gapwalk. Los valores permitidos son WIN (CRLF)/UNIX (LF)/LINUX (LF). Los demás valores se ignoran y se utiliza la propiedad de separador de líneas del sistema.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`enableActivePgmIdCache`	valor booleano	false	Especifica si se habilitará la caché local del ID del programa activo. Utilice esta característica con cuidado porque los recursos de JICS se pueden compartir entre los programas y los usuarios. Cualquier administrador puede cambiar esos recursos de manera externa y la memoria caché local instalada podría invalidarse.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mq.queues.default.syncpoint`	valor booleano	false	Especifica el comportamiento por defecto de los comandos MQ PUT cuando no están establecidos ni MQPMO_SYNCPOINT ni MQPMO_NO_SYNCPOINT. Cuando se establece en verdadero, actúa como si `MQPMO_SYNCPOINT` y los mensajes NO se confirman directamente durante el comando PUT. Si se establece en falso, actúa como si `MQPMO_NO_SYNCPOINT` y los mensajes se confirman directamente durante el comando PUT.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`dataSimplifier.byteRangeBoundsCheck`	valor booleano	false	Cuando se establece en verdadero, garantiza que no ByteRange se cree con valores incorrectos. El valor predeterminado es false.	
`file.stdoutIntoLogger`	valor booleano	false	Especifica si se habilitará la escritura en el registrador en lugar del flujo de salida predeterminado del sistema en los archivos `SYSPRINT` y `SYSPUNCH` predeterminados.	
`tempFilesDirectory`	cadena	null	Especifica el nombre de la ubicación de la carpeta de los archivos temporales que se generan.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`cleanTempFilesDirectoryAtStartup`	booleano	true	Especifica si se purgará el contenido de la carpeta de archivos temporales al iniciar la aplicación.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`tempFolderPattern`	cadena	null	Especifica un patrón que se utilizará para crear dinámicamente el nombre de la carpeta temporal en función de la siguiente información predefinida y personalizable. HOST: el nombre del host. JOBID: el ID del trabajo. HASHCODE: el código hash del contexto del trabajo. TIMESTAMP: el patrón que se usará para obtener la marca de tiempo. El nombre de destino de la carpeta temporales TMP_DIR_{}. tempFolderPattern Por	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
			ejemplo, en el caso del siguiente patrón, el nombre empezará con el identificador del trabajo y terminará con la «marca de tiempo» tempFolderPattern: JOBID, host=XXXXX, HASHCODE, timestamp =YYYYMMDDHHMMSS. Si la propiedad no `tempFolderPattern` se añade al archivo YAML o está vacía, el nombre de la carpeta temporal será «TMP_DIR_» + this.hashcode () (). DefaultJobContext	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`database.cursor.raise.already.opened.error`	valor booleano	false	Especifica si se habilitará la generación del error 502 de SQLCODE al abrir un cursor que ya está abierto.	
`jics.spool.smtp.hostname`	cadena	null	Especifica el host del servidor SMTP. Ejemplo: `smtp.xxx.com`	
`jics.spool.smtp.port`	cadena	null	Especifica el puerto del servidor SMTP. Ejemplo: 25	
`jics.spool.smtp.password`	cadena	null	Especifica la contraseña de inicio de sesión del servidor SMTP.	
`jics.spool.smtp.username`	cadena	null	Especifica el nombre de usuario del servidor SMTP.	
`jics.spool.smtp.debug`	valor booleano	false	Especifica el modo de depuración del servidor SMTP.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk-application.security`	cadena	`disabled`	Cambia la configuración de seguridad global (autenticación XSS, CORS, CSRF, OAUTH...). Los valores permitidos son `disabled` y `enabled`.	
`gapwalk-application.identity`	cadena	null	Método de autenticación global. El valor recomendado es `oauth`. Los valores permitidos son `json` y `oauth`. Esta opción es obligatoria cuando `gapwalk-application.security` es `enabled`.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk-application.security.issuerUri`	cadena	null	El URI de emisores el proveedor de identidades (IdP). Esta opción es obligatoria cuando `gapwalk-application.identity` es `oauth`.	
`gapwalk-application.security.allowedOrigins`	string[]	`null`	La lista de orígenes que se permitirá. Esta opción requiere que `gapwalk-application.identity` se establezca en `oauth`.	
`gdgDirectoryPath`	cadena	`output/gdg`	La ruta del directorio GDG es el directorio donde se almacenan los archivos gdg.	4.6.0

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk-application.security.claimGroupName`	cadena	`cognito:groups`	El atributo de notificación que contiene la lista de todos los grupos a los que pertenece el usuario. Use `cognito:groups` para Amazon Cognito o cualquier otra cadena para un IdP extranjero.	
`gapwalk-application.security.userAttributeName`	cadena	`username`	El nombre del atributo de notificación utilizado para identificar la solicitud de un usuario. Use `username` para Amazon Cognito, `preferred_username` para Keycloak o cualquier otra cadena para un IdP extranjero.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk-application.security.localhostWhitelistingEnabled`	booleano	true	Especifica si se habilitará la autenticación a partir de cualquier solicitud de `localhost` .	
`gapwalk-application.defaultSuperAdminUserName`	cadena	`sadmin`	Cuando `gapwalk-application.security` está deshabilitado, especifica el nombre de superusuario local predeterminado.	
`gapwalk-application.defaultSuperAdminUserPwd`	cadena	`sadmin`	Cuando `gapwalk-application.security` está deshabilitado, especifica la contraseña de superusuario local predeterminada.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk-application.security.filterURIs`	cadena	`disabled`	Cambia la configuración de filtrado. URIs Los valores permitidos son `disabled` y `enabled`.	
`gapwalk-application.security.blockedURIs`	string[]	null	La lista de cosas URIs a bloquear. Esta opción es obligatoria cuando `gapwalk-application.security.filterURIs` es `enabled`.	
`jics.redis.*`	Propiedades de Redis admitidas		Especifica las propiedades de configuración de la fábrica de conexiones del servidor de Redis de JICS; consulte the section called “Propiedades de Redis admitidas”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`spring.aws.client.jics.redis.secret`	cadena	null	Especifica el ARN secreto de credenciales para la fábrica de conexiones del servidor de Redis de JICS; consulte the section called “AWS Secretos de Blu Age Runtime”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jcl.checkpoint.enabled`	valor booleano	false	Especifica si el mecanismo de puntos de comprobación de JCL está habilitado o para permitir el reinicio del trabajo. Los puntos de comprobación de JCL se crean y guardan en el registro en memoria al inicio de cada paso o de la invocación del programa principal. Todos los puntos de comprobación de paso se conservan al final del trabajo, si se ha definido un proveedor de persistencia.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jcl.checkpoint.expireTimeout`	número	-1	Especifica el tiempo que se deben retener los puntos de comprobación de JCL en el proveedor de persistencia o en el registro en memoria.	
`jcl.checkpoint.expireTimeoutUnit`	cadena	SECONDS	Especifica la unidad de duración de tiempo de la propiedad `jcl.checkpoint.expireTimeout` . Valores constantes de enumeración admitidos: java.util.concurrent. TimeUnit.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jcl.checkpoint.provider`	cadena	null	Especifica el proveedor de persistencia del mecanismo de puntos de comprobación de JCL. Los valores permitidos son `redis`.	
`jcl.checkpoint.redis.*`	Propiedades de Redis admitidas		Especifica las propiedades de configuración del proveedor de persistencia REDIS del mecanismo de puntos de comprobación de JCL; consulte the section called “Propiedades de Redis admitidas”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`spring.aws.client.jcl.checkpoint.redis.secret`	cadena	null	Especifica el ARN secreto de credenciales para el proveedor de persistencia de Redis del mecanismo de puntos de comprobación de JCL; consulte the section called “AWS Secretos de Blu Age Runtime”.	
`gapwalk.ssl.enabled`	valor booleano	false	Se indica para establecer las siguientes propiedades `gapwalk.ssl.*` en las propiedades actuales del sistema JVM si aún no están establecidas al iniciar la aplicación.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk.ssl.trustStore`	cadena	null	Establezca el valor en la propiedad del sistema `javax.net.ssl.trustStore` si aún no se ha establecido al iniciar la aplicación.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk.ssl.trustStorePassword`	cadena	null	Establezca el valor en la propiedad del sistema `javax.net.ssl.trustStorePassword` si aún no se ha configurado al iniciar la aplicación. Como alternativa, se recomienda encarecidamente el uso de AWS secretos, como se explica en the section called “Administrador de secretos para la configuración de contraseñas SSL”.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk.ssl.trustStoreType`	cadena	null	Establezca el valor en la propiedad del sistema `javax.net.ssl.trustStoreType` si aún no se ha configurado al iniciar la aplicación.	
`gapwalk.ssl.keyStore`	cadena	null	Establezca el valor en la propiedad del sistema `javax.net.ssl.keyStore` si aún no se ha configurado al iniciar la aplicación.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`gapwalk.ssl.keyStorePassword`	cadena	null	Establezca el valor en la propiedad del sistema `javax.net.ssl.keyStorePassword` si aún no se ha configurado al iniciar la aplicación. Como alternativa, se recomienda encarecidamente el uso de AWS secretos, como se explica en. the section called “Administrador de secretos para la configuración de contraseñas SSL”	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mq.queues`	cadena	`sqs`	Especifica qué agente de colas admitido se utilizará entre `sqs` usando Amazon SQS, `rabbitmq`, usando Rabbit MQ en las instalaciones `jms` usando IBMMQ en las instalaciones.	
`mq.queues.jmsMQQueueManagers[N]`			Cuando `mq.queues` es `jms`, permite especificar una lista de conexiones de IBM MQ. `mq.queues.jmsMQQueueManagers[0]` para la primera conexión, `mq.queues.jmsMQQueueManagers[1]` para la segunda y así sucesivamente.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mq.queues.jmsMQQueueManagers[N].jmsMQQueueManager`	cadena	null	El nombre del administrador de colas de IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQAppName`	cadena	null	El nombre de la aplicación de IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQChannel`	cadena	null	El nombre del canal de IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQHost`	cadena	null	El nombre de host de IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQPort`	número	null	El puerto de IBMMQ.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mq.queues.jmsMQQueueManagers[N].jmsMQUserid`	cadena	null	El nombre de usuario de IBMMQ.	
`mq.queues.jmsMQQueueManagers[N].jmsMQPassword`	cadena	null	La contraseña de usuario de IBMMQ. Como alternativa, se recomienda encarecidamente el uso de AWS secretos, como se explica en. the section called “Administrador de secretos para la configuración de contraseñas de IBM MQ”	
`mq.queues.jmsMQQueueManagers[N].jmsMQMaxPoolSize`	número	0	El tamaño de grupo máximo de IBMMQ. Con 0, se habilita un número infinito de conexiones físicas.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mq.queues.jmsMQQueueManagers[N].jmsMQSSLCipher`	cadena	null	El conjunto de cifrado SSL de IBMMQ. Un ejemplo podría ser `"*TLS120RHIGHER"` . Consulte la documentación oficial de TLS CipherSpecs y de las clases CipherSuites de IBM MQ para JMS para obtener más información.	
`mq.queues.non.jms.client`	valor booleano	false	Indique si el cliente de destino al que se van a enviar los mensajes no es JMS. El formato MQ nativo se usará para clientes que no sean JMS, mientras que el RFH2 formato se usará para JMS.	4.5.0

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
			Cuando `mq.queues` es `rabbitmq`, el nombre de host de IBMMQ.	
`mq.queues.rabbitMQHost`			El nombre de host de MQ de Rabbit.	
`mq.queues.rabbitMQVirtualHost`			El nombre de host virtual de MQ de Rabbit.	
`mq.queues.rabbitMQPort`			El puerto MQ de Rabbit.	
`mq.queues.rabbitMQUsername`			El usuario de MQ de Rabbit.	
`mq.queues.rabbitMQPassword`			La contraseña de MQ de Rabbit.	
`mf.runtime.switch.N`	booleano	true	Permite la inserción de valores nulos en archivos secuenciales de líneas de MF Nature.	4.4.0

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`mf.runtime.switch.T`	valor booleano	false	Permite la inserción de caracteres de tabulación en archivos secuenciales de líneas naturales de MF.	4.4.0
`gapwalk.database.support.useSavePointToRestoreFail`	valor booleano	false	Permite la recuperación de transacciones en caso de error mediante el uso de puntos de almacenamiento en las consultas de inserción. La activación de esta propiedad puede afectar al rendimiento de la base de datos. Puede anular esta configuración para consultas específicas mediante la configuración de query-to-database mapeo.	4.6.0

Propiedades disponibles para las aplicaciones web opcionales

En función de la aplicación modernizada, es posible que necesite configurar una o más aplicaciones web opcionales que admitan dependencias como z/OS, AS/400 o IMS/MFS. The following tables contain lists of the available key/value parámetros para configurar cada aplicación web opcional.

gapwalk-utility-pgm.war

Esta aplicación web opcional contiene soporte para los programas de utilidades de Z/OS.

Esta tabla proporciona una vista exhaustiva de los parámetros clave/valores para esta aplicación.

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`logging.config`	Ruta	classpath:logback-utility.xml	Clave estándar para la referencia al archivo de configuración del logback. También hay disponibles otras claves de registro estándar.	
`spring.jta.enabled`	valor booleano	false	Clave estándar. Si el modo de soporte del origen de datos no es static-xa, la configuración automática de las transacciones JTA de Spring debe estar deshabilitada.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`spring.datasource.primary.jndi-name`	cadena	jdbc/primary	El nombre JNDI (Interfaz de Nombrado y Directorio Java) del origen de datos principal, si se utiliza JNDI.	
`primary.datasource -driver-class-name -url -username -password`	Origen de datos estándar de Spring con subclaves		Contiene la información de conexión de la base de datos de la aplicación, si no se utiliza JNDI. Debe tener la misma configuración que en el archivo YAML de la aplicación modernizada. Como alternativa, se recomienda encarecidamente el uso de AWS secretos, como se explica en. the section called “Base de datos de clientes”	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`encoding`	cadena	ASCII	La codificación utilizada en los programas de utilidades. Espera una codificación válida`CP1047`,, `IBM930ASCII`,`UTF`	
`sysPunchEncoding`	cadena	ASCII	El juego de caracteres que codifica el syspunch. Espera una codificación válida`CP1047`,`IBM`	
`systin.encoding`	cadena	ASCII	El conjunto de caracteres de codificación del conjunto de datos de archivos SYSTIN. Espera una codificación válida`CP1047`,, `IBM930ASCII`,`UTF`	4.5.0

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`zonedMode`	cadena	EBCDIC_STRICT	El modo para codificar o decodificar tipos de datos zonificados. Los valores permitidos son`EBCDIC_STRICT` /`EBCDIC_MODIFIED` /`AS400`.	
`idcams.encoding.forced`	cadena		La codificación utilizada en el programa de utilidad IDCAMS. Espera una codificación válida`CP1047`,, `IBM930ASCII`,`UTF`	4.4.0
`unload.chunkSize`	número	0	Tamaño de fragmento utilizado para la utilidad de descarga.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`unload.computeRecordSizeIfNull`	valor booleano	false	Determina si se debe calcular el tamaño del registro si no se especifica. Si se especifica, el valor permanece sin cambios.	
`unload.sqlCodePointShift`	número	0	El punto de cambio en el código SQL para la utilidad de descarga Ejecuta el proceso de cambio de caracteres. Se requiere cuando la base de datos de destino DB2 es Postgresql.	
`unload.columnFiller`	cadena	espacio	El rellenador de columnas de la utilidad de descarga.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`unload.varCharIsNull`	valor booleano	false	Utilice este parámetro en el programa INFTILB; si `true` se establece en, todos los campos que no admiten valores nulos con valores en blanco (espacio) devolverán una cadena vacía.	
`unload.useDatabaseConfiguration`	valor booleano	false	Especifica si se debe usar la configuración de fecha u hora de application-main.yml en la utilidad de descarga.	
`unload.format.date`	cadena	MM/dd/yyyy	Si `unload.useDatabaseConfiguration` está activado, el formato de fecha que se va a utilizar en la utilidad de descarga.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`unload.format.time`	cadena	Hh.mm.ss	Si `unload.useDatabaseConfiguration` está activado, el formato de hora que se utilizará en la utilidad de descarga.	
`unload.format.timestamp`	cadena	yyyy-MM-dd-HH.mm.ss.ssssss	Si `unload.useDatabaseConfiguration` está activado, el formato de marca de tiempo que se utilizará en la utilidad de descarga.	
`unload.nbi.whenNull`	hexadecimal	6F	El valor del indicador de bytes nulos (NBI) que se agrega cuando el valor de la base de datos es nulo.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`unload.nbi.whenNotNull`	hexadecimal	00	El valor del indicador de bytes nulos (NBI) que se agrega cuando el valor de la base de datos no es nulo.	
`unload.nbi.writeNullIndicator`	valor booleano	false	Especifica si se debe escribir el indicador nulo en el archivo de salida de descarga.	
`unload.bmc.useInto`	valor booleano	false	Especifica si se gestionará la palabra clave de control bmc INTO para la utilidad de descarga.	
`unload.fetchSize`	número	0	Permite ajustar el tamaño de la búsqueda al manipular los cursores en la utilidad de descarga.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`unload.noPad`	booleano	true	Indica que los campos de caracteres de longitud variable (VARCHAR) deben descargarse sin rellenarlos hasta alcanzar la longitud máxima.	4.5.0
`treatLargeNumberAsInteger`	valor booleano	false	Especifica si se deben tratar los números grandes como `Integer`. Se tratan como `BigDecimal` por defecto.	
`load.batchSize`	número	0	El tamaño del lote de la utilidad de carga.	
`load.format.localDate`	cadena	dd/MM/yyydd.MM.aaaa\\aaaa-MM-dd	El formato de hora local de la utilidad de carga que se va a utilizar.	
`load.format.localTime`	cadena	HH:mm:ss\HH.mm.ss	El formato de hora local de la utilidad de carga que se va a utilizar.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`load.format.dbDate`	cadena	yyyy-MM-dd	El formato de base de datos de la utilidad de carga que se va a utilizar.	
`load.format.dbTime`	cadena	HH:mm:ss	La hora de la base de datos de la utilidad de carga que se va a utilizar.	
`load.sqlCodePointShift`	número	0s	El punto de cambio en el código SQL para la utilidad de carga Ejecuta el proceso de cambio de caracteres. Necesario cuando la base de datos de destino es Postgresql. DB2	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`load.applyRollback`	valor booleano	false	Establezca este parámetro en `true` para indicar que desea que el servicio revierta los cambios en la tabla si detecta un error al cargar los datos en la base de datos.	
`forcedDate`	cadena		Fuerza la fecha a la fecha proporcionada, si la hay.	
`frozenDate`	booleano	true	Especifica si se debe congelar la fecha. Solo se aplica si también `forcedDate` está establecida.	

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`jcl.type`	cadena	mvs	Tipo de archivo .jcl. Los valores permitidos son `jcl` / `vse`. Los comandos PRINT/REPRO de la utilidad IDCAMS devuelven 4 si el archivo está vacío en el caso de jcl que no es vse.	
`hasGraphic`	valor booleano	false	Si la utilidad INFUTILB necesita gestionar columnas GRÁFICAS. DB2	
`convertGraphicDataToFullWidth`	booleano	true	Especifica si se convertirán los datos gráficos a un formato de ancho completo.	

gapwalk-cl-command.guerra

Esta aplicación web opcional contiene soporte para los programas de utilidades del AS/400.

Esta tabla proporciona una vista exhaustiva de los parámetros clave/valores para esta aplicación.

Clave	Tipo	Valor predeterminado	Descripción
`logging.config`	Ruta	classpath:logback-utility.xml	Clave estándar para la referencia al archivo de configuración del logback. También hay disponibles otras claves de registro estándar.
`spring.jta.enabled`	valor booleano	false	Clave estándar. Si el modo de soporte del origen de datos no es static-xa, la configuración automática de las transacciones JTA de Spring debe estar deshabilitada.
`spring.datasource.primary.jndi-name`	cadena	jdbc/primary	El nombre JNDI (Interfaz de Nombrado y Directorio Java) del origen de datos principal, si se utiliza JNDI.
`primary.datasource + -driver-class-name + -url + -username + -password`	Origen de datos estándar de Spring con subclaves		Contiene la información de conexión de la base de datos de la aplicación, si no se utiliza JNDI. Debe tener la misma configuración que en el archivo YAML de la aplicación modernizada.

Clave	Tipo	Valor predeterminado	Descripción
			Como alternativa, se recomienda encarecidamente el uso de AWS secretos, como se explica en. the section called “Base de datos de clientes”
`encoding`	cadena	ASCII	La codificación utilizada en los programas de utilidades. Espera una codificación válida `CP1047IBM930ASCIIUTF-8`.
`zonedMode`	cadena	EBCDIC_STRICT	El modo para codificar o decodificar tipos de datos zonificados. Los valores permitidos son`EBCDIC_STRICT /EBCDIC_MODIFIED /AS400`.

Clave	Tipo	Valor predeterminado	Descripción
`commands-off`	cadena		Lista de comandos a desactivar, separados por comas. Los valores permitidos son `PGM_BASIC,RCVMSG,SNDRCVF,CHGVAR,Q`. Útil cuando se desea deshabilitar o sobrescribir un programa existente. `PGM_BASIC` es un programa específico de AWS Blu Age Runtime diseñado con fines de depuración.
`forcedDate`	cadena		Fuerza la fecha a la fecha proporcionada, si la hay.

gapwalk-hierarchical-support.guerra

Esta aplicación web opcional contiene soporte para transacciones de IMS/MFS.

Esta tabla proporciona una vista exhaustiva de los parámetros clave/valores para esta aplicación.

Clave	Tipo	Valor predeterminado	Descripción
`logging.config`	Ruta	classpath:logback-utility.xml	Clave estándar para la referencia al archivo de configuración del logback. También hay disponibles otras

Clave	Tipo	Valor predeterminado	Descripción
			claves de registro estándar.
`spring.jta.enabled`	valor booleano	false	Clave estándar. Si el modo de soporte del origen de datos no es static-xa, la configuración automática de las transacciones JTA de Spring debe estar deshabilitada.
`jhdb.configuration.context.encoding`	cadena		La codificación JHDB (base de datos jerárquica de Java). Espera una cadena de codificación válida`CP1047`,`IBM930`,`ASCII`,

Clave	Tipo	Valor predeterminado	Descripción
`jhdb.checkpointPersistence`	cadena	none	El modo de persistencia del punto de control. Los valores permitidos sonnone /add /end. Utilice add para conservar los puntos de control cuando se crea y se agrega uno nuevo al registro. Utilice end para conservar el punto de control al cerrar el servidor. Cualquier otro valor deshabilita la persistencia. Tenga en cuenta que cada vez que se añada un nuevo punto de control al registro, todos los puntos de comprobación existentes se serializarán y el archivo se borrará. No es un anexo a los datos existentes en el archivo. Por lo tanto, dependiendo del número de puntos de control, puede tener algunos efectos en el rendimiento.

Propiedades disponibles para la aplicación cliente

Es posible que su aplicación modernizada requiera configuraciones de propiedades específicas para la aplicación Spring del cliente. Estas propiedades inicializan los beans de las clases empaquetadas en archivos JAR en tiempo de ejecución. El `application-profile.yaml` archivo, en el que se establece el valor del perfil durante la generación de la aplicación, permite configurar estas propiedades. En la siguiente tabla se enumeran los parámetros clave/valor disponibles para configurar la aplicación web cliente que utiliza fuentes de clases empaquetadas en el entorno de ejecución de Gapwalk

Clave	Tipo	Valor predeterminado	Descripción	Versión de lanzamiento
`blu4iv.dtaara.library.disable`	valor booleano	false	Controla el uso de la biblioteca en el contexto de las operaciones del área de datos. Si se establece en True, el uso de la bibliotec a está deshabilitado para las operaciones del área de datos, pero esto no afecta al uso de QTemp. Si se establece en falso, se tiene en cuenta la bibliotec a al realizar operaciones CRUD en el área de datos.	4.5.0

Propiedades de caché de Redis disponibles en AWS Blu Age Runtime

Puedes usar este documento para obtener información sobre las cachés de Redis en AWS Blu Age Runtime, junto con la configuración de Gapwalk, las propiedades de Redis compatibles y cómo el archivo `application-main.yml` puede hacer referencia al ARN secreto de las cachés de Redis.

Redis almacena en caché en Blu Age Runtime AWS

Los servidores Redis se pueden usar como cachés para varias funciones de la aplicación AWS Blu Age Gapwalk, como:

AWS Funciones de Blu Age Runtime que utilizan el almacenamiento en caché de Redis	Descripción
Caché de Blusam	Una caché de Blusam de Redis para leer los registros con eficiencia, mediante una estrategi a de escritura diferida, con el fin de optimizar las cargas de trabajo de escritura intensiva que se encuentran en las cargas útiles por lotes.
Bloqueos de Blusam	Una memoria caché para bloqueos distribuidos de conjuntos de datos y registros.
Catálogo de conjuntos de datos	La caché del conjunto de datos del catálogo.
Caché de sesión	Una caché de Redis para. HttpSession La caché almacena el nombre de usuario, el estado del diálogo con la interfaz de Angular y la información específica sobre el «dialecto » (BMS, MFS, 00). AS4
Rastreador de sesiones	Un caché de sesiones activas con el nombre de usuario y la información asociados. session-creation-time
Caché de JICS	Una caché para las definiciones de recursos de JICS.
Colas de TS	Almacenamiento para colas de TS.

AWS Funciones de Blu Age Runtime que utilizan el almacenamiento en caché de Redis	Descripción
Puntos de comprobación de JCL	Caché de puntos de comprobación de JCL.
Bloqueos de archivos de Gapwalk	Una caché para los bloqueos de archivos distribuidos por trabajo.
Bloqueos de Blu4iv	Almacenamiento para bloqueos de registros de Blu4iv.

Configuración de Gapwalk de Redis

Se utiliza la configuración global de Redis si se especifica `redis` como el mecanismo de almacenamiento en caché y no se proporciona ninguna configuración de Redis para la característica específica. Esta configuración le permite utilizar la misma configuración para varias cachés de Redis de manera simultánea.

En el siguiente ejemplo, la caché de conjuntos de datos de Blusam y la caché de JICS utilizan la configuración `gapwalk.redis` (`redis.server1`) porque su tipo de caché está establecido en `redis` y no se especifica ninguna propiedad de Redis implícita en the section called “Definiciones de recursos de JICS” y the section called “Definiciones de recursos de JICS”. Sin embargo, la caché de bloqueos de Blusam utilizará una configuración de Redis diferente (`redis.server2`) porque sus propiedades de Redis están definidas de manera explícita.

```
...

 gapwalk:
   redis:
     hostName: redis.server1
  port: 6379
 ...

 bluesam:
   # Redis bluesam cache
   cache: redis
   # Redis locks cache
   locks:
     cache: redis
  hostName: redis.server2
```

```
 port: 6379
...
# Redis jics cache
jics:
  resource-definitions:
    store-type: redis
...
```

Para habilitar la configuración global de Redis, añada la siguiente configuración en `main-application.yml`.

```
gapwalk:
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Propiedades de Redis admitidas

La siguiente tabla muestra las propiedades de Redis compatibles con las cachés de Redis globales y específicas en AWS Blu Age Runtime.

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
`mode`	No	El modo de ejecución de Redis.	`standalone | cluster`	`standalone`
`hostname`	Sí	La dirección IP o el nombre de host del servidor de Redis.	cadena	null
`port`	Sí	El número de puerto en el que el servidor de Redis escucha las conexiones.	int	nulo
`username`	No	El nombre de usuario para la autenticación.	cadena	null
`password`	No	La contraseña para la autenticación.	cadena	empty string
`useSsl`	No	Especifica si se habilitará el cifrado SSL/TLS para la conexión de Redis.	valor booleano	false
`database`	No	El número de la base de datos de Redis que se utilizará. Redis admite varias bases de	int	0

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
		datos lógicas y esta propiedad especifica cuál se usará.		
`maxTotal`	No	El número máximo de conexiones permitidas en el grupo de conexiones de Redis.	int	128
`maxIdle`	No	El número máximo de conexione s inactivas permitidas en el grupo de conexiones de Redis.	int	128
`minIdle`	No	El número mínimo de conexiones inactivas que se mantendrán en el grupo de conexiones de Redis.	int	16

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
`testOnBorrow`	No	Un valor booleano que indica si se validarán las conexiones antes de tomarlas prestadas del grupo.	booleano	true
`testOnReturn`	No	Un valor booleano que indica si se validarán las conexiones antes de devolverlas al grupo.	booleano	true
`testWhileIdle`	No	Un valor booleano que indica si se validarán periódicamente las conexiones inactivas del grupo.	booleano	true
`testOnCreate`	No	Un valor booleano que indica si se validarán las conexiones cuando se creen.	booleano	true

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
`minEvictableIdleTimeMillis`	No	Cantidad de tiempo mínima (en milisegundos) que una conexión inactiva debe permanecer en el grupo antes de que se pueda expulsar.	long	60000L
`timeBetweenEvictionRunsMillis`	No	El tiempo (en milisegundos) entre ejecuciones sucesivas del subproceso expulsor de conexión inactiva.	long	30000L
`numTestsPerEvictionRun`	No	El número máximo de conexiones para probar durante cada ejecución del subproceso expulsor de conexiones inactivas.	int	-1

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
`blockWhenExhausted`	No	Valor booleano que indica si se debe bloquear y esperar a que una conexión esté disponible cuando el grupo esté agotado.	booleano	true
`nettyThreads`	No	El número de subprocesos de Netty que se usarán para gestionar las conexiones de Redis.	int	32
`subscriptionsPerConnection`	No	El número máximo de suscripciones permitidas por conexión de Redis.	int	10
`subscriptionConnectionPoolSize`	No	El número máximo de conexiones permitidas en el grupo de conexiones de suscripción de Redis.	int	100

Nombre de la propiedad	¿Obligatorio?	Descripción	Valores	Predeterminado/a
`pageSizeInBytes`	No	El tamaño de página predeterminado en bytes para las operaciones de Redis.	long	262144000
`readTimeout`	No	El tiempo de espera de lectura en milisegundos para las operaciones de Redis.	long	2000
`timeToLiveMillis`	No	El tiempo (en milisegundos) durante el que una entrada de la caché permanece en la memoria caché antes de que se considere caducada y eliminada. Si no se especifica esta propiedad, las entradas de la caché no caducarán automáticamente de forma predeterminada.	long	-1

Propiedades de caché de Redis

Caché de Blusam de Redis

```
bluesam:
   cache: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                        # Optional
     username:                               # Optional
     password: ""                            # Optional
     useSsl: false                           # Optional
     database: 0                             # Optional
     maxTotal: 128                           # Optional
     maxIdle: 128                            # Optional
     minIdle: 16                             # Optional
     testOnBorrow: true                      # Optional
     testOnReturn: true                      # Optional
     testWhileIdle: true                     # Optional
     testOnCreate: true                      # Optional
     minEvictableIdleTimeMillis: 60000       # Optional
     timeBetweenEvictionRunsMillis: 30000    # Optional
     numTestsPerEvictionRun: -1              # Optional
     blockWhenExhausted: true                # Optional
     nettyThreads: 32                        # Optional
     subscriptionsPerConnection: 10          # Optional
     subscriptionConnectionPoolSize: 100     # Optional
     pageSizeInBytes: 8192                   # Optional
     readTimeout: 2000                       # Optional
     timeToLiveMillis: 60000                 # Optional
```

Caché de Blusam de Redis

```
bluesam:
   locks:
     cache: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
       hostName: localhost
       port: 6379
```

```
      mode: standalone                          # Optional
      username:                                 # Optional
      password: ""                              # Optional
      useSsl: false                             # Optional
      database: 0                               # Optional
      maxTotal: 128                             # Optional
      maxIdle: 128                              # Optional
      minIdle: 16                               # Optional
      testOnBorrow: true                        # Optional
      testOnReturn: true                        # Optional
      testWhileIdle: true                       # Optional
      testOnCreate: true                        # Optional
      minEvictableIdleTimeMillis: 60000         # Optional
      timeBetweenEvictionRunsMillis: 30000      # Optional
      numTestsPerEvictionRun: -1                # Optional
      blockWhenExhausted: true                  # Optional
      nettyThreads: 32                          # Optional
      subscriptionsPerConnection: 10            # Optional
      subscriptionConnectionPoolSize: 100       # Optional
      pageSizeInBytes: 8192                     # Optional
      readTimeout: 2000                         # Optional
```

Caché de sesión

```
spring:
  session:
    store-type: redis
# If the following redis properties are not specified gapwalk.redis configuration will
be used for this cache
jics:
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
```

```
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Definiciones de recursos de JICS

```
jics:
   resource-definitions:
     store-type: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     maxTotal: 128                          # Optional
     maxIdle: 128                           # Optional
     minIdle: 16                            # Optional
     testOnBorrow: true                     # Optional
     testOnReturn: true                     # Optional
     testWhileIdle: true                    # Optional
     testOnCreate: true                     # Optional
     minEvictableIdleTimeMillis: 60000      # Optional
     timeBetweenEvictionRunsMillis: 30000   # Optional
     numTestsPerEvictionRun: -1             # Optional
     blockWhenExhausted: true               # Optional
     nettyThreads: 32                       # Optional
     subscriptionsPerConnection: 10         # Optional
     subscriptionConnectionPoolSize: 100    # Optional
     pageSizeInBytes: 8192                  # Optional
```

```
    readTimeout: 2000                       # Optional
```

Colas TS de JICS

```
jics:
   parameters:
     tsqimpl: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   queues:
     ts:
       redis:
         hostName: localhost
         port: 6379
         mode: standalone                       # Optional
         username:                              # Optional
         password: ""                           # Optional
         useSsl: false                          # Optional
         database: 0                            # Optional
         maxTotal: 128                          # Optional
         maxIdle: 128                           # Optional
         minIdle: 16                            # Optional
         testOnBorrow: true                     # Optional
         testOnReturn: true                     # Optional
         testWhileIdle: true                    # Optional
         testOnCreate: true                     # Optional
         minEvictableIdleTimeMillis: 60000      # Optional
         timeBetweenEvictionRunsMillis: 30000   # Optional
         numTestsPerEvictionRun: -1             # Optional
         blockWhenExhausted: true               # Optional
         nettyThreads: 32                       # Optional
         subscriptionsPerConnection: 10         # Optional
         subscriptionConnectionPoolSize: 100    # Optional
         pageSizeInBytes: 8192                  # Optional
         readTimeout: 2000                      # Optional
```

Rastreador de sesiones

```
session-tracker:
   store-type: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
```

```
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Puntos de comprobación de JCL

```
jcl:
   checkpoint:
     provider: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostname: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     maxTotal: 128                          # Optional
     maxIdle: 128                           # Optional
     minIdle: 16                            # Optional
     testOnBorrow: true                     # Optional
```

```
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Bloqueos de archivos de Gapwalk

```
filesLocks:
   enabled: true
   retryTime: 1000
   MaxRetry: 5
   provider: redis
 # If the following redis properties are not specified gapwalk.redis configuration will
 be used for this cache
   redis:
     hostName: localhost
     port: 6379
     mode: standalone                       # Optional
     username:                              # Optional
     password: ""                           # Optional
     useSsl: false                          # Optional
     database: 0                            # Optional
     pool:
       maxTotal: 128                        # Optional
       maxIdle: 128                         # Optional
       minIdle: 16                          # Optional
       testOnBorrow: true                   # Optional
       testOnReturn: true                   # Optional
       testWhileIdle: true                  # Optional
       testOnCreate: true                   # Optional
       minEvictableIdleTimeMillis: 60000    # Optional
       timeBetweenEvictionRunsMillis: 30000 # Optional
       numTestsPerEvictionRun: -1           # Optional
       blockWhenExhausted: true             # Optional
       nettyThreads: 32                     # Optional
```

```
      subscriptionsPerConnection: 10          # Optional
      subscriptionConnectionPoolSize: 100     # Optional
      pageSizeInBytes: 8192                   # Optional
      readTimeout: 2000                       # Optional
```

Bloqueos de Blu4iv

```
blu4iv.lock: redis
blu4iv.lock.timeout: 10 #(in millisecondes)
 # If the following redis properties are not specified gapwalk.redis configuration
will be used for this cache
blu4iv.lock.redis:
      hostName: localhost
      port: 6379
      mode: standalone                        # Optional
      username:                               # Optional
      password: ""                            # Optional
      useSsl: false                           # Optional
      database: 0                             # Optional
      maxTotal: 128                           # Optional
      maxIdle: 128                            # Optional
      minIdle: 16                             # Optional
      testOnBorrow: true                      # Optional
      testOnReturn: true                      # Optional
      testWhileIdle: true                     # Optional
      testOnCreate: true                      # Optional
      minEvictableIdleTimeMillis: 60000       # Optional
      timeBetweenEvictionRunsMillis: 30000    # Optional
      numTestsPerEvictionRun: -1              # Optional
      blockWhenExhausted: true                # Optional
      nettyThreads: 32                        # Optional
      subscriptionsPerConnection: 10          # Optional
      subscriptionConnectionPoolSize: 100     # Optional
      pageSizeInBytes: 8192                   # Optional
      readTimeout: 2000                       # Optional
```

Catálogo de conjuntos de datos

```
datasimplifier:
   catalogImplementation: redis
   # If the following redis properties are not specified gapwalk.redis configuration
 will be used for this cache
```

```
  redis:
    hostName: localhost
    port: 6379
    mode: standalone                        # Optional
    username:                               # Optional
    password: ""                            # Optional
    useSsl: false                           # Optional
    database: 0                             # Optional
    maxTotal: 128                           # Optional
    maxIdle: 128                            # Optional
    minIdle: 16                             # Optional
    testOnBorrow: true                      # Optional
    testOnReturn: true                      # Optional
    testWhileIdle: true                     # Optional
    testOnCreate: true                      # Optional
    minEvictableIdleTimeMillis: 60000       # Optional
    timeBetweenEvictionRunsMillis: 30000    # Optional
    numTestsPerEvictionRun: -1              # Optional
    blockWhenExhausted: true                # Optional
    nettyThreads: 32                        # Optional
    subscriptionsPerConnection: 10          # Optional
    subscriptionConnectionPoolSize: 100     # Optional
    pageSizeInBytes: 8192                   # Optional
    readTimeout: 2000                       # Optional
```

Administrador de secretos de las cachés de Redis

El archivo `application-main.yaml` puede hacer referencia al ARN secreto de las cachés de Redis. Para obtener información sobre cómo realizar la integración AWS Secrets Manager para recuperar de forma segura los detalles de la conexión de Redis en tiempo de ejecución, consultethe section called “AWS Secretos de Blu Age Runtime”.

Configuración de la seguridad para las aplicaciones de Gapwalk

En los siguientes temas se describe cómo administrar las aplicaciones de Gapwalk.

Es su responsabilidad proporcionar la configuración correcta para garantizar que el uso del marco de AWS Blu Age sea seguro.

Todas las características relacionadas con la seguridad están deshabilitadas de manera predeterminada. Para habilitar la autenticación (y CSRF, XSS, CSP, etc.), establezca `gapwalk-`

`application.security` en `enabled` y `gapwalk-application.security.identity` en `oauth`.

Temas

- Configuración de la accesibilidad de URI para las aplicaciones de Gapwalk
- Configurar la autenticación para las aplicaciones de Gapwalk

Configuración de la accesibilidad de URI para las aplicaciones de Gapwalk

En este tema se describe cómo configurar el filtrado de las aplicaciones URIs de Gapwalk. Esta característica no requiere un nuevo proveedor de identidades (IdP).

Para bloquear una lista de URIs, añada las dos líneas siguientes a la `application-main.yml` de su aplicación modernizada*URI-1*, *URI-2* sustituyéndolas, etc., por las URIs que desee bloquear.

```
gapwalk-application.security.filterURIs: enabled
gapwalk-application.security.blockedURIs: URI-1, URI-2, URI-3
```

Configurar la autenticación para las aplicaciones de Gapwalk

Para configurar la OAuth2 autenticación de su aplicación Gapwalk, debe configurar un proveedor de identidad (IdP) e integrarlo con su aplicación. En esta guía se describen los pasos para utilizar Amazon Cognito o Keycloak como IdP. Con Amazon Cognito, puede actualizar el archivo de configuración de la aplicación con los detalles del grupo de usuarios de Cognito. Con Keycloak, puedes controlar el acceso a tu aplicación APIs y a los recursos en función de las funciones asignadas al usuario.

Temas

- Configurar la OAuth2 autenticación de Gapwalk con Amazon Cognito
- Configura la autenticación de Gapwalk con Keycloak OAuth2

Configurar la OAuth2 autenticación de Gapwalk con Amazon Cognito

En este tema se describe cómo configurar la OAuth2 autenticación para las aplicaciones de Gapwalk que utilizan Amazon Cognito como proveedor de identidad (IdP).

Requisitos previos

En este tutorial utilizaremos Amazon Cognito como IdP y PlanetDemo como proyecto modernizado.

Puede utilizar cualquier otro proveedor de identidades externo. La ClientRegistration información debe obtenerse de su IDP y es necesaria para la autenticación de Gapwalk. Para obtener más información, consulte la Guía para desarrolladores de Amazon Cognito.

La información: ClientRegistration

client-id

El ID de la ClientRegistration. En nuestro ejemplo, lo será PlanetsDemo.

client-secret

Su secreto de cliente.

authorization endpoint

El URI del punto de conexión de autorización para el servidor de autorizaciones.

token endpoint

El URI del punto de conexión del token para el servidor de autorización.

jwks endpoint

El URI utilizado para obtener la clave web JSON (JWK) que contiene las claves para validar la firma web JSON emitida por el servidor de autorización.

redirect URI

URI al que el servidor de autorización redirige al usuario final si se le concede acceso.

Configuración de Amazon Cognito

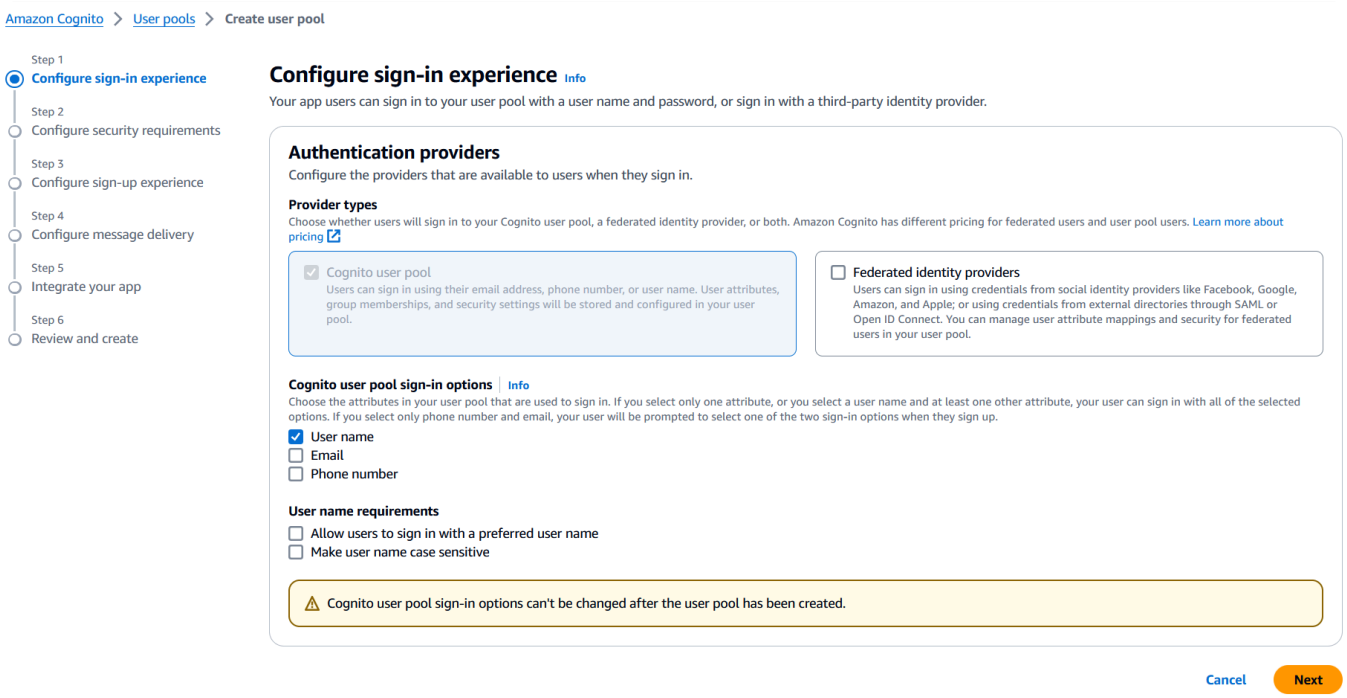
En primer lugar, crearemos y configuraremos un usuario y un grupo de usuarios de Amazon Cognito que utilizaremos con nuestra aplicación de Gapwalk implementada con fines de prueba.

> **Note**
>
> Si utiliza otro IdP, puede saltarse este paso.

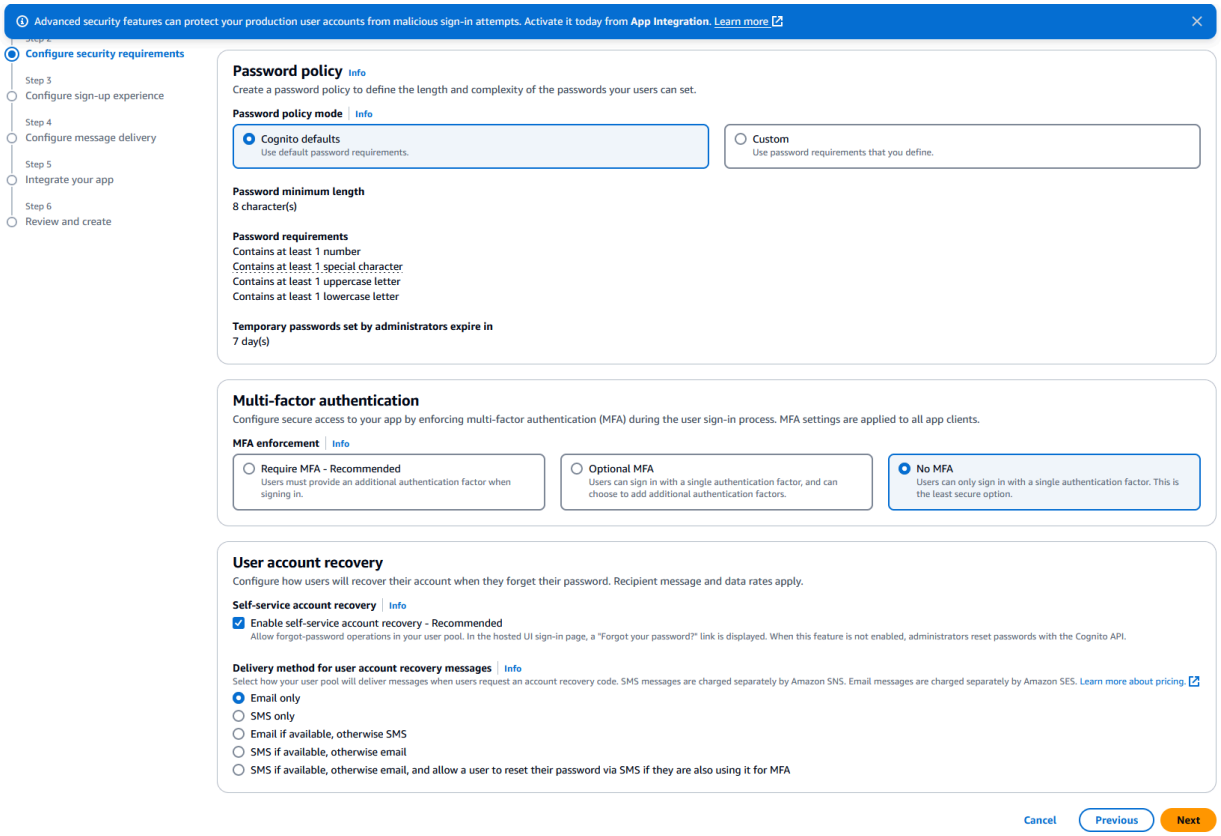
Creación de un grupo de usuarios

1. Vaya a Amazon Cognito en AWS Management Console y autentíquese con sus credenciales. AWS

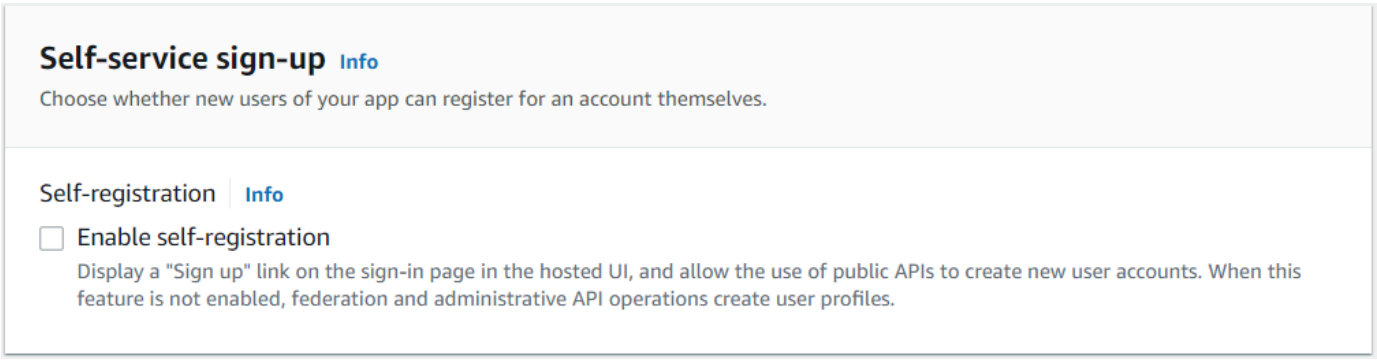
2. Elija Grupos de usuarios.
3. Elija Crear un grupo de usuarios.
4. En Configurar la experiencia de inicio de sesión, mantenga el tipo de proveedor predeterminado Grupo de usuarios de Cognito. Puede elegir una o varias Opciones de inicio de sesión del grupo de usuarios de Cognito; por ahora, elija Nombre de usuario y, a continuación, Siguiente.



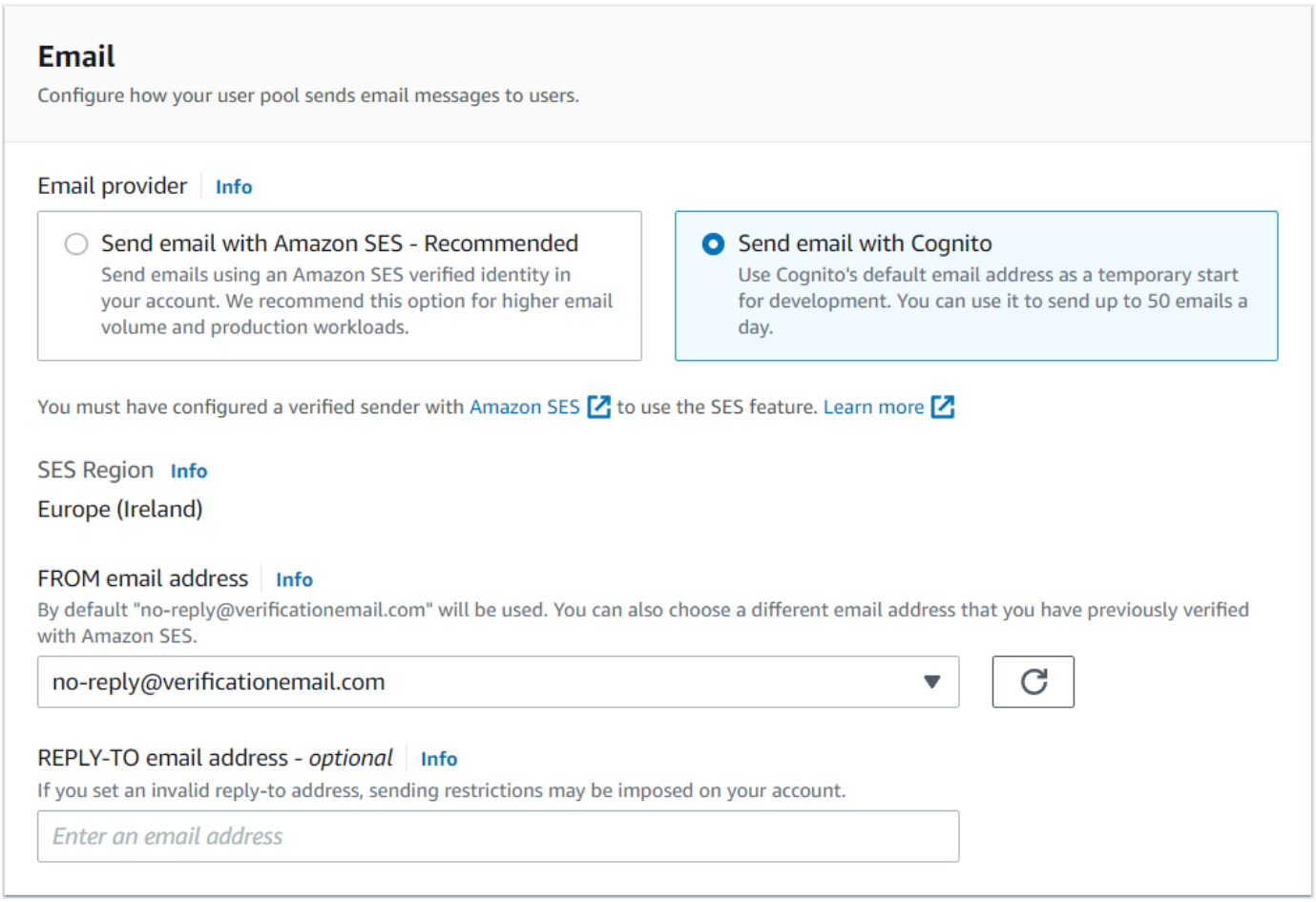
5. En Configurar los requisitos de seguridad, mantenga los valores predeterminados y deshabilite Autenticación multifactor seleccionando Sin MFA y, a continuación, elija Siguiente.



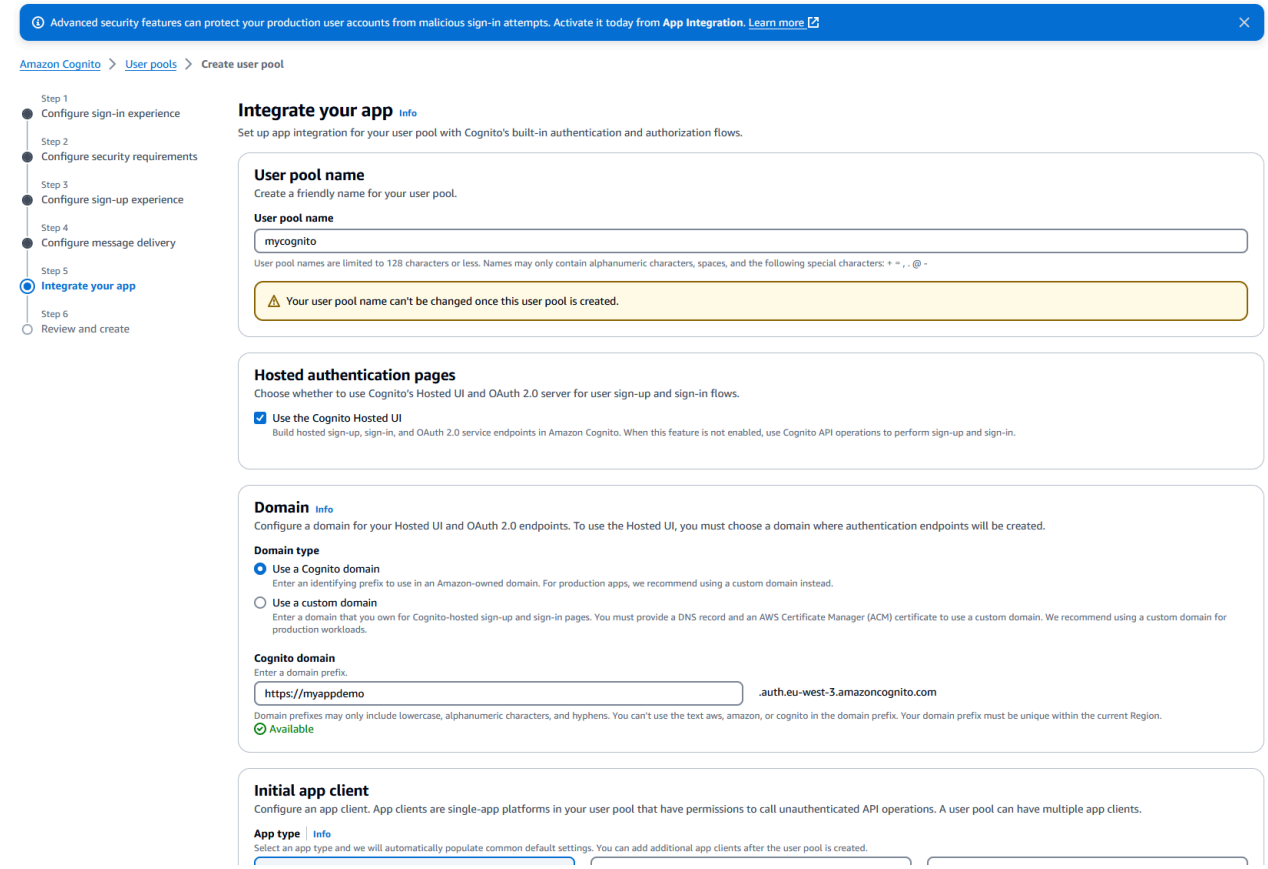
6. Como seguridad, deshabilite Habilitar el registro automático y, a continuación, seleccione Siguiente.



7. Elija Enviar correo electrónico con Cognito y, a continuación, elija Siguiente.



8. En Integrar su aplicación, especifique un nombre para su grupo de usuarios. En Páginas de autenticación alojadas, elija Usar la interfaz de usuario alojada de Cognito.



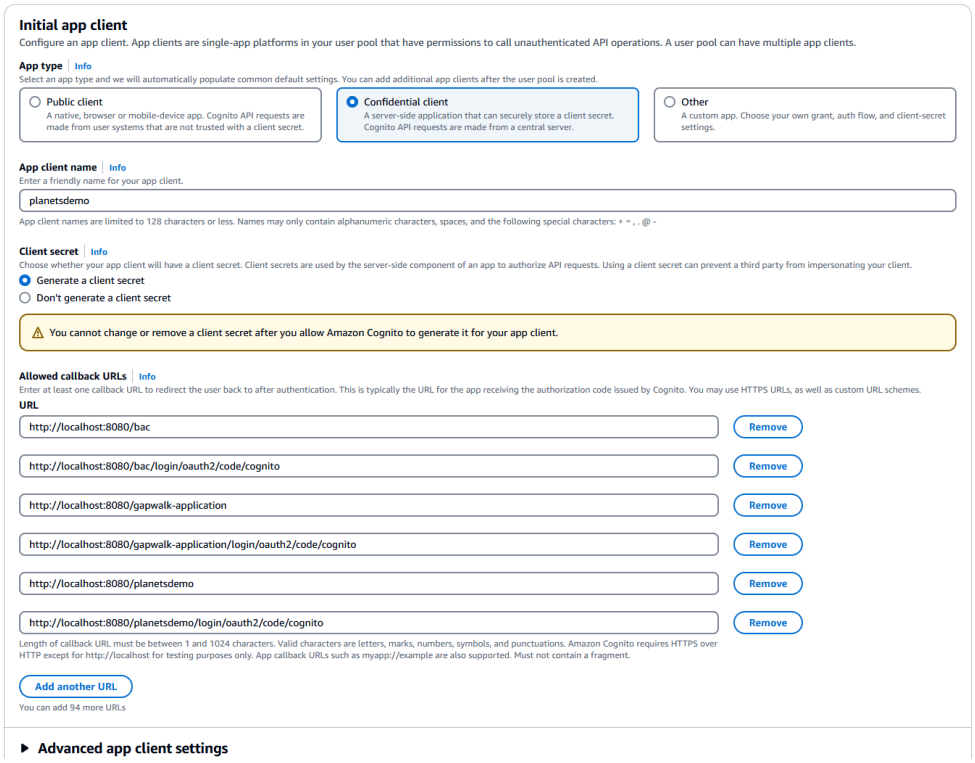
9. Para simplificar, en Dominio, elija Utilizar un dominio de Cognito e introduzca un prefijo de dominio; por ejemplo, `https://planetsdemo`. La aplicación de demostración debe añadirse como cliente.

   a. En Cliente de aplicación inicial, elija Cliente confidencial. Introduzca un nombre de cliente de aplicación, por ejemplo **`planetsdemo`**, y, a continuación, elija Generar un secreto de cliente.

   b. En URL de devolución de llamadas permitidas, introduzca la URL a la que redirigir al usuario tras la autenticación. La URL debe finalizar con `/login/oauth2/code/cognito`. Por ejemplo, para nuestra aplicación y aplicaciones de backend de Gapwalk y BAC:

```
http://localhost:8080/bac
      http://localhost:8080/bac/login/oauth2/code/cognito
      http://localhost:8080/gapwalk-application
      http://localhost:8080/gapwalk-application/login/oauth2/code/cognito
      http://localhost:8080/planetsdemo
      http://localhost:8080/planetsdemo/login/oauth2/code/cognito
```

   Puede editar la URL más adelante.



c. En Permitido cierre de sesión, URLs introduzca la URL de la página de cierre de sesión a la que desea que Amazon Cognito redirija cuando la aplicación cierre sesión de los usuarios. Por ejemplo, para las aplicaciones de backend de Gapwalk y BAC:

```
http://localhost:8080/bac/logout
http://localhost:8080/gapwalk-application/logout
http://localhost:8080/planetsdemo/logout
```

Puede editar la URL más adelante.

d. Mantenga los valores predeterminados de las secciones Configuración avanzada del cliente de aplicación y Permisos avanzados de lectura y escritura de atributos.

e. Elija Siguiente.

10. En Revisar y crear, verifique sus opciones y, a continuación, elija Crear un grupo de usuarios.

Para obtener más información, consulte Creación de un grupo de usuarios.

Creación de usuarios

Como el registro automático está deshabilitado, cree un usuario de Amazon Cognito. Vaya a la consola de Amazon Cognito. Elija el grupo de usuarios que ha creado y, a continuación, en Usuarios, elija Crear usuario.

En Información del usuario, elija Enviar una invitación por email, introduzca un nombre de usuario y una dirección de correo electrónico y, a continuación, seleccione Generar una contraseña. Seleccione la opción Crear usuario.

Creación de roles

En la pestaña Grupos, cree 3 grupos (SUPER_ADMIN, ADMIN y USER) y asocie su usuario a uno o más de estos grupos. Posteriormente, la aplicación de Gapwalk asigna estos roles a ROLE_SUPER_ADMIN, ROLE_ADMIN y ROLE_USER para poder acceder a algunas llamadas API REST restringidas.

Integración de Amazon Cognito en la aplicación de Gapwalk

Ahora que el grupo de usuarios y el usuario de Amazon Cognito están preparados, vaya al archivo `application-main.yml` de la aplicación modernizada y agregue el siguiente código:

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: https://cognito-idp.<region-id>.amazonaws.com/
<pool-id>
gapwalk-application.security.domainName: <your-cognito-domain>
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          cognito:
            client-id: <client-id>
            client-name: <client-name>
            client-secret: <client-secret>
            provider: cognito
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "<redirect-uri>"
        provider:
          cognito:
            issuer-uri: ${gapwalk-application.security.issuerUri}
```

```
            authorization-uri: ${gapwalk-application.security.domainName}/oauth2/
authorize
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/
jwks.json
            token-uri: ${gapwalk-application.security.domainName}/oauth2/token
            user-name-attribute: username
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/.well-known/jwks.json
```

Reemplace los siguientes marcadores de posición como se indica:

1. Vaya a Amazon Cognito en AWS Management Console y autentíquese con sus credenciales. AWS
2. Elija Grupos de usuarios y el grupo de usuarios que ha creado. Puede encontrar su ID *`pool-id`* en el grupo de usuarios.
3. Selecciona Integración de aplicaciones, donde encontrarás la tuya*`your-cognito-domain`*, y luego ve a Clientes y análisis de aplicaciones y elige tu aplicación.
4. En el cliente de aplicaciones: YourApp, puedes encontrar las opciones *`client-nameclient-id`*, y *`client-secret`* (Mostrar el secreto del cliente).
5. *`region-id`*corresponde al ID de AWS región en el que creó su usuario y grupo de usuarios de Amazon Cognito. Ejemplo: `eu-west-3`.
6. Para, *`redirect-uri`* introduzca el URI que especificó para la URL de devolución de llamada permitida. En nuestro ejemplo, es `http://localhost:8080/planetsdemo/login/oauth2/code/cognito`.

Ahora puede implementar su aplicación de Gapwalk y usar el usuario creado anteriormente para iniciar sesión en su aplicación.

Configura la autenticación de Gapwalk con Keycloak OAuth2

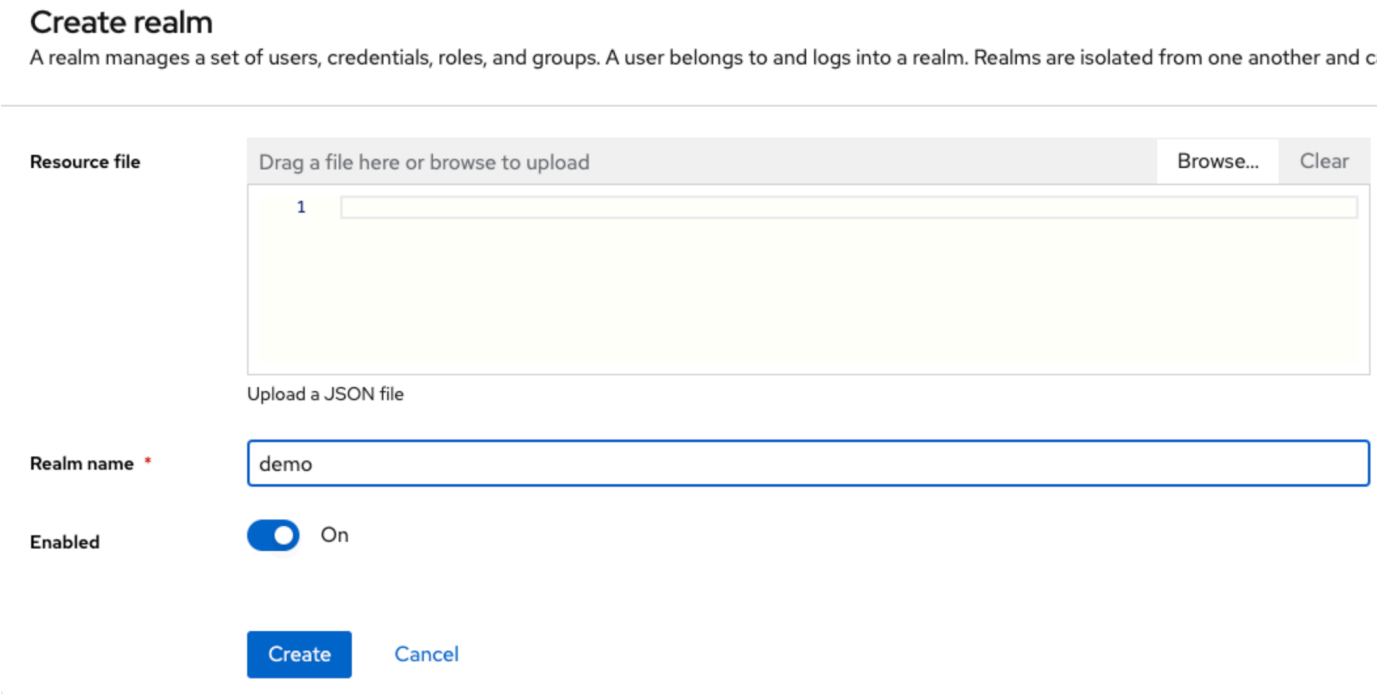
En este tema se describe cómo configurar la OAuth2 autenticación para las aplicaciones de Gapwalk que utilizan Keycloak como proveedor de identidad (IdP). En este tutorial, utilizaremos 24.0.0.

Requisitos previos

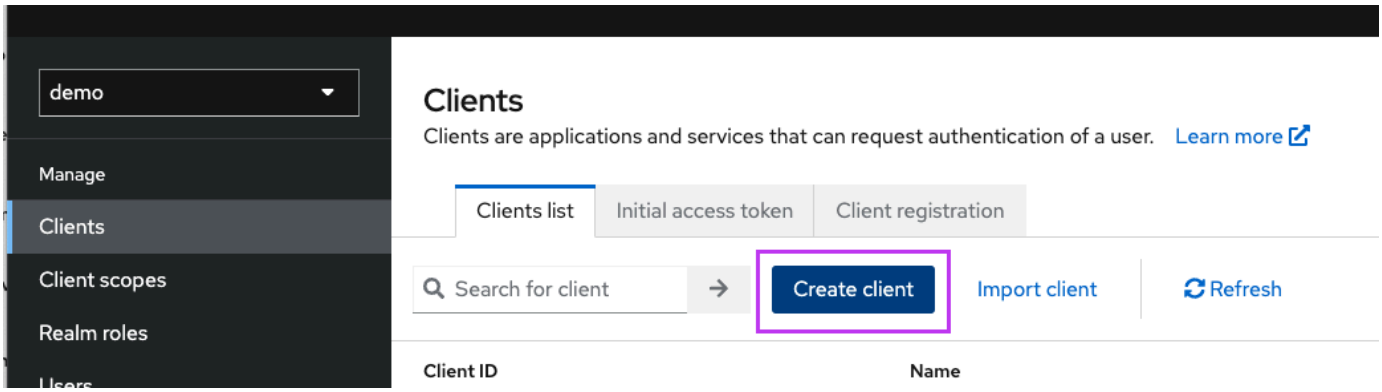
- Keycloak
- Aplicación Gapwalk

Configuración de Keycloak

1. Vaya al panel de Keycloak en el navegador web. Las credenciales predeterminadas son admin/admin. Vaya a la barra de navegación superior izquierda y cree un dominio con el nombre **`demo`**, tal como se muestra en la imagen siguiente.



2. Cree un cliente con el nombre **`app-demo`**.



Reemplace `localhost:8080` por la dirección de su aplicación de Gapwalk

General settings

Client ID * app-demo

Name

Description

Always display in UI Off

Access settings

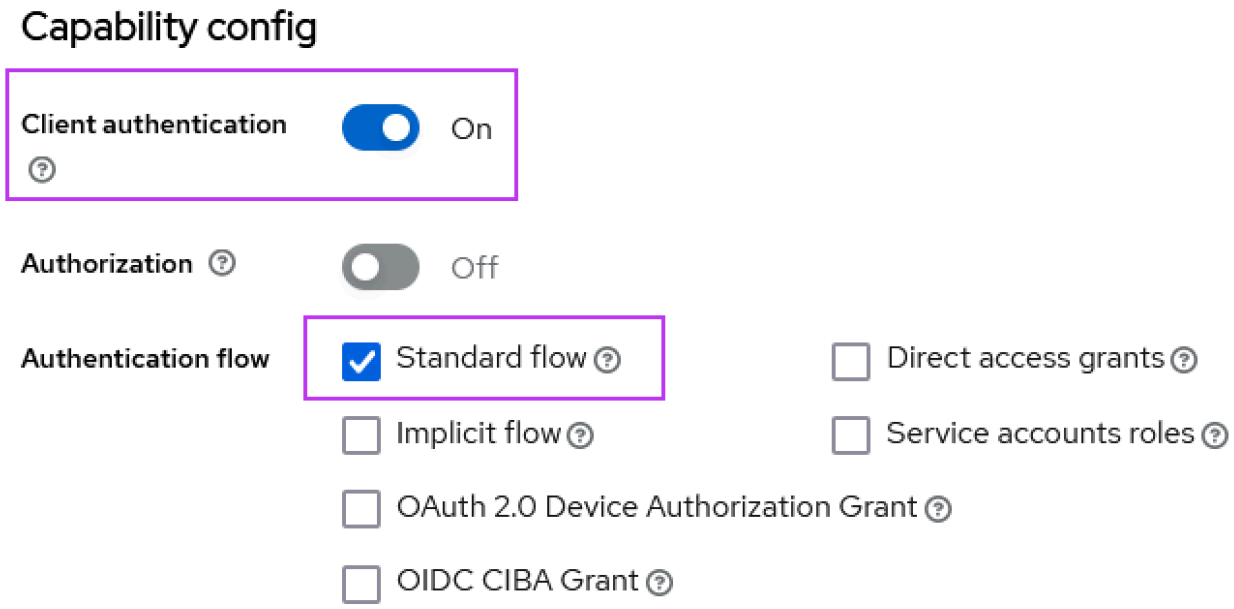
Root URL http://localhost:8080

Home URL

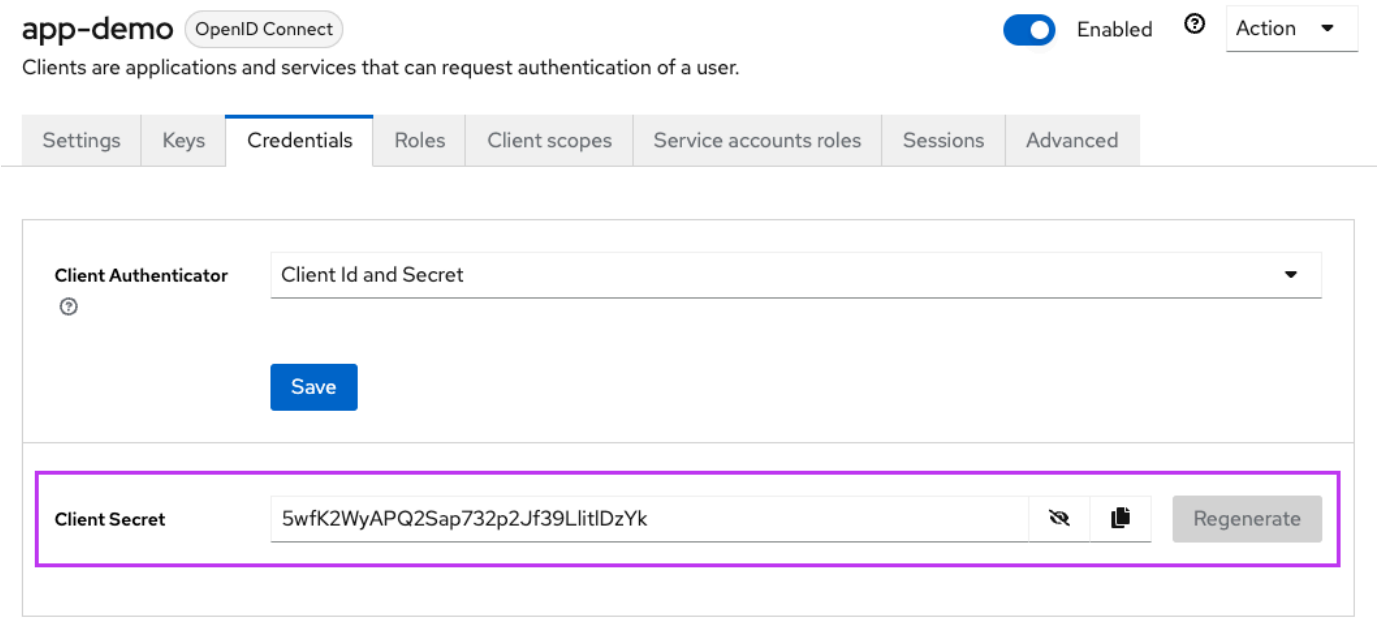
Valid redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*
- Add valid redirect URIs

Valid post logout redirect URIs
- http://localhost:8080/*
- https://localhost:8080/*
- Add valid post logout redirect URIs

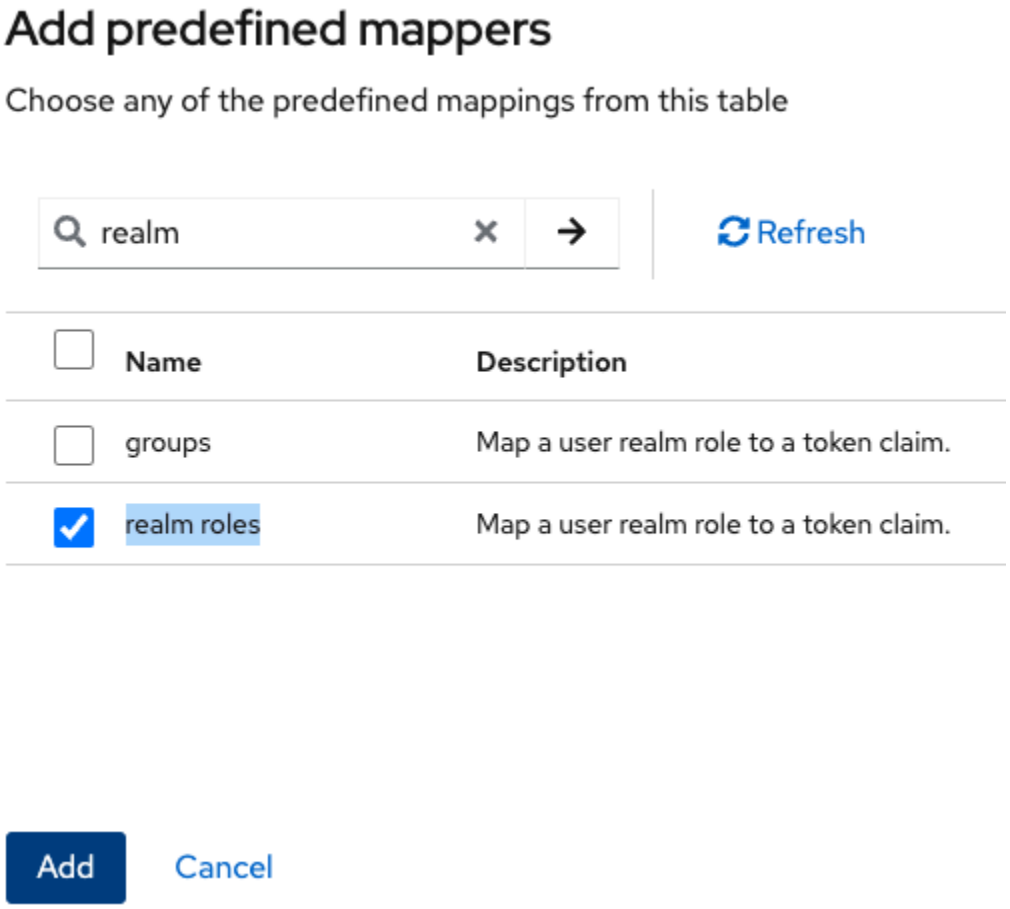
Web origins
- +
- Add web origins



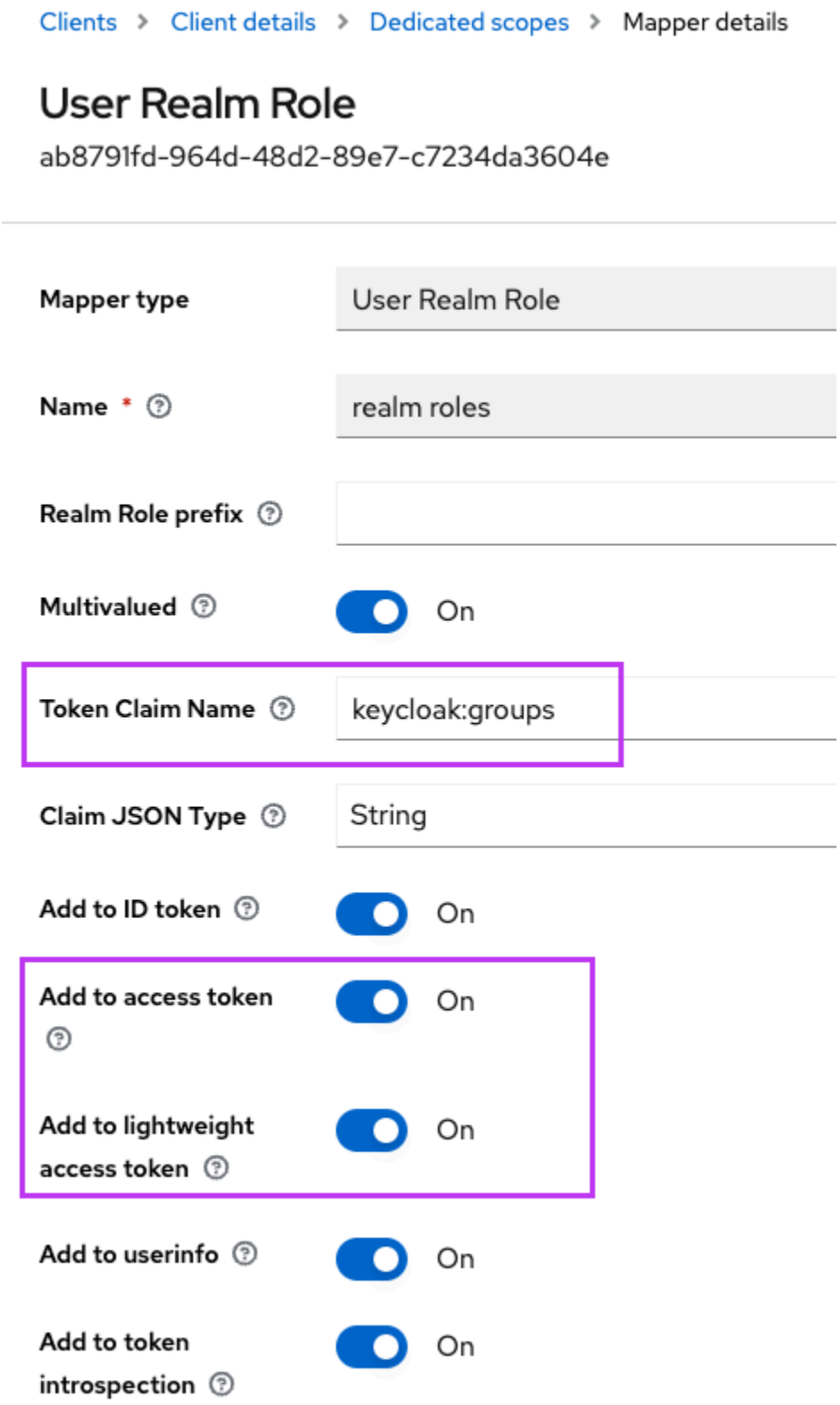
3. Para obtener su secreto de cliente, elija Clientes, app-demo y Credenciales.



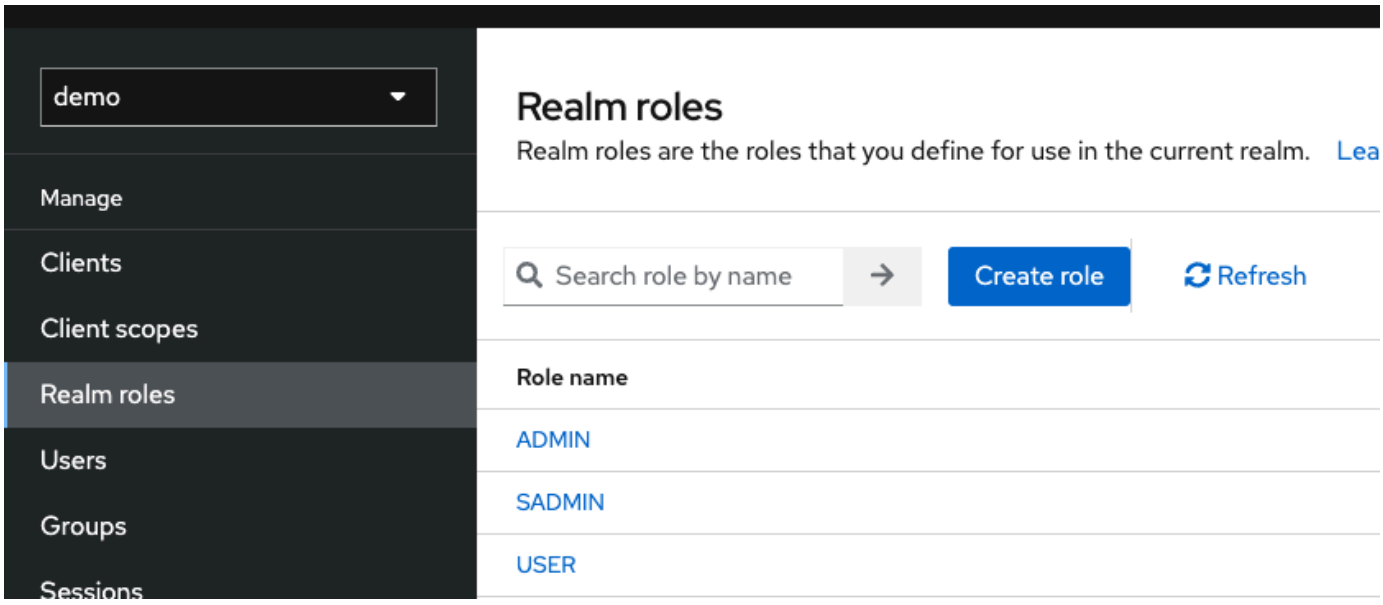
4. Elija Clientes, Ámbitos de clientes y Agregar asignador predefinido. Elija los roles de dominio.



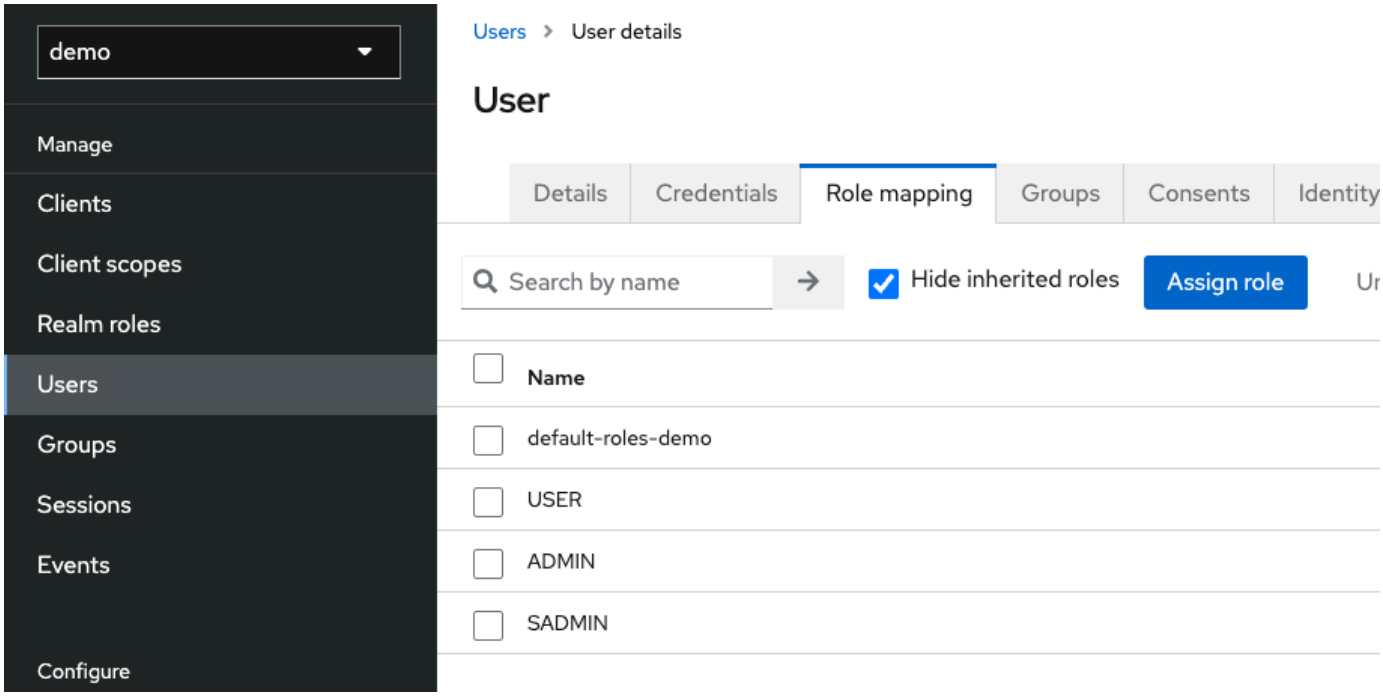
5. Edite su rol de dominio con la configuración que se muestra en la imagen siguiente.


Clients > Client details > Dedicated scopes > Mapper details
User Realm Role
ab8791fd-964d-48d2-89e7-c7234da3604e
Mapper type
User Realm Role
Name *
realm roles
Realm Role prefix
Multivalued
On
Token Claim Name
keycloak:groups
Claim JSON Type
String
Add to ID token
On
Add to access token
On
Add to lightweight access token
On
Add to userinfo
On
Add to token introspection
On

6. Recuerde el Nombre de notificación de token definido. Necesitará este valor en la definición de configuración de Gapwalk para la propiedad `gapwalk-application.security.claimGroupName`.



7. Elige Roles de dominios y cree 3 roles: **`SUPER_ADMIN`**, **`ADMIN`** y **`USER`**. Estos roles se asignan posteriormente a `ROLE_SUPER_ADMIN`, `ROLE_ADMIN` y `ROLE_USER` por la aplicación de Gapwalk para poder acceder a algunas llamadas API de REST restringidas.



Integración de Keycloak en la aplicación de Gapwalk

Edite el archivo `application-main.yml` de la siguiente manera:

```
gapwalk-application.security: enabled
gapwalk-application.security.identity: oauth
gapwalk-application.security.issuerUri: http://<KEYCLOAK_SERVER_HOSTNAME>/realms/
<YOUR_REALM_NAME>
gapwalk-application.security.claimGroupName: "keycloak:groups"

gapwalk-application.security.userAttributeName: "preferred_username"
# Use "username" for cognito,
#     "preferred_username" for keycloak
#      or any other string
gapwalk-application.security.localhostWhitelistingEnabled: false

spring:
  security:
    oauth2:
      client:
        registration:
          demo:
            client-id: <YOUR_CLIENT_ID>
            client-name: Demo App
            client-secret: <YOUR_CLIENT_SECRET>
            provider: keycloak
            authorization-grant-type: authorization_code
            scope: openid
            redirect-uri: "{baseUrl}/login/oauth2/code/{registrationId}"
        provider:
          keycloak:
            issuer-uri: ${gapwalk-application.security.issuerUri}
            authorization-uri: ${gapwalk-application.security.issuerUri}/protocol/
openid-connect/auth
            jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
            token-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/token
            user-name-attribute: ${gapwalk-application.security.userAttributeName}
      resourceserver:
        jwt:
          jwk-set-uri: ${gapwalk-application.security.issuerUri}/protocol/openid-
connect/certs
```

Sustituya*<KEYCLOAK_SERVER_HOSTNAME>*, *<YOUR_REALM_NAME><YOUR_CLIENT_ID>*, y *<YOUR_CLIENT_SECRET>* por el nombre de host del servidor Keycloak, el nombre de dominio, el ID de cliente y el secreto de cliente.

AWS Tiempo de ejecución Blu Age APIs

El AWS Blu Age Runtime utiliza varias aplicaciones web para exponer los puntos finales REST, lo que proporciona formas de interactuar con las aplicaciones modernizadas mediante clientes REST (por ejemplo, para llamar a las tareas mediante un programador).

El propósito de este documento es enumerar los puntos de conexión REST disponibles, proporcionando detalles sobre:

- Su rol
- La forma de utilizarlos correctamente

La lista de puntos de conexión está organizada en categorías, según la naturaleza del servicio prestado y la aplicación web que muestre los puntos de conexión.

Suponemos que ya tiene conocimientos básicos sobre el uso de puntgos de conexión REST (utilizando herramientas específicas como POSTMAN, Thunder Client, CURL, navegadores web, etc.) o escribir su propio código para realizar una llamada a la API.

Temas

- Puntos finales disponibles para el usuario al construir URLs
- Puntos finales para la aplicación Gapwalk en Blu Age AWS
- Puntos de conexión REST de la consola de aplicaciones de Blusam
- Administre la consola de aplicaciones JICS en AWS Blu Age
- Estructuras de datos para usuarios de AWS Blu Age

Puntos finales disponibles para el usuario al construir URLs

En este tema se enumeran URLs las rutas raíz de los puntos finales. Cada aplicación web que aparece a continuación define una ruta raíz, compartida por todos los puntos de conexión. Luego, cada punto de conexión agrega su propia ruta dedicada. La URL resultante que se utilizará es el

resultado de la concatenación de las rutas. Por ejemplo, si consideramos el primer punto de conexión de la aplicación Gapwalk, tenemos:

- `/gapwalk-application` para la ruta raíz de la aplicación web.
- `/scripts` para la ruta de punto de conexión dedicada.

La URL resultante que se utilizará será `http://server:port/gapwalk-application/scripts`

server

apunta al nombre del servidor (el que aloja la aplicación web en cuestión).

port

el puerto expuesto por el servidor.

Puntos finales para la aplicación Gapwalk en Blu Age AWS

En este tema, obtenga más información sobre los puntos de conexión de la aplicación web Gapwalk. Utilizan la ruta raíz `/gapwalk-application`.

Temas

- Terminales relacionados con trabajos por lotes (modernizados JCLs y similares)
- Métricas para puntos de conexión
- Otros puntos de conexión
- Puntos de conexión relacionados con las colas de trabajos

Terminales relacionados con trabajos por lotes (modernizados JCLs y similares)

Los trabajos por lotes se pueden ejecutar de forma sincrónica o asincrónica (consulte los detalles a continuación). Los trabajos por lotes se ejecutan mediante scripts groovy que son el resultado de la modernización de los scripts heredados (JCL).

Temas

- Enumere los scripts implementados
- Lanzar un script de forma sincrónica

- Lanzar un script de forma asincrónica
- Listado de scripts activados
- Recuperar los detalles de la ejecución del trabajo
- Lista los scripts lanzados de forma asincrónica que se pueden eliminar
- Lista los scripts lanzados de forma sincrónica que se pueden eliminar
- Eliminar una ejecución de trabajo determinada
- Listado de los puntos de comprobación existentes para la reiniciabilidad
- Reiniciar un trabajo (de forma sincrónica)
- Reiniciar un trabajo (de forma asincrónica)
- Establecer el límite de subprocesos para la ejecución de trabajos asincrónica

Enumere los scripts implementados

- Método compatible: GET
- Ruta: `/scripts`
- Argumentos: ninguno
- Este punto de conexión devuelve la lista de scripts groovy desplegados en el servidor, en forma de cadena. Este punto de conexión está diseñado principalmente para ser utilizado desde un navegador web, ya que la cadena resultante es una página HTML, con enlaces activos (un enlace por script que se pueda iniciar; consulte el ejemplo siguiente).

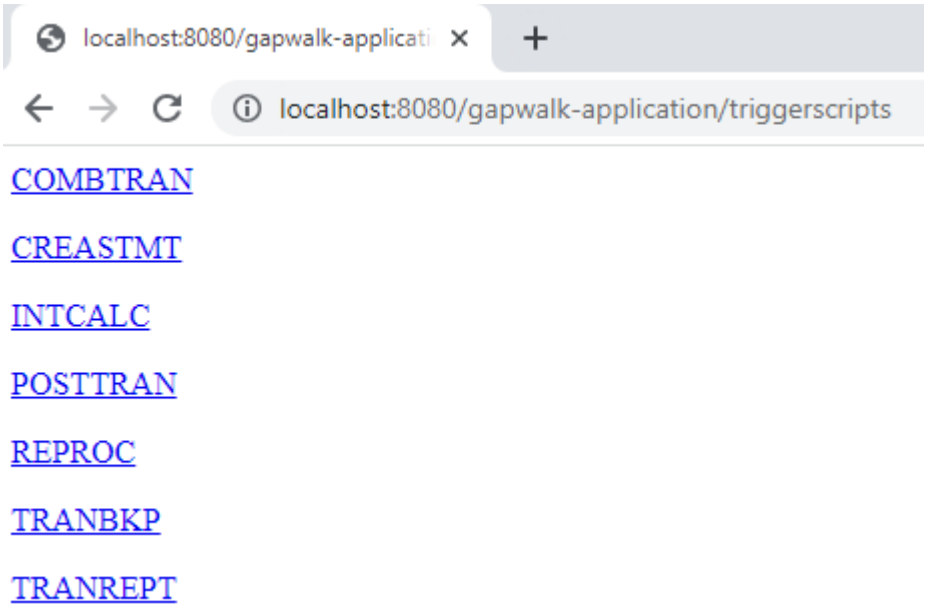
Respuesta de ejemplo:

```
<p><a href=./script/COMBTRAN>COMBTRAN</a></p><p><a href=./script/CREASTMT>CREASTMT</
a></p><p><a href=./script/INTCALC>INTCALC</a></p><p><a href=./script/
POSTTRAN>POSTTRAN</a></p><p><a href=./script/REPROC>REPROC</a></p><p><a href=./script/
TRANBKP>TRANBKP</a></p><p><a href=./script/TRANREPT>TRANREPT</a></p><p><a href=./
script/functions>functions</a></p>
```

> **Note**
>
> Los enlaces representan la URL que se utilizará para lanzar cada script de la lista de forma sincrónica.

- Método compatible: GET
- Ruta: `/triggerscripts`
- Argumentos: ninguno
- Este punto de conexión devuelve la lista de scripts groovy desplegados en el servidor, en forma de cadena. Este punto de conexión está diseñado principalmente para ser utilizado desde un navegador web, ya que la cadena resultante es una página HTML, con enlaces activos (un enlace por script que se pueda iniciar; consulte el ejemplo siguiente).

  A diferencia de la respuesta anterior del punto de conexión, los enlaces representan la URL que se debe utilizar para lanzar cada script de la lista de forma asincrónica.



Lanzar un script de forma sincrónica

Este punto de conexión tiene dos variantes con rutas dedicadas para el uso de GET y POST (ver más abajo).

- Método compatible: GET
- Ruta: `/script/{scriptId:.+}`
- Método compatible: POST
- Ruta: `/post/script/{scriptId:.+}`
- Argumentos:
  - identificador del script que se va a lanzar

- opcionalmente: parámetros para pasarlos al script, utilizando los parámetros de solicitud (vistos como un `Map<String,String>`). Los parámetros dados se añadirán automáticamente a los enlaces del script groovy invocado.

- La llamada lanzará el script con el identificador dado, utilizando parámetros adicionales si se proporcionan y esperará a que se complete la ejecución del script antes de devolver un mensaje (`String`) que será:
  - "Done" (si la ejecución del trabajo se realizó sin problemas).
  - Un mensaje de error de JSON con detalles sobre lo que ha fallado durante la ejecución del trabajo. Se pueden obtener más detalles de los registros del servidor para saber qué ha fallado en la ejecución del trabajo.

```
{
    "exitCode": -1,
    "stepName": "STEP15",
    "program": "CBACT04C",
    "status": "Error"
}
```

Al observar los registros del servidor, podemos darnos cuenta de que se trata de un problema de implementación (el programa esperado no se ha implementado correctamente, por lo que no se puede encontrar, lo que provoca un error en la ejecución del trabajo):

```
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - --> executing script INTCALC
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.BatchWebController          - Bound jobContext 419695287 - GDGEventsQueueHandler :907380469
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Added jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] to Sync Script Control Tower.
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - a65c2791-864f-43c9-972a-b5f2353389e6 - worker :Thread-26 [1547512424]
2023-06-09 10:27:28 default        INFO  - c.n.b.g.r.j.s.JobExecutor               - Triggered script: INTCALC - [a65c2791-864f-43c9-972a-b5f2353389e6] - jobContext [419695287]
2023-06-09_10-27-29-613 | [JOB] INTCALC - Started
2023-06-09_10-27-29-651 | [STEP] STEP15 - Started
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09 10:27:29 default        ERROR - c.n.b.g.r.c.i.ExecutionControllerImpl    - Could not find program "CBACT04C" in the program registry.
2023-06-09_10-27-29-760 | Program not found => not executed !
2023-06-09_10-27-29-761 | [STEP] STEP15 - Ended
2023-06-09_10-27-29-772 | [JOB] INTCALC - Ended
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - Job [419695287] - starting final operation
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.j.s.DefaultJobContext         - End of job [419695287]
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Removed jobExecutor [a65c2791-864f-43c9-972a-b5f2353389e6] from Script Control Tower.
2023-06-09 10:27:29 default        INFO  - c.n.b.g.r.s.ScriptControlTower          - Remaining jobExecutors:0
```

> **Note**
>
> Las llamadas sincrónicas deben reservarse para tareas de corta duración. Los trabajos que se ejecutan durante mucho tiempo deberían lanzarse más bien de forma asincrónica (consulte el punto de conexión específico a continuación).

Lanzar un script de forma asincrónica

- Métodos compatibles: GET/POST
- Ruta: `/triggerscript/{scriptId:.+}`
- Argumentos:
  - identificador del script que se va a lanzar
  - opcionalmente: parámetros para pasarlos al script, utilizando los parámetros de solicitud (vistos como un `Map<String,String>`). Los parámetros indicados se añadirán automáticamente al archivo https://docs.groovy-lang.org/latest/html/api/groovy/lang/Binding.html [bindings] del groovy script invocado.
- A diferencia del modo sincrónico anterior, el punto de conexión no espera a que finalice la ejecución del trabajo para enviar una respuesta. La ejecución de la tarea se inicia al mismo tiempo, si se encuentra un subproceso disponible para hacerlo, y se envía inmediatamente una respuesta al programa que hace la llamada, con el identificador de ejecución de la tarea, un identificador único que representa la ejecución de la tarea, que se puede utilizar para consultar el estado de la ejecución de la tarea o forzar la finalización de una ejecución de tareas que se supone que no funciona correctamente. El formato de la respuesta es:

  ```
  Triggered script <script identifier> [unique job execution id] @ <date and time>
  ```

- Dado que la ejecución asincrónica del trabajo se basa en un número fijo y limitado de subprocesos, es posible que la ejecución del trabajo no se inicie si no se encuentra ningún subproceso disponible. En ese caso, el mensaje devuelto se ve del siguiente modo:

  ```
  Script [<script identifier>] NOT triggered - Thread limit reached (<actual thread
   limit>) - Please retry later or increase thread limit.
  ```

  Consulte el siguiente punto de conexión `settriggerthreadlimit` para obtener información sobre cómo aumentar el límite de subprocesos.

Respuesta de ejemplo:

```
Triggered script INTCALC [d43cbf46-4255-4ce2-aac2-79137573a8b4] @ 06-12-2023 16:26:15
```

El identificador único de ejecución de tareas permite recuperar rápidamente las entradas de registro relacionadas en los registros del servidor, si es necesario. También lo utilizan varios otros puntos de conexión que se detallan a continuación.

Listado de scripts activados

- Método compatible: GET
- Rutas: `/triggeredscripts/{status:.+}`, `/triggeredscripts/{status:.+}/{namefilter}`
- Argumentos:
  - Estado (obligatorio): el estado de los scripts activados para su recuperación. Los valores posibles son los siguientes:
    - `all`: muestra todos los detalles de la ejecución de los trabajos, independientemente de si los trabajos siguen ejecutándose o no.
    - `running`: muestra solo los detalles de los trabajos que se están ejecutando actualmente.
    - `done`: muestra solo los detalles de los trabajos cuya ejecución ha finalizado.
    - `killed`: solo muestra los detalles de los trabajos cuya ejecución se ha interrumpido forzosamente mediante el punto final específico (ver más abajo).
    - `triggered`: muestra solo los detalles de los trabajos que se han activado pero que aún no se han lanzado.
    - `failed`: muestra solo los detalles de los trabajos cuya ejecución se ha marcado como fallida.
    - _namefilter (opcional) _: recupera solo las ejecuciones del identificador de script dado.
- Devuelve una colección de detalles de las ejecuciones de los trabajos en formato JSON. Para obtener más información, consulte Estructura de mensajes de detalles de ejecución del trabajo.

Respuesta de ejemplo:

```
[
    {
      "scriptId": "INTCALC",
      "caller": "127.0.0.1",
      "identifier": "d43cbf46-4255-4ce2-aac2-79137573a8b4",
      "startTime": "06-12-2023 16:26:15",
      "endTime": "06-12-2023 16:26:15",
      "status": "DONE",
```

```
    "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
\"CBACT04C\", \"status\": \"Error\" }",
    "executionMode": "ASYNCHRONOUS"
  }
 ]
```

Recuperar los detalles de la ejecución del trabajo

- Método compatible: GET
- Ruta: `/getjobexecutioninfo/{jobexecutionid:.+}`
- Argumentos:
  - jobexecutionid (obligatorio): el identificador único de ejecución de tareas para recuperar los detalles de ejecución de tareas correspondientes.
- Devuelve una cadena JSON que representa los detalles de la ejecución de una sola tarea (consulte Estructura de mensajes de detalles de ejecución del trabajo) o una respuesta vacía si no se pudieron encontrar detalles de ejecución de la tarea para el identificador dado.

Lista los scripts lanzados de forma asincrónica que se pueden eliminar

- Método compatible: GET
- Ruta: `/killablescripts`
- Devuelve una colección de identificadores de ejecución de tareas que se han lanzado de forma asincrónica y que aún se están ejecutando y que se pueden cerrar por la fuerza (consulte el punto de conexión `/kill` a continuación).

Lista los scripts lanzados de forma sincrónica que se pueden eliminar

- Método compatible: GET
- Ruta: `/killablesyncscripts`
- Devuelve una colección de identificadores de ejecución de tareas que se han lanzado de forma sincrónica y que aún se están ejecutando y que se pueden cerrar por la fuerza (consulte el punto de conexión `/kill` a continuación).

Eliminar una ejecución de trabajo determinada

- Método compatible: GET

- Ruta: `/kill/{identifier:.+}`
- Argumento: identificador de ejecución de la tarea (obligatorio): el identificador único de ejecución de la tarea que apunta a la ejecución de la tarea cuya finalización se va a forzar.
- Devuelve un mensaje textual en el que se detalla el resultado del intento de forzar la finalización de la ejecución de la tarea. El mensaje contendrá el identificador del script, el identificador único de la ejecución de la tarea, y la fecha y hora en que se produjo la finalización de la ejecución de la tarea. Si no se encuentra ninguna ejecución de trabajo en ejecución para el identificador indicado, se devolverá un mensaje de error en su lugar.

⚠ Warning

- El motor de ejecución hace todo lo posible para acabar de forma adecuada con la ejecución de la tarea objetivo. Por lo tanto, la respuesta del punto final /kill puede tardar un poco en llegar a la persona que llama, ya que el tiempo de ejecución de AWS Blu Age intentará minimizar el impacto empresarial de la interrupción del trabajo.
- Eliminar por la fuerza la ejecución de un trabajo no debe hacerse a la ligera, ya que puede tener consecuencias comerciales directas, incluida la posible pérdida o corrupción de datos. Debe reservarse para los casos en los que la ejecución de una determinada tarea no se haya realizado correctamente y los medios de corrección de datos estén claramente identificados.
- Destruir un trabajo debería dar lugar a nuevas investigaciones (análisis post mortem) para averiguar qué fue lo que salió mal y tomar las medidas correctivas adecuadas.
- En cualquier caso, el intento de interrumpir un trabajo en ejecución se anotará en los registros del servidor con mensajes de nivel de advertencia.

Listado de los puntos de comprobación existentes para la reiniciabilidad

La reiniciabilidad de los trabajos se basa en la capacidad de los scripts de registrar puntos de comprobación en el `CheckpointRegistry` para seguir el progreso de la ejecución del trabajo. Si la ejecución de una tarea no finaliza correctamente y se han registrado los puntos de comprobación de reinicio, basta con reiniciar la ejecución de la tarea desde el último punto de comprobación registrado conocido (sin tener que ejecutar los pasos previos anteriores al punto de comprobación).

- Método compatible: GET

- Ruta: `/restarts/{scriptId}/{jobId}`
- Argumentos:
  - scriptId (opcional: cadena): el script que se reinicia.
  - jobId (opcional: cadena): el identificador único de una ejecución de trabajo.
- Devuelve una lista con formato JSON de puntos de reinicio existentes, que se pueden utilizar para reiniciar un trabajo cuya ejecución no ha finalizado correctamente, o para desencadenar un reinicio retrasado al omitir los pasos ejecutados anteriormente. Si ningún script ha registrado ningún punto de comprobación, el contenido de la página será No registered checkpoints.

Reiniciar un trabajo (de forma sincrónica)

- Método compatible: GET
- Ruta: `/restart/{hashcode}/{scriptId}/{skipflag}`
- Argumentos:
  - hashcode (entero: obligatorio): reinicia la ejecución más reciente de un trabajo, utilizando el código hash proporcionado como valor de punto de comprobación (consulte el punto de conexión `/restarts` anterior para obtener más información sobre cómo recuperar un valor de punto de comprobación válido).
  - scriptId (opcional: cadena): el script que se reinicia.
  - skipflag (opcional, booleano): omite la ejecución del paso (punto de comprobación) seleccionado y reinicia desde el paso inmediatamente posterior (de haberlo).
- Devoluciones: consulte la descripción de devolución de `/script` anterior.

Reiniciar un trabajo (de forma asincrónica)

- Método compatible: GET
- Ruta: `/triggerrestart/{hashcode}/{scriptId}/{skipflag}`
- Argumentos:
  - hashcode (entero: obligatorio): reinicia la ejecución más reciente de un trabajo, utilizando el código hash proporcionado como valor de punto de comprobación (consulte el punto de conexión `/restarts` anterior para obtener más información sobre cómo recuperar un valor de punto de comprobación válido).
  - scriptId (opcional: cadena): el script que se reinicia.

  - skipflag (opcional, booleano): omite la ejecución del paso (punto de comprobación) seleccionado y reinicia desde el paso inmediatamente posterior (de haberlo).
- Devoluciones: consulte la descripción de devolución de `/triggerscript` anterior.

Establecer el límite de subprocesos para la ejecución de trabajos asincrónica

La ejecución asincrónica del trabajo se basa en un grupo dedicado de subprocesos en la JVM. Ese grupo tiene un límite fijo en cuanto al número de subprocesos disponibles. El usuario tiene la capacidad de ajustar el límite de acuerdo con las capacidades del host (cantidad de memoria disponible CPUs, etc.). De forma predeterminada, el límite de subprocesos está establecido en 5 subprocesos.

- Método compatible: GET
- Ruta: `/settriggerthreadlimit/{threadlimit:.+}`
- Argumento (entero): el nuevo límite de subprocesos que se aplicará. Debe ser un entero estrictamente positivo.
- Devuelve un mensaje (`String`) con el nuevo límite de subprocesos y el anterior, o un mensaje de error si el valor límite de subprocesos proporcionado no es válido (no es un entero estrictamente positivo).

Respuesta de ejemplo:

```
Set thread limit for Script Tower Control to 10 (previous value was 5)
```

Contar las ejecuciones de trabajos desencadenados en curso

- Método compatible: GET
- Ruta: `/countrunningtriggeredscripts`
- Devuelve un mensaje que indica el número de trabajos en ejecución lanzados de forma asincrónica y el límite de subprocesos (es decir, el número máximo de trabajos activados que se pueden ejecutar simultáneamente).

Respuesta de ejemplo:

```
0 triggered script(s) running (limit =10)
```

> **Note**
>
> Se puede utilizar para comprobar, antes de lanzar un trabajo, si no se ha alcanzado el límite de subprocesos (lo que impediría lanzar el trabajo).

Purgar la información sobre las ejecuciones de trabajos

La información sobre las ejecuciones de los trabajos permanece en la memoria del servidor mientras el servidor esté activo. Puede ser conveniente purgar la información más antigua de la memoria, pues ya no es relevante; este es el propósito de este punto de conexión.

- Método compatible: GET
- Ruta: `/purgejobinformation/{age:.+}`
- Argumentos: un valor entero estrictamente positivo que representa la antigüedad en horas de la información que se va a purgar.
- Devuelve un mensaje con la siguiente información:
  - Nombre del archivo de purga en el que se almacena la información de ejecución de los trabajos purgados con fines de archivado.
  - Número de información de ejecución de tareas purgadas.
  - Número de información restante sobre la ejecución del trabajo en la nota

Métricas para puntos de conexión

JVM

Este punto de conexión devuelve las métricas disponibles relacionadas con la JVM.

- Método compatible: GET
- Ruta: `/metrics/jvm`
- Argumentos: ninguno
- Devuelve un mensaje con la siguiente información:
  - threadActiveCount: Número de hilos activos.
  - jvmMemoryUsed: Memoria utilizada activamente por la máquina virtual Java.
  - jvmMemoryMax: Memoria máxima permitida para la máquina virtual Java.
  - jvmMemoryFree: Memoria disponible que la máquina virtual Java no está utilizando actualmente.

Sesión

Este punto de conexión devuelve las métricas relacionadas con las sesiones HTTP abiertas en ese momento.

- Método compatible: GET
- Ruta: `/metrics/session`
- Argumentos: ninguno
- Devuelve un mensaje con la siguiente información:
  - sessionCount: número de sesiones de usuario activas que actualmente mantiene el servidor.

Lote

- Método compatible: GET
- Ruta: `/metrics/batch`
- Argumentos:
  - startTimestamp (opcional, número): marca de tiempo inicial para el filtrado de datos.
  - startTimestamp (opcional, número): marca de tiempo final para el filtrado de datos.
  - page (opcional, número): número de página para la paginación.
  - pageSize (opcional, número): número de elementos por página en la paginación.
- Devuelve un mensaje con la siguiente información:
  - content: lista de métricas de ejecución por lotes.
  - pageNumber: número de página actual en la paginación.
  - pagesize: número de elementos mostrados por página.
  - totalPages: número total de páginas disponibles.
  - numberOfElements: Recuento de elementos de la página actual.
  - last: marca booleana para la última página.
  - first: marca booleana para la primera página.

Transacción

- Método compatible: GET
- Ruta: `/metrics/transaction`

- Argumentos:
  - startTimestamp (opcional, número): marca de tiempo inicial para el filtrado de datos.
  - startTimestamp (opcional, número): marca de tiempo final para el filtrado de datos.
  - page (opcional, número): número de página para la paginación.
  - pageSize (opcional, número): número de elementos por página en la paginación.
- Devuelve un mensaje con la siguiente información:
  - content: lista de métricas de ejecución de transacciones.
  - pageNumber: número de página actual en la paginación.
  - pagesize: número de elementos mostrados por página.
  - totalPages: número total de páginas disponibles.
  - numberOfElements: Recuento de elementos de la página actual.
  - last: marca booleana para la última página.
  - first: marca booleana para la primera página.

Otros puntos de conexión

Utilice estos puntos de conexión para listar los programas o servicios registrados, conocer el estado y administrar las transacciones del JICS.

Temas

- Visualización de los programas registrados
- Listado de servicios registrados
- Estado
- Listado de las transacciones JICS disponibles
- Lanzar una transacción de JICS
- Lanzar una transacción de JICS (alternativa)
- Enumeración de las sesiones activas

Visualización de los programas registrados

- Método compatible: GET
- Ruta: `/programs`

- Devuelve la lista de programas registrados, en forma de página html. Cada programa se designa mediante su identificador de programa principal. Se devuelven a la lista tanto los programas antiguos modernizados como los programas de utilidades (IDCAMS, IEBGENER, etc.). Tenga en cuenta que los programas de utilidades disponibles dependerán de las aplicaciones web de utilidades que se hayan implementado en su servidor Tomcat. Por ejemplo, es posible que los programas de compatibilidad de utilidades de z/OS no estén disponibles para los activos modernizados de iSeries, ya que no son relevantes.

Listado de servicios registrados

- Método compatible: GET
- Ruta: `/services`
- Devuelve la lista de servicios de tiempo de ejecución registrados, en forma de página html. El motor de ejecución de AWS Blu Age ofrece estos servicios como utilidades, que se pueden utilizar, por ejemplo, en scripts geniales. Los servicios de carga de Blusam (para crear conjuntos de datos de Blusam a partir de conjuntos de datos antiguos) entran en esa categoría.

Respuesta de ejemplo:

```
<p>BluesamESDSFileLoader</p><p>BluesamKSDSFileLoader</p><p>BluesamRRDSFileLoader</p>
```

Estado

- Método compatible: GET
- Ruta: `/`
- Devuelve un mensaje sencillo que indica que la aplicación Gapwalk está activa y en ejecución (`Jics application is running.`)

Listado de las transacciones JICS disponibles

- Método compatible: GET
- Ruta: `/transactions`
- Devuelve una página html con una lista de todas las transacciones JICS disponibles. Esto solo tiene sentido para entornos con elementos JICS (modernización de elementos CICS heredados).

Respuesta de ejemplo:

```
<p>INQ1</p><p>MENU</p><p>MNT2</p><p>ORD1</p><p>PRNT</p>
```

Lanzar una transacción de JICS

- Métodos compatibles: GET,POST
- Ruta: `/jicstransrunner/{jtrans:.+}`
- Argumentos:
  - Identificador de transacción JICS (cadena, obligatorio): identificador de la transacción JICS que se va a lanzar (8 caracteres como máximo)
  - obligatorio: datos de entrada adicionales para pasarlos a la transacción, en forma de Map<String, Object>. El contenido de este mapa se utilizará para alimentar la COMMAREA que consumirá la transacción del JICS. El mapa puede estar vacío si no se requieren datos para ejecutar la transacción.
  - opcional: entradas de encabezados HTTP para personalizar el entorno de ejecución de la transacción en cuestión. Se admiten las siguientes claves de encabezado:
    - `jics-channel`: el nombre del JICS CHANNEL que utilizará el programa que se lanzará al lanzar esta transacción.
    - `jics-container`: el nombre del JICS CONTAINER que se utilizará para el lanzamiento de esta transacción de JICS.
    - `jics-startcode`: el STARTCODE (cadena, de hasta 2 caracteres) que se utilizará al iniciar la transacción con el JICS. Consulte STARTCODE para ver los valores posibles (desplácese hacia abajo en la página).
    - `jicxa-xid`: el XID (estructura XID del identificador de transacción X/Open) de una “transacción global” (XA), iniciada por el programa que hace la llamada, en la que participará el lanzamiento actual de la transacción del JICS.
- Devuelve una serialización de JSON `com.netfective.bluage.gapwalk.rt.shared.web.TransactionResultBean`, que representa la salida del inicio de la transacción del JICS.

Para obtener más información sobre los detalles de la estructura, consulte Estructura de resultados del lanzamiento de la transacción.

Lanzar una transacción de JICS (alternativa)

- Métodos compatibles: GET,POST
- ruta: `/jicstransaction/{jtrans:.+}`
- Argumentos:

  identificador de transacción JICS (cadena, obligatorio)

  identificador de la transacción JICS que se va a lanzar (8 caracteres como máximo)

  obligatorio: datos de entrada adicionales para pasarlos a la transacción, en forma de Map<String, Object>.

  El contenido de este mapa se utilizará para alimentar la COMMAREA que consumirá la transacción del JICS. El mapa puede estar vacío si no se requieren datos para ejecutar la transacción.

  opcional: entradas de encabezados HTTP para personalizar el entorno de ejecución de la transacción en cuestión.

  Se admiten las siguientes claves de encabezado:

  - `jics-channel`: el nombre del JICS CHANNEL que utilizará el programa que se lanzará al lanzar esta transacción.
  - `jics-container`: el nombre del JICS CONTAINER que se utilizará para el lanzamiento de esta transacción de JICS.
  - `jics-startcode`: el STARTCODE (cadena, de hasta 2 caracteres) que se utilizará al iniciar la transacción con el JICS. Para ver los valores posibles, consulte STARTCODE (navegue hacia abajo en la página).
  - `jicxa-xid`: el XID (estructura XID del identificador de transacción X/Open) de una "transacción global" (XA), iniciada por el programa que hace la llamada, en la que participará el lanzamiento actual de la transacción del JICS.
- Devuelve una serialización de JSON `com.netfective.bluage.gapwalk.rt.shared.web.RecordHolderBean`, que representa la salida del inicio de la transacción del JICS. Los detalles de la estructura se pueden consultar en Estructura de resultados del registro de lanzamiento de la transacción.

Enumeración de las sesiones activas

- Métodos compatibles: GET,POST

- ruta: `/activesessionlist`
- Argumentos: ninguno
- Devuelve una lista de `com.netfective.bluage.gapwalk.application.web.sessiontracker.SessionTrackerObje` en la serialización de JSON, que representa la lista de sesiones de usuario activas. Si el seguimiento de sesiones está deshabilitado, se devolverá una lista vacía.

Puntos de conexión relacionados con las colas de trabajos

Las colas de trabajos son el soporte de la Era AWS Azul para el mecanismo de presentación de AS4 00 trabajos. Las colas de trabajos se utilizan en AS4 00 para ejecutar trabajos en grupos de subprocesos específicos. Una cola de trabajos se define mediante un nombre y un número máximo de subprocesos que corresponde al número máximo de programas que se pueden ejecutar simultáneamente en esa cola. Si se envían más trabajos a la cola que el número máximo de subprocesos, los trabajos esperarán a que haya un subproceso disponible.

Para obtener una lista exhaustiva del estado de un trabajo en cola, consulte Los posibles estados de un trabajo en una cola son:.

Las operaciones en las colas de trabajos se gestionan a través de los siguientes puntos de conexión específicos. Puede invocar estas operaciones desde la URL de la aplicación Gapwalk con la siguiente URL raíz: `http://server:port/gapwalk-application/jobqueue`.

Temas

- Listado de las colas disponibles
- Iniciar o reiniciar una cola de trabajo
- Enviar un trabajo para su lanzamiento
- Enumeración de todos los trabajos enviados
- Liberación de todos los trabajos que están “en espera”
- Liberación de todos los trabajos que estén “en espera” de un nombre de trabajo determinado
- Liberación de un trabajo determinado de un número de trabajo
- Envío de un trabajo en una programación repetida
- Enumeración de todos los trabajos repetitivos enviados
- Cancelación de la programación de un trabajo de repetición

Listado de las colas disponibles

- Método compatible: GET
- Ruta: `list-queues`
- Devuelve la lista de las colas disponibles junto con su estado, como una lista JSON de valores clave.

Respuesta de ejemplo:

```
{"Default":"STAND_BY","queue1":"STARTED","queue2":"STARTED"}
```

Los posibles estados de una cola de trabajos son:

STAND_BY

la cola de trabajos está esperando a que se inicie.

STARTED

la cola de trabajos está activa y funcionando.

UNKNOWN

no se puede determinar el estado de la cola de trabajos.

Iniciar o reiniciar una cola de trabajo

- Método compatible: POST
- Ruta: `/restart/{name}`
- Argumento: el nombre de la cola que se va a iniciar o reiniciar, en forma de cadena: obligatorio.
- El punto de conexión no devuelve nada, sino que se basa en el estado de http para indicar el resultado de la operación de inicio o reinicio:

  HTTP 200

  la operación de inicio o reinicio ha ido bien: la cola de trabajos indicada ahora está STARTED.

  HTTP 404

  la cola de trabajos no existe.

HTTP 503

se ha producido una excepción durante el intento de inicio o reinicio (se deben inspeccionar los registros del servidor para averiguar qué ha fallado).

Enviar un trabajo para su lanzamiento

- Método compatible: POST
- Ruta: `/submit`
- Argumento: obligatorio como cuerpo de la solicitud, una serialización JSON de un objeto `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage`. Para obtener más información, consulte Entrada de envío y de programación del trabajo.
- Devuelve: un JSON que contiene el `SubmitJobMessage` original y un registro que indica si el trabajo se ha enviado o no.

Enumeración de todos los trabajos enviados

- Método compatible: GET
- Ruta: `/list-jobs?status={status}&size={size}&page={page}&sort={sort}`
- Argumentos:
  - page: número de página que se recuperará (predeterminado = 1)
  - size: tamaño de la página (predeterminado = 50, máximo = 300)
  - sort: el orden de los trabajos. (predeterminado = executionId). En la actualidad, executionId es el único valor admitido
  - status: (opcional) si está presente, filtrará el estado.
- Devuelve una lista de todos los trabajos programados, como cadena JSON. Para ver un ejemplo de respuesta, consulte Lista de respuestas a los trabajos programados.

Liberación de todos los trabajos que están “en espera”

- Método compatible: POST
- Ruta: `/release-all`
- Devuelve un mensaje que indica el resultado de la operación de intento de liberación. Aquí hay dos posibles casos:

- HTTP 200 y el mensaje "All job released with success!" si todos los trabajos se han publicado correctamente.
- HTTP 503 y un mensaje "Jobs not released. Se ha producido un error desconocido. See log for more details" si algo ha salido mal en el intento de publicación.

Liberación de todos los trabajos que estén “en espera” de un nombre de trabajo determinado

Para un nombre de trabajo determinado, se pueden enviar varios trabajos con diferentes números de trabajo (la unicidad de una ejecución de trabajo se concede por un par <job name, job number>). El punto de conexión intentará publicar todos los trabajos presentados con el nombre indicado, que estén “en espera”.

- Método compatible: POST
- Ruta: `/release/{name}`
- Argumentos: el nombre del trabajo que se va a buscar, en forma de cadena. Obligatorio.
- Devuelve un mensaje que indica el resultado de la operación de intento de liberación. Aquí hay dos posibles casos:
  - HTTP 200 y el mensaje "Jobs in group <nombre> (<número de trabajos publicados>) released with success!" si los trabajos se han publicado correctamente.
  - HTTP 503 y un mensaje "Jobs in group <nombre> not released. An unknown error occured. See log for more details" si algo ha salido mal en el intento de publicación.

Liberación de un trabajo determinado de un número de trabajo

El punto de conexión intentará liberar la publicación de trabajo única que está “en espera” para el par en cuestión <job name, job number>.

- Método compatible: POST
- Ruta: `/release/{name}/{number}`
- Argumentos:

  nombre

  el nombre del trabajo que se va a buscar, en forma de cadena. Obligatorio.

  number

  el número de trabajo que se va a buscar, como un entero. Obligatorio.

returns

un mensaje que indica la salida de la operación de intento de liberación. Aquí hay dos posibles casos:

- HTTP 200 y el mensaje ""Job <nombre/número> released with success!" si el trabajo se ha publicado correctamente.
- HTTP 503 y un mensaje "Job <name/number>>not released. An unknown error occured. See log for more details" si algo ha salido mal en el intento de publicación.

Envío de un trabajo en una programación repetida

Programe un trabajo que se ejecutará con una programación repetida.

- Método compatible: POST
- Ruta: `/schedule`
- Argumento: el cuerpo de la solicitud debe contener una serialización JSON de un objeto `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage`.

Enumeración de todos los trabajos repetitivos enviados

- Método compatible: GET
- Ruta: `/schedule/list?status={status}&size={size}&page={page}&sort={sort}`
- Argumentos:
  1. page: número de página que se recuperará (predeterminado = 1)
  2. size: tamaño de la página (predeterminado = 50, máximo = 300)
  3. sort: el orden de los trabajos. (predeterminado = id). id es el único valor admitido por el momento.
  4. status: (opcional) si está presente, filtrará el estado. Los valores posibles son los que se mencionan en la sección 1.
  5. status: (opcional) si está presente, filtrará el estado. Los valores posibles son los que se mencionan en la sección 1.
  6. Devuelve una lista de todos los trabajos programados, como cadena JSON.

Cancelación de la programación de un trabajo de repetición

Elimina un trabajo que se creó según en una programación repetida. El estado de la programación del trabajo está establecido en INACTIVE.

- Método compatible: GET
- Ruta: `/schedule/remove/{schedule_id}`
- Argumento: `schedule_id`, el identificador del trabajo programado que se eliminará.

Puntos de conexión REST de la consola de aplicaciones de Blusam

En esta sección, puede obtener información sobre la consola de aplicaciones de Blusam, que es una API diseñada para simplificar la administración de conjuntos de datos de VSAM modernizados. Los puntos de conexión de la aplicación web de Blusam utilizan la ruta raíz `/bac`.

Temas

- Puntos de conexión relacionados con conjuntos de datos
- Puntos de conexión relacionados con conjuntos de datos en bloque
- Registros
- Máscaras
- Otro
- Puntos de conexión de administración de usuarios de BAC

Puntos de conexión relacionados con conjuntos de datos

Utilice los siguientes puntos de conexión para crear o administrar un conjunto de datos específico.

Temas

- Creación de un conjunto de datos
- Carga de un archivo
- Carga de un conjunto de datos (POST)
- Carga de un conjunto de datos (GET)
- Puede cargar un conjunto de datos desde un bucket de Amazon S3.
- Exportar el conjunto de datos a un bucket de Amazon S3.

- Borrar un conjunto de datos
- Eliminar un conjunto de datos
- Contar los registros del conjunto de datos

Creación de un conjunto de datos

Puede utilizar este punto de conexión para crear una definición de conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/createDataSet`
- Argumentos:

  name

  (obligatorio, cadena): el nombre del conjunto de datos.

  type

  (obligatorio, cadena): el tipo de conjunto de datos. Los valores posibles son `ESDS`, `KSDS`, `RRDS`.

  recordSize

  (opcional, cadena): tamaño máximo de cada registro del conjunto de datos.

  fixedLength

  (opcional, booleano): indica si la longitud de los registros es fija.

  compression

  (opcional, booleano): indica si el conjunto de datos está comprimido.

  cacheEnable

  (opcional, booleano): indica si el almacenamiento en caché está habilitado para el conjunto de datos.

  alternativeKeys

  (opcional, lista de claves):

  - offset (obligatorio, número)
  - length (obligatorio, número)

    - name (obligatorio, número)
- Devuelve un archivo JSON que representa el conjunto de datos recién creado.

Solicitud de ejemplo:

```
POST /api/services/rest/bluesamservice/createDataSet
{
  "name": "DATASET",
  "checked": false,
  "records": [],
  "primaryKey": {
    "name": "PK"
  },
  "alternativeKeys": [
    {
      "offset": 10,
      "length": 10,
      "name": "ALTK_0"
    }
  ],
  "type": "ESDS",
  "recordSize": 10,
  "compression": true,
  "cacheEnable": true
}
```

Respuesta de ejemplo:

```
{
    "dataSet": {
      "name": "DATASET",
      "checked": false,
      "nbRecords": 0,
      "keyLength": -1,
      "recordSize": 10,
      "compression": false,
      "fixLength": true,
      "type": "ESDS",
      "cacheEnable": false,
      "cacheWarmup": false,
      "cacheEviction": "100ms",
      "creationDate": 1686744961234,
```

```
      "modificationDate": 1686744961234,
      "records": [],
      "primaryKey": {
        "name": "PK",
        "offset": null,
        "length": null,
        "columns": null,
        "unique": true
      },
      "alternativeKeys": [
        {
          "offset": 10,
          "length": 10,
          "name": "ALTK_0"
        }
      ],
      "readLimit": 0,
      "readEncoding": null,
      "initCharacter": null,
      "defaultCharacter": null,
      "blankCharacter": null,
      "strictZoned": null,
      "decimalSeparator": null,
      "currencySign": null,
      "pictureCurrencySign": null
    },
    "message": null,
    "result": true
  }
```

Carga de un archivo

Este punto de conexión permite cargar archivos al servidor. El archivo se almacena en una carpeta temporal que corresponde a cada usuario específico. Utilice este punto de conexión cada vez que necesite cargar un archivo.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/upload`
- Argumentos:

archivo

(obligatorio, datos de varias partes/formulario): el archivo que se va a cargar.

- Devuelve un valor booleano que refleja el estado de la carga

Carga de un conjunto de datos (POST)

Después de utilizar `createDataSet` para crear la definición del conjunto de datos, puede cargar los registros asociados al archivo cargado en un conjunto de datos específico.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/loadDataSet`
- Argumentos:

  name

  (obligatorio, cadena): el nombre del conjunto de datos.
- Devuelve el estado de la solicitud y del conjunto de datos cargado.

Carga de un conjunto de datos (GET)

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/loadDataSet`
- Argumentos:

  name

  (obligatorio, cadena): el nombre del conjunto de datos.

  archivo de conjunto de datos

  (obligatorio, cadena): el nombre del archivo de conjunto de datos.
- Devuelve el estado de la solicitud y del conjunto de datos cargado.

Puede cargar un conjunto de datos desde un bucket de Amazon S3.

Carga un conjunto de datos mediante un archivo listcat de un bucket de Amazon S3.

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/loadDataSetFromS3`
- Argumentos:

  listcatFileS3Location

  (obligatorio, cadena): la ubicación de Amazon S3 del archivo listcat.

  datasetFileS3Location

  (obligatorio, cadena): la ubicación de Amazon S3 del archivo de conjunto de datos.

  region

  (obligatorio, cadena): el Amazon S3 Región de AWS donde se almacenan los archivos.
- Devuelve el conjunto de datos recién creado

Solicitud de ejemplo:

```
/BAC/api/services/rest/bluesamservice/loadDataSetFromS3?region=us-
east-1&listcatFileS3Location=s3://bucket-name/listcat.json&datasetFileS3Location=s3://
bucket-name/dataset.DAT
```

Exportar el conjunto de datos a un bucket de Amazon S3.

Exporta un conjunto de datos al bucket de Amazon S3 especificado.

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/exportDataSetToS3`
- Argumentos:

  s3Location

  (obligatorio, cadena): la ubicación de Amazon S3 a la que exportar el conjunto de datos.

  datasetName

  (obligatorio, cadena): el nombre del conjunto de datos que se va a exportar.

region

(obligatorio, cadena): el Región de AWS del bucket de Amazon S3.

kmsKeyId

(opcional, cadena): el AWS KMS ID que se utilizará para cifrar el conjunto de datos exportado al bucket de Amazon S3.

- Devuelve el conjunto de datos exportado

Solicitud de ejemplo:

```
/BAC/api/services/rest/bluesamservice/exportDataSetToS3?region=eu-
west-1&s3Location=s3://bucket-name/dump&datasetName=dataset
```

Borrar un conjunto de datos

Borra todos los registros de un conjunto de datos.

- Métodos compatibles: POST, GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/clearDataSet`
- Argumentos:

  nombre

  (obligatorio, cadena): el nombre del conjunto de datos que se va a borrar.
- Devuelve: el estado de la solicitud.

Eliminar un conjunto de datos

Elimina la definición y los registros del conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/deleteDataSet`
- Argumentos:

nombre

(obligatorio, cadena): el nombre del conjunto de datos que se va a eliminar.

- Devuelve el estado de la solicitud y del conjunto de datos eliminado.

Contar los registros del conjunto de datos

Este punto de conexión devuelve el número de registros asociados a un conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/countRecords`
- Argumentos:

  name

  (obligatorio, cadena): el nombre del conjunto de datos.
- Devuelve: el número de registros

Puntos de conexión relacionados con conjuntos de datos en bloque

Utilice los siguientes puntos de conexión para crear o administrar varios conjuntos de datos a la vez.

Temas

- Exportación de conjuntos de datos (GET)
- Exportación de conjuntos de datos (POST)
- Creación de varios conjuntos de datos
- Lista de todos los conjuntos de datos
- Lista directa de todos los conjuntos de datos
- Lista directa de todos los conjuntos de datos por página
- Flujo de conjunto de datos
- Eliminación de todos los conjuntos de datos
- Obtención de las definiciones de conjuntos de datos del archivo listcat
- Obtener las definiciones de conjuntos de datos del archivo listcat cargado

- Obtención de un conjunto de datos
- Carga de listcat desde el archivo JSON

Exportación de conjuntos de datos (GET)

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/exportDataSet`
- Argumentos:

  datasetName

  (obligatorio, cadena): el nombre del conjunto de datos que se va a exportar.

  datasetOutputFile

  (obligatorio, cadena): la ruta de la carpeta donde desea almacenar el conjunto de datos exportado en el servidor.

  rdw

  (obligatorio, booleano): si desea que la palabra descriptora del registro (RDW) forme parte de los registros exportados. Si el conjunto de datos tiene registros de longitud fija, se omite el valor de este parámetro.
- Devuelve el estado de la solicitud y la ruta al archivo que contiene el conjunto de datos exportado (de haberlo). Si el conjunto de datos es nulo en la respuesta, significa que el sistema no ha podido localizar un conjunto de datos con el nombre especificado.

Exportación de conjuntos de datos (POST)

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/exportDataSet`
- Argumentos:

  dumpParameters

  (obligatorio, BACRead parámetros): parámetros de lectura de Bluesam.
- Devuelve el estado del conjunto de datos exportado.

Creación de varios conjuntos de datos

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/createAllDataSets`
- Argumentos:
  - Lista de conjuntos de datos

    nombre

    (obligatorio, cadena): el nombre del conjunto de datos.

    type

    (obligatorio, cadena): el tipo de conjunto de datos. Los valores posibles son `ESDS`, `KSDS`, `RRDS`.

    recordSize

    (opcional, cadena): tamaño máximo de cada registro del conjunto de datos.

    fixedLength

    (opcional, booleano): indica si la longitud de los registros es fija.

    compression

    (opcional, booleano): indica si el conjunto de datos está comprimido.

    cacheEnable

    (opcional, booleano): indica si el almacenamiento en caché está habilitado para el conjunto de datos.
- Devuelve: el estado de la solicitud y el conjunto de datos recién creado.

Lista de todos los conjuntos de datos

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/listDataSet`
- Argumentos: ninguno
- Devuelve el estado de la solicitud y de la lista de conjuntos de datos.

Lista directa de todos los conjuntos de datos

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/directListDataSet`
- Argumentos: ninguno
- Devuelve el estado de la solicitud y de la lista de conjuntos de datos.

Lista directa de todos los conjuntos de datos por página

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/directListDataSetByPage`
- Argumentos:

  datasetName

  (obligatorio, cadena): el nombre del conjunto de datos.

  pageNumber

  (obligatorio, int): el número de página.

  pageSize

  (obligatorio, int): el tamaño de la página.
- Devuelve el estado de la solicitud y de la lista de conjuntos de datos.

Flujo de conjunto de datos

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/streamDataset`
- Argumentos:

  datasetName

  (obligatorio, cadena): el nombre del conjunto de datos.
- Devuelve: un flujo de los conjuntos de datos solicitados.

Eliminación de todos los conjuntos de datos

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/removeAll`
- Argumentos: ninguno
- Devuelve: un valor booleano que representa el estado de la solicitud.

Obtención de las definiciones de conjuntos de datos del archivo listcat

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/getDataSetsDefinitionFromListcat`
- Argumentos:

  paramFilePath

  (obligatorio, cadena): la ruta al archivo listcat.
- Devuelve: una lista de conjuntos de datos

Obtener las definiciones de conjuntos de datos del archivo listcat cargado

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/getDataSetsDefinitionFromUploadedListcat`
- Argumentos: ninguno
- Devuelve: una lista de conjuntos de datos

Obtención de un conjunto de datos

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/getDataSet`
- Argumentos:

name

(obligatorio, cadena): el nombre del conjunto de datos.

- Devuelve el conjunto de datos solicitado.

Carga de listcat desde el archivo JSON

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/loadListcatFromJsonFile`
- Argumentos:

  filePath

  (obligatorio, cadena): la ruta al archivo listcat.
- Devuelve: una lista de conjuntos de datos

Registros

Utilice los siguientes puntos de conexión para crear o administrar registros en un conjunto de datos.

Temas

- Creación de un registro
- Lectura de un conjunto de datos
- Eliminación de un registro
- Actualización de un registro
- Guardar un registro
- Validación de un registro
- Obtención de un árbol de registros

Creación de un registro

Puede utilizar este punto de conexión para crear un nuevo registro.

- Método compatible: POST

- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/createRecord`
- Argumentos:

  conjunto de datos

  (obligatorio DataSet): el objeto del conjunto de datos

  mask

  (obligatorio, mask): el objeto de máscara.
- Devuelve: el estado de la solicitud y el registro creado.

Lectura de un conjunto de datos

Puede usar este punto de conexión para leer un conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/readDataSet`
- Argumentos:

  conjunto de datos

  (obligatorio DataSet): el objeto del conjunto de datos.
- Devuelve el estado de la solicitud y del conjunto de datos con los registros.

Eliminación de un registro

Puede usar este punto de conexión para eliminar un registro de un conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/deleteRecord`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

record

(obligatorio, Record): el registro que se va a eliminar

- Devuelve el estado de la eliminación.

Actualización de un registro

Puede usar este punto de conexión para actualizar un registro asociado a un conjunto de datos.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/updateRecord`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

  record

  (obligatorio, Record): el registro que se va a actualizar
- Devuelve el estado de la solicitud y del conjunto de datos con los registros.

Guardar un registro

Puede usar este punto de conexión para guardar un registro en un conjunto de datos y usar una máscara.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/saveRecord`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

  record

  (obligatorio, Record): el registro que se va a guardar

- Devuelve el estado de la solicitud y del conjunto de datos con los registros.

Validación de un registro

Utilice este punto de conexión para validar un registro.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/validateRecord`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos
- Devuelve el estado de la solicitud y del conjunto de datos con los registros.

Obtención de un árbol de registros

Utilice este punto de conexión para obtener el árbol jerárquico de un registro.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/getRecordTree`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

  record

  (obligatorio, Record): el registro que se recuperará
- Devuelve el estado de la solicitud y el árbol jerárquico del registro solicitado.

Máscaras

Utilice los siguientes puntos de conexión para cargar o aplicar máscaras a un conjunto de datos.

Temas

- Cargar máscaras
- Aplicar máscara
- Aplicar filtro de máscara

Cargar máscaras

Puede usar este punto de conexión para recuperar todas las máscaras asociadas a un conjunto de datos específico.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/loadMasks`
- Variables de ruta:

  recordSize: .../loadMasks/{recordSize}

  (opcional, numérico): el tamaño del registro, filtra las máscaras cargadas que coinciden con este tamaño de registro
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos
- Devuelve el estado de la solicitud y de la lista de máscaras.

Aplicar máscara

Puede usar este punto de conexión para aplicar una máscara a un conjunto de datos específico.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/applyMask`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

mask

(obligatorio, máscara): el objeto del conjunto de datos

- Devuelve el estado de la solicitud y el conjunto de datos con la máscara aplicada.

Aplicar filtro de máscara

Puede usar este punto de conexión para aplicar una máscara y un filtro a un conjunto de datos específico.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/crud/applyMaskFilter`
- Argumentos:

  conjunto de datos

  (obligatorio, DataSet): el objeto del conjunto de datos

  mask

  (obligatorio, máscara): el objeto del conjunto de datos
- Devuelve el estado de la solicitud y el conjunto de datos con la máscara y el filtro aplicados.

Otro

Utilice los siguientes puntos de conexión para administrar la caché de un conjunto de datos o comprobar las características del conjunto de datos

Temas

- Comprobación de la memoria caché de preparación
- Comprobar que la caché está habilitada
- Habilitar caché
- Comprobación de la memoria caché RAM asignada
- Comprobar la persistencia
- Comprobar los tipos de conjuntos de datos compatibles

- Comprobar el estado del servidor

Comprobación de la memoria caché de preparación

Comprueba si la caché de preparación está habilitada para un conjunto de datos específico.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/warmupCache`
- Argumentos:

  name

  (obligatorio, cadena): el nombre del conjunto de datos.
- Devuelve: true si la memoria caché de preparación está habilitada y false en caso contrario.

Comprobar que la caché está habilitada

Comprueba si la caché está habilitada para un conjunto de datos específico.

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/isEnableCache`
- Argumentos: ninguno
- Devuelve true si el almacenamiento en caché está habilitado.

Habilitar caché

- Método compatible: POST
- Requiere autenticación y los roles ROLE_ADMIN y ROLE_SUPER_ADMIN.
- Ruta: `/api/services/rest/bluesamservice/enableDisableCache/{enable}`
- Argumentos:

  enable

  (obligatorio, booleano): si se establece en true, habilitará el almacenamiento en caché.

- Devuelve: nada

Comprobación de la memoria caché RAM asignada

Puede usar este punto de conexión para recuperar la memoria caché RAM asignada.

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/allocatedRamCache`
- Argumentos: ninguno
- Devuelve: el tamaño de la memoria en forma de cadena

Comprobar la persistencia

- Método compatible: GET
- Requiere autenticación y el rol ROLE_USER.
- Ruta: `/api/services/rest/bluesamservice/persistence`
- Argumentos: ninguno
- Devuelve: la persistencia utilizada como cadena

Comprobar los tipos de conjuntos de datos compatibles

- Método compatible: GET
- Ruta: `/api/services/rest/bluesamservice/getDataSetTypes`
- Requiere autenticación y el rol ROLE_USER.
- Argumentos: ninguno
- Devuelve: la lista de tipos de conjuntos de datos admitidos en forma de lista de cadenas.

Comprobar el estado del servidor

- Método compatible: GET
- Ruta: `/api/services/rest/bluesamserver/serverIsUp`
- Argumentos: ninguno

- Devuelve: nada. El código de estado de respuesta HTTP 200 indica que el servidor está en funcionamiento.

Puntos de conexión de administración de usuarios de BAC

Utilice los siguientes puntos de conexión para administrar las interacciones de los usuarios.

Temas

- Registro de usuarios
- Verificación de si existe al menos un usuario en el sistema
- Registro de usuarios nuevos
- Obtención de la información de los usuarios
- Enumeración de usuarios
- Eliminación de un usuario
- Cierre de sesión del usuario actual

Registro de usuarios

- Método compatible: POST
- Ruta: `/api/services/security/servicelogin/login`
- Argumentos: ninguno
- Devuelve la serialización JSON de un objeto `com.netfective.bluage.bac.entities.SignOn`, que representa al usuario cuyas credenciales se proporcionan en la solicitud actual. La contraseña está oculta en la vista del objeto devuelto. Se muestran las funciones asignadas al usuario.

Respuesta de ejemplo:

```
{
     "login": "some-admin",
     "password": null,
     "roles": [
       {
         "id": 0,
         "roleName": "ROLE_ADMIN"
       }
```

```
    ]
  }
```

Verificación de si existe al menos un usuario en el sistema

- Método compatible: GET
- Ruta: `/api/services/security/servicelogin/hasAccount`
- Argumentos: ninguno
- Devuelve el valor booleano `true` si se ha creado al menos un usuario distinto de la superadministrador predeterminado. De lo contrario, devuelve `false`.

Registro de usuarios nuevos

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/recorduser`
- Argumentos: la serialización en JSON de un objeto `com.netfective.bluage.bac.entities.Sign0n` que representa al usuario que se agregará al almacenamiento. Se deben definir los roles de usuario; de lo contrario, es posible que el usuario no pueda utilizar las funciones y los puntos de conexión de BAC.
- Devuelve el valor booleano `true` si el usuario se ha creado correctamente. De lo contrario, devuelve `false`.
- JSON de solicitud de ejemplo:

```
{
    "login": "simpleuser",
    "password": "simplepassword",
    "roles": [
      {
        "id": 2,
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Los siguientes son los dos valores válidos para `roleName`:

- `ROLE_ADMIN`: puede administrar los recursos y usuarios de Blusam.
- `ROLE_USER`: puede administrar los recursos de Blusam, pero no los usuarios.

Obtención de la información de los usuarios

- Método compatible: GET
- Ruta: `/api/services/security/servicelogin/userInfo`
- Argumentos: ninguno
- Devuelve el nombre de usuario y el rol del usuario actualmente conectado.

Enumeración de usuarios

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/listusers`
- Argumentos: ninguno
- Devuelve una lista de `com.netfective.bluage.bac.entities.SignOn`, serializada como JSON.

Eliminación de un usuario

> **Important**
>
> Esta acción no se puede deshacer. El usuario eliminado no podrá volver a conectarse a la aplicación de BAC.

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/deleteuser`
- Argumentos: la serialización en JSON de un objeto `com.netfective.bluage.bac.entities.SignOn` que representa al usuario que se eliminará del almacenamiento.
- Devuelve el valor booleano `true` si el usuario se ha eliminado correctamente.

Cierre de sesión del usuario actual

- Método compatible: GET
- Ruta: `/api/services/security/servicelogout/logout`
- Argumentos: ninguno
- Devuelve el mensaje JSON `{"success":true}` si la sesión del usuario se ha cerrado correctamente. La sesión HTTP relacionada será invalidada.

Administre la consola de aplicaciones JICS en AWS Blu Age

El componente JICS es el soporte de la Era AWS Azul para la modernización de los recursos antiguos del CICS. La aplicación web de la consola de aplicaciones JICS está dedicada a administrar los recursos del JICS. Los siguientes puntos de conexión permiten realizar las tareas de administración sin tener que interactuar con la interfaz de usuario de JAC. Siempre que un punto de conexión requiera autenticación, la solicitud deberá incluir los detalles de autenticación (normalmente, nombre de usuario/contraseña, tal y como exige la autenticación básica). Los puntos de conexión de la aplicación web de la consola de aplicaciones JICS utilizan la ruta raíz `/jac/`.

Temas

- Administración de recursos del JICS
- Otro
- Puntos de conexión de administración de usuarios de JAC

Administración de recursos del JICS

Todos los puntos de conexión siguientes están relacionados con la administración de los recursos del JICS, lo que permite a los administradores del JICS gestionar los recursos a diario.

Temas

- Muestre las listas y los grupos del JICS
- Recuperación de los recursos del JICS
- Lista los grupos de JICS
- Lista de grupos de JICS para una lista determinada
- Lista de recursos del JICS para un grupo determinado

- Lista de los recursos del JICS para un grupo determinado (también puede usar un nombre)
- Edición de los grupos propios de varias listas
- Eliminar una lista
- Eliminación de un grupo
- Eliminar una transacción
- Eliminar un programa
- Eliminar un archivo
- Eliminar un objeto TDQUEUE
- Eliminar un objeto TSMODEL
- Eliminación de elementos
- Crear una lista
- Crear un grupo
- Consideraciones comunes sobre la creación de recursos
- Crear una transacción
- Crear un programa
- Crear un archivo
- Crear un objeto TDQUEUE
- Crear un objeto TSMODEL.
- Creación de elementos
- Actualizar una lista
- Actualizar un grupo
- Consideraciones sobre la actualización de los recursos comunes
- Actualizar una transacción
- Actualizar un programa
- Actualizar un archivo
- Actualice un objeto TDQUEUE
- Actualizar un objeto TSMODEL
- Actualización de elementos
- Creación o actualización de elementos

- Recuperación de elementos
- Funcionamiento de CRUD de JICS

Muestre las listas y los grupos del JICS

La lista y los grupos son los principales recursos de contenedores propietarios del componente JICS. Todos los recursos del JICS deben pertenecer a un grupo. Los grupos pueden pertenecer a listas, pero no es obligatorio. Es posible que las listas ni siquiera existan en un entorno JICS determinado, pero la mayoría de las veces, las listas están ahí para ofrecer una capa adicional de organización de los recursos. Para obtener más información sobre la organización de los recursos del CICS, consulte los recursos del CICS.

- Método compatible: GET
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/listJicsListsAndGroups`
- Argumentos: ninguno
- Devuelve: una lista de JicsContainer objetos serializados, tanto LISTAS como GRUPOS, en formato JSON.

Respuesta de ejemplo:

```
[
    {
      "name": "Resources",
      "children": [
        {
          "jacType": "JACList",
          "name": "MURACHS",
          "isActive": true,
          "children": [
            {
              "jacType": "JACGroup",
              "name": "MURACHS",
              "isActive": true,
              "children": []
            }
          ]
```

```
        },
        {
          "jacType": "JACGroup",
          "name": "TEST",
          "isActive": true,
          "children": []
        }
      ],
      "isExpanded": true
    }
  ]
```

Recuperación de los recursos del JICS

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/retrieveJicsResources`
- Argumentos: una carga útil de JSON que representa los recursos del JICS que desea recuperar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.request.RetrieveOperationRequest`.
- Devuelve: una lista de objetos serializados. JicsResource Los objetos se devuelven sin ningún orden en particular y son de diferentes tipos, como PROGRAM, TRANSACTION, FILE, etc.

Lista los grupos de JICS

- Método compatible: GET
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/listJicsGroups`
- Argumentos: ninguno
- Devuelve una lista de JicsContainer objetos serializados (GRUPOS) en formato JSON. Los grupos se devuelven sin su propia información de LIST.

Respuesta de ejemplo:

```
[
```

```
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
      "isActive": true,
      "children": []
    },
    {
      "jacType": "JACGroup",
      "name": "TEST",
      "isActive": true,
      "children": []
    }
  ]
```

Lista de grupos de JICS para una lista determinada

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/listGroupsForList`
- Argumentos: una carga útil de JSON, que representa la lista de JICS cuyos grupos está buscando. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACList`.

  Solicitud de ejemplo:

```
{
    "jacType":"JACList",
    "name":"MURACHS",
    "isActive":true
  }
```

- Devuelve una lista de JicsContainer objetos serializados (GRUPOS) en formato JSON, que se adjuntan a la LISTA dada. Los grupos se devuelven sin su propia información de LIST.

  Respuesta de ejemplo:

```
[
    {
      "jacType": "JACGroup",
      "name": "MURACHS",
```

```
      "isActive": true,
      "children": []
    }
  ]
```

Lista de recursos del JICS para un grupo determinado

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/listResourcesForGroup`
- Argumentos: una carga útil de JSON, que representa el JICS GROUP cuyos recursos está buscando. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACGroup`. No es necesario especificar todos los campos para el grupo, pero el nombre sí es obligatorio.

  Solicitud de ejemplo:

```
{
    "jacType":"JACGroup",
    "name":"MURACHS",
    "isActive":true
  }
```

- Devuelve una lista de JicsResource objetos serializados, propiedad del GRUPO dado. Los objetos se devuelven sin ningún orden en particular y son de diferentes tipos, como PROGRAM, TRANSACTION, FILE, etc.

Lista de los recursos del JICS para un grupo determinado (también puede usar un nombre)

- Método compatible: POST
- Requiere autenticación
- Ruta: `/api/services/rest/jicsservice/listResourcesForGroupName`
- Argumentos: el nombre del GROUP propietario de los recursos que busca.
- Devuelve: una lista de JicsResource objetos serializados, propiedad del GRUPO dado. Los objetos se devuelven sin ningún orden en particular y son de diferentes tipos, como PROGRAM, TRANSACTION, FILE, etc.

Edición de los grupos propios de varias listas

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/editGroupsList`
- Argumentos: una representación en JSON de una colección de listas con grupos secundarios;

  Solicitud de ejemplo:

```
[
  {
        "jacType": "JACList",
        "name": "MURACHS",
        "isActive": true,
        "children": [
          {
            "jacType": "JACGroup",
            "name": "MURACHS",
            "isActive": true,
            "children": []
          },
          {
            "jacType": "JACGroup",
            "name": "TEST",
            "isActive": true,
            "children": []
          }
        ]
  }
]
```

  Antes de esta edición, solo el grupo denominado MURACHS pertenecía a la lista denominada MURACHS. Con esta edición, agregamos el grupo denominado TEST a la lista denominada MURACHS.

- Devuelve un valor booleano. Si el valor es “true”, las modificaciones de LISTS se han conservado correctamente en el almacenamiento de JICS subyacente.

Eliminar una lista

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteList`
- Argumentos: una carga JSON, que representa la lista de JICS que se va a eliminar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACList`.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de LIST se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminación de un grupo

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteGroup`
- Argumentos: una carga JSON, que representa el grupo de JICS que se va a eliminar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACGroup`.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de GROUP se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminar una transacción

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteTransaction`
- Argumentos: una carga JSON, que representa la transacción de JICS que se va a eliminar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de TRANSACTION se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminar un programa

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteProgram`
- Argumentos: una carga JSON, que representa el programa de JICS que se va a eliminar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de PROGRAM se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminar un archivo

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteFile`
- Argumentos: una carga JSON, que representa el archivo de JICS que se va a eliminar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACFile`.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de FILE se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminar un objeto TDQUEUE

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteTDQueue`
- Argumentos: una carga JSON que representa el TDQUEUE del JICS que se va a eliminar. Esta es la serialización en JSON de un archivo `com.netfective.bluage.jac.entities. JACTDQueue`objeto.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación del objeto TDQUEUE se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminar un objeto TSMODEL

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteTSModel`
- Argumentos: una carga JSON que representa el TSMODEL del JICS que se va a eliminar. Esta es la serialización en JSON de un `com.netfective.bluage.jac.entities. JACTSModel`objeto.
- Devuelve un valor booleano. Si el valor es “true”, la eliminación de TSMODEL se ha realizado correctamente en el almacenamiento de JICS subyacente.

Eliminación de elementos

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/deleteElements`
- Argumentos: una carga útil JSON que representa los elementos JICS que se eliminarán.
- Devuelve un valor booleano donde `true` indica que la eliminación se ha realizado correctamente en el almacenamiento de JICS subyacente.

Crear una lista

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createList`
- Argumentos: una carga JSON, que representa la lista de JICS que se va a eliminar. Esta es la serialización en JSON de un `com.netfective.bluage.jac.entities. JACList`objeto.
- Devuelve un valor booleano. Si el valor es “true”, la LIST se ha creado correctamente en el almacenamiento de JICS subyacente.

> **Note**
>
> La lista siempre se creará vacía. Adjuntar grupos a la lista requerirá otra operación.

Crear un grupo

- Método compatible: POST
- Requiere autenticación y los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createGroup`
- Argumentos: una carga JSON, que representa el grupo de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACGroup`.
- Devuelve un valor booleano. Si el valor es “true”, el grupo se ha creado correctamente en el almacenamiento JICS subyacente.

> **Note**
>
> El grupo siempre se creará vacío. Adjuntar recursos al grupo requerirá operaciones adicionales (la creación de recursos los asociará automáticamente a un grupo determinado).

Consideraciones comunes sobre la creación de recursos

Todos los puntos de conexión siguientes están relacionados con la creación de recursos de JICS y comparten algunas limitaciones comunes: en la carga de la solicitud que se va a enviar al punto de conexión, se debe valorar el campo `groupName`.

Restricción de propiedad del grupo:

No se puede crear ningún recurso sin estar asociado a un grupo existente, y el punto de conexión utiliza el groupName para recuperar el grupo al que se adjuntará este recurso. El `groupName` debe apuntar al nombre de un grupo existente. Se enviará un mensaje de error con el estado HTTP 400 si el `groupName` no apunta a un grupo existente en el almacenamiento subyacente del JICS.

Restricción de unicidad dentro de un grupo:

Un recurso determinado con un nombre determinado tiene que ser único en un grupo determinado. La comprobación de la unicidad la realizará cada punto de conexión de creación de recursos. Si la carga dada no respeta la restricción de unicidad, el punto de conexión enviará una respuesta HTTP STATUS 400 (BAD REQUEST); consulte el ejemplo de respuesta que aparece a continuación.

Ejemplo de carga útil: intentamos crear la transacción ARIT en el grupo TEST, pero ya existe una transacción con ese nombre en este grupo.

```
{
    "jacType":"JACTransaction",
    "name":"ARIT",
    "groupName":"TEST",
    "isActive":true
  }
```

Recibe la siguiente respuesta de error:

```
{
    "timestamp": 1686759054510,
    "status": 400,
    "error": "Bad Request",
    "path": "/jac/api/services/rest/jicsservice/createTransaction"
  }
```

Al inspeccionar los registros de los servidores, se confirmará el origen del problema:

```
2023-06-14 18:10:54 default         TRACE - o.s.w.m.HandlerMethod
        - Arguments: [java.lang.IllegalArgumentException: Transaction already
 present in the group, org.springframework.security.web.header.HeaderWriterFilter
$HeaderWriterResponse@e34f6b8]
2023-06-14 18:10:54 default         ERROR - c.n.b.j.a.WebConfig                    -
 400
java.lang.IllegalArgumentException: Transaction already present in the group
 at
 com.netfective.bluage.jac.server.services.rest.impl.JicsServiceImpl.createElement(JicsServiceI
```

Crear una transacción

- Método compatible: POST

- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createTransaction`
- Argumentos: una carga JSON, que representa la transacción de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Devuelve un valor booleano. Si el valor es “true”, la TRANSACTION se ha creado correctamente en el almacenamiento de JICS subyacente.

Crear un programa

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createProgram`
- Argumentos: una carga JSON, que representa el programa de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Devuelve un valor booleano. Si el valor es “true”, el PROGRAM se ha creado correctamente en el almacenamiento de JICS subyacente.

Crear un archivo

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createFile`
- Argumentos: una carga JSON, que representa el archivo de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACFile`.
- Devuelve un valor booleano. Si el valor es “true”, el FILE se ha creado correctamente en el almacenamiento de JICS subyacente.

Crear un objeto TDQUEUE

- Método compatible: POST

- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createTDQueue`
- Argumentos: una carga JSON, que representa el objeto TDQUEUE de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTDQueue`.
- Devuelve un valor booleano. Si el valor es “true”, el objeto TDQUEUE se ha creado correctamente en el almacenamiento de JICS subyacente.

Crear un objeto TSMODEL.

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createTSModel`
- Argumentos: una carga JSON, que representa el objeto TSMODEL de JICS que se va a crear. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTSModel`.
- Devuelve un valor booleano donde `true` indica que la creación de elementos se ha realizado correctamente en el almacenamiento de JICS subyacente.

Creación de elementos

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/createElements`
- Argumentos: una carga útil JSON, que representa los elementos JICS que se crearán.
- Devuelve un valor booleano. Si el valor es “true”, los elementos se han creado correctamente en el almacenamiento de JICS subyacente.

Actualizar una lista

- Método compatible: POST

- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateList`
- Argumentos: una carga JSON, que representa la lista de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACList`. No es necesario proporcionar los elementos secundarios de la LIST; el mecanismo de actualización de la LIST no los tendrá en cuenta.
- Devuelve un valor booleano. Si el valor es “true”, la lista se ha actualizado correctamente en el almacenamiento de JICS subyacente.

Si se actualiza el indicador “isActive” de la lista, se propagará a todos los elementos que sean propiedad de la lista, es decir, a todos los grupos que sean propiedad de la lista y a todos los recursos que pertenezcan a dichos grupos. Esta es una forma cómoda de desactivar una gran cantidad de recursos con una sola operación, en varios GROUPS.

Actualizar un grupo

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateGroup`
- Argumentos: una carga JSON, que representa el grupo de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACGroup`. No es necesario proporcionar los elementos secundarios del GROUP; el mecanismo de actualización del grupo no lo tendrá en cuenta.
- Devuelve un valor booleano. Si el valor es “true”, el GROUP se ha actualizado correctamente en el almacenamiento de JICS subyacente.

> **Note**
>
> Si se actualiza el indicador “isActive” del grupo, se propagará a todos los elementos que sean propiedad del grupo, es decir, a todos los recursos que pertenezcan al grupo. Esta es una forma cómoda de desactivar una gran cantidad de recursos con una sola operación en un GROUP dado.

Consideraciones sobre la actualización de los recursos comunes

Los siguientes puntos de conexión están relacionados con la actualización de recursos de JICS. Mediante el campo `groupName`, puede cambiar el GROUP propietario de cualquier JICS RESOURCE, siempre que el valor del campo señale a un grupo existente en el almacenamiento de JICS subyacente (de lo contrario, recibirá una respuesta de BAD REQUEST [HTTP STATUS 400] del punto de conexión).

Actualizar una transacción

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateTransaction`
- Argumentos: una carga JSON, que representa la transacción de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTransaction`.
- Devuelve un valor booleano. Si el valor es “true”, la TRANSACTION se ha actualizado correctamente en el almacenamiento de JICS subyacente.

Actualizar un programa

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateProgram`
- Argumentos: una carga JSON, que representa el programa de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACProgram`.
- Devuelve un valor booleano. Si el valor es “true”, el PROGRAM se ha actualizado correctamente en el almacenamiento de JICS subyacente.

Actualizar un archivo

- Método compatible: POST

- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateFile`
- Argumentos: una carga JSON, que representa el archivo de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACFile`.
- Devuelve un valor booleano. Si el valor es “true”, el FILE se ha actualizado correctamente en el almacenamiento de JICS subyacente.

Actualice un objeto TDQUEUE

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateTDQueue`
- Argumentos: una carga JSON, que representa el objeto TDQUEUE de JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTDQueue`.
- Devuelve un valor booleano. Si el valor es «verdadero», significa que TDQueue se actualizó correctamente en el almacenamiento JICS subyacente.

Actualizar un objeto TSMODEL

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateTSModel`
- Argumentos: una carga útil de JSON, que representa el TSMODEL del JICS que se va a actualizar. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.JACTSModel`.
- Devuelve un valor booleano. Si el valor es “true”, el objeto TSMODEL se ha actualizado correctamente en el almacenamiento de JICS subyacente.

Actualización de elementos

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/updateElements`
- Argumentos: carga útil de JSON, que representa los elementos que se actualizarán.
- Devuelve un valor booleano donde `true` indica que la actualización de los elementos se ha realizado correctamente en el almacenamiento de JICS subyacente.

Creación o actualización de elementos

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/upsertElements`
- Argumentos: carga útil de JSON que representa los elementos que se crearán o actualizarán.
- Devuelve un valor booleano donde `true` indica que la creación o actualización de elementos se ha realizado correctamente en el almacenamiento de JICS subyacente.

Recuperación de elementos

- Método compatible: GET
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.
- Ruta: `/api/services/rest/jicsservice/retrieveElements`
- Argumentos: ninguno
- Devuelve una lista de todos los recursos JICS serializados.

Funcionamiento de CRUD de JICS

- Método compatible: POST
- Requiere autenticación y uno de los siguientes roles: ROLE_ADMIN, ROLE_SUPER_ADMIN, ROLE_USER.

- Ruta: `/api/services/rest/jicsservice/jicsCrudOperation`
- Argumentos: carga útil de JSON que representa los recursos de JICS que busca. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.request.JicsCrudOperationRequest`.
- Devuelve una carga útil de JSON que representa la respuesta. Esta es la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.request.JicsCrudOperationResponse`.

Otro

Temas

- Estado del servidor de JICS

Estado del servidor de JICS

- Método compatible: GET
- Ruta: `/api/services/rest/jicsserver/serverIsUp`
- Argumentos: ninguno
- Devuelve: nada. Una respuesta HTTP STATUS 200 indica que el servidor JICS está en funcionamiento.

Puntos de conexión de administración de usuarios de JAC

Utilice los siguientes puntos de conexión para administrar las interacciones de los usuarios.

Temas

- Registrar a un usuario .
- Probar si existe al menos un usuario en el sistema
- Grabar a un usuario nuevo
- Información del usuario
- Mostrar usuarios
- Eliminar un usuario
- Cierre de sesión del usuario actual

Registrar a un usuario .

- Método compatible: POST
- Ruta: `/api/services/security/servicelogin/login`
- Argumentos: ninguno
- Devuelve la serialización JSON de un objeto `com.netfective.bluage.jac.entities.SignOn`, que representa al usuario cuyas credenciales se proporcionan en la solicitud actual. La contraseña está oculta en la vista del objeto devuelto. Se muestran las funciones asignadas al usuario.

Respuesta de ejemplo:

```
{
    "login": "some-admin",
    "password": null,
    "roles": [
      {
        "id": 0,
        "roleName": "ROLE_ADMIN"
      }
    ]
  }
```

Probar si existe al menos un usuario en el sistema

- Método compatible: GET
- Ruta: `/api/services/security/servicelogin/hasAccount`
- Argumentos: ninguno
- Devuelve el valor booleano `true` si se ha creado al menos un usuario distinto de la superadministrador predeterminado. De lo contrario, devuelve `false`.

Grabar a un usuario nuevo

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/recorduser`

- Argumentos: la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.SignOn`, que representa al usuario que se va a añadir al almacenamiento. Se deben definir los roles del usuario; de lo contrario, es posible que el usuario no pueda utilizar las funciones y los puntos de conexión del JAC.
- Devuelve el valor booleano `true` si el usuario se ha creado correctamente. De lo contrario, devuelve `false`.

Solicitud de ejemplo:

```
{
    "login": "simpleuser",
    "password": "simplepassword",
    "roles": [
      {
        "id": 2,
        "roleName": "ROLE_USER"
      }
    ]
  }
```

Al grabar a un usuario nuevo, solo se pueden usar los siguientes roles:

- ROLE_ADMIN: puede administrar los recursos y usuarios de JICS.
- ROLE_USER: puede administrar los recursos del JICS pero no los usuarios.

Información del usuario

- Método compatible: GET
- Ruta: `/api/services/security/servicelogin/userInfo`
- Argumentos: ninguno
- Devuelve el nombre de usuario y los roles del usuario conectado actualmente.

Mostrar usuarios

- Método compatible: GET
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/listusers`

- Argumentos: ninguno
- Devuelve una lista de `com.netfective.bluage.jac.entities.SignOn`, serializada como JSON.

Eliminar un usuario

- Método compatible: POST
- Requiere autenticación y el rol ROLE_ADMIN.
- Ruta: `/api/services/security/servicelogin/deleteuser`
- Argumentos: la serialización en JSON de un objeto `com.netfective.bluage.jac.entities.SignOn` que representa al usuario que se eliminará del almacenamiento.
- Devuelve el valor booleano `true` si el usuario se ha eliminado correctamente.

> **Important**
>
> Esta acción no se puede deshacer. El usuario eliminado no podrá volver a conectarse a la aplicación JAC.

Cierre de sesión del usuario actual

- Método compatible: GET
- Ruta: `/api/services/security/servicelogout/logout`
- Argumentos: ninguno
- Devuelve el mensaje JSON `{"success":true}` si la sesión del usuario se ha cerrado correctamente. La sesión HTTP relacionada será invalidada.

Estructuras de datos para usuarios de AWS Blu Age

Puede obtener información sobre las diversas estructuras de datos del motor AWS Blu Age en la siguiente sección.

Temas

- Estructura de mensajes de detalles de ejecución del trabajo
- Estructura de resultados del lanzamiento de la transacción
- Estructura de resultados del registro de lanzamiento de la transacción
- Los posibles estados de un trabajo en una cola son:
- Entrada de envío y de programación del trabajo
- Lista de respuestas a los trabajos programados
- Lista de respuestas de trabajos recurrentes

Estructura de mensajes de detalles de ejecución del trabajo

Los detalles de la ejecución de cada trabajo tendrán los siguientes campos:

ScriptId

el identificador del script llamado.

caller

dirección IP del programa que llama.

identificador

identificador único de ejecución del trabajo.

startTime

la fecha y hora de inicio de la ejecución del trabajo.

endTime

la fecha y hora de fin de la ejecución del trabajo.

status

un estado de la ejecución del trabajo. Un valor posible entre:

- `DONE`: la ejecución del trabajo ha finalizado con normalidad.
- `TRIGGERED`: la ejecución del trabajo se ha activado pero aún no se ha iniciado.
- `RUNNING`: la ejecución del trabajo se está ejecutando.
- `KILLED`: la ejecución del trabajo ha sido cancelada.

- `FAILED`: la ejecución del trabajo ha fallado.

executionResult

un mensaje para resumir el resultado de la ejecución del trabajo. Este mensaje puede ser un mensaje simple si la ejecución del trabajo aún no ha finalizado o una estructura JSON con los siguientes campos:

- exitCode: código de salida numérico; los valores negativos indican situaciones de fallo.
- program: último programa lanzado por el trabajo.
- status: un valor posible entre:
  - `Error`: cuando exitCode = -1; corresponde a un error (técnico) que se produce durante la ejecución del trabajo.
  - `Failed`: cuando exitcode = -2; corresponde a un fallo que se produce durante la ejecución de un programa de servicio (como una situación ABEND).
  - `Succeeded`: cuando exitCode >= 0;
- stepName: nombre del último paso ejecutado en el trabajo.

executionMode

SYNCHRONOUS o ASYNCHRONOUS, según la forma en que se haya iniciado el trabajo.

Salida de ejemplo:

```
{
    "scriptId": "INTCALC",
    "caller": "127.0.0.1",
    "identifier": "97d410be-efa7-4bd3-b7b9-d080e5769771",
    "startTime": "06-09-2023 11:42:41",
    "endTime": "06-09-2023 11:42:42",
    "status": "DONE",
    "executionResult": "{ \"exitCode\": -1, \"stepName\": \"STEP15\", \"program\":
 \"CBACT04C\", \"status\": \"Error\" }",
    "executionMode": "ASYNCHRONOUS"
  }
```

Estructura de resultados del lanzamiento de la transacción

La estructura podría contener los siguientes campos:

outCome

una cadena que representa el resultado de la ejecución de la transacción. Los valores posibles son los siguientes:

- `Success`: la ejecución de la transacción ha finalizado correctamente.
- `Failure`: la ejecución de la transacción no ha finalizado correctamente, se han producido algunos problemas.

commarea

una cadena que representa el valor final de COMMAREA, como una matriz de bytes codificada en byte64. Puede ser una cadena vacía.

containerRecord

(Opcional) Cadena que representa el contenido del registro de CONTAINER como matriz de bytes codificada en byte64.

serverDescription

Puede contener información sobre el servidor que ha atendido la solicitud (con fines de depuración). Puede ser una cadena vacía.

abendCode

(Opcional) Si el programa al que se hace referencia en la transacción iniciada está ausente, el valor del código de abend se devolverá en forma de cadena en este campo.

Respuestas de ejemplo

Success

```
{
    "outCome": "Success",
    "commarea": "",
    "serverDescription": ""
  }
```

Failure

```
{
    "outCome": "Failure",
    "commarea": "",
```

```
    "serverDescription": "",
    "abendCode": "AEIA"
  }
```

Estructura de resultados del registro de lanzamiento de la transacción

La estructura podría contener los siguientes campos:

recordContent

una cadena que representa el contenido del registro del COMMAREA como una matriz de bytes codificada en byte64.

containerRecord

una cadena que representa el contenido del registro del CONTAINER como una matriz de bytes codificada en byte64.

serverDescription

Puede contener información sobre el servidor que ha atendido la solicitud (con fines de depuración). Puede ser una cadena vacía.

Respuestas de ejemplo

Success

```
{
    "recordContent": "",
    "serverDescription": ""
}
```

Los posibles estados de un trabajo en una cola son:

En una cola, los trabajos pueden tener el siguiente estado:

ACTIVE

El trabajo se está ejecutando actualmente en la cola.

EXECUTION_WAIT

El trabajo está esperando a que haya un subproceso disponible.

SCHEDULED

La ejecución del trabajo está programada en una fecha y hora específicas.

HOLD

El trabajo está esperando a ser publicado antes de ejecutarse.

COMPLETED

El trabajo se ha ejecutado correctamente.

ERROR

La ejecución del trabajo ha fallado.

UNKNOWN

Se desconoce el estado.

Entrada de envío y de programación del trabajo

La entrada de envío y de programación del trabajo es la serialización en JSON de un objeto `com.netfective.bluage.gapwalk.rt.jobqueue.SubmitJobMessage`. El ejemplo de entrada que aparece a continuación muestra todos los campos de dicho bean.

Ejemplo de entrada para envío de trabajo:

```
{
    "messageQueueName":null,
    "scheduleDate":null,
    "scheduleTime":null,
    "programName":"PTA0044",
    "programParams":
     {"wmind":"B"},
    "localDataAreaValue":"",
    "userName":"USER1",
    "jobName":"PTA0044",
    "jobNumber":9,
    "jobPriority":5,
    "executionDate":"20181231",
    "jobQueue":"queue1",
    "jobOnHold":false
```

```
}
```

Ejemplo de entrada para programación de trabajo:

```
{
     "scheduleCron": "*/2 * * * * ?",
     "programName":"LOGPGM",
     "programParams": {
         "cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\", \"Every 2
 seconds!\"]"
     },
     "localDataAreaValue":"",
     "userName":"PVO",
     "jobName":"LOGGERJOB",
     "jobPriority":5,
     "jobQueue":"queue1",
     "scheduleMisfirePolicy": 4,
     "startTime": "2003/05/04 07:00:00.000 GMT-06:00",
     "endTime": "2003/05/04 07:00:07.000 GMT-06:00"
 }
```

jobNumber

si el número de trabajo es 0, el número de trabajo se generará automáticamente utilizando el siguiente número de la secuencia de números de trabajo. Ese valor debe establecerse en 0 (excepto para fines de prueba).

jobPriority

La prioridad de trabajo predeterminada en AS4 00 es 5. El rango válido es de 0 a 9, siendo 0 la prioridad más alta.

jobOnHold

Si un trabajo se envía en espera, no se ejecutará de inmediato, sino cuando alguien lo “publique”. Se puede publicar un trabajo mediante la API de REST (/release o /release-all).

scheduleDate y scheduleTime

Si estos valores no son null, el trabajo se ejecutará en la fecha y hora especificadas.

Date

Se puede proporcionar con formato MMddyy o dd MMyyyy (el tamaño de la entrada determinará el formato que se utilice)

Tiempo

Se puede proporcionar con formato HHmm o HHmmss (el tamaño de la entrada determinará qué formato se utilizará)

programParams

Esto se pasará al programa como mapa.

scheduleMisfirePolicy

Define la estrategia utilizada cuando un desencadenador no se activa correctamente. A continuación se muestran los posibles valores:

1. Libera la primera activación incorrecta y descarta las demás activaciones incorrectas.
2. Envíe un trabajo en espera para la primera activación incorrecta y descarte las demás activaciones incorrectas.
3. Descarte la activación incorrecta.
4. Libere todas las activaciones incorrectas. La cola de trabajos ejecutará todos los trabajos.

Lista de respuestas a los trabajos programados

Esta es la estructura del punto de conexión de la cola de trabajos de la lista de trabajos. El mensaje de envío de trabajo que se ha utilizado para enviar ese trabajo forma parte de la respuesta. Este se puede utilizar para realizar un seguimiento, realizar pruebas o volver a enviarlo. Cuando se complete un trabajo, también se rellenarán la fecha de inicio y la fecha de finalización.


```
[
  {
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "HOLD",
    "jobDelay": 0,
    "startDate": null,
    "endDate": null,
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
```

```
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": null,
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 1,
    "jobScheduledId": 0,
    "jobScheduledAt": null
  },
  {
    "jobName": "PTA0044",
    "userName": "USER1",
    "jobNumber": 9,
    "jobPriority": 5,
    "status": "COMPLETED",
    "jobDelay": 0,
    "startDate": "2022-10-13T22:48:34.025+00:00",
    "endDate": "2022-10-13T22:52:54.475+00:00",
    "jobQueue": "queue1",
    "message": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "programName": "PTA0044",
      "programParams": {"wmind": "B"},
      "localDataAreaValue": "",
      "userName": "USER1",
      "jobName": "PTA0044",
      "jobNumber": 9,
      "jobPriority": 5,
      "executionDate": "20181231",
      "jobQueue": "queue1",
      "jobOnHold": true,
      "scheduleCron": "*/20 * * * * ?",
```

```
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
    "executionId": 2,
    "jobScheduledId": 0,
    "jobScheduledAt": null
  }
]
```

Lista de respuestas de trabajos recurrentes

Esta es la estructura del punto final de la cola de the /schedule/list trabajos.

```
[
  {
    "id": 1,
    "status": "ACTIVE",
    "jobNumber": 1,
    "userName": "PVO",
    "msg": {
      "messageQueueName": null,
      "scheduleDate": null,
      "scheduleTime": null,
      "startTime": "2024/03/07 21:12:00.000 UTC",
      "endTime": "2024/03/07 21:13:59.000 UTC",
      "programName": "LOGPGM",
      "programParams": {"cl_sbmjob_param_json": "[\"./output/schedule-job-log.txt\",
 \"Every 20 seconds!\"]"},
      "localDataAreaValue": "",
      "userName": "PVO",
      "jobName": "LOGGERJOB",
      "jobNumber": 1,
      "jobScheduleId": 1,
      "jobPriority": 5,
      "executionDate": null,
      "jobQueue": "queue1",
      "jobOnHold": false,
      "scheduleCron": "*/20 * * * * ?",
      "save": false,
      "scheduleMisfirePolicy": 4,
      "omitdates": null
    },
```

```
    "lastUpdatedAt": "2024-03-07T21:11:13.282+00:00",
    "lastUpdatedBy": ""
  }
]
```

Configurar AWS Blu Age Runtime (no gestionado)

En esta sección se explican los pasos para configurar AWS Blu Age Runtime (no gestionado) en su AWS infraestructura. Antes de configurar su AWS Blu Age Runtime (no gestionado) para las aplicaciones, comprenda los requisitos previos, las regiones y los intervalos, y la configuración de las CloudWatch alarmas para configurar y gestionar su entorno de ejecución.

Temas

- AWS Requisitos previos de Blu Age Runtime
- Incorporación de AWS Blu Age Runtime
- Requisitos de configuración de infraestructura para el tiempo de ejecución de AWS Blu Age (no administrado)
- AWS artefactos de Blu Age Runtime
- Implemente AWS Blu Age Runtime en Amazon EC2
- Implemente AWS Blu Age Runtime en contenedores de Amazon ECS y Amazon EKS
- Prueba la PlanetsDemo aplicación

AWS Requisitos previos de Blu Age Runtime

AWS Blu Age Runtime (no gestionado) está disponible en varias the section called “AWS Notas de lanzamiento de Blu Age” versiones. Si tiene proyectos de modernización en curso, es posible que necesite versiones incrementales del tiempo de ejecución para fines de implementación y pruebas. Para definir sus necesidades, póngase en contacto con su gerente de entrega de AWS Blu Age.

Antes de comenzar el proceso de incorporación de AWS Blu Age Runtime (no gestionado), haga lo siguiente:

- Asegúrese de tener una AWS cuenta.
- Asegúrese de tener una aplicación modernizada rediseñada con Blu Age. AWS
- Elija una AWS región y una de las opciones de procesamiento compatibles con AWS Blu Age Runtime (no administrado).

- Elige la versión de AWS Blu Age Runtime que quieras usar.
- Revise the section called “Requisitos de configuración de infraestructura” y valide los componentes adicionales necesarios para ejecutar el AWS Blu Age Runtime (no administrado).

> **Note**
>
> Si desea probar las funciones de AWS Blu Age Runtime (no gestionado), puede utilizar la aplicación de demostración`Planets Demo`, que puede descargar desde PlanetsDemo - v1.zip.

Incorporación de AWS Blu Age Runtime

Para empezar, crea un AWS Supportestuche para solicitar la incorporación y acceder a AWS Blu Age Runtime. Incluye en tu solicitud tu Cuenta de AWS ID, la AWS región que deseas usar y una opción de procesamiento, además de la versión de AWS Blu Age Runtime. Si no estás seguro de qué versión necesitas, ponte en contacto con tu responsable de entregas de AWS Blu Age. Si ya tiene el código fuente de la aplicación generado por las herramientas de refactorización de la modernización del AWS mainframe, anote el valor de la `gapwalk.version` etiqueta en el `pom.xml` archivo de su base de código modernizada.

> **Note**
>
> El AWS Blu Age Runtime está disponible en dos versiones principales: versiones preliminares y versiones Alpha. Para determinar qué versión utilizarthe section called “AWS Control de versiones de Blu Age”, consulta o ponte en contacto con tu responsable de entregas de AWS Blu Age.

Regiones y segmentos de AWS Blu Age Runtime (no gestionado)

Almacenamos los artefactos de AWS Blu Age Runtime (no gestionados) en diferentes depósitos de Amazon S3 por región y por elección de procesamiento. Para acceder al depósito de AWS Blu Age Runtime (no administrado), usa el nombre que aparece en la siguiente tabla. Región de AWS

Región de AWS	Bucket de versión	Bucket de versión preliminar alfa
Este de EE. UU. (Ohio)	aws-bluage-runtime-artifacts-055777665268-Este-EE. UU.	aws-bluage-runtime-artifacts-dev-055777665268-us-east-2
Este de EE. UU. (Norte de Virginia)	aws-bluage-runtime-artifacts-139023371234-Estados Unidos-Este-1	aws-bluage-runtime-artifacts-dev-139023371234-uso-este-1
Oeste de EE. UU. (Norte de California)	aws-bluage-runtime-artifacts-788454048782-Estados Unidos-Oeste-1	aws-bluage-runtime-artifacts-dev-788454048782-us-west-1
Oeste de EE. UU. (Oregón)	aws-bluage-runtime-artifacts-836771190483-Estados Unidos-Oeste-2	aws-bluage-runtime-artifacts-dev-836771190483-us-west-2
Canadá (centro)	aws-bluage-runtime-artifacts-637423580979-ca-central-1	aws-bluage-runtime-artifacts-dev-637423580979-ca-central-1
Europa (Irlanda)	aws-bluage-runtime-artifacts-925278190477-eu-west-1	aws-bluage-runtime-artifacts-dev-925278190477-eu-west-1
Europa (Londres)	aws-bluage-runtime-artifacts-767397831990-eu-west-1	aws-bluage-runtime-artifacts-dev-767397831990-eu-west-1
Europa (París)	aws-bluage-runtime-artifacts-673009995881-eu-west-3	aws-bluage-runtime-artifacts-dev-673009995881-eu-west-3
Europa (Fráncfort)	aws-bluage-runtime-artifacts-485196800481-eu-central-1	aws-bluage-runtime-artifacts-dev-485196800481-eu-central-1
Europa (Estocolmo)	aws-bluage-runtime-artifacts-654654484534-eu-norte-1	aws-bluage-runtime-artifacts-dev-654654484534-eu-norte-1

Región de AWS	Bucket de versión	Bucket de versión preliminar alfa
Europa (Milán)	aws-bluage-runtime-artifacts-654654328338-eu-sur-1	aws-bluage-runtime-artifacts-dev-654654328338-eu-sur-1
Europa (España)	aws-bluage-runtime-artifacts-905417994954-eu-sur-2	aws-bluage-runtime-artifacts-dev-905417994954-eu-sur-2
América del Sur (São Paulo)	aws-bluage-runtime-artifacts-737536804457-sa-este-1	aws-bluage-runtime-artifacts-dev-737536804457-sa-este-1
Asia-Pacífico (Tokio)	aws-bluage-runtime-artifacts-445578176276-ap-northeast-1	aws-bluage-runtime-artifacts-dev-445578176276-ap-northeast-1
Asia-Pacífico (Seúl)	aws-bluage-runtime-artifacts-381492221498-ap-northeast-2	aws-bluage-runtime-artifacts-dev-381492221498-ap-northeast-2
Asia-Pacífico (Osaka)	aws-bluage-runtime-artifacts-905418229615-ap-northeast-3	aws-bluage-runtime-artifacts-dev-905418229615-ap-northeast-3
Asia-Pacífico (Singapur)	aws-bluage-runtime-artifacts-767397774613-ap-southeast-1	aws-bluage-runtime-artifacts-dev-767397774613-ap-sudeste-1
Asia-Pacífico (Sídney)	aws-bluage-runtime-artifacts-726160321909-ap-southeast-2	aws-bluage-runtime-artifacts-dev-726160321909-ap-sudeste-2
Asia-Pacífico (Bombay)	aws-bluage-runtime-artifacts-905418353577-ap-sur-1	aws-bluage-runtime-artifacts-dev-905418353577-ap-sur-1
África (Ciudad del Cabo)	aws-bluage-runtime-artifacts-992382777663-af-sur-1	aws-bluage-runtime-artifacts-dev-992382777663-af-sur-1

Región de AWS	Bucket de versión	Bucket de versión preliminar alfa
Israel (Tel Aviv)	aws-bluage-runtime-artifacts-471112516508-il-central-1	aws-bluage-runtime-artifacts-dev-471112516508-il-central-1

Se usa para enumerar el contenido del depósito AWS CLI

Una vez incorporado o incorporada, puede ver una lista del contenido del bucket ejecutando el siguiente comando AWS CLI en un terminal.

`aws s3 ls bucket-name`

`bucket-name`Sustitúyalo por el nombre del depósito correspondiente Región de AWS a usted de la tabla anterior.

Este comando devuelve una lista de carpetas que corresponden a diferentes versiones del motor de ejecución (no administrado) de AWS Blu Age Runtime, como la siguiente para un bucket de versiones:

```
PRE 3.10.0/
PRE 4.0.0/
```

O lo siguiente para un bucket de compilación:

```
PRE 4.1.0-alpha.8/
PRE 4.1.0-alpha.9/
```

Le recomendamos que utilice la última versión disponible. Si eso no es posible, utilice la versión de tiempo de ejecución que se validó durante la fase de refactorización de la aplicación. Para ver una lista de los marcos disponibles para una versión específica, ejecute el siguiente comando:

`aws s3 ls s3://bucket-name/version/Framework/`

`bucket-name`Sustitúyelo por el nombre del depósito correspondiente a su versión Región de AWS y `version` por el nombre que desee. A continuación se muestran dos ejemplos:

Para un bucket de versión:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/
```

El comando devuelve una lista de marcos, como la siguiente:

```
2024-04-08 16:11:19  152040176 aws-bluage-runtime-4.0.0.tar.gz
2024-04-08 16:11:50         45 aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256
2024-04-08 16:11:52  176518889 aws-bluage-webapps-4.0.0.tar.gz
2024-04-08 16:12:28         45 aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256
```

Para un bucket de compilación:

```
aws s3 ls s3://aws-bluage-runtime-artifacts-dev-139023371234-us-
east-1/4.1.0-alpha.9/Framework/
```

El comando devuelve una lista de marcos, como la siguiente:

```
2024-04-09 20:23:34  152304534 aws-bluage-runtime-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:05         45 aws-bluage-runtime-4.1.0-alpha.9.tar.gz.checksumSHA256
2024-04-09 20:24:07  176262381 aws-bluage-webapps-4.1.0-alpha.9.tar.gz
2024-04-09 20:24:42         45 aws-bluage-webapps-4.1.0-alpha.9.tar.gz.checksumSHA256
```

Descarga del marco

Puede descargar el marco, por ejemplo, para actualizar la versión de AWS Blu Age Runtime en una EC2 instancia de Amazon existente.

```
aws s3 cp s3://bucket-name/version/Framework/ folder-of-your-choice --
recursive
```

Donde:

folder-of-your-choice

ruta de la carpeta en la que vaya a descargar el marco.

Por ejemplo: `aws s3 cp s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/ . --recursive`

Este comando produce el resultado siguiente:

```
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-runtime-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/
Framework/aws-bluage-webapps-4.0.0.tar.gz.checksumSHA256 to ./aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-webapps-4.0.0.tar.gz to ./aws-bluage-webapps-4.0.0.tar.gz
download: s3://aws-bluage-runtime-artifacts-139023371234-us-east-1/4.0.0/Framework/aws-
bluage-runtime-4.0.0.tar.gz to ./aws-bluage-runtime-4.0.0.tar.gz
```

Puede obtener una lista de los archivos del marco de la siguiente manera:

```
ls -l
```

Este comando produce el resultado siguiente:

```
total 230928
-rw-rw-r-- 1 cloudshell-user cloudshell-user 152040176 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:11 aws-bluage-
runtime-4.0.0.tar.gz.checksumSHA256
-rw-rw-r-- 1 cloudshell-user cloudshell-user 176518889 Apr  8 16:11 aws-bluage-
webapps-4.0.0.tar.gz
-rw-rw-r-- 1 cloudshell-user cloudshell-user        45 Apr  8 16:12 aws-bluage-
webapps-4.0.0.tar.gz.checksumSHA256
```

> **Note**
>
> El acceso a los artefactos puede interrumpirse temporalmente y las versiones pueden eliminarse por motivos de seguridad. Te recomendamos encarecidamente que guardes los artefactos que utilices en tu propia cuenta. La versión local debe usarse como referencia en sus arquitecturas internas.

Requisitos de configuración de infraestructura para el tiempo de ejecución de AWS Blu Age (no administrado)

En este tema se describe la configuración de infraestructura mínima necesaria para ejecutar AWS Blu Age Runtime (no gestionado). Los siguientes procedimientos describen cómo configurar AWS

Blu Age Runtime (no administrado) en el equipo que prefiera para implementar una aplicación modernizada en AWS Blu Age Runtime. Los recursos que cree deben estar en una Amazon VPC que tenga una subred dedicada al dominio de su aplicación.

Temas

- Requisitos de infraestructura
- Tipos de EC2 instancias de Amazon para AWS Blu Age Runtime (en Amazon EC2)
- Ejecutando AWS Blu Age Runtime en Amazon EC2
- Ejecución de AWS Blu Age Runtime en Amazon ECS en Amazon EC2
- Ejecutar AWS Blu Age Runtime en Amazon EKS en Amazon EC2
- Ejecución de AWS Blu Age Runtime en Amazon ECS gestionado por AWS Fargate

Requisitos de infraestructura

Creación de un grupo de seguridad

Si planea trabajar con EC2 instancias de Amazon en Amazon EKS, omita este procedimiento porque el proceso de creación del clúster de Amazon EKS crea un grupo de seguridad en su nombre. Utilice ese grupo de seguridad en los siguientes procedimientos en lugar de crear otro nuevo.

1. Abra la consola de Amazon VPC en https://console.aws.amazon.com/vpc/.
2. En el panel de navegación izquierdo, en Seguridad, elija Grupos de seguridad.
3. En el panel central, elija Crear grupos de seguridad.
4. En el campo Nombre del grupo de seguridad, escriba **`M2BluagePrivateLink-SG`**.
5. En la sección Inbound rules (Reglas de entrada), elija Add rule (agregar regla).
6. En Tipo, seleccione HTTPS.
7. En Origen, introduzca el CIDR de su VPC.
8. En la sección Reglas salientes, elija Agregar regla.
9. En Tipo, seleccione HTTPS.
10. En Destino, escriba **`0.0.0.0/0`**.
11. Elija Creación de grupo de seguridad.

Crear un punto de conexión de VPC de Amazon

1. Abra la consola de Amazon VPC en https://console.aws.amazon.com/vpc/.
2. En el panel de navegación izquierdo, en Nube virtual privada, seleccione Puntos de conexión.
3. En el panel central, elija Crear punto de conexión.
4. En la sección Servicios, introduzca **SQS** en el campo de búsqueda y, a continuación, seleccione el servicio de Amazon SQS que corresponda a su región.
5. En VPC, seleccione la Amazon VPC que creó en el paso anterior.
6. En la sección Subredes, seleccione la subred que creó para el dominio de su aplicación.
7. En la sección Grupos de seguridad, seleccione el grupo de seguridad del procedimiento anterior.
8. Elija Crear punto de conexión.

Creación de una política de IAM

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación izquierdo, en Administración de acceso, seleccione Políticas.
3. En el panel central, elija Crear política.
4. En la sección Editor de políticas, elija JSON.
5. Sustituya todo el JSON que vea en el editor por el siguiente JSON.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": [
                "sqs:GetQueueUrl",
                "sqs:ReceiveMessage",
                "sqs:SendMessage"
            ],
            "Resource": "*"
        }
    ]
}
```

> **Note**
>
> Si necesitas más detalles para personalizar tu política, ponte en contacto con tu gerente de entregas o de cuentas de AWS Blu Age.

6. Elija Siguiente.
7. Escriba un nombre para la política y elija Crear política.

Creación de un rol de IAM

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, en Administración del acceso, elija Roles.
3. En el panel central, elija Crear rol.
4. En la sección Casos de uso, en función de la opción de computación que elija, seleccione una de las siguientes opciones:
   - EC2(para Amazon EC2 y Amazon EKS en Amazon EC2)
   - Elastic Container Service y, después, EC2Rol para Elastic Container Service (para Amazon ECS en Amazon EC2)
   - Elastic Container Service y, a continuación, Tarea de Elastic Container Service (para Amazon ECS administrado por Fargate)
5. Elija Siguiente.
6. En el campo de búsqueda, introduzca el nombre de la política que creó antes.
7. Seleccione la casilla de verificación a la izquierda de su política.

> **Note**
>
> Si no puede agregar una política, termine de crear el rol y, a continuación, actualícelo para agregar la política.

8. Elija Siguiente.
9. Ingrese un nombre para el rol y, a continuación, seleccione Crear rol.

Tipos de EC2 instancias de Amazon para AWS Blu Age Runtime (en Amazon EC2)

La siguiente es una lista de los tipos de EC2 instancias de Amazon que puede usar para AWS Blu Age Runtime (en Amazon EC2) al crear EC2 instancias de Amazon o al definir los nodos de trabajo de Amazon EKS.

Compruebe que la instancia que le interesa esté disponible en la región que desee implementar.

```
t3.small
t3.medium
t3.large
t3.xlarge
t3.2xlarge
t2.small
t2.medium
t2.large
t2.xlarge
t2.2xlarge
r7a.medium
r7a.large
r7a.xlarge
r7a.2xlarge
r7a.4xlarge
r7a.8xlarge
r7a.12xlarge
r7a.16xlarge
r7a.24xlarge
r7a.32xlarge
r7a.48xlarge
r7a.metal-48xl
r7i.large
r7i.xlarge
r7i.2xlarge
r7i.4xlarge
r7i.8xlarge
r7i.12xlarge
r7i.16xlarge
r7i.24xlarge
r7i.48xlarge
r7i.metal-24xl
r7i.metal-48xl
r6i.xlarge
r6i.large
```

```
r6i.4xlarge
r6i.2xlarge
r5b.xlarge
r5b.large
r5b.2xlarge
r3.xlarge
m6i.xlarge
m6i.large
m6i.8xlarge
m6i.4xlarge
m6i.2xlarge
m6i.16xlarge
m5zn.xlarge
m5zn.large
m5zn.3xlarge
m5zn.2xlarge
m5.xlarge
m5.large
m5.8xlarge
m5.4xlarge
m5.2xlarge
m5.16xlarge
m5.12xlarge
c6i.xlarge
c6i.large
c6i.8xlarge
c6i.4xlarge
c6i.2xlarge
c6i.16xlarge
c5.xlarge
c5.large
c5.9xlarge
c5.4xlarge
c5.2xlarge
c5.18xlarge
c5.12xlarge
```

Ejecutando AWS Blu Age Runtime en Amazon EC2

Para crear una EC2 instancia de Amazon, sigue estos pasos.

Crear una EC2 instancia de Amazon

1. Abre la EC2 consola de Amazon en https://console.aws.amazon.com/ec2/.
2. Seleccione Iniciar instancia.
3. En Tipo de instancia, seleccione uno de los tipos que se enumeran en the section called “Tipos de EC2 instancias de Amazon para AWS Blu Age Runtime (en Amazon EC2)”.
4. En la sección Par de claves, seleccione o bien un par de claves existente o bien cree uno nuevo.
5. En la sección Configuración de red, elija Seleccionar grupo de seguridad existente.
6. Para los grupos de seguridad comunes, elija M2 BluagePrivateLink -SG.
7. Amplíe la sección Detalles avanzados.
8. En Perfil de instancia de IAM, seleccione el rol de IAM que creó antes.
9. Seleccione Iniciar instancia.

Instalar la aplicación en la EC2 instancia de Amazon

1. Cuando el estado de la EC2 instancia de Amazon cambie a En ejecución, conéctese a la instancia.
2. Instale los siguientes componentes de software en la instancia:

   - Entorno de ejecución de Java (JRE) 17.
   - Apache Tomcat 10.
   - AWS Blu Age Runtime (en Amazon EC2). Instale el motor de ejecución AWS Blu Age en la raíz de la carpeta de instalación de Apache Tomcat (algunos archivos se añadirán y otros se sobrescribirán).

   Para instalar las aplicaciones web adicionales que se incluyen con el archivo AWS Blu Age Runtime, configure una instancia secundaria del servidor Apache Tomcat y descomprima el archivo de aplicaciones web en esa ubicación. Para obtener instrucciones detalladas, consulta the section called “AWS artefactos de Blu Age Runtime”.

Ejecución de AWS Blu Age Runtime en Amazon ECS en Amazon EC2

1. Cree un clúster de Amazon ECS con EC2 instancias de Amazon como infraestructura subyacente. Consulte Introducción a Windows en Amazon EC2 en la Guía para desarrolladores de Amazon Elastic Container Service.
2. Especifique el rol de IAM que creó en los pasos anteriores.
3. Elija uno de los tipos de instancia que se enumeran en the section called “Tipos de EC2 instancias de Amazon para AWS Blu Age Runtime (en Amazon EC2)”.
4. En Configuración de red para EC2 instancias de Amazon, elige el grupo de seguridad que creaste en los pasos anteriores.

Ejecutar AWS Blu Age Runtime en Amazon EKS en Amazon EC2

1. Cree un clúster de Amazon EKS. Consulte Creación de un clúster de Amazon EKS en la Guía del usuario de Amazon EKS.
2. Como se ha mencionado anteriormente, se crea un grupo de seguridad en su nombre. Puede utilizar ese grupo de seguridad al crear el punto de conexión de VPC de Amazon.
3. Cree un grupo de nodos. Especifique el rol de IAM que creó en los pasos anteriores.
4. Elija uno de los tipos de instancia que se enumeran en the section called “Tipos de EC2 instancias de Amazon para AWS Blu Age Runtime (en Amazon EC2)”.
5. Amazon EKS asignará automáticamente el grupo de seguridad a las EC2 instancias de Amazon generadas.

Ejecución de AWS Blu Age Runtime en Amazon ECS gestionado por AWS Fargate

Cree un clúster de Amazon ECS con AWS Fargate (sin servidor) como infraestructura subyacente. Consulte Introducción a Fargate en la Amazon Elastic Container Service Developer Guide.

AWS artefactos de Blu Age Runtime

AWS Los artefactos de Blu Age Runtime son los componentes para implementar y ejecutar aplicaciones modernizadas. Este documento describe los diferentes tipos de artefactos disponibles, sus ubicaciones de almacenamiento y cómo acceder a ellos.

AWS Artefactos de Blu Age Runtime (no gestionados)

Acceso y almacenamiento de artefactos

Los artefactos de AWS Blu Age Runtime para las implementaciones no administradas se almacenan en depósitos S3 específicos de la región. Cada versión tiene su propia carpeta dedicada, lo que facilita la administración y el acceso a las versiones.

Hay dos tipos de depósitos:

Cubetas de liberación

Los buckets de lanzamiento contienen directorios para las versiones implementadas más recientemente y siguen la convención de nomenclatura:. `aws-bluage-runtime-artifacts-<accountId>-<region>`

Depósitos previos al lanzamiento

Los cubos de versiones preliminares contienen directorios para las versiones alfa correspondientes a las últimas versiones preliminares incrementales de corta duración y siguen la convención de nomenclatura:. `convention: aws-bluage-runtime-artifacts-dev-<accountId>-<region>`

El acceso a los paquetes de producción y de prelanzamiento se otorga de forma independiente. Para obtener más información sobre cómo solicitar el acceso y más detalles sobre la organización de los buckets de S3, consulte. the section called “Incorporación de AWS Blu Age Runtime ”

Contenido del artefacto

Tanto en la sección de lanzamiento como en la versión preliminar, encontrarás:

aws-bluage-runtime-x.y.z.tar.gz

Este archivo, donde x.y.z representa el número de versión (`major.minor.patch`según el control de versiones semántico, consultethe section called “AWS Control de versiones de Blu Age”), y contiene los componentes principales de Blu Age Runtime esenciales para ejecutar aplicaciones de AWS Blu Age, entre los que se incluyen: AWS

- Gapwalk: un componente crucial del AWS Blu Age Runtime, diseñado para cerrar la brecha entre las aplicaciones heredadas y los entornos modernos nativos de la nube. Sirve como capa de compatibilidad que permite que las aplicaciones modernizadas por AWS Blu Age se ejecuten de forma eficaz en las plataformas contemporáneas.

- bluage.bin: El archivo binario principal del motor de ejecución de AWS Blu Age. Este archivo es fundamental para el funcionamiento del motor de ejecución.
- Todas las bibliotecas y archivos de soporte necesarios para el funcionamiento de AWS Blu Age Runtime.

aws-bluage-webapps-x.y.z.tar.gz

Este archivo, en el que x.y.z sigue el mismo esquema de control de versiones que el anterior, incluye las aplicaciones web y las bibliotecas necesarias para gestionar y controlar las implementaciones de Blu Age: AWS

- Archivo WAR de BAC (consola Blusam), que se utiliza para supervisar la base de datos de Blusam.
- Archivo WAR JAC (consola JICS), utilizado para monitorear la base de datos JICS.
- Bibliotecas de apoyo necesarias.

Archivos adicionales

- Archivos de suma de verificación que permiten verificar la integridad de ambos archivos de Blu Age siguiendo la convención de nomenclatura:
  - Para Runtime: `aws-bluage-runtime-x.y.z.tar.gz.checksumSHA256`
  - Para aplicaciones web: `aws-bluage-webapps-x.y.z.tar.gz.checksumSHA256`
- Los archivos de informes CVE (solo para las versiones de lanzamiento) muestran lo presente CVEs en esta versión y siguen la convención de nomenclatura:
  - Para Runtime: `Bluage-Runtime-x.y.z-CVEs.txt`
  - Para aplicaciones web: `Bluage-Webapps-x.y.z-CVEs.txt`

Para obtener más información sobre cómo se abordan las vulnerabilidades de seguridad, consulte la descripción general de la versión AWS Mainframe Modernization Refactorization with AWS Blu Age.

> **Note**
>
> Si bien nos esforzamos por lanzar nuestros productos sin ellos CVEs, es CVEs posible que aparezcan nuevos productos más adelante. El archivo del informe CVE se actualiza periódicamente para reflejar el estado más reciente.

Desarrollador: AWS Blu Age Runtime Artifacts

Acceso y almacenamiento de artefactos

Los artefactos de AWS Blu Age Developer Runtime se almacenan en depósitos S3 dedicados. Este tiempo de ejecución incluye la versión preliminar y la versión preliminar de la versión Alpha. El acceso a estos artefactos se gestiona mediante las solicitudes de la caja de herramientas de AWS Blu Age. Una vez que se procese y apruebe tu solicitud, se te concederá acceso a la sección correspondiente de entre las Cuenta de AWS especificadas en tu solicitud.

Grupo de tiempo de ejecución para desarrolladores

El segmento principal del Developer Runtime es: `s3://toolbox-dev-runtime`

Para obtener información más detallada sobre la solicitud de acceso y la comprensión de la estructura de los buckets, consulte la documentación especial y de desarrolladores de AWS Blu Age Runtimes.

Contenido del artefacto

Los artefactos relacionados con el tiempo de ejecución para desarrolladores suelen incluir:

gapwalk-x.y.z-dev.tar.gz

Este archivo contiene la versión de desarrollo del componente Gapwalk, que es una parte crucial del Blu Age Runtime. AWS Está diseñado para unir las aplicaciones heredadas con los entornos modernos nativos de la nube.

gapwalk-runtime-x.y.z-javadoc.zip

Este archivo zip contiene la documentación del entorno de ejecución de Gapwalk. JavaDoc JavaDocproporciona documentación API detallada, que es especialmente útil para los desarrolladores que trabajan en la integración o ampliación del tiempo de ejecución de Gapwalk.

gapwalk-webapps-x.y.z-javadoc.zip

Al igual que el motor de ejecución JavaDoc, este archivo zip contiene la JavaDoc documentación específica para las aplicaciones web de Gapwalk. Esta documentación es crucial para los desarrolladores que trabajan con los componentes basados en la web del sistema Gapwalk o los personalizan.

Implemente AWS Blu Age Runtime en Amazon EC2

En los temas de esta sección, encontrarás información sobre cómo configurar AWS Blu Age Runtime (no gestionado) en Amazon EC2, cómo actualizar la versión en tiempo de ejecución, cómo supervisar tu despliegue mediante CloudWatch alarmas de Amazon y cómo añadir dependencias con licencia. Estas instrucciones se aplican tanto al crear EC2 instancias de Amazon como al utilizar Amazon ECS en Amazon EC2 o Amazon EKS en Amazon EC2.

Temas

- Configurar AWS Blu Age Runtime (no gestionado) en Amazon EC2
- Actualiza el tiempo de ejecución de AWS Blu Age en Amazon EC2
- Configura las CloudWatch alarmas de Amazon de AWS Blu Age Runtime (en Amazon EC2)
- Configura dependencias con licencia en AWS Blu Age Runtime en Amazon EC2

Configurar AWS Blu Age Runtime (no gestionado) en Amazon EC2

En este tema se explica cómo configurar e implementar la aplicación de PlanetsDemo ejemplo mediante AWS Blu Age Runtime (no gestionado) en Amazon EC2.

Temas

- Requisitos previos
- Configuración
- Prueba de la aplicación implementada

Requisitos previos

Antes de comenzar, asegúrese de que cumple los requisitos previos que se indican a continuación:

- Configure el AWS CLI archivo siguiendo los pasos que se indican en Configuración de la AWS CLI.
- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Crea una EC2 instancia de Amazon con uno de los tipos de instancias compatibles. Para obtener más información, consulte Introducción a las instancias de Amazon EC2 Linux.
- Asegúrese de poder conectarse a la EC2 instancia de Amazon correctamente, por ejemplo, mediante SSM.

> **Note**
>
> A lo largo de esta guía, se asume que la ruta de instalación de Tomcat es. `/m2-anywhere/tomcat-gapwalk/velocity` Asegúrese de utilizar esta ruta cuando siga las instrucciones que aparecen a continuación o adapte la siguiente instrucción a la ruta que elija.

- Descarga y extrae AWS Blu Age Runtime (en Amazon EC2). Copia el contenido del directorio velocity a`/m2-anywhere/tomcat-gapwalk/velocity`. Asegúrese de colocar el `bluage.bin` archivo exactamente en la ubicación especificada por la variable de entorno CATALINA_HOME descrita en CATALINA_HOME y CATALINA_BASE en la documentación de Apache Tomcat. Para obtener instrucciones sobre cómo recuperar los artefactos de AWS Blu Age Runtime, incluida información sobre el almacenamiento, el acceso y el contenido, consulte. the section called “AWS artefactos de Blu Age Runtime”
- Descargue el archivo PlanetsDemo de la aplicación.
- Descomprima el archivo y cargue la aplicación en el bucket de Amazon S3 de su elección.
- Cree una base de datos PostgreSQL de Amazon Aurora para JICS. El motor de ejecución de AWS Blu Age ejecutará automáticamente el `PlanetsDemo-v1/jics/sql/initJics.sql` script durante el primer inicio. Para obtener información sobre cómo crear una base de datos de PostgreSQL de Amazon Aurora, consulte Creación de un clúster de base de datos de Aurora PostgreSQL y conexión a él.

Configuración

Para configurar la aplicación PlanetsDemo de muestra, complete los siguientes pasos.

1. Conéctese a su EC2 instancia de Amazon y vaya a la `conf` carpeta situada debajo de la carpeta de instalación de Apache Tomcat 10. Abra el archivo `catalina.properties` para editarlo y sustituya la línea que empieza por `common.loader` por la siguiente línea.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Vaya a la carpeta `/m2-anywhere/tomcat-gapwalk/velocity /webapps/webapps`.

3. Copie los PlanetsDemo archivos binarios disponibles en la `PlanetsDemo-v1/webapps/` carpeta del bucket de Amazon S3 mediante el siguiente comando.

```
aws s3 cp s3://path-to-demo-app-webapps/ . --recursive
```

> **Note**
>
> `path-to-demo-app-webapps`Sustitúyalo por el URI de Amazon S3 correcto para el depósito en el que descomprimió el PlanetsDemo archivo anteriormente.

4. Copie el contenido de la carpeta `PlanetsDemo-v1/config/` a `/m2-anywhere/tomcat-gapwalk/velocity /config/`.
5. Proporcione la información de conexión para la base de datos que creó como parte de los requisitos previos en el siguiente fragmento de código del archivo `application-main.yml`. Para obtener más información, consulte Creación de un clúster de base de datos de Aurora PostgreSQL y conexión a él.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

6. Inicie el servidor de Apache Tomcat y compruebe los registros.

```
/m2-anywhere/tomcat-gapwalk/velocity/startup.sh

tail -f /m2-anywhere/tomcat-gapwalk/velocity/logs/catalina.log
```

Si encuentra códigos de error que empiecen por una C seguida de un número, como CXXXX, anote los mensajes de error. Por ejemplo, el código de error C5102 es un error común que indica una configuración de infraestructura incorrecta.

Prueba de la aplicación implementada

Para ver un ejemplo de cómo probar la PlanetsDemo aplicación, consulte. the section called “Prueba la PlanetsDemo aplicación”

Actualiza el tiempo de ejecución de AWS Blu Age en Amazon EC2

Esta guía describe cómo actualizar el tiempo de ejecución de AWS Blu Age en Amazon EC2.

Temas

- Requisitos previos
- Actualice el tiempo de ejecución de AWS Blu Age en la EC2 instancia de Amazon
- Actualice el tiempo de ejecución de AWS Blu Age en un contenedor

Requisitos previos

Antes de comenzar, asegúrese de que cumple los siguientes requisitos previos:

- Para comprobar si hay instrucciones específicas para su versión, consulte the section called “Actualización de AWS Blu Age”.
- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Asegúrese de tener una EC2 instancia de Amazon que contenga la última versión de AWS Blu Age Runtime. Para obtener más información, consulte Introducción a las instancias de Amazon EC2 Linux.
- Asegúrese de poder conectarse a la EC2 instancia de Amazon correctamente, por ejemplo, mediante SSM.
- Descarga la versión de AWS Blu Age Runtime a la que deseas actualizar. Para obtener más información, consulte the section called “ Configurar AWS Blu Age Runtime (no gestionado)” El marco consta de dos archivos binarios: `aws-bluage-runtime-x.x.x.x.tar.gz` y `aws-bluage-webapps-x.x.x.x.tar.gz`.

Actualice el tiempo de ejecución de AWS Blu Age en la EC2 instancia de Amazon

Complete los siguientes pasos para actualizar el tiempo de ejecución de AWS Blu Age.

1. Conéctate a tu EC2 instancia de Amazon y cambia el usuario a su ejecutando el siguiente comando.

```
sudo su
```

En este tutorial, necesitará el privilegio de superusuario para ejecutar los comandos.

2. Cree dos carpetas, una para cada archivo binario.
3. Dé a cada carpeta el nombre del archivo binario.
4. Copie cada archivo binario en la carpeta correspondiente.

> **Warning**
>
> Al extraer cada binario, se obtiene una carpeta con el mismo nombre. Por lo tanto, si extrae ambos archivos binarios en la misma ubicación uno tras otro, sobrescribirá el contenido.

5. Para extraer los archivos binarios, utilice los siguientes comandos: Ejecute los comandos en cada carpeta:

```
tar xvf aws-bluage-runtime-x.x.x.x.tar.gz
tar xvf aws-bluage-webapps-x.x.x.x.tar.gz
```

6. Detenga los servicios de Apache Tomcat con los siguientes comandos.

```
systemctl stop tomcat.service
systemctl stop tomcat-webapps.service
```

7. Sustituya el contenido de `<your-tomcat-path>/shared/` por el contenido de `aws-bluage-runtime-x.x.x.x/velocity/shared/`.
8. Sustituya `<your-tomcat-path>/webapps/gapwalk-application.war` por `aws-bluage-runtime-x.x.x.x/velocity/webapps/gapwalk-application.war`.
9. Reemplace los archivos war de `<your-tomcat-path>/webapps/`, es decir `bac.war` y `jac.war`, por los mismos archivos de `aws-bluage-webapps-x.x.x.x/velocity/webapps/`.
10. Inicie los servicios de Apache Tomcat ejecutando los siguientes comandos.

```
systemctl start tomcat.service
systemctl start tomcat-webapps.service
```

11. Consulte los registros.

Para comprobar el estado de la aplicación implementada, ejecute los siguientes comandos.

```
curl http://localhost:8080/gapwalk-application/
```

Debería aparecer el siguiente mensaje.

```
Jics application is running
```

```
curl http://localhost:8181/jac/api/services/rest/jicsservice/
```

Debería aparecer el siguiente mensaje.

```
Jics application is running
```

```
curl http://localhost:8181/bac/api/services/rest/bluesamserver/serverIsUp
```

La respuesta debe estar vacía.

El tiempo de ejecución de AWS Blu Age se actualizó correctamente.

Actualice el tiempo de ejecución de AWS Blu Age en un contenedor

Complete los siguientes pasos para actualizar el tiempo de ejecución de AWS Blu Age.

1. Reconstruye tu imagen de Docker con la versión de AWS Blu Age Runtime que desees. Para obtener instrucciones, consulte the section called “Configurar AWS Blu Age Runtime (no gestionado) en Amazon EC2”.
2. Envíe la imagen de Docker a su repositorio de Amazon ECR.
3. Detenga y reinicie el servicio de Amazon ECS o Amazon EKS.
4. Consulte los registros.

El tiempo de ejecución de AWS Blu Age se actualizó correctamente.

Configura las CloudWatch alarmas de Amazon de AWS Blu Age Runtime (en Amazon EC2)

Puede configurarlo CloudWatch para recibir el registro de su aplicación y añadir una alarma que le avise de posibles errores. Esto le permite tener notificaciones más visibles siempre que las

aplicaciones implementadas encuentren excepciones. Las siguientes secciones le ayudarán a comprender y obtener información sobre la configuración del CloudWatch registro y la configuración de las alarmas.

Despliegue del CloudWatch registro

De forma predeterminada, el motor de ejecución de AWS Blu Age contiene un archivo de registro denominado`logback-cloudwatch.yml`. A este archivo se hace referencia en el archivo `application-main.yml`, pero esta referencia está comentada.

```
# logging:
#  config: classpath:logback-cloudwatch.xml
```

Ambos archivos están en la carpeta de configuración y, al eliminar los comentarios de las líneas anteriores, se puede activar la función. CloudWatch el registro se puede configurar, como se explica en las siguientes secciones.

Configuración del CloudWatch registro

El archivo `logback-cloudwatch.xml` predeterminado tiene el siguiente contenido.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE configuration>
<configuration>

    <appender name="console" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </encoder>
    </appender>

    <appender name="cloudwatch"
 class="com.netfective.bluage.runtime.cloudwatchlogger.CloudWatchAppender">
        <logGroup>BluAgeRuntimeOnEC2-Logs</logGroup>
        <logStream>%date{yyyy-MM-dd,UTC}.%instanceId.%uuid</logStream>
        <layout>
            <pattern>%date{yyyy-MM-dd HH:mm:ss.SSS,UTC}  %level --- [%thread{15}]
 %logger{40} : %msg%n%xThrowable</pattern>
        </layout>
        <appender-ref ref="console" />
```

```
    </appender>

    <root level="INFO">
        <appender-ref ref="cloudwatch" />
    </root>
</configuration>
```

Todo lo que está fuera del elemento <appender name="cloudwatch"/> es la configuración estándar de logback. Hay dos anexadores en este archivo: un anexador de consola para enviar los registros a la consola y un CloudWatch anexador al que enviar los registros. CloudWatch

El atributo de `level` del elemento `root` especifica el nivel de registro de toda la aplicación.

Los valores obligatorios dentro de la etiqueta <appender name="cloudwatch"/> son:

- <logGroup/>:Establece el nombre del grupo de registros en CloudWatch. Si no se especifica ningún valor, el predeterminado es `BluAgeRuntimeOnEC2-Logs`. Si el grupo de registro no existe, se creará automáticamente. Este comportamiento se puede cambiar mediante la configuración, que se describe a continuación.
- <logStream/>: Establece el nombre del LogStream (dentro del grupo de registros) en. CloudWatch

Valores opcionales:

- <region/>: sobrescribe la región en la que se escribirá el flujo de registro. De forma predeterminada, los registros van a la misma región que la EC2 instancia.
- <layout/>: el patrón que utilizarán los mensajes de registro.
- <maxbatchsize/>: el número máximo de mensajes de registro a los que se debe enviar CloudWatch por operación.
- <maxbatchtimemillis/>: el tiempo en milisegundos que se permite escribir CloudWatch los registros.
- <maxqueuewaittimemillis/>: el tiempo en milisegundos que se tarda en intentar insertar las solicitudes en la cola de registros interna.
- <internalqueuesize/>: el tamaño máximo de la cola interna.
- <createlogdests/>: crea un grupo y un flujo de registro si no existen.
- <initialwaittimemillis/>: la cantidad de tiempo que el subproceso debe permanecer en reposo al startup. Esta espera inicial permite una acumulación inicial de registros.

- <maxeventmessagesize/>: el tamaño máximo de un evento de registro. Los registros que superen este tamaño no se enviarán.
- <truncateeventmessages/>: trunca los mensajes que son demasiado largos.
- <printrejectedevents/>: habilita el appender de emergencia.

CloudWatch configuración

Para que la configuración anterior envíe los registros correctamente CloudWatch, actualice el rol de perfil de su instancia de Amazon EC2 IAM para concederle permisos adicionales para el grupo de registros `BluAgeRuntimeOnEC2-Logs` y sus flujos de registros:

- `logs:CreateLogStream`
- `logs:DescribeLogStreams`
- `logs:CreateLogGroup`
- `logs:PutLogEvents`
- `logs:DescribeLogGroups`

Configuración de alarmas

Gracias a CloudWatch los registros, puede configurar diferentes métricas y alarmas, en función de su aplicación y sus necesidades. En concreto, puede configurar alarmas proactivas para las alertas de uso, a fin de recibir alertas en caso de que se produzcan errores que puedan poner a su aplicación en un período de gracia (y, en última instancia, impedir que funcione). Para ello, puede añadir en los registros una métrica relativa a la cadena «Error C5001", que muestre los errores en la conexión con el sistema de control AWS Blu Age. A continuación, puede definir una alarma que reaccione a esta métrica.

Configura dependencias con licencia en AWS Blu Age Runtime en Amazon EC2

Esta guía describe cómo configurar dependencias con licencia adicionales que puedes usar con AWS Blu Age Runtime en Amazon EC2.

Temas

- Requisitos previos
- Descripción general
- Configure las dependencias para las aplicaciones web JAC y BAC

Requisitos previos

Antes de comenzar, asegúrese de que cumple los requisitos previos que se indican a continuación:

- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Asegúrate de tener una EC2 instancia de Amazon que contenga la última versión de AWS Blu Age Runtime (en Amazon EC2). Para obtener más información, consulte Introducción a las instancias de Amazon EC2 Linux.
- Asegúrese de poder conectarse a la EC2 instancia de Amazon correctamente, por ejemplo, mediante SSM.
- Obtenga las siguientes dependencias de sus orígenes.

Base de datos de Oracle

Proporcione un controlador de base de datos de Oracle. Probamos la funcionalidad de AWS Blu Age Runtime (en Amazon EC2) con la versión ojdbc11-23.3.0.23.09.jar, pero podría ser compatible con una versión más reciente.

Conexión MQ de IBM

Suministre un cliente MQ de IBM. Probamos la funcionalidad de AWS Blu Age Runtime (en Amazon EC2) con la versión com.ibm.mq.jakarta.client-9.3.4.1.jar, pero podría ser compatible con una versión más reciente.

Con esta versión de dependencias, suministre también las siguientes dependencias transitivas:

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Archivos de impresora DDS

Proporcione la biblioteca de informes de Jasper https://com (munity.jaspersoft). com/download-jaspersoft/community-edición). Probamos la funcionalidad de AWS Blu Age Runtime (en Amazon EC2) con jasperreports-6.16.0.jar, pero es posible que una versión más reciente sea compatible.

Con esta versión de dependencias, suministre también las siguientes dependencias transitivas:

- castor-core-1.4.1. jar
- castor-core-1.4.1. jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7. js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar
- commons-beanutils-1.9.4.jar
- commons-collections-3.2.2.jar

Descripción general

Para instalar las dependencias, complete los pasos siguientes.

1. Conéctate a tu EC2 instancia de Amazon y cambia el usuario a su ejecutando el siguiente comando.

   ```
   sudo su
   ```

   En este tutorial, necesitará el privilegio de superusuario para ejecutar los comandos.
2. Vaya a la carpeta `<your-tomcat-path>/extra/`.

   ```
   cd <your-tomcat-path>/extra/
   ```
3. Copie cualquiera de las dependencias anteriores según sea necesario en esta carpeta.
4. Pare e inicie el tomcat.service ejecutando los siguientes comandos.

   ```
   systemctl stop tomcat.service
   ```

   ```
   systemctl start tomcat.service
   ```
5. Compruebe el estado del servicio para asegurarse de que se está ejecutando.

   ```
   systemctl status tomcat.service
   ```

6. Verifique los registros.

Configure las dependencias para las aplicaciones web JAC y BAC

1. Si su base de datos JICS está alojada en Oracle, debe proporcionar el controlador de la base de datos de Oracle. `<your-tomcat-path>/extra`
2. Cree la carpeta si aún no está presente.
3. Detenga y reinicie el servidor de Apache Tomcat.
4. Verifique los registros.

Implemente AWS Blu Age Runtime en contenedores de Amazon ECS y Amazon EKS

Puede utilizar los temas de esta sección para aprender a configurar AWS Blu Age Runtime en contenedores para implementarlo en Amazon ECS (gestionado por Amazon EC2 o AWS Fargate) y Amazon EKS gestionado por Amazon EC2, cómo actualizar la versión en tiempo de ejecución, cómo supervisar el despliegue mediante CloudWatch las alarmas de Amazon y cómo añadir dependencias con licencia.

> **Note**
>
> Esto no es compatible con Amazon EKS gestionado por AWS Fargate.

Temas

- Configurar AWS Blu Age Runtime en un contenedor
- Actualice el tiempo de ejecución de AWS Blu Age en un contenedor
- Configura CloudWatch las alarmas de Amazon para AWS Blu Age Runtime en un contenedor
- Configure las dependencias con licencia en AWS Blu Age Runtime en un contenedor

Configurar AWS Blu Age Runtime en un contenedor

En este tema se explica cómo configurar e implementar la aplicación de PlanetsDemo ejemplo mediante AWS Blu Age Runtime en un contenedor docker.

AWS Blu Age Runtime en contenedor está disponible para Amazon ECS gestionado por Amazon EC2, Amazon ECS gestionado por AWS Fargate Amazon y Amazon EKS gestionado por Amazon EC2. No es compatible con Amazon EKS gestionado por AWS Fargate.

Temas

- Requisitos previos
- Configuración
- Prueba de la aplicación implementada

Requisitos previos

Antes de comenzar, asegúrese de que cumple los requisitos previos que se indican a continuación:

- Configure el AWS CLI archivo siguiendo los pasos que se indican en Configuración de la AWS CLI.
- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Descargue los binarios de AWS Blu Age Runtime. Para obtener instrucciones, consulte the section called “Incorporación de AWS Blu Age Runtime ”.
- Descargue los binarios de Apache Tomcat 10.
- Descarga el archivo de PlanetsDemo la aplicación.
- Cree una base de datos de PostgreSQL de Amazon Aurora para JICS y ejecute la consulta `PlanetsDemo-v1/jics/sql/initJics.sql` en ella. Para obtener información sobre cómo crear una base de datos de PostgreSQL de Amazon Aurora, consulte Creación de un clúster de base de datos de Aurora PostgreSQL y conexión a él.

Configuración

Para configurar la aplicación PlanetsDemo de muestra, complete los siguientes pasos.

1. Tras descargar los binarios de Apache Tomcat, extraiga el contenido y vaya a la carpeta `conf`. Abra el archivo `catalina.properties` para editarlo y sustituya la línea que empieza por `common.loader` por la siguiente línea.

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/
*.jar","${catalina.home}/lib","${catalina.home}/lib/*.jar","${catalina.home}/
```

```
shared","${catalina.home}/shared/*.jar","${catalina.home}/extra","${catalina.home}/
extra/*.jar"
```

2. Comprima la carpeta de Apache Tomcat mediante el comando tar para crear un archivo tar.gz.
3. Prepare un Dockerfile para crear su imagen personalizada a partir de los binarios de tiempo de ejecución proporcionados y los binarios del servidor de Apache Tomcat. Consulte el siguiente Dockerfile de ejemplo. El objetivo es instalar Apache Tomcat 10, seguido de AWS Blu Age Runtime (para Amazon ECS gestionado por AWS Fargate) extraído de la raíz del directorio de instalación de Apache Tomcat 10 y, a continuación, instalar la aplicación modernizada de muestra denominada. PlanetsDemo

> **Note**
>
> El contenido de los scripts install-gapwalk.sh e install-app.sh, que se utilizan en este Dockerfile de ejemplo, aparece después del Dockerfile.

```
FROM --platform=linux/x86_64 amazonlinux:2

RUN mkdir -p /workdir/apps
WORKDIR /workdir
COPY install-gapwalk.sh .
COPY install-app.sh .
RUN chmod +x install-gapwalk.sh
RUN chmod +x install-app.sh

# Install Java and AWS CLI v2-y
RUN yum install sudo java-17-amazon-corretto unzip tar -y
RUN sudo yum remove awscli -y
RUN curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o
 "awscliv2.zip"
RUN sudo unzip awscliv2.zip
RUN sudo ./aws/install

# Installation dir
RUN mkdir -p /usr/local/velocity/installation/gapwalk
# Copy PlanetsDemo archive to a dedicated apps dir
COPY PlanetsDemo-v1.zip /workdir/apps/

# Copy resources (tomcat, blu age runtime) to installation dir
COPY tomcat.tar.gz /usr/local/velocity/installation/tomcat.tar.gz
```

```
COPY aws-bluage-runtime-4.x.x.tar.gz /usr/local/velocity/installation/gapwalk/
gapwalk.tar.gz

# run relevant installation scripts
RUN ./install-gapwalk.sh
RUN ./install-app.sh

EXPOSE 8080
EXPOSE 8081
# ...

WORKDIR /bluage/tomcat.gapwalk/velocity
# Run Command to start Tomcat server
CMD ["sh", "-c", "sudo bin/catalina.sh run"]
```

A continuación se muestra el contenido de `install-gapwalk.sh`.

```


# Vars
TEMP_DIR=/bluage-on-fargate/tomcat.gapwalk/temp

# Install
echo "Installing Gapwalk and Tomcat"
sudo rm -rf /bluage-on-fargate
mkdir -p ${TEMP_DIR}
# Copy Blu Age runtime and tomcat archives to temporary extraction dir
sudo cp /usr/local/velocity/installation/gapwalk/gapwalk.tar.gz ${TEMP_DIR}
sudo cp /usr/local/velocity/installation/tomcat.tar.gz ${TEMP_DIR}
# Create velocity dir
mkdir -p /bluage/tomcat.gapwalk/velocity
# Extract tomcat files
tar -xvf ${TEMP_DIR}/tomcat.tar.gz -C ${TEMP_DIR}
# Copy all tomcat files to velocity dir
cp -fr ${TEMP_DIR}/apache-tomcat-10.x.x/* /bluage/tomcat.gapwalk/velocity
# Remove default webapps of Tomcat
rm -f /bluage-on-fargate/tomcat.gapwalk/velocity/webapps/*
# Extract Blu Age runtime at velocity dir
tar -xvf ${TEMP_DIR}/gapwalk.tar.gz -C /bluage/tomcat.gapwalk
# Remove temporary extraction dir
sudo rm -rf ${TEMP_DIR}
```

A continuación se muestra el contenido de `install-app.sh`.

```
#!/bin/sh

APP_DIR=/workdir/apps
TOMCAT_GAPWALK_DIR=/bluage-on-fargate/tomcat.gapwalk

unzip ${APP_DIR}/PlanetsDemo-v1.zip -d ${APP_DIR}
cp -r ${APP_DIR}/webapps/* ${TOMCAT_GAPWALK_DIR}/velocity/webapps/
cp -r ${APP_DIR}/config/* ${TOMCAT_GAPWALK_DIR}/velocity/config/
```

4. Proporcione la información de conexión para la base de datos que creó como parte de los requisitos previos en el siguiente fragmento de código del archivo `application-main.yml`, que se encuentra en la carpeta `{TOMCAT_GAPWALK_DIR}/config`. Para obtener más información, consulte Creación de un clúster de base de datos de Aurora PostgreSQL y conexión a él.

```
datasource:
   jicsDs:
     driver-class-name :
     url:
     username:
     password:
     type :
```

5. Compile y envíe la imagen a su repositorio de Amazon ECR. Para ver más instrucciones, consulte Inserción de una imagen de Docker en la Guía del usuario de Amazon Elastic Container Registry. A continuación, según su situación, cree un pod de Amazon EKS o una definición de tarea de Amazon ECS con su imagen de Amazon ECR e impleméntela en su clúster. Para ver ejemplos de cómo crearlos, consulte Creación de una definición de tareas con la consola en la Guía para desarrolladores de Amazon Elastic Container Service (Amazon ECS) e Implementación de una aplicación de ejemplo en la Guía del usuario de Amazon EKS.
6. En concreto, en el caso de Amazon ECS administrado por AWS Fargate, al crear la definición de tarea, utilice el rol de IAM que creó como parte de la configuración de la infraestructura inicial. A continuación, al crear el servicio, expanda la sección Redes y configure la VPC, las subredes y el grupo de seguridad que creó como parte de la configuración de la infraestructura inicial. Consulte los requisitos de configuración de la infraestructura para AWS Blu Age Runtime (no administrado).

Prueba de la aplicación implementada

Para ver un ejemplo de cómo probar la PlanetsDemo aplicación, consultethe section called “Prueba la PlanetsDemo aplicación”.

Actualice el tiempo de ejecución de AWS Blu Age en un contenedor

Esta guía describe cómo actualizar el tiempo de ejecución de AWS Blu Age en un contenedor. Para ello, primero debe completar algunos requisitos previos y, a continuación, trabajar con la imagen de Docker para actualizar el tiempo de ejecución de AWS Blu Age.

Temas

- Requisitos previos
- Actualiza el tiempo de ejecución de AWS Blu Age

Requisitos previos

Antes de comenzar, asegúrese de que cumple los siguientes requisitos previos:

- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Descarga la versión de AWS Blu Age Runtime a la que deseas actualizar. Para obtener más información, consulte the section called “Incorporación de AWS Blu Age Runtime ”. El marco consta de dos archivos binarios: `aws-bluage-runtime-x.x.x.x.tar.gz` y `aws-bluage-webapps-x.x.x.x.tar.gz`.

Actualiza el tiempo de ejecución de AWS Blu Age

Complete los siguientes pasos para actualizar el tiempo de ejecución de AWS Blu Age.

1. Reconstruye tu imagen de Docker con la versión de AWS Blu Age Runtime que desees. Para obtener instrucciones, consulte the section called “Configura AWS Blu Age Runtime en un contenedor”.
2. Envíe la imagen de Docker a su repositorio de Amazon ECR.
3. Detenga y reinicie el servicio de Amazon ECS o Amazon EKS.
4. Verifique los registros.

El tiempo de ejecución de AWS Blu Age se actualizó correctamente.

Configura CloudWatch las alarmas de Amazon para AWS Blu Age Runtime en un contenedor

Puede configurarlo CloudWatch para que las notificaciones sean más visibles cada vez que las aplicaciones implementadas encuentren excepciones. Esto le ayuda a supervisar el registro de aplicaciones al CloudWatch que se redirige y a añadir una alarma para advertirle de posibles errores.

Configuración de alarmas

Con CloudWatch los registros, puede configurar cualquier cantidad de métricas y alarmas, en función de la aplicación y de sus necesidades.

En concreto, puede configurar alarmas proactivas para alertas de uso directamente durante la creación del clúster, de modo que reciba notificaciones cuando se produzcan errores. Para resaltar los errores en la conexión al sistema de control AWS Blu Age, añada una métrica relativa a la cadena «Error C» en los registros. A continuación, puede definir una alarma que reaccione a esta métrica.

Configure las dependencias con licencia en AWS Blu Age Runtime en un contenedor

Esta guía describe cómo configurar dependencias con licencia adicionales que puede usar con el tiempo de ejecución de AWS Blu Age en contenedor.

Temas

- Requisitos previos
- Descripción general

Requisitos previos

Antes de comenzar, asegúrese de que cumple los requisitos previos que se indican a continuación:

- Complete the section called “AWS Requisitos previos de Blu Age Runtime” y the section called “Incorporación de AWS Blu Age Runtime ”.
- Obtenga las siguientes dependencias de su origen.

Base de datos de Oracle

Proporcione un controlador de base de datos de Oracle. Por ejemplo, ojdbc11-23.3.0.23.09.jar.

Conexión MQ de IBM

Suministre un cliente MQ de IBM. Por ejemplo, com.ibm.mq.jakarta.client-9.3.4.1.jar.

Con esta versión de dependencias, suministre también las siguientes dependencias transitivas:

- bcprov-jdk15to18-1.76.jar
- bcpkix-jdk15to18-1.76.jar
- bcutil-jdk15to18-1.76.jar

Archivos de impresora DDS

Proporcione la biblioteca de informes de Jasper (munity.jaspersoft)https://com. com/download-jaspersoft/community-edición). Por ejemplo, jasperreports-6.16.0.jar, pero una versión más reciente podría ser compatible.

Con esta versión de dependencias, suministre también las siguientes dependencias transitivas:

- castor-core-1.4.1. jar
- castor-core-1.4.1. jar
- commons-digester-2.1.jar
- ecj-3.21.0.jar
- itext-2.1.7. js8.jar
- javax.inject-1.jar
- jcommon-1.0.23.jar
- jfreechart-1.0.19.jar
- commons-beanutils-1.9.4.jar
- commons-collections-3.2.2.jar

Descripción general

Para instalar las dependencias, complete los pasos siguientes.

1. Copie cualquiera de las dependencias anteriores según sea necesario en la carpeta de compilación de imágenes de Docker.
2. Si su base de datos JICS está alojada en Oracle, proporcione el controlador de la base de datos Oracle. *your-tomcat-path*/extra

3. En su Dockerfile, copie estas dependencias en *your-tomcat-path*/extra.
4. Cree su imagen de Docker y, a continuación, envíela a Amazon ECR.
5. Detenga y reinicie el servicio de Amazon ECS o Amazon EKS.
6. Consulte los registros.

Prueba la PlanetsDemo aplicación

Para comprobar el estado de la PlanetsDemo aplicación desplegada, ejecute los siguientes comandos después de load-balancer-DNS-name reemplazarla y web-binary-name con los valores correctos para la configuración. listener-port

```
curl http://load-balancer-DNS-name:listener-port/gapwalk-application/
```

Si la aplicación está ejecutándose, verá el siguiente mensaje de salida: Jics application is running.

A continuación, ejecute los comandos siguientes:

```
curl http://load-balancer-DNS-name:listener-port/jac/api/services/rest/jicsservice/
```

Si la aplicación está ejecutándose, verá el siguiente mensaje de salida: Jics application is running.

```
Jics application is running
```

Si ha configurado Blusam, puede esperar una respuesta vacía al ejecutar el siguiente comando.

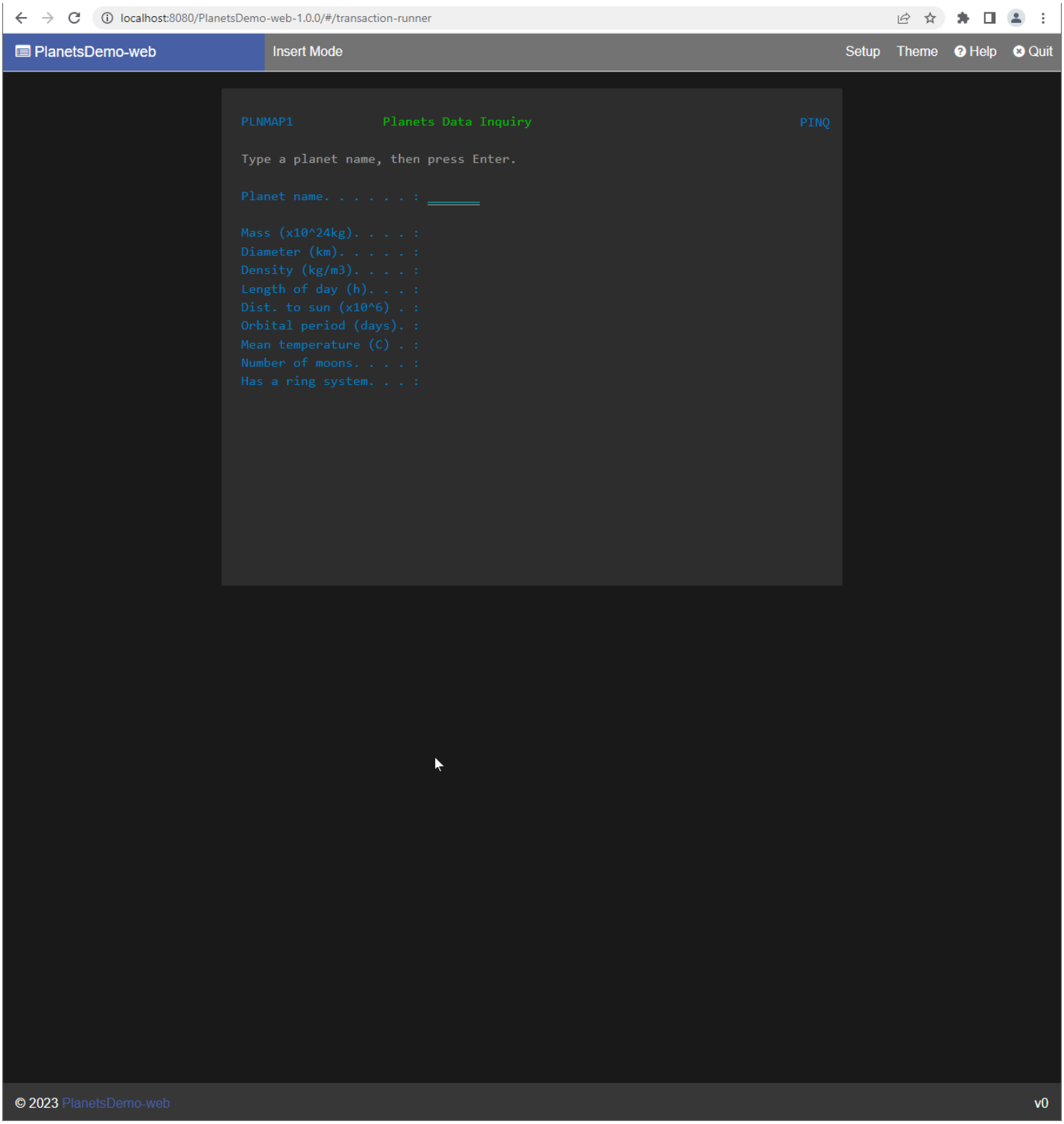
```
curl http://load-balancer-DNS-name:listener-port/bac/api/services/rest/bluesamserver/
serverIsUp
```

Anote el nombre del binario web (PlanetsDemo-web-1.0.0, si no ha cambiado). Para acceder a la PlanetsDemo aplicación, utilice una URL con el siguiente formato.

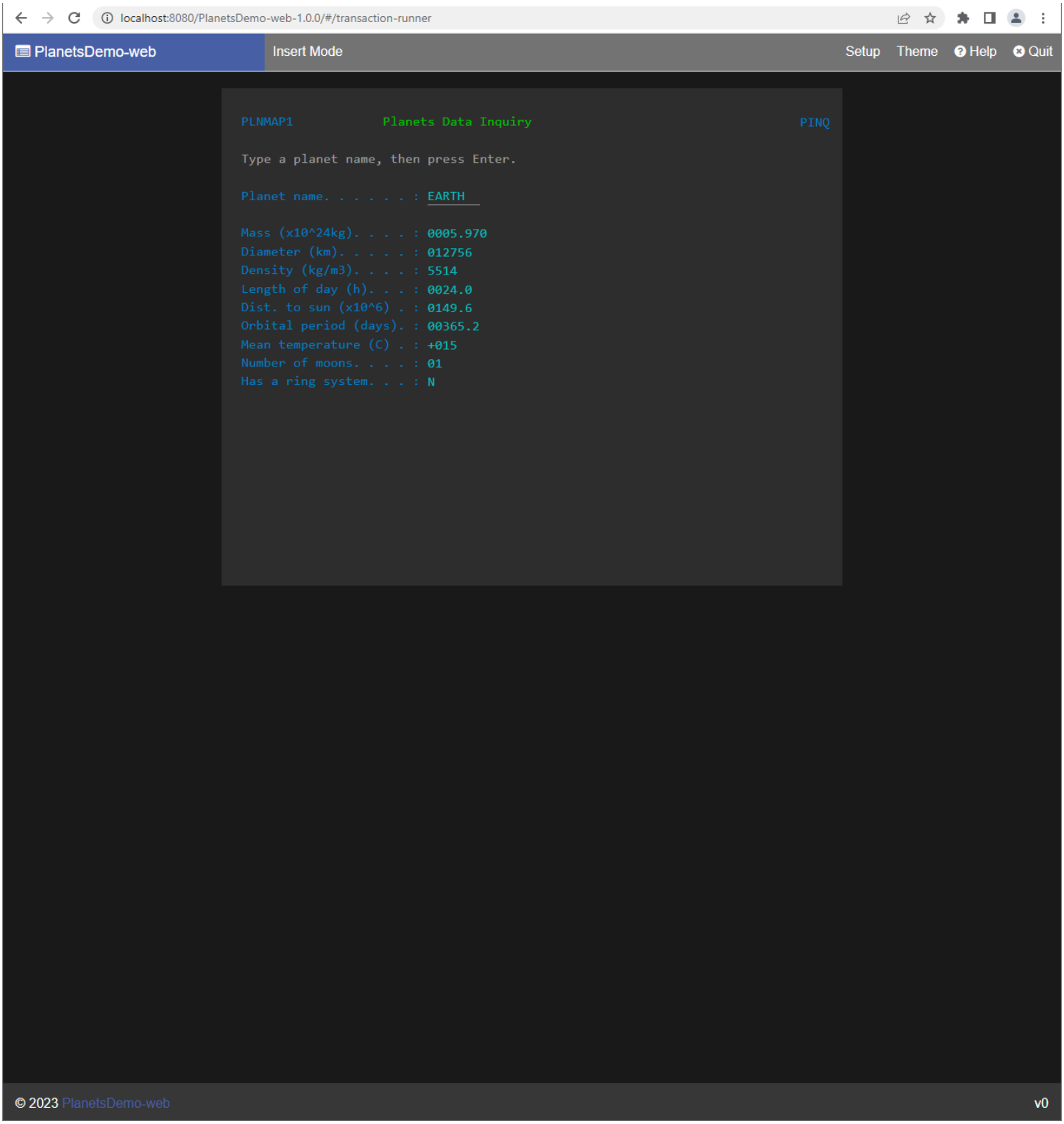
```
https://load-balancer-DNS-name:listener-port/web-binary-name
```

Cuando se inicie la PlanetsDemo aplicación, aparecerá la página de inicio.

En el cuadro de texto, escriba PINQ y, a continuación, pulse Intro. Aparece la página de consulta de datos.



Por ejemplo, escriba EARTH en el campo PlanetsDemo del nombre y, a continuación, presione Entrar. Aparece la página del planeta que haya introducido.



AWS Blu Age Runtime está disponible en las siguientes regiones: EE.UU. Este (Ohio), EE.UU. Este (Norte de Virginia), EE.UU. Oeste (Norte de California), EE.UU. Oeste (Oregón), Canadá (Central), Región Europa (Irlanda), Región Europa (Londres), Región Europa (París), Europa (Fráncfort), Región Europa (Estocolmo), Región Europa (Milán), Región Europa (España), Sudamérica (São

Paulo), Asia Pacífico (Tokio), Asia Pacífico (Seúl), Asia Pacífico (Osaka), Asia Pacífico (Singapur), Asia Pacífico (Sídney), Asia Pacífico (Bombay), África (Ciudad del Cabo) e Israel (Tel Aviv).

Modificación del código fuente con el IDE para desarrolladores de Blu Age

Si utiliza el motor AWS de ejecución AWS Blu Age gestionado, puede utilizar Blu Age Developer para modificar el código fuente generado. Puede que quiera hacerlo si necesita actualizar el código modernizado por alguna razón o si una parte del código fuente heredado no se ha podido modernizar. Puedes acceder a Blu Age Developer a través de Amazon AppStream 2.0. En esta sección se describe cómo configurar Blu Age Developer en la AppStream versión 2.0. También se explica cómo utilizar Blu Age Developer para actualizar el código fuente mediante la aplicación de ejemplo PlanetsDemo.

Temas

- Tutorial: Configurar AppStream 2.0 para el IDE de desarrolladores de AWS Blu Age
- Tutorial: Utilice AWS Blu Age Developer en AppStream 2.0

Tutorial: Configurar AppStream 2.0 para el IDE de desarrolladores de AWS Blu Age

AWS La modernización del mainframe proporciona varias herramientas a través de Amazon AppStream 2.0. AppStream 2.0 es un servicio de streaming de aplicaciones seguro y totalmente gestionado que permite transmitir aplicaciones de escritorio a los usuarios sin tener que volver a escribirlas. AppStream La versión 2.0 proporciona a los usuarios acceso instantáneo a las aplicaciones que necesitan, con una experiencia de usuario fluida y con gran capacidad de respuesta en el dispositivo que elijan. El uso de la AppStream versión 2.0 para alojar herramientas específicas del motor de ejecución ofrece a los equipos de aplicaciones de los clientes la posibilidad de utilizar las herramientas directamente desde sus navegadores web e interactuar con los archivos de las aplicaciones almacenados en depósitos o repositorios de Amazon S3. CodeCommit

Para obtener información sobre la compatibilidad de los navegadores en la AppStream versión 2.0, consulte System Requirements and Feature Support (navegador web) en la Guía de administración de Amazon AppStream 2.0. Si tiene problemas al utilizar la AppStream versión 2.0, consulte Solución de problemas de usuario de la AppStream versión 2.0 en la Guía de administración de Amazon AppStream 2.0.

Este documento describe cómo configurar el IDE para desarrolladores de AWS Blu Age en una flota AppStream 2.0.

Temas

- Requisitos previos
- Paso 1: Crear un bucket de Amazon S3
- Paso 2: Adjuntar una política al bucket de S3
- Paso 3: Cargar los archivos a un bucket de Amazon S3
- Paso 4: Descargar plantillas AWS CloudFormation
- Paso 5: Crea la flota con AWS CloudFormation
- Paso 6: Acceder a una instancia
- Eliminar recursos

Requisitos previos

Para los usuarios primerizos, haga lo siguiente:

1. Dirígete a la consola AppStream 2.0 de https://console.aws.amazon.com/appstream2/tu casa.
2. Elija Comenzar.
3. Elija Omitir.

> **Important**
>
> Amazon AppStream 2.0 usa roles de IAM para administrar sus recursos de AppStream 2.0 y AWS creará estos roles cuando lo haga.

A continuación, descarga el archivo comprimido que contiene los artefactos que necesitas para configurar el IDE para desarrolladores de AWS Blu Age en la AppStream versión 2.0.

> **Note**
>
> Es un archivo grande. Si tienes problemas con el tiempo de espera de la operación, te recomendamos que utilices una EC2 instancia de Amazon para mejorar el rendimiento de

carga y descarga. Para obtener más información sobre cómo lanzar una EC2 instancia de Amazon y conectarse a ella, consulta Comenzar con Amazon EC2.

Paso 1: Crear un bucket de Amazon S3

Cree un bucket de Amazon S3 Región de AWS igual que la flota AppStream 2.0 que va a crear. Este bucket contendrá los artefactos que necesita para completar este tutorial. Para obtener más información sobre los buckets, consulte Crear un bucket.

Paso 2: Adjuntar una política al bucket de S3

Adjunte la siguiente política al bucket que ha creado para este tutorial. Para obtener más información sobre cómo adjuntar una política a un bucket de S3, consulte Añadir una política de bucket.

Asegúrese de reemplazar `amzn-s3-demo-bucket` por el nombre real del bucket que ha creado.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "AllowAppStream2.0ToRetrieveObjects",
        "Effect": "Allow",
        "Principal": {
            "Service": "appstream.amazonaws.com"
        },
        "Action": "s3:GetObject",
        "Resource": "arn:aws:s3:::amzn-s3-demo-bucket/*"
    }]
}
```

Paso 3: Cargar los archivos a un bucket de Amazon S3

Descomprima los archivos que ha descargado en el Requisito previo y suba la carpeta `appstream` a su bucket. Al cargar esta carpeta, se crea la estructura correcta en el bucket. Para obtener más información, consulte Carga de objetos en la Guía del usuario de Amazon S3.

Paso 4: Descargar plantillas AWS CloudFormation

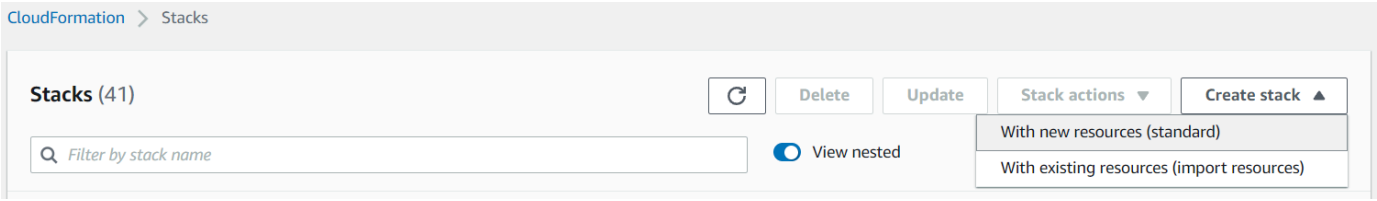
Descarga las siguientes AWS CloudFormation plantillas. Necesita estas plantillas para crear y completar la flota AppStream 2.0.

- cfn-m2-.yaml appstream-elastic-fleet-linux
- cfn-m2- appstream-bluage-dev-tools -linux.yaml
- cfn-m2 appstream-bluage-shared-linux - .yaml
- appstream-chrome-linuxcfn-m2-.yaml
- appstream-eclipse-jee-linuxcfn-m2-.yaml
- appstream-pgadmin-linuxcfn-m2-.yaml

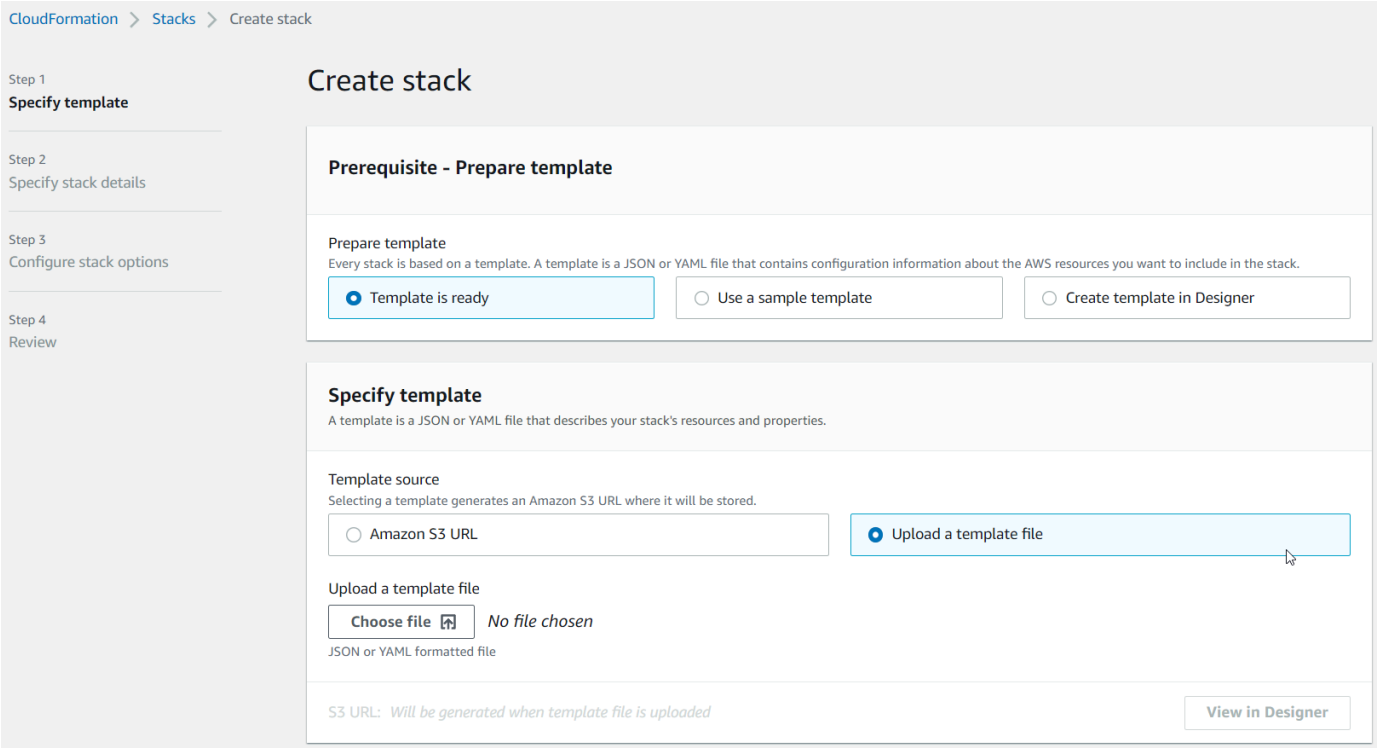
Paso 5: Crea la flota con AWS CloudFormation

En este paso, utilizarás la `cfn-m2-appstream-elastic-fleet-linux.yaml` AWS CloudFormation plantilla para crear una flota AppStream 2.0 y apilarla para alojar el IDE para desarrolladores de AWS Blu Age. Tras crear la flota y la pila, ejecutará las demás AWS CloudFormation plantillas que descargó en el paso anterior para instalar el IDE para desarrolladores y otras herramientas necesarias.

1. Ve a AWS CloudFormation la consola AWS de administración y selecciona Stacks.
2. Seleccione Pilas, seleccione Crear pila y elija la opción Con recursos nuevos (estándar).



3. En Crear pila, selecciona Elegir una plantilla existente y Cargar un archivo de plantilla:



4. Seleccione Elegir archivo y navegue al archivo `cfn-m2-appstream-elastic-fleet-linux.yaml`. Elija Siguiente.
5. En Especificar los detalles de la pila, indique la siguiente información.

   - Un nombre para la pila.
   - Su grupo de seguridad predeterminado y dos subredes de ese grupo de seguridad.

   > **Note**
   >
   > Las dos subredes del grupo de seguridad deben estar en zonas de disponibilidad diferentes.

6. Elija Siguiente.
7. Navegue hacia abajo en la página y elija Acepto que AWS CloudFormation podría crear recursos de IAM con nombres personalizados. .
8. Elija Siguiente.
9. Revisa los detalles y selecciona Enviar.
10. Tras crear la flota, cree CloudFormation pilas con todas las demás plantillas descargadas para terminar de configurar las aplicaciones. Asegúrese de actualizarlas BucketNamecada vez para que apunten al bucket S3 correcto. Puedes editarlo BucketNameen la CloudFormation consola.

Como alternativa, puede editar los archivos de plantilla directamente y actualizar la propiedad `S3Bucket`.

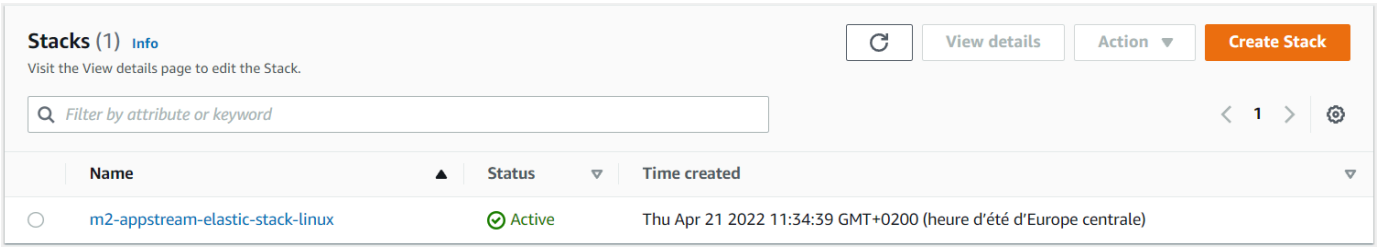
> **Note**
>
> Las plantillas descargadas esperan encontrar los activos en un bucket de S3 con una estructura de carpetas denominada `appstream/bluage/developer-ide/`. El depósito debe estar en el Región de AWS mismo lugar que la flota que has creado.

> **Important**
>
> Ejecute todos los CloudFormation scripts descargados en el paso 4 para configurar la aplicación correctamente.

Paso 6: Acceder a una instancia

Tras crear e iniciar la flota, puede crear un enlace temporal para acceder a la flota a través del cliente nativo.

1. Navegue hasta la AppStream versión 2.0 en AWS Management Console y elija la pila creada anteriormente:



2. En la página de detalles de la pila, elija la pila y, a continuación, seleccione Asociar flota.
3. En la pantalla, selecciona la flota que creaste e iniciaste anteriormente.
4. Elija Asociar.
5. Selecciona la pila asociada y, en el menú Acciones, selecciona Crear URL de streaming, introduce un seudónimo arbitrario y una fecha de caducidad de la URL y, a continuación, selecciona Obtener URL. Obtendrá una URL que puede usar para transmitir en streaming a un navegador o al cliente nativo. Le recomendamos que haga streaming en el cliente nativo.

Eliminar recursos

Para conocer el procedimiento para limpiar la pila y las flotas creadas, consulte Crear una flota y una pila AppStream 2.0.

Cuando hayas eliminado los objetos AppStream 2.0, tú o el administrador de la cuenta también podéis limpiar los depósitos de S3 para guardar la configuración de la aplicación y las carpetas de inicio.

> **Note**
>
> La carpeta principal de un usuario determinado es única en todas las flotas, por lo que es posible que tengas que conservarla si hay otras pilas AppStream 2.0 activas en la misma cuenta.

No puedes usar la consola AppStream 2.0 para eliminar usuarios. Para ello, debe utilizar la API de servicio con la AWS CLI. Para obtener más información, consulte Administración de grupos de usuarios en la Guía de administración de Amazon AppStream 2.0.

Tutorial: Utilice AWS Blu Age Developer en AppStream 2.0

Este tutorial te muestra cómo acceder a AWS Blu Age Developer en la AppStream versión 2.0 y cómo usarlo con una aplicación de muestra para que puedas probar sus funciones. Cuando haya acabado este tutorial, podrá seguir los mismos pasos con sus propias aplicaciones.

Temas

- Paso 1: Crear una base de datos
- Paso 2: Acceder al entorno
- Paso 3: Configurar el tiempo de ejecución
- Paso 4: Iniciar el IDE de Eclipse
- Paso 5: Configurar un proyecto de Maven
- Paso 6: Configurar un servidor de Tomcat
- Paso 7: Implementar en Tomcat
- Paso 8: Crear la base de datos de JICS
- Paso 8: iniciar y probar la aplicación

- Paso 10: Depurar la aplicación
- Eliminar recursos

Paso 1: Crear una base de datos

En este paso, utilizará Amazon RDS para crear una base de datos PostgreSQL administrada que la aplicación de ejemplo utilizará para almacenar la información de configuración.

1. Abra la consola de Amazon RDS.
2. Seleccione Bases de datos > Crear base de datos.
3. Elija Creación estándar > PostgreSQL, deje la versión predeterminada y, a continuación, elija Capa gratuita.
4. Elija el identificador de la instancia de base de datos.
5. En Configuración de credenciales, seleccione Administrar credenciales principales en AWS Secrets Manager. Para obtener más información, consulte Administración de contraseñas con Amazon RDS y AWS Secrets Manager en la Guía del usuario de Amazon RDS.
6. Asegúrese de que la VPC sea la misma que la que utiliza para la instancia AppStream 2.0. Puede solicitar este valor a su administrador.
7. En Nuevo nombre del grupo de seguridad de VPC, elija Crear nuevo.
8. Establezca Acceso público en Sí.
9. Deje los demás valores predeterminados. Revise estos valores.
10. Elija Crear base de datos.

Para que el servidor de base de datos sea accesible desde su instancia, seleccione el servidor de base de datos en Amazon RDS. En Conectividad y seguridad, elija el grupo de seguridad de VPC para el servidor de base de datos. Este grupo de seguridad se ha creado previamente para usted y debe tener una descripción similar a la de Creado por la consola de administración de RDS. Seleccione Acción > Editar reglas de entrada, elija Agregar regla y cree una regla de tipo PostgreSQL. Para el origen de la regla, utilice el grupo de seguridad predeterminado. Puede empezar a escribir el nombre del origen en el campo Origen y aceptar el ID sugerido. Por último, seleccione Guardar reglas.

Paso 2: Acceder al entorno

En este paso, accederá al entorno de desarrollo de AWS Blu Age en la AppStream versión 2.0.

1. Póngase en contacto con su administrador para conocer la forma correcta de acceder a su instancia AppStream 2.0. Para obtener información general sobre posibles clientes y configuraciones, consulte Métodos y clientes de acceso AppStream 2.0 en la Guía de administración de Amazon AppStream 2.0. Considere la posibilidad de utilizar el cliente nativo para disfrutar de la mejor experiencia.
2. En la AppStream versión 2.0, elija Escritorio.

Paso 3: Configurar el tiempo de ejecución

En este paso, configuras el tiempo de ejecución de AWS Blu Age. Debe configurar el tiempo de ejecución en el primer lanzamiento y de nuevo si se le notifica una actualización del tiempo de ejecución. Este paso rellena la carpeta `.m2`.

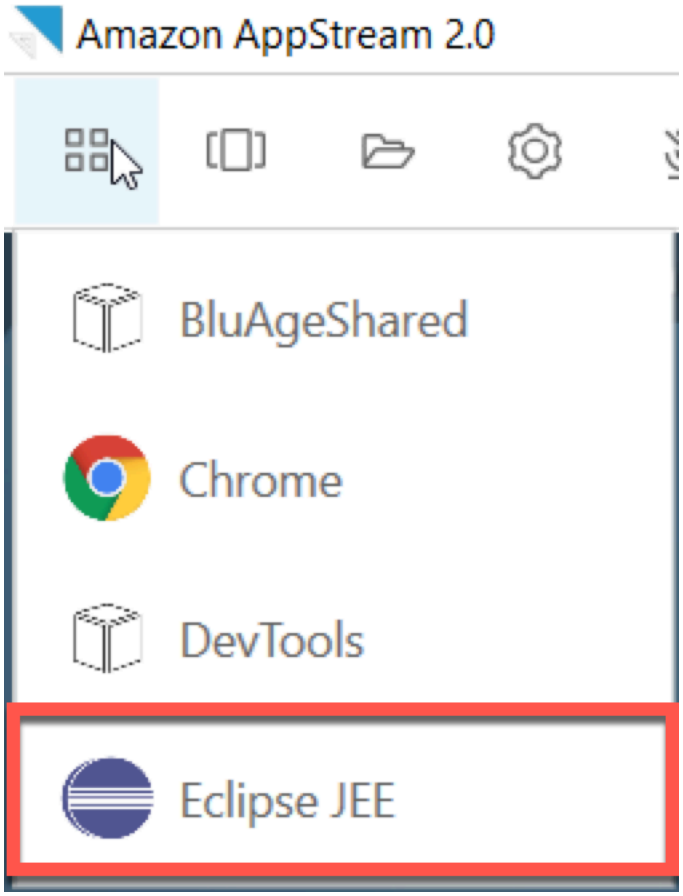
1. Seleccione Aplicaciones en la barra de menús y, a continuación, elija Terminal.
2. Escriba el siguiente comando:

```
~/_install-velocity-runtime.sh
```

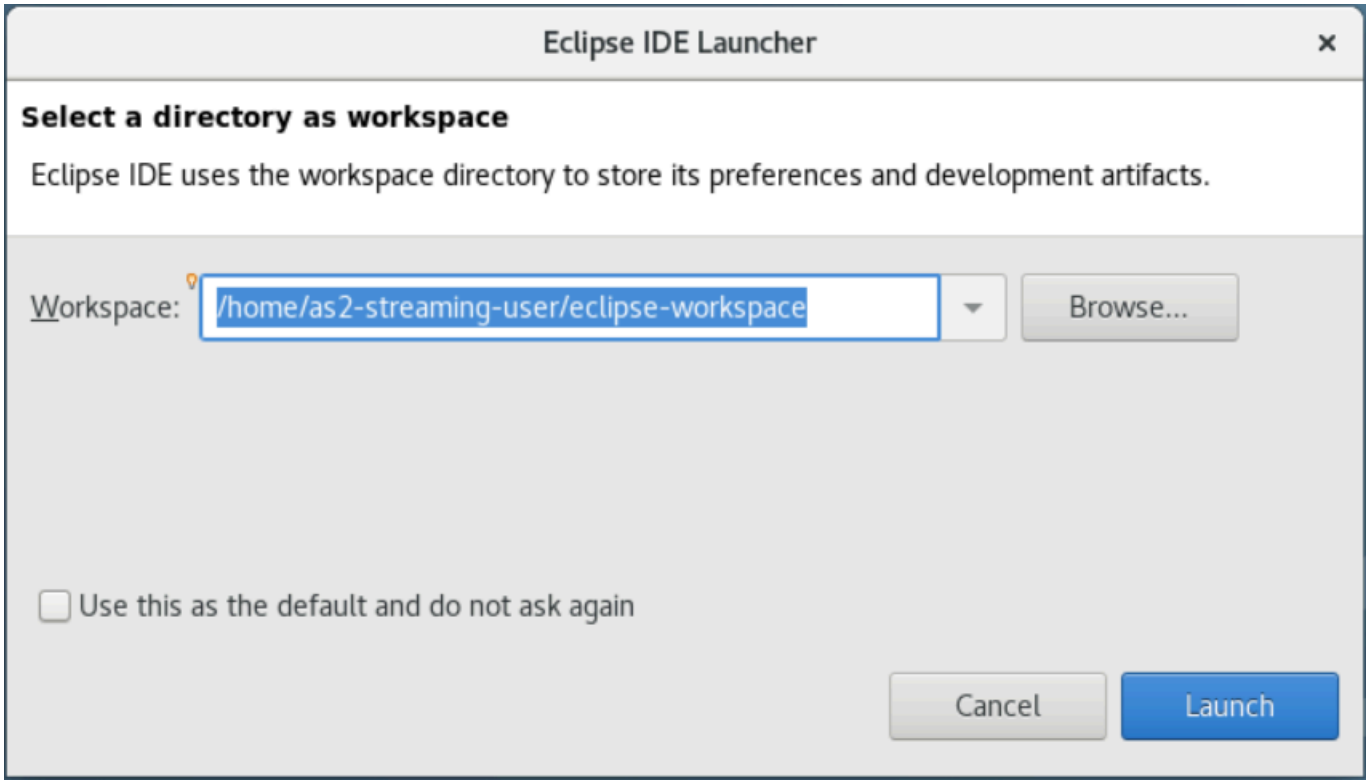
Paso 4: Iniciar el IDE de Eclipse

En este paso, iniciará el IDE de Eclipse y elegirá la ubicación en la que desee crear un espacio de trabajo.

1. En la AppStream versión 2.0, elija el icono de iniciar aplicación en la barra de herramientas y, a continuación, elija Eclipse JEE.



2. Cuando se abra el lanzador, introduzca la ubicación en la que quiere crear su espacio de trabajo y elija Launch.



Si lo desea, puede lanzar Eclipse desde la línea de comandos, de la siguiente manera:

```
~/eclipse &
```

Paso 5: Configurar un proyecto de Maven

En este paso, importará un proyecto de Maven para la aplicación de demostración Planets.

1. Sube el PlanetsDemoarchivo -pom.zip a tu carpeta de inicio. Para ello, puede utilizar la característica “My Files” del cliente nativo.
2. Utilice la herramienta de línea de comandos de `unzip` para extraer los archivos.
3. Desplácese por la carpeta descomprimida y abra el `pom.xml` raíz de su proyecto en un editor de texto.
4. Edite la propiedad `gapwalk.version` para que coincida con el motor de tiempo de ejecución de AWS Blu Age instalado.

   Si no conoce con seguridad la versión instalada, ejecute el siguiente comando en un terminal:

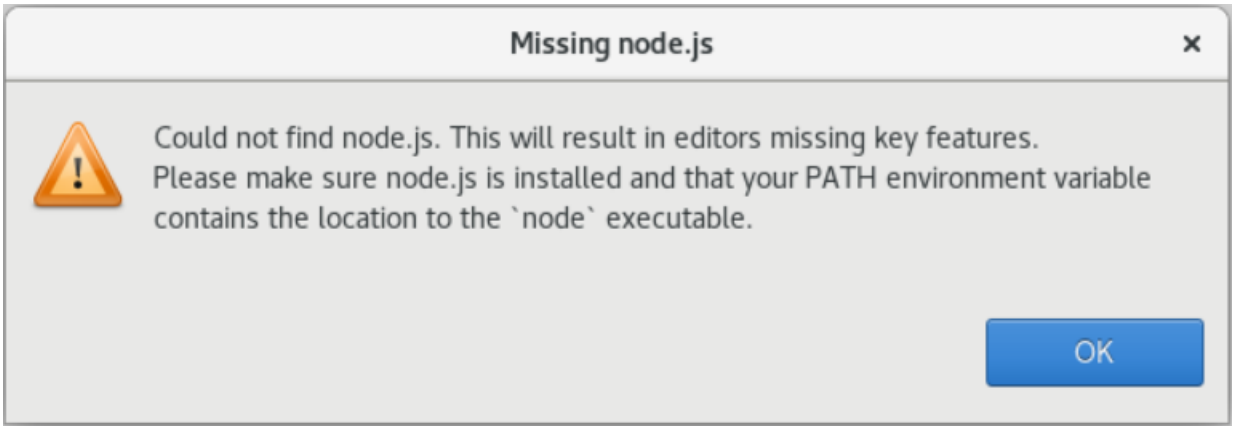
   ```
   cat ~/runtime-version.txt
   ```

Este comando imprime la versión de tiempo de ejecución disponible actualmente, por ejemplo `3.1.0-b3257-dev`.

> **Note**
>
> No incluya el sufijo `-dev` en `gapwalk.version`. Por ejemplo, un valor válido sería `<gapwalk.version>3.1.0-b3257</gapwalk.version>`.

5. En Eclipse, elija File e Import. En la ventana de diálogo Import, expanda Maven y elija Existing Maven Projects. Elija Next.
6. En Import Maven Projects, proporcione la ubicación de los archivos extraídos y elija Finish.

   Puede omitir la siguiente ventana emergente de forma segura. Maven descarga una copia local de `node.js` para compilar la parte angular (*-web) del proyecto:

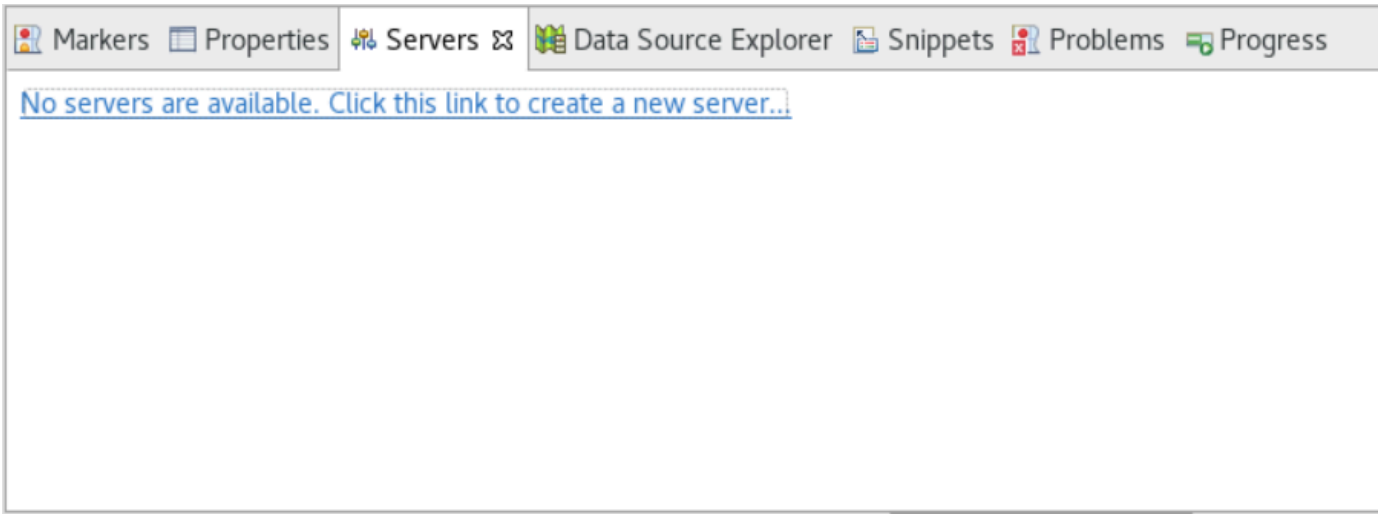


   Espere hasta el final de la compilación. Puede seguir la compilación en la vista de Progreso.
7. En Eclipse, seleccione el proyecto y elija Run as. Después, seleccione Maven install. Una vez que la instalación de Maven se haya realizado correctamente, se creará el archivo `war` en `PlanetsDemoPom/PlanetsDemo-web/target/PlanetsDemo-web-1.0.0.war`.

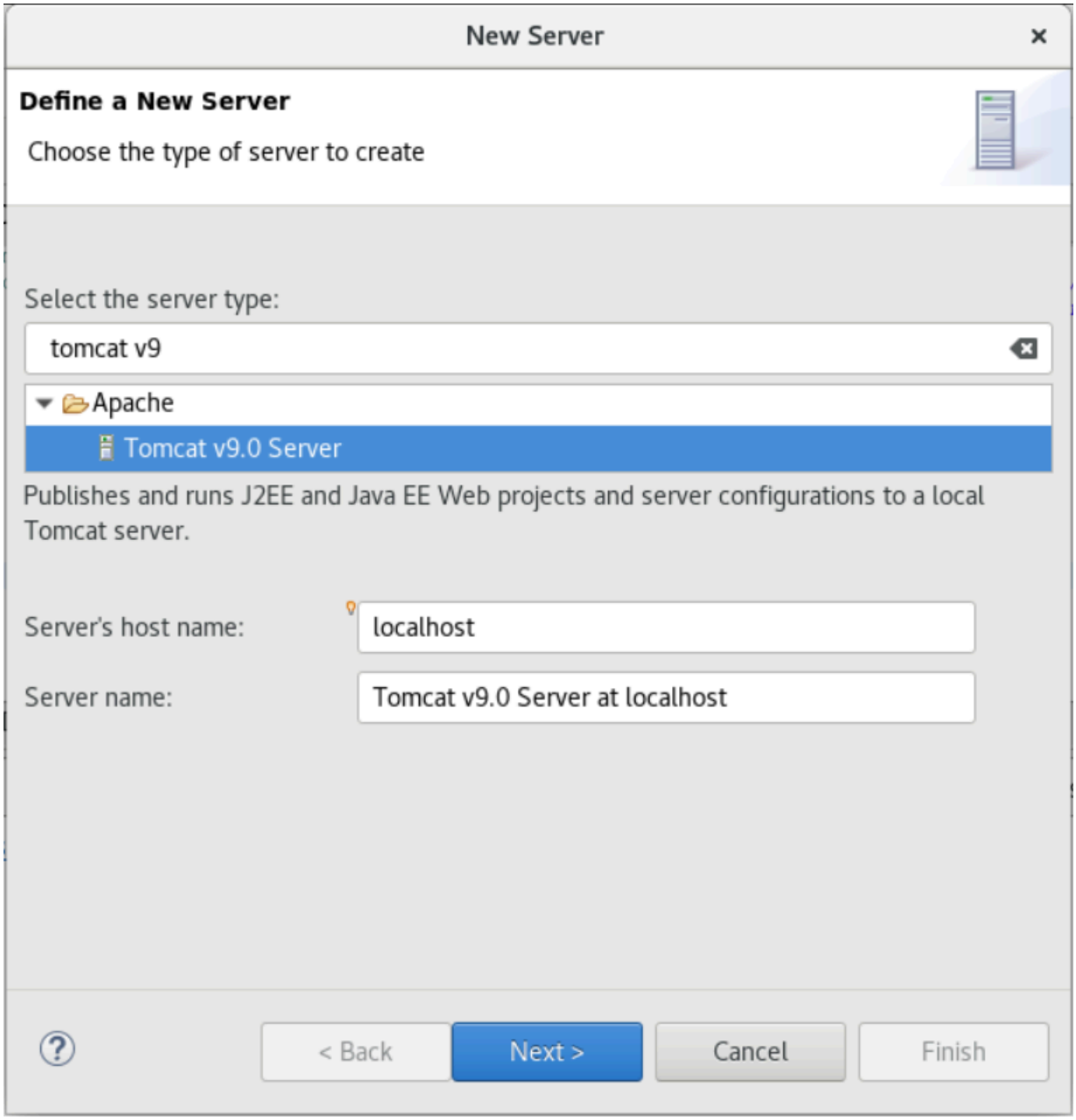
Paso 6: Configurar un servidor de Tomcat

En este paso, configurará un servidor de Tomcat en el que implementará e iniciará la aplicación compilada.

1. En Eclipse, seleccione Window > Show View > Servers para mostrar la vista Servers:

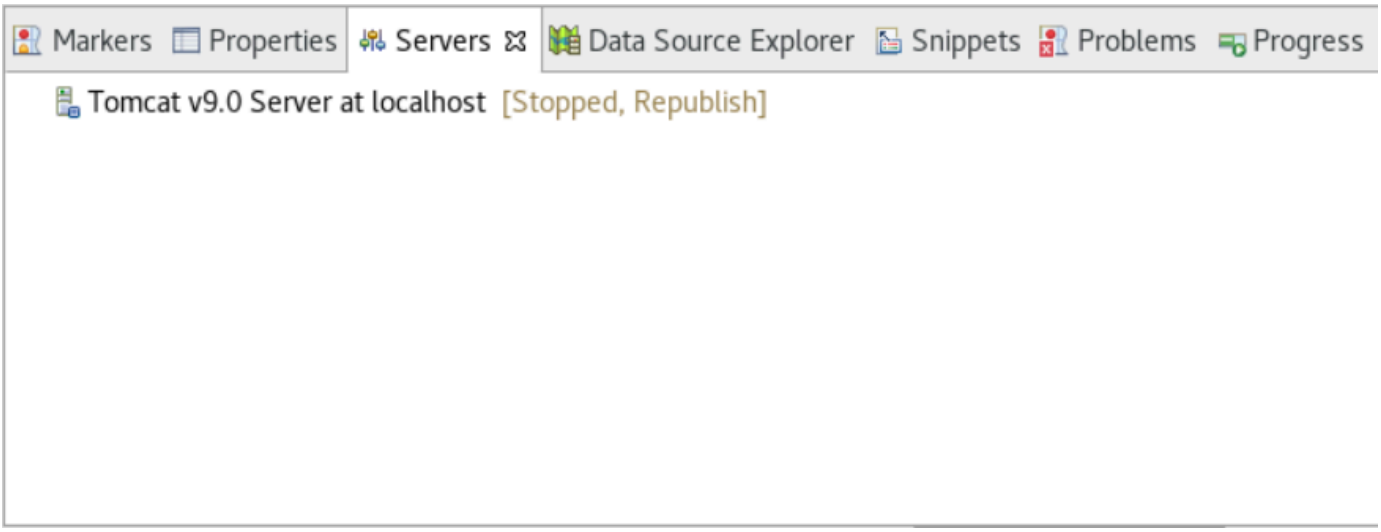


2. Seleccione No servers are available. Haga clic en este enlace para crear un nuevo servidor. Se abrirá el asistente New Server. En el campo Select the server type del asistente, escriba tomcat v9 y elija Tomcat v9.0 Server. A continuación, elija Siguiente.



3. Seleccione Browse y elija la carpeta tomcat situada en la raíz de la carpeta de inicio. Deje el JRE en su valor predeterminado y seleccione Finish.

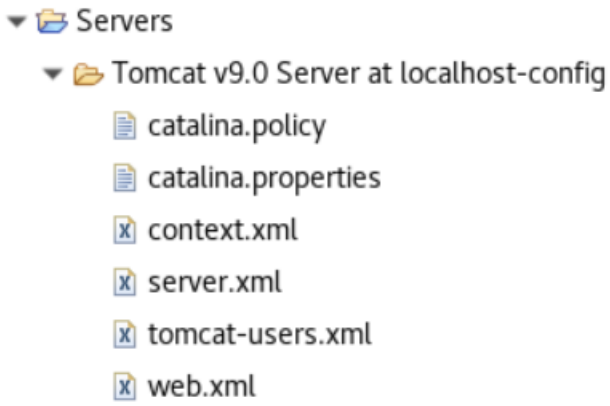
   Se ha creado un proyecto Servers en el espacio de trabajo y ahora hay un servidor Tomcat v9.0 disponible en la vista Servers. Aquí es donde se implementará e iniciará la aplicación compilada:



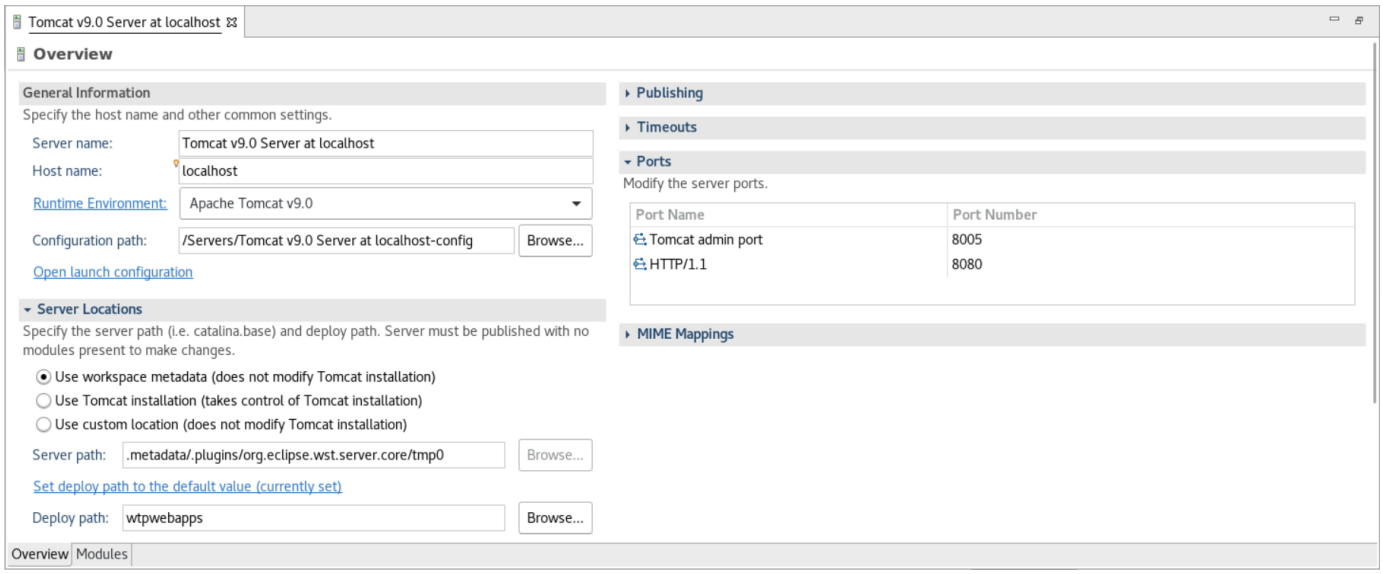
Paso 7: Implementar en Tomcat

En este paso, implementará la aplicación de ejemplo Planets en el servidor de Tomcat para poder ejecutar la aplicación.

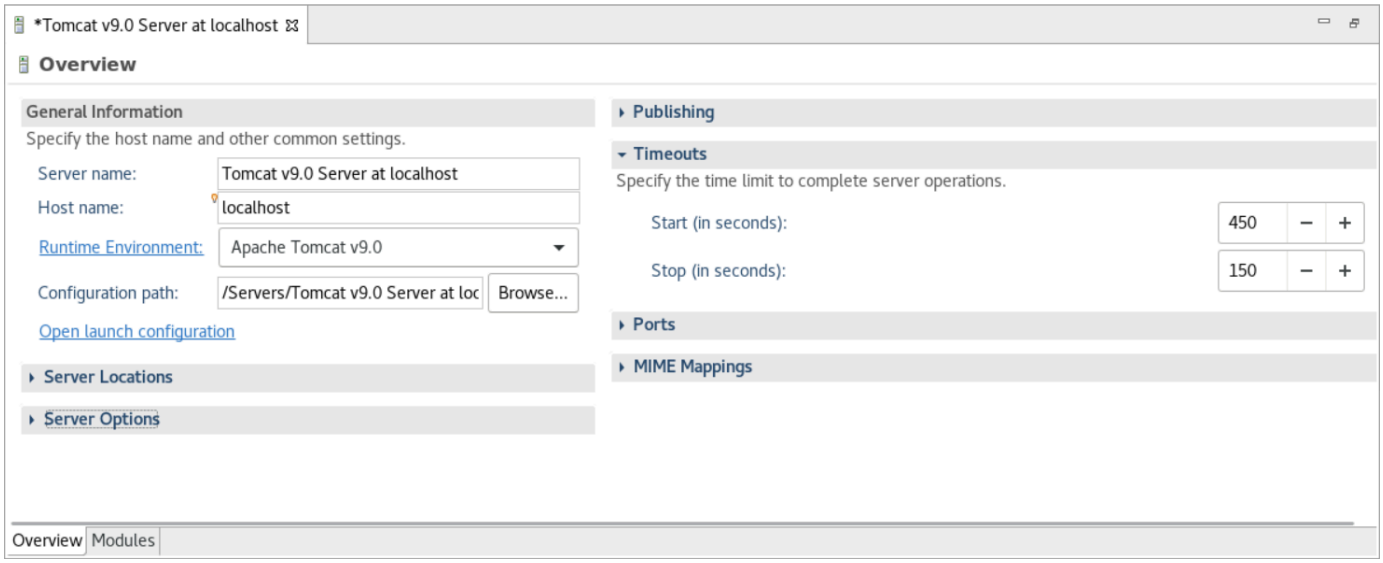
1. Seleccione el archivo `PlanetsDemo-web` y elija Run As > Maven install. Vuelva a seleccionar `PlanetsDemo-web` y elija Refresh para asegurarse de que la interfaz compilada con npm esté correctamente compilada en un archivo .war y Eclipse la detecte.
2. Cargue el PlanetsDemoarchivo -runtime.zip a la instancia y descomprima el archivo en una ubicación accesible. Esto garantiza que la aplicación de demostración pueda acceder a las carpetas y archivos de configuración que necesite.
3. Copie el contenido de `PlanetsDemo-runtime/tomcat-config` en la subcarpeta `Servers/Tomcat v9.0...` que ha creado para su servidor Tomcat:



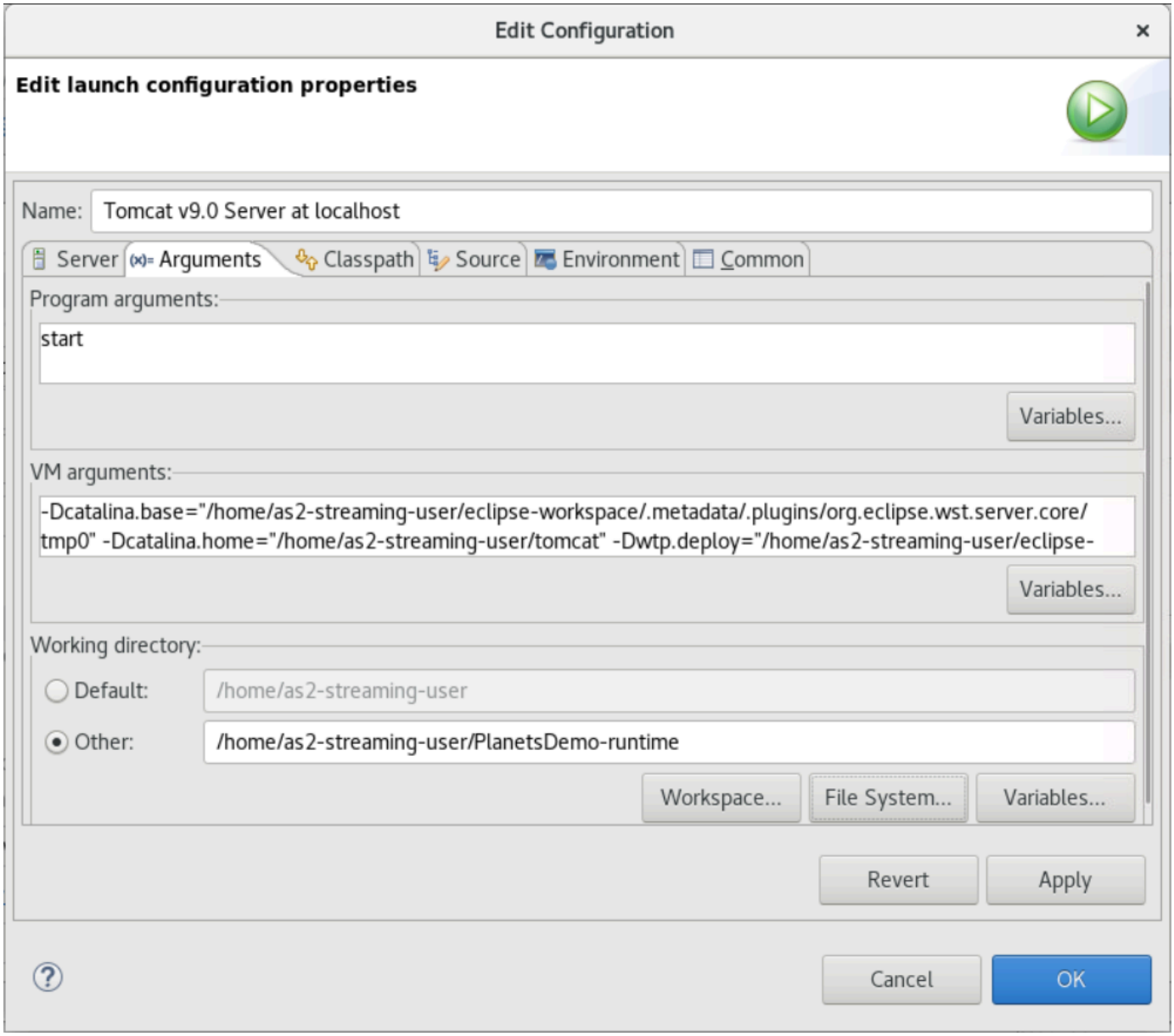
4. Abra la entrada del servidor `tomcat v9.0` en la vista Servers. Aparece el editor de propiedades del servidor:



5. En la pestaña Overview, aumente los valores de Timeouts a 450 segundos para Start y 150 segundos para Stop, como se muestra a continuación:



6. Seleccione Open launch configuration. Aparecerá el asistente. En el asistente, vaya a la carpeta Arguments y, en Working directory, elija Other. Elija File System y navegue hasta la carpeta `PlanetsDemo-runtime` descomprimida anteriormente. Esta carpeta debe contener una subcarpeta directa llamada config.

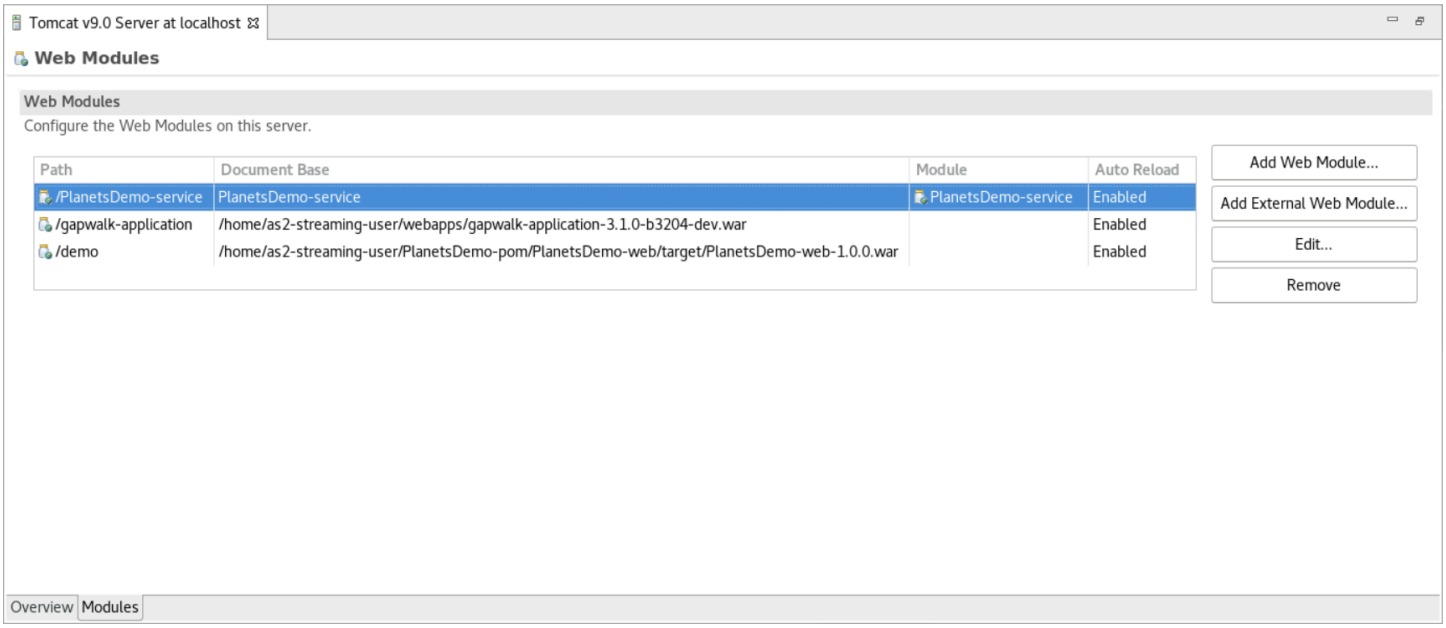


7. Seleccione la pestaña Modules del editor de propiedades del servidor y realice los siguientes cambios:

   - Seleccione Add Web Module and añada `PlanetsDemo-service`.
   - Elija Add External Web Module. Aparece la ventana Add Web Module. Realice los siguientes cambios:
     - En Document base, elija Browse y vaya a `~/webapps/gapwalk-application...war`
     - En Path, introduzca `/gapwalk-application`.
   - Seleccione Aceptar.
   - Vuelva a seleccionar Add External Web Module y realice los siguientes cambios:

    - En Document base, introduzca la ruta a la interfaz .war (in `PlanetsDemo-web/target`)
    - En Path, introduzca `/demo`
  - Seleccione Aceptar.
  - Guarde las modificaciones del editor (Ctrl + S).

El editor debe de mostrar un contenido similar al del siguiente ejemplo.



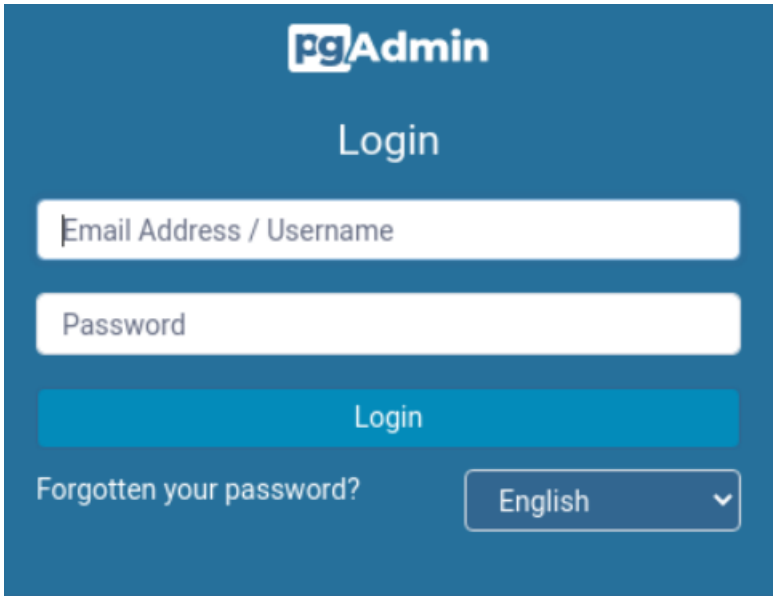
Paso 8: Crear la base de datos de JICS

En este paso, se conectará a la base de datos que ha creado enPaso 1: Crear una base de datos.

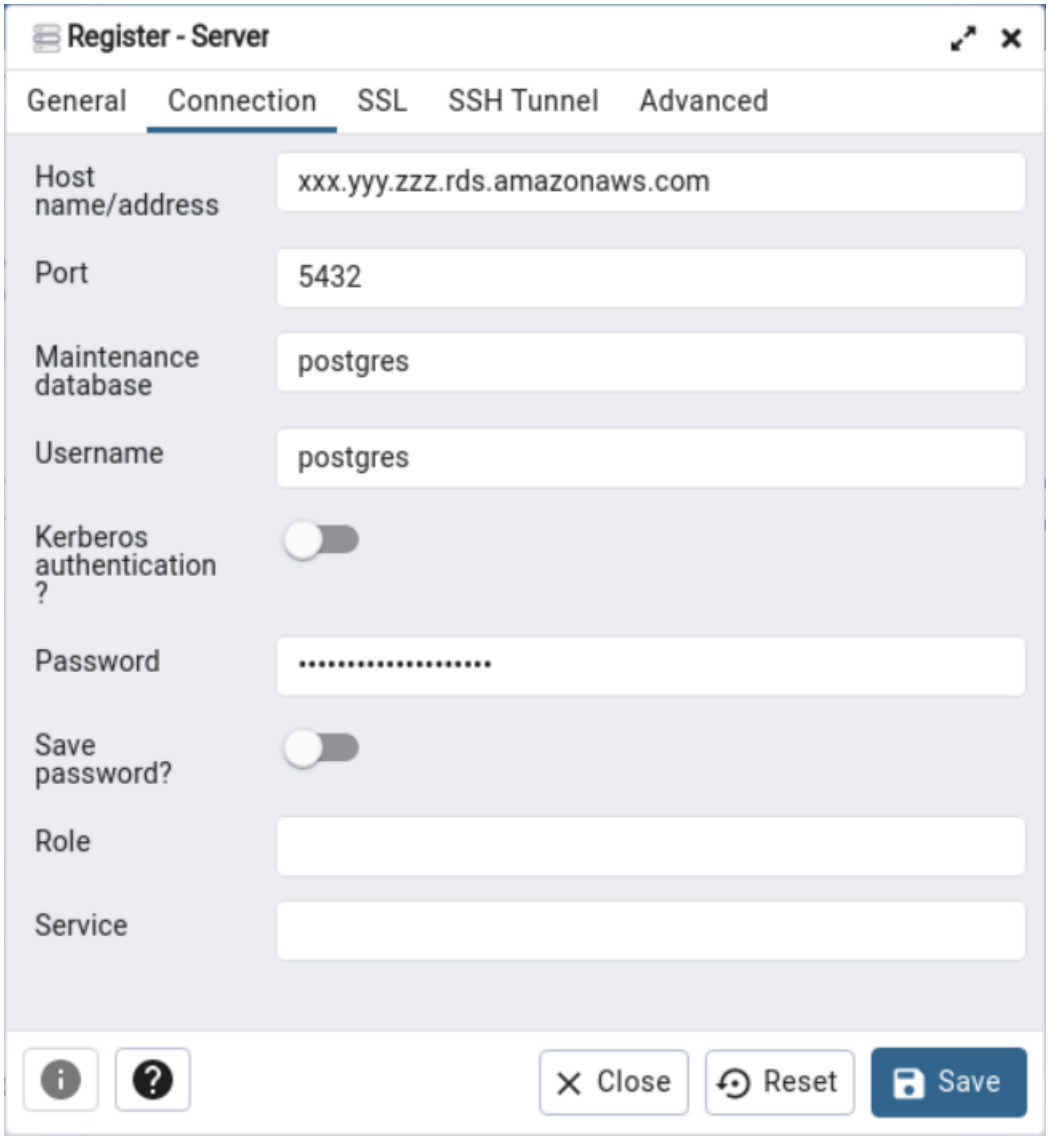
1. Desde la instancia AppStream 2.0, ejecute el siguiente comando en una terminal para iniciar: `pgAdmin`

   ```
   ./pgadmin-start.sh
   ```

2. Elija una dirección de correo electrónico y una contraseña como identificadores para el inicio de sesión. Tome nota de la URL proporcionada (normalmente es http://127.0.0.1:5050). Lance Google Chrome en la instancia, copie y pegue la URL en el navegador e inicie sesión con sus identificadores.



3. Después de iniciar sesión, elija Add New Server e introduzca la información de conexión a la base de datos creada anteriormente de la siguiente manera.

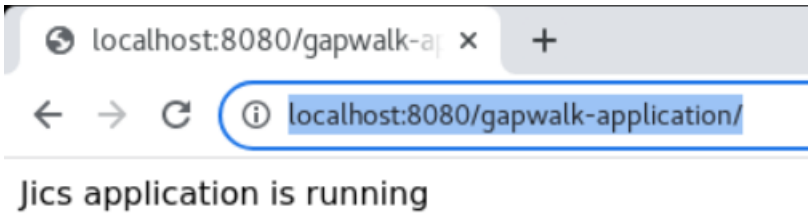


4. Cuando se conecte al servidor de base de datos, utilice Object > Create > Database y cree una nueva base de datos llamada jics.
5. Edite la información de conexión a la base de datos que ha utilizado la aplicación de demostración. Esta información se define en `PlanetsDemo-runtime/config/application-main.yml`. Busque la entrada `jicsDs`. Para recuperar los valores de `username` y `password`, vaya a la base de datos en la consola de Amazon RDS. En la pestaña Configuración, en ARN de credenciales maestras, seleccione Administrar en Secrets Manager. A continuación, en la consola de Secrets Manager, en el secreto, selecciona Recuperar valor del secreto.

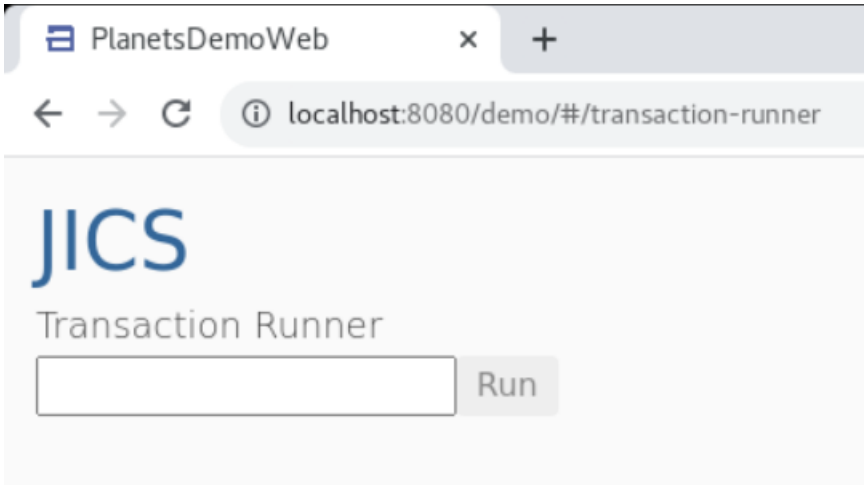
Paso 8: iniciar y probar la aplicación

En este paso, iniciará el servidor de Tomcat y la aplicación de demostración para poder probarla.

1. Para iniciar el servidor de Tomcat y las aplicaciones implementadas anteriormente, seleccione la entrada del servidor en la vista Servers y elija Start. Aparece una consola que muestra los registros de inicio.
2. Compruebe el estado del servidor en la vista Servers o espere a que aparezca el mensaje de Server startup in [xxx] milliseconds en la consola. Cuando se inicie el servidor, compruebe que gapwalk-application esté correctamente implementada. Para ello, acceda a la URL http://localhost:8080/gapwalk-application en un navegador Google Chrome. Debería ver lo siguiente:



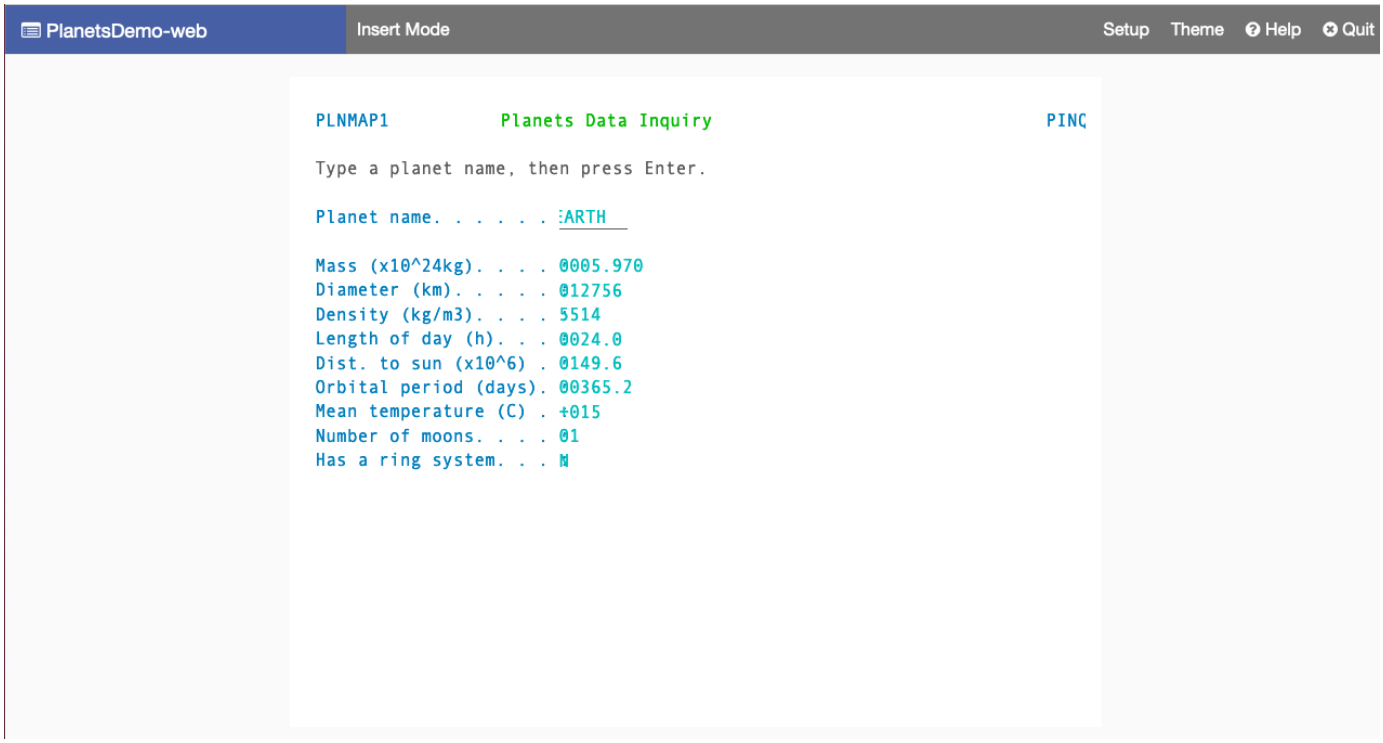
3. Acceda a la interfaz de la aplicación implementada desde Google Chrome en http://localhost:8080/demo. Debería aparecer la página Transaction Launcher siguiente.



4. Para iniciar la transacción de la aplicación, introduzca `PINQ` en el campo de entrada y elija Run (o presione Intro).

   Debería aparecer la pantalla de la aplicación de demostración.

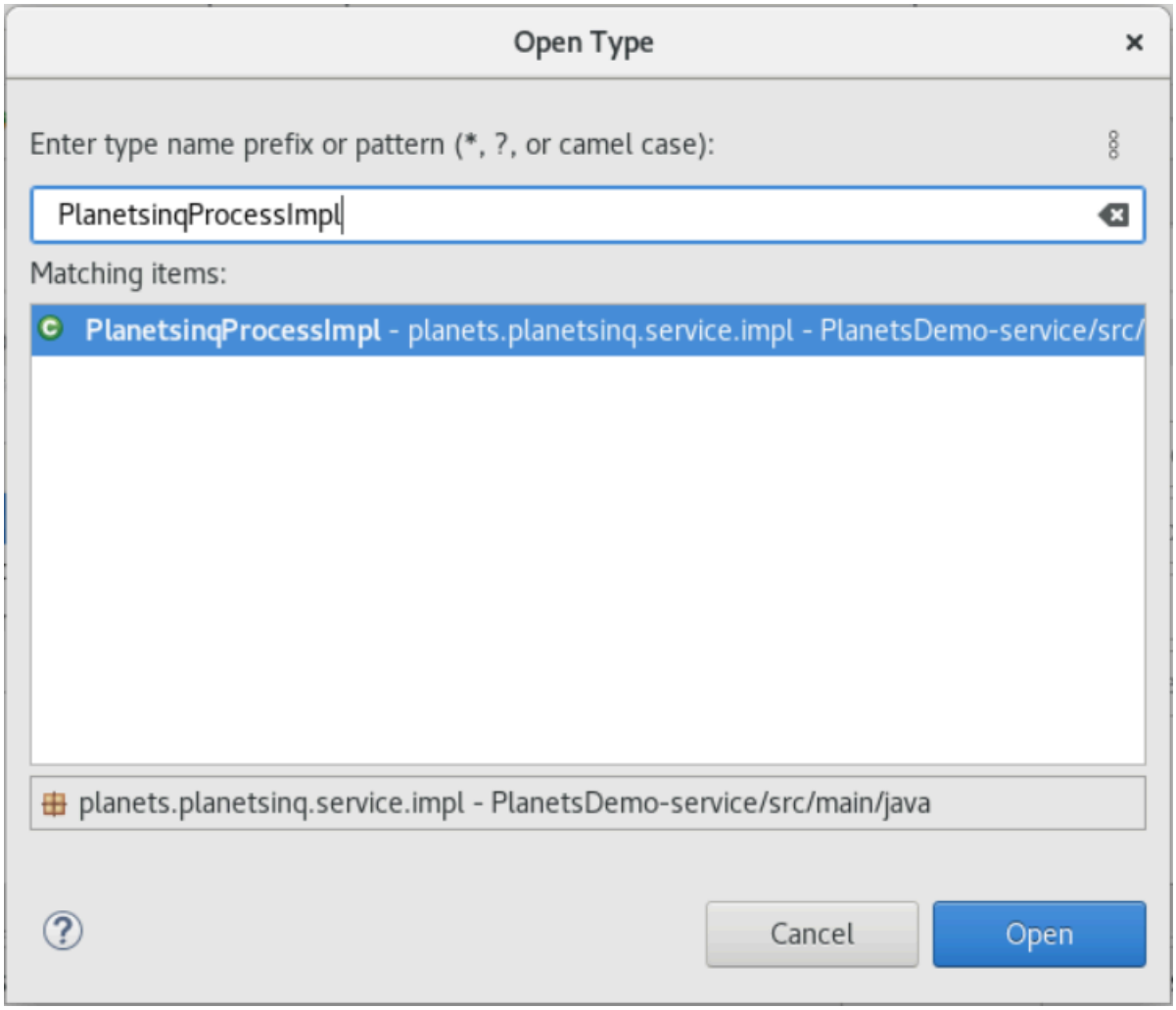
5. Escriba el nombre de un planeta en el campo correspondiente y pulse Intro.



Paso 10: Depurar la aplicación

En este paso, probará las características de depuración estándar de Eclipse. Estas características están disponibles cuando se trabaja en una aplicación modernizada.

1. Para abrir la clase de servicio principal, pulse Ctrl + Mayús + T. Después, introduzca `PlanetsinqProcessImpl`.



2. Navegue hasta el método `searchPlanet` y coloque allí un punto de ruptura.
3. Seleccione el nombre del servidor y elija Restart in Debug.
4. Repita los pasos anteriores. Es decir, acceda a la aplicación, introduzca el nombre de un planeta y pulse Intro.

   Eclipse detendrá la aplicación en el método `searchPlanet`. Ahora puede examinarlo.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para que no se le cobre por ellos. Realice los siguientes pasos:

- Si la aplicación Planets sigue ejecutándose, deténgala.
- Elimine la base de datos que ha creado en Paso 1: Crear una base de datos. Para obtener más información, consulte Eliminación de una instancia de base de datos.

AWS Preguntas frecuentes sobre Blu Age

General

1. ¿Cuál es el objetivo principal de la capacidad de refactorización de AWS Blu Age?

   La capacidad de refactorización refactoriza el código monolítico heredado para convertirlo en Java utilizando aplicaciones distribuidas contemporáneas que utilizan lenguajes y marcos modernos, siguiendo un patrón de refactorización automatizado. Este patrón implica analizar automáticamente el código heredado, comprender su funcionalidad y convertirlo en código moderno equivalente, preservando al mismo tiempo la lógica empresarial. El proceso incluye modernizar no solo el código, sino también toda la pila de aplicaciones, las dependencias y la infraestructura mediante herramientas y procesos automatizados. La solución tiene como objetivo acelerar la modernización y, al mismo tiempo, mantener la equivalencia funcional y el rendimiento. Esto incluye la transformación del código de la aplicación y las bases de datos y almacenes de datos asociados, al tiempo que se implementan las mejores prácticas y patrones de diseño en la nube.

2. ¿Qué aplicaciones de mainframe admite AWS Blu Age?

   AWS Actualmente, Blu Age apoya la modernización de IBM z/OS aplicaciones de mainframe escritas en COBOL, PL/I, JCL (Job Control Language) y que se basan en el administrador de transacciones CICS (Customer Information Control System), pantallas BMS (Basic Mapping Support), pantallas IMS MFS, bases de datos, DB2 bases de datos IMS, archivos planos, archivos de datos GDG (Generation Data Groups) y VSAM (Virtual Storage Access Method). Para obtener más información, consulte Blu Insights.AWS

3. ¿Qué lenguajes de mainframe puede modernizar AWS Blu Age?

AWS Blu Age transforma el código COBOL y PL/I en Java, en Groovy, JCLs en pantallas (BMS o MFS) en HTML (con Sass) y (aplicaciones angulares; por el momento, React no es compatible), lo que permite JavaScript modernizar las aplicaciones de mainframe antiguas para convertirlas en arquitecturas nativas de la nube. Estas tecnologías se eligen por su adopción generalizada, su sólido ecosistema y sus capacidades nativas de la nube. Angular proporciona una capa de interfaz de usuario moderna y con capacidad de respuesta que reemplaza a las antiguas interfaces de pantalla verde. Permite la creación de aplicaciones web dinámicas y fáciles de usar a las que se puede acceder desde diferentes dispositivos y plataformas. Su arquitectura basada en componentes permite un desarrollo front-end escalable y fácil de mantener. La transformación da como resultado aplicaciones distribuidas que siguen los patrones arquitectónicos modernos y las mejores prácticas.

4. ¿Cómo equilibra AWS Blu Age las limitaciones heredadas con los beneficios de la nube?

   AWS Blu Age logra el equilibrio al preservar la lógica y la funcionalidad empresariales fundamentales y, al mismo tiempo, introducir capacidades nativas de la nube. Garantiza que las aplicaciones modernizadas mantengan la lógica empresarial tradicional necesaria y, al mismo tiempo, aprovechen la escalabilidad de la nube, la agilidad y las prácticas operativas modernas. Este enfoque ayuda a las organizaciones a mantener la continuidad empresarial y, al mismo tiempo, a aprovechar las ventajas de la infraestructura de nube.

5. ¿Qué función desempeña la arquitectura orientada a los servicios en la aplicación modernizada?

   La arquitectura orientada a los servicios desempeña un papel fundamental a la hora de dividir las aplicaciones monolíticas en componentes modulares más fáciles de administrar. AWS Blu Age crea aplicaciones orientadas a servicios y a objetos que facilitan una mejor capacidad de mantenimiento y escalabilidad. Este enfoque arquitectónico permite a las organizaciones lograr una mayor eficiencia empresarial y prepararse para la posible adopción de microservicios en el futuro.

6. ¿Qué aspectos del conjunto de aplicaciones se incluyen en el proceso de refactorización?

   El proceso de refactorización incluye el paquete de software completo: código de aplicación, dependencias, bases de datos e infraestructura (por ejemplo, opciones de almacenamiento en caché, soporte de mensajería, etc.). Abarca la transformación de los lenguajes de programación antiguos, los sistemas de bases de datos, los archivos de datos y los componentes de infraestructura asociados. Este enfoque integral garantiza que todos los aspectos de la aplicación se modernicen de manera coherente, lo que da como resultado una pila de aplicaciones modernas totalmente transformada.

7. ¿Elimina el proceso de modernización de la AWS Blu Age la necesidad de realizar pruebas o controles de calidad en la aplicación Java modernizada?

   No, el proceso de modernización de la AWS Blu Age no elimina la necesidad de realizar pruebas o controles de calidad en la aplicación Java modernizada.

8. ¿Qué significa AWS Blu Age JAC?

   JAC son las siglas de JICS Administration Console

9. ¿Cómo puedo acceder a las herramientas de AWS Blu Age?

   AWS Se puede acceder a las herramientas de Blu Age a través de la consola de AWS mediante AWS Mainframe Modernization (M2) Refactor, y el acceso a las funciones se basa en su nivel de acreditación. Comience con el Centro de Transformación para evaluar la refactorización automática de Java de su código fuente. Para obtener una guía detallada, consulte la documentación de AWS Blu Insights. Tras la modernización, puede implementar aplicaciones mediante opciones de tiempo de ejecución gestionado o no gestionado. Para obtener más información sobre estas opciones de implementación, consulte la documentación sobre la modernización AWS del mainframe.

10. ¿Cómo dimensionar (carga de trabajo y cronograma) un proyecto?

    Consulte las estimaciones de AWS Blu Insights para obtener más información al respecto o póngase en contacto con su administrador de cuentas.

11. ¿Existen requisitos específicos para mantener las soluciones migradas a Java AWS Blu Age?

    No, no hay requisitos específicos para mantener las soluciones migradas de Java AWS Blu Age.

12. ¿Cuáles son las especificaciones técnicas y la compatibilidad del código generado por AWS Blu Age?

    AWS El código generado por Blu Age está diseñado con características técnicas específicas y una amplia compatibilidad. Si bien no es compatible con JPA, utiliza la ejecución directa de SQL con consultas externalizadas. El código se basa en bibliotecas específicas del tiempo de ejecución para la equivalencia funcional, la generación de servicios web y las implementaciones de MQ. El código generado se puede importar a cualquier IDE de Java para su desarrollo, prueba, compilación e implementación, aunque las bibliotecas necesarias se deben importar en consecuencia. Si bien Maven está integrado de forma predeterminada con el servicio de modernización de AWS mainframe para los procesos de compilación, se pueden usar herramientas alternativas como Gradle modificando el formato de empaquetado después de la

transformación. La plataforma ofrece flexibilidad en términos de herramientas de desarrollo y control de código fuente, con formación disponible para los equipos de desarrollo que gestionan el código. Para obtener más información, consulte la arquitectura de alto nivel de AWS Blu Age Runtime.

AWS Blu Age Runtime

1. ¿Dónde puedo encontrar información sobre AWS Blu Age Runtime?

   Consulte la documentación sobre la configuración del tiempo de ejecución de AWS Blu Age (no administrado) sobre el tiempo de ejecución no administrado, que detalla la incorporación del proceso de configuración, la recuperación de artefactos, la implementación, etc.

2. ¿Dónde puedo encontrar AWS Blu Age Runtime para desarrolladores?

   El AWS Blu Age Runtime para desarrolladores está disponible en Blu Age Toolbox para personas con certificación L3.

3. ¿Las dependencias JAR de AWS Blu Age se cargan en el repositorio Maven del cliente para su desarrollo local?

   Las bibliotecas se pueden importar EC2 mediante una AMI que se puede usar para configurar el entorno de desarrollo, prueba y producción. Se capacitará y capacitará al equipo para mantener o mejorar el código de aplicación generado. Para obtener más información, consulte la arquitectura de alto nivel de AWS Blu Age Runtime.

4. ¿A qué se refiere el término «Gapwalk» en los tarros de AWS Blu Age Runtime distribuidos?

   Para obtener información sobre Gapwalk, consulta los artefactos de AWS Blu Age Runtime.

5. ¿Cómo solicitar acceso al AWS Blu Age Runtime no gestionado?

   Sigue las instrucciones que aparecen al incorporar AWS Blu Age Runtime para solicitar acceso al AWS Support centro.

6. ¿Cuáles son los tiempos de ejecución compatibles con las aplicaciones refactorizadas de AWS Blu Age?

   Para explorar la gama completa de opciones de tiempo de ejecución para sus aplicaciones modernizadas, le recomendamos que consulte la guía de opciones de tiempo de ejecución de Blu Age.

7. ¿Cuándo se usa el AWS Blu Age Runtime?

Se necesita un motor de ejecución de AWS Blu Age para poder ejecutar aplicaciones AWS refactorizadas de Blu Age. Durante los proyectos de refactorización basados en AWS Blu Age, se necesita un tiempo de ejecución para probar las aplicaciones refactorizadas. Una vez finalizado el proyecto de refactorización, también se necesita un tiempo de ejecución para mantener, probar y ejecutar las aplicaciones refactorizadas de Blu Age en producción. AWS

8. ¿Cómo se AWS distribuyen los nuevos lanzamientos de Blu Age Runtime? AWS

   En el caso de M2 Managed Runtime, las actualizaciones, incluidos los parches y las versiones secundarias y principales, están disponibles en la AWS consola y AWS CLI. Incluyen actualizaciones del sistema operativo, cambios en el motor y en las dependencias, normalmente dentro de los 30 días siguientes a la disponibilidad general. AWS es responsable de los componentes compatibles y aplica las actualizaciones a las instancias de modernización AWS del mainframe de forma automática. Y lo mismo ocurre con otros entornos, como Custom Runtime, la AMI de Linux y los entornos locales.

9. ¿Con qué frecuencia se publican nuevas versiones principales y secundarias del entorno de ejecución AWS Blu Age?

   Las nuevas versiones se publican una o dos veces al mes, y los clientes pueden decidir cuándo y cómo actualizar sus instancias de tiempo de ejecución. Para obtener más información, consulte la página de versiones de AWS Blu Age.

10. ¿Cómo se AWS proporciona soporte para AWS Blu Age Runtime?

    Support se proporciona a través de AWS Support, donde los problemas se resuelven mediante la presentación de un ticket, y se aplica el SLA estándar. Para obtener más información, consulte el ciclo de vida de los componentes de modernización AWS del mainframe.

11. ¿Qué implica el tiempo de ejecución de AWS Blu Age para la modernización de AWS mainframes?

    El motor de ejecución de AWS Blu Age incluye bibliotecas de herramientas para acelerar la modernización, facilitar las integraciones en la nube y mejorar la calidad y el mantenimiento del código. También permite una mayor automatización de la modernización al facilitar las transiciones entre las arquitecturas heredadas y las arquitecturas de nube. El motor de ejecución permite gestionar las representaciones de memoria de los verbos y estructuras de datos tradicionales mediante expresiones idiomáticas de Java. Permite crear aplicaciones modernizadas basadas en técnicas de programación orientadas a objetos y capaces de reproducir los flujos de control heredados. Moderniza los conjuntos de datos VSAM heredados o el soporte de bases de

datos jerárquicas IMS mediante una base de datos relacional como Amazon Aurora. Sustituye en Java a las utilidades del sistema tradicionales (IDCAMS, IEBGENER, DFSORT, etc.) y a los sistemas de administración de transacciones tradicionales (CICS, IMS). Facilita las integraciones en la nube con el almacenamiento en caché en Amazon ElastiCache y el soporte para soluciones de AWS mensajería (SQS, Kinesis).

12¿ AWS Blu Age Runtime es compatible con arquitecturas de ordenadores que no son x86?

Actualmente, AWS Blu Age Runtime solo es compatible con arquitecturas y procesos informáticos basados en x86. AWS Blu Age Runtime no admite la computación basada en ARM ni en Graviton.

13¿Cómo pueden los clientes mantenerse informados sobre las versiones de AWS Blu Age Runtime, incluidas las notificaciones de nuevas versiones y el acceso al historial de versiones y las notas de las versiones?

Las nuevas versiones de AWS Blu Age Runtime están subidas a nuestra página de lanzamiento oficial. Recomendamos visitar esta página con regularidad, idealmente cada 3 meses, para ver las últimas versiones y actualizaciones. En cuanto al acceso al historial de versiones y a las notas de publicación, la disponibilidad depende de la fecha end-of-life (EOL) de cada versión principal. Para obtener información detallada sobre las fechas de caducidad, la planificación de la actualización de las versiones y el acceso a la información histórica, consulta el ciclo de vida de AWS Blu Age.

14¿Cuáles son los componentes principales de la arquitectura de alto nivel de AWS Blu Age Runtime?

La arquitectura AWS Blu Age Runtime consta de dos tipos de componentes principales. En primer lugar, están las bibliotecas Java (archivos jar) almacenadas en una carpeta compartida (accesible para el cargador de clases del servidor de aplicaciones) que proporcionan soporte para construcciones y declaraciones heredadas. En segundo lugar, están las aplicaciones web (archivos de guerra) que contienen aplicaciones basadas en Spring y que proporcionan marcos y servicios a programas modernizados. El motor de ejecución también incluye: un registro de programas que recopila todos los programas para su invocación y llamadas entre programas, y un registro de scripts que recopila todos los scripts de tareas modernizados. Estos componentes funcionan en conjunto para proporcionar un punto de entrada y un marco de ejecución unificados basados en REST para las aplicaciones modernizadas. El motor de ejecución y la aplicación modernizada se implementan juntos en un servidor de aplicaciones (por ejemplo, Tomcat).

15¿Cómo configurar la carpeta compartida que contiene los artefactos de AWS Blu Age Runtime?

Los artefactos de AWS Blu Age Runtime (jar) deben recopilarse en una carpeta compartida a la que pueda acceder el cargador de clases del servidor de aplicaciones. En el caso de un

servidor Tomcat, la configuración se realiza modificando el archivo de configuración normal denominado catalina.properties. Por ejemplo, si creó la carpeta compartida como una carpeta denominada «shared», en la carpeta tomcat, tendrá que modificar la entrada common.loader en catalina.properties para que el cargador de clases de tomcat pueda acceder a la carpeta compartida, de la siguiente manera:

```
common.loader="${catalina.base}/lib","${catalina.base}/lib/*.jar","${catalina.home}/
lib","${catalina.home}/lib/*.jar","${catalina.home}/shared","${catalina.home}/shared/
*.jar"
```

16¿Cómo gestiona AWS Blu Age Runtime la apatridia y la gestión de sesiones?

AWS Blu Age Runtime implementa la gestión de apatridia y sesiones a través de múltiples mecanismos. Para las sesiones HTTP, utiliza una identificación basada en cookies con almacenamiento en caché externo para el contexto del usuario. Las sesiones se pueden almacenar en varios almacenes de datos, incluidos Amazon ElastiCache, el clúster de Redis o los mapas en memoria. El diseño sin estado garantiza que la mayoría de los estados no transitorios se almacenen externamente en una «fuente única de información» común, lo que permite una alta disponibilidad y un escalado horizontal. Este enfoque, combinado con el equilibrio de carga y las sesiones compartidas, permite distribuir el diálogo orientado al usuario en varios nodos.

17¿Qué función desempeñan las aplicaciones web en el entorno de ejecución de AWS Blu Age?

Las aplicaciones web de AWS Blu Age Runtime cumplen múltiples funciones clave. Proporcionan marcos de ejecución que reproducen entornos y monitores de transacciones heredados (como los lotes de JCL, CICS o IMS). Ofrecen puntos de entrada basados en REST `gapwalk-application.war` para activar y controlar transacciones, programas y lotes. Además, emulan los programas proporcionados por el sistema operativo y los programas «controladores» especializados de los que dependen las aplicaciones antiguas para acceder a servicios como IMS DB o a los diálogos de usuario a través de MFS.

18¿Cómo se registran y administran los programas en AWS Blu Age Runtime?

Los programas de AWS Blu Age Runtime se registran a través de un ProgramRegistry sistema que se rellena al iniciar el servidor. Cada programa implementa la interfaz del programa y está marcado como un componente de Spring. Los programas se registran utilizando sus identificadores, con la posibilidad de introducir múltiples entradas si un programa tiene varios identificadores. El proceso de registro es automático y se registra en los registros de Tomcat. ProgramRegistryEsto permite que otros programas y scripts localicen y llamen a los programas registrados, manteniendo la modularidad y la interconectividad del sistema modernizado.

19¿Cómo se gestiona la configuración en las aplicaciones AWS Blu Age Runtime?

La configuración en AWS Blu Age Runtime se gestiona a través de archivos YAML que utilizan las capacidades del marco Spring Boot. Se utilizan dos archivos de configuración principales: application-main.yml para la configuración del marco y para las opciones específicas del cliente. `application-profile.yml` El sistema sigue la lógica de prioridad de Spring, lo que permite anular la configuración por varios medios. Se puede proporcionar una configuración adicional a través de JNDI para las bases de datos y los parámetros de la línea de comandos, lo que ofrece flexibilidad en la administración de la configuración. La configuración de los registradores se realiza mediante archivos de configuración xml de registro.

20¿Qué papel desempeñan los administradores de secretos en la configuración de AWS Blu Age Runtime?

Los administradores de secretos de AWS Blu Age Runtime protegen los datos de configuración confidenciales, como las credenciales de las bases de datos y las contraseñas de la caché de Redis. Permiten almacenar datos críticos en AWS secretos y hacer referencia a ellos en los archivos de configuración YAML. El sistema admite diferentes tipos de secretos, incluidos los secretos de bases de datos que rellenan automáticamente todos los campos relevantes y los secretos de contraseña única para los recursos protegidos por contraseña. Este enfoque mejora la seguridad al mantener los datos confidenciales separados de la configuración de la aplicación.

21¿Cómo pueden los desarrolladores escribir sus propios programas compatibles con AWS Blu Age Runtime?

Los desarrolladores pueden crear programas compatibles con AWS Blu Age Runtime implementando la interfaz del programa y siguiendo patrones específicos. El programa debe declararse como un componente de Spring, implementar los métodos necesarios y estar debidamente registrado en el. ProgramRegistry Los desarrolladores deben crear clases complementarias de contexto y configuración, gestionar los identificadores del programa y garantizar una integración adecuada con el marco de Spring. La implementación debe seguir las convenciones de AWS Blu Age Runtime para la estructura y ejecución del programa.

22¿Cómo gestiona AWS Blu Age Runtime los errores de ejecución del programa?

AWS Blu Age Runtime maneja los errores de ejecución del programa a través de múltiples mecanismos. En el caso de los trabajos por lotes, captura el estado de la ejecución, los códigos de salida y la información detallada sobre los errores en los detalles de la ejecución del trabajo. La gestión de errores incluye códigos de salida específicos (-1 para errores técnicos, -2 para errores del programa de servicio) y un registro detallado en los registros de Tomcat. El sistema se puede

configurar para anular las transacciones en caso de excepciones de tiempo de ejecución y ofrece opciones de notificación y recuperación de errores. Se puede acceder a los detalles de los errores a través de los puntos finales de REST para su supervisión y solución de problemas.

23 ¿Qué funciones de monitoreo de AWS Blu Age Runtime están disponibles para los trabajos por lotes?

AWS Blu Age Runtime proporciona capacidades de monitoreo para trabajos por lotes a través de varios puntos finales. Realiza un seguimiento del estado de ejecución de los trabajos, las horas de inicio y finalización, el modo de ejecución y los resultados detallados. El sistema ofrece puntos finales para enumerar los scripts activados, recuperar los detalles de la ejecución de los trabajos y monitorear los trabajos que se están ejecutando actualmente. Los puntos finales de Metrics proporcionan estadísticas de JVM, recuentos de sesiones y métricas detalladas de ejecución por lotes. La plataforma también admite la paginación y el filtrado temporal de los datos de monitoreo.

24 ¿Cómo se rastrean y gestionan los estados de ejecución de los trabajos de AWS Blu Age Runtime?

Los estados de ejecución de los trabajos se rastrean a través de un completo sistema de estados que incluye estados como DONE, TRIGGERED, RUNNING, KILLED y FAILED. Cada ejecución de tareas recibe un identificador único para el seguimiento y mantiene información detallada sobre la ejecución, incluida la hora de inicio, la hora de finalización, la información de la persona que llama y los resultados de la ejecución. El sistema proporciona puntos finales REST para consultar el estado de los trabajos, gestionar los trabajos en ejecución y recuperar el historial de ejecuciones. La información de estado permanece en la memoria del servidor y se puede purgar según su antigüedad para administrar los recursos.

25 ¿Cómo gestiona AWS Blu Age Runtime las interacciones externas del sistema?

El tiempo de ejecución gestiona las interacciones externas del sistema a través de varios mecanismos, incluidos los puntos finales REST para la integración de servicios, el soporte para colas de mensajes (SQS, RabbitMQ, IBM MQ) y las opciones de conectividad de bases de datos. Permite emular las interacciones de los sistemas antiguos mediante componentes especializados, es compatible con SSL/TLS para garantizar la seguridad de las comunicaciones e incluye funciones para gestionar sistemas de archivos externos. El sistema también admite la integración con proveedores de autenticación externos y se puede configurar para interactuar con varios servicios de terceros.

26 ¿Cómo se gestiona la autenticación en AWS Blu Age Runtime?

AWS Blu Age Runtime admite varios métodos de autenticación, OAuth2 siendo el mecanismo principal. Se puede integrar con proveedores de identidad como Amazon Cognito o Keycloak. La configuración de la autenticación se administra a través del archivo de configuración principal denominado application-main.yml, donde se pueden definir los ajustes de seguridad, los proveedores de identidad y los métodos de autenticación. El sistema admite funciones como la protección XSS, CORS y CSRF, y se puede configurar tanto para la seguridad global como para la seguridad de puntos finales específicos. Para el desarrollo, también está disponible un sistema de autenticación local con credenciales de superadministrador predeterminadas.

27 ¿Cómo garantiza AWS Blu Age Runtime la alta disponibilidad?

AWS Blu Age Runtime garantiza una alta disponibilidad a través de varios mecanismos. Implementa la apatridia al almacenar los estados no transitorios en un almacenamiento compartido externo, lo que permite que varias instancias de aplicaciones trabajen juntas. El sistema admite el equilibrio de carga y las sesiones compartidas, lo que permite distribuir las solicitudes entre varios nodos. Para el almacenamiento de datos, puede utilizar bases de datos y sistemas de almacenamiento en caché de alta disponibilidad. La arquitectura admite la conmutación por error automática y se puede implementar en varias zonas de disponibilidad para aumentar la confiabilidad.

28 ¿Qué componente se utiliza para reproducir las transacciones distribuidas de CICS con las aplicaciones de AWS Blu Age?

El AWS Blu Age Runtime proporciona un punto final dedicado que permite invocar las transacciones JICS existentes como parte de una transacción global (compatible con XA). Las dos fases subyacentes confirman que el soporte se basa en el componente de software Atomikos.

29 ¿Cómo se llama AWS Blu Age a las clases que se utilizan para definir el comportamiento específico de un programa?

Cada programa está vinculado a una clase de configuración dedicada que permite especificar los comportamientos específicos del programa. Para obtener más información sobre las convenciones de nomenclatura y ubicación, consulte la estructura de la aplicación modernizada de AWS Blu Age

30 ¿Qué codificación tiene el siguiente orden de secuencia de caracteres: espacio, caracteres en minúscula, caracteres en mayúscula, números?

Conjuntos de caracteres que pertenecen a la familia de variantes EBCDIC (como CP1 047, etc.). CP297

31 ¿Cómo se utiliza el tiempo de ejecución gestionado por AWS Blu Age?

¿Con el AWS Management Console AWS CLI, el o el AWS APIs

32 ¿Cuáles son los precios de AWS Blu Age Runtime?

AWS Mainframe Modernization-core-hours (consulte los precios de modernización de AWS mainframe).

33 ¿Cuál es el mecanismo que se utiliza para pasar los datos sin procesar a través de HTTP a los puntos finales del programa?

Cadenas codificadas en Base64.

34 ¿Cómo ejecuta un usuario un trabajo por lotes?

Mediante una llamada HTTP a uno de los puntos finales de lote dedicados (consulte la página de documentación de los puntos finales de lote).

35 ¿Qué terminal de ejecución de AWS Blu Age es el punto de entrada principal desde la aplicación principal de interfaz web?

```
/transaction
```

36 ¿Qué significa AWS Blu Age JICS?

El JICS de AWS Blu Age es el componente de tiempo de ejecución que se utiliza para respaldar la modernización de los recursos del CICS. Las definiciones de los recursos se almacenan en un almacén de datos dedicado. Para administrarlos, utilice la API REST o la consola de aplicaciones JICS. Para obtener más información, consulte Administrar la consola de aplicaciones JICS en AWS Blu Age.

37 ¿Qué mecanismos de almacenamiento en caché de AWS Blu Age Runtime están disponibles?

AWS Blu Age Runtime admite varios mecanismos de almacenamiento en caché, incluidos Redis y. EhCache Se recomienda Redis para entornos de producción, ya que proporciona almacenamiento en caché persistente compartido en varios nodos. EhCache está disponible para implementaciones independientes con almacenamiento en caché local volátil integrado. El sistema admite el almacenamiento en caché de varios componentes, incluidos los datos de Blusam, la información de la sesión, los recursos del JICS y las colas de almacenamiento temporal. La configuración de la caché se puede personalizar para diferentes casos de uso y requisitos de rendimiento.

38 ¿Cómo calculamos el precio de una implementación de AWS Blu Age Runtime para modernizar un AWS mainframe?

AWS proporciona estimaciones a los clientes en función de sus requisitos y la arquitectura objetivo.

39. ¿Cuál es el precio de AWS Mainframe Modernization AWS Blu Age Runtime?

    AWS Mainframe Modernization ofrece dos modelos de precios para AWS Blu Age: una opción de tiempo de ejecución administrado que incluye el tiempo de ejecución, los recursos de procesamiento, el almacenamiento interno y la automatización, y una opción de tiempo de ejecución no administrado que cubre únicamente el tiempo de ejecución de AWS Blu Age en sí. Para AWS las implementaciones, ambas utilizan una pay-as-you-go estructura de precios. Para obtener la información más up-to-date detallada sobre los precios, se recomienda consultar la página oficial de precios de modernización de mainframe de AWS.

40. ¿Qué sucede si necesitamos implementar una aplicación refactorizada de AWS Blu Age en una infraestructura que no aparece en la lista del entorno de ejecución compatible?

    Si necesita implementar una aplicación refactorizada de AWS Blu Age en una infraestructura que no figure en la lista del entorno de ejecución compatible, hay varias opciones disponibles. En primer lugar, compruebe si su infraestructura es compatible con las opciones de implementación existentes, como Amazon EKS Anywhere u otras plataformas de organización de contenedores. Si es así, es posible que pueda utilizar el AWS Blu Age Runtime (no administrado). En el caso de infraestructuras no compatibles, recomendamos consultar con un especialista en AWS mainframe para explorar soluciones personalizadas o posibles adaptaciones. También puede enviar una solicitud de características del producto (PFR) para ampliar el soporte de infraestructura. Es posible que haya opciones de facturación alternativas disponibles para las implementaciones no estándar. Póngase en contacto con su AWS representante para analizar sus necesidades específicas y el mejor enfoque para su entorno.

41. ¿Cómo se licencia el AWS Blu Age Runtime? ¿Es de código abierto?

    AWS Blu Age Runtime no es de código abierto. Se distribuye por AWS IP como un servicio nativo de la nube. Hay dos opciones de implementación:

    a. AWS Con Blu Age Managed, el tiempo de ejecución se despliega en un servicio AWS gestionado dedicado, que aprovecha un entorno totalmente preconfigurado y listo para el despliegue, sin necesidad de configuración ni administración.

    b. AWS Blu Age Non Managed, que se puede implementar en su propia AWS arquitectura personalizada basada en Amazon o EC2 Amazon ECS/AWS Fargate, que debe aprovisionar y configurar usted mismo. Ambas opciones conllevan gastos de tiempo de ejecución, que se incluyen en las estimaciones del proyecto que se le proporcionan. Como se trata de un servicio

gestionado con Soporte acceso, no necesitas el código fuente. Para obtener más información sobre los precios, consulte la página de precios de modernización de AWS mainframe.

42. ¿Cómo se gestionan los cambios y las actualizaciones de los marcos y bibliotecas de AWS Blu Age?

    AWS Los marcos y bibliotecas de Blu Age se actualizan mediante procesos regulares de generación e implementación de código. Estas actualizaciones se administran como parte del ciclo de vida de modernización del AWS mainframe, que incluye actualizaciones de versiones y soporte del equipo de AWS Blu Age o de socios certificados. Para obtener información detallada sobre el control de versiones, los procesos de actualización y los plazos de soporte, consulte la documentación sobre el ciclo de vida de la modernización de los AWS mainframes.

Datos

1. ¿Qué opciones de bases de datos están disponibles para las aplicaciones modernizadas en relación con la modernización de la base de datos antigua?

   Las aplicaciones modernizadas pueden utilizar varias opciones de bases de datos modernas, como PostgreSQL, Amazon Aurora, RDS for PostgreSQL, base de datos Oracle, MS-SQL e IBM Db2. Estas opciones ofrecen flexibilidad a la hora de elegir el sistema de base de datos más adecuado en función de los requisitos específicos, al tiempo que aprovechan las ventajas de los sistemas de gestión de bases de datos modernos y de las funciones nativas de la nube.

2. ¿Para qué sirve la cobertura de transformación de IBM Db2 z/OS ¿a Postgres DDL?

   Transformación completa (incluidas las restricciones de la base de datos).

3. ¿ AWS Blu Age admite la generación de datos grupales (GDG)?

   Sí, se admite el uso de GDG en lotes, con el apoyo de generaciones relativas y absolutas y de estrategias de limpieza automática.

4. ¿Admite AWS Blu Age conjuntos de datos concatenados?

   Sí, se admite el uso de conjuntos de datos concatenados en lotes. Con la concatenación en acción, se pueden leer varios conjuntos de datos como un único conjunto de datos. Tenga en cuenta que los conjuntos de datos de Blusam no pueden formar parte de una concatenación.

5. ¿Cuál es el proceso que se aplica a las consultas SQL?

   Se ajusta durante la transformación del código, en función de la base de datos de destino.

6. ¿Qué opciones se aplican si hay varias bases de datos para una aplicación?

   Configure la base de datos de destino para cada consulta y defina todas las bases de datos de la aplicación y de Apache Tomcat.

7. ¿Se puede desactivar Blusam?

   Sí, en el archivo de configuración principal y no se requiere ninguna base de datos (para obtener más información, consulte la página de documentación de configuración de Blusam).

8. ¿Qué API de AWS Blu Age se utiliza para reemplazar bases de datos como IMS DB?

   La API JHDB (Java Jerarical DataBase).

9. ¿Qué producto de AWS Blu Age se puede usar para migrar datos y bases de datos antiguos a un sistema moderno de administración de bases de datos relacionales (RDBMS)?

   AWS Herramienta de modernización de Blu Age DB (Data Migrator).

10. ¿Qué es AWS Blu Age Data Simplifier y qué problema resuelve en la modernización?

    Data Simplifier es una biblioteca básica de AWS Blu Age que aborda el desafío de gestionar los patrones de acceso a la memoria heredados en Java. Proporciona estructuras que admiten el acceso a la memoria de bajo nivel, los tipos de datos heredados (como zonificados, empaquetados o alfanuméricos) y los basados en clases y mixtos, lo que los hace accesibles a los desarrolladores de Java structured/raw memory access that are common in mainframe applications but not natively available in Java. The library exposes these features through familiar Java patterns like getters/setters y APIs, al mismo tiempo, mantiene la funcionalidad tradicional.

11. ¿Cómo gestiona AWS Blu Age los diseños de memoria y las estructuras de datos antiguos?

    AWS Blu Age gestiona los diseños de memoria antiguos a través de la interfaz Record, que proporciona una abstracción de matrices de bytes con un tamaño fijo. Para los datos estructurados, como los «01 elementos de datos» de COBOL, utiliza RecordEntitysubclases que se generan automáticamente durante la modernización. Estas clases mantienen la estructura jerárquica de los datos heredados, y cada elemento tiene una relación padre-hijo. El sistema admite tanto el acceso a la memoria sin procesar como los patrones de acceso estructurados, lo que preserva la flexibilidad de los sistemas heredados y, al mismo tiempo, proporciona una interfaz de programación moderna.

12. ¿Cómo aborda AWS Blu Age la modernización de los conjuntos de datos de VSAM?

    El componente Blusam proporciona soporte para la modernización de los conjuntos de datos de VSAM, con una API dedicada, terminales y una aplicación web de administración (BAC: consola

de administración de Blusam). Blusam se basa en una base de datos relacional como backend (PostgreSQL, ya sea con RDS o Aurora).

Transformación

1. ¿Dónde puedo encontrar detalles sobre el proceso de transformación?

   Consulte la documentación de AWS Blu Insights.

2. ¿Cuáles son los nombres de los módulos generados por AWS Blu Age?

   Servicio, entidades, web y herramientas.

3. ¿Por qué se eligió Java/Spring como una de las tecnologías objetivo de AWS Blu Age?

   Se eligió Java/Spring como tecnología objetivo debido a su adopción generalizada, su gran reserva de talentos y sus sólidas capacidades empresariales. El ecosistema de Java ofrece amplias bibliotecas, marcos y herramientas que respaldan el desarrollo de aplicaciones modernas. El marco Spring proporciona funciones de nivel empresarial, capacidades nativas de la nube y sigue las mejores prácticas del sector, lo que lo hace ideal para aplicaciones modernizadas.

4. ¿Cómo se llama el proyecto principal que contiene los módulos generados por AWS Blu Age?

   El nombre del proyecto principal tiene el sufijo «-pom» y se puede definir en el Centro de Transformación mediante la propiedad Transform denominada project.

5. ¿Cómo gestiona AWS Blu Age la modernización de los planificadores antiguos, si los tiene?

   Blu Age no moderniza los activos de los programadores antiguos. AWS Se tienen en cuenta durante la fase de evaluación para ayudar a identificar posibles artefactos faltantes.

6. ¿Cuáles son los requisitos para depurar el código generado con AWS Blu Age?

   Cualquier entorno de desarrollo integrado (IDE) compatible con Java, como Eclipse o VisualCode. JetBrain

Implementación

1. ¿Qué entornos están disponibles para implementar la aplicación modernizada con AWS Blu Age?

   Windows Server, servidor Linux y contenedor Docker Linux.

2. ¿Pueden las aplicaciones refactorizadas de AWS Blu Age ejecutarse en cualquier infraestructura?

Si bien las aplicaciones refactorizadas de AWS Blu Age no están diseñadas para ejecutarse en ninguna infraestructura, ofrecen una flexibilidad significativa en las opciones de implementación. Estas aplicaciones se pueden implementar en varias plataformas informáticas, incluidos los servicios gestionados en la nube, la informática sin servidor y la infraestructura local. AWS Blu Age ofrece opciones de tiempo de ejecución gestionado y no gestionado, lo que permite a las organizaciones elegir entre una comodidad totalmente gestionada y un control personalizado en función de sus necesidades y requisitos específicos. Esta flexibilidad permite moverse fácilmente entre las infraestructuras compatibles, lo que hace que las aplicaciones refactorizadas de AWS Blu Age se adapten en gran medida a los diferentes entornos de implementación. Para obtener más información, consulte la documentación sobre las opciones de tiempo de ejecución de AWS Blu Age.

3. ¿Qué configuración de MQ admite AWS Blu Age?

   SQS, IBM WebSphere MQ.

4. ¿En qué servidores de aplicaciones puede un usuario implementar la lógica de aplicaciones empresariales de Java con el tiempo de ejecución no gestionado de AWS Mainframe Modernization?

   Apache Tomcat, versión superior o igual a la 10.1.

5. ¿Cómo se integra la aplicación refactorizada con otras como Amazon Servicios de AWS Aurora?

   La aplicación modernizada se integra Servicios de AWS y permite la transformación a soluciones de bases de datos nativas de la nube, como Amazon Aurora y RDS para PostgreSQL. AWS Blu Age garantiza la integración entre las aplicaciones modernizadas y permite a las organizaciones utilizar las capacidades de Servicios de AWS la nube. Esta integración se extiende tanto al almacenamiento de datos como a los servicios de aplicaciones dentro del AWS ecosistema. Más allá del almacenamiento de bases de datos, AWS Blu Age Runtime se integra con varios Servicios de AWS , como Amazon ElastiCache para el almacenamiento en caché de Redis, AWS Secrets Manager para la gestión de la configuración, y AWS Mainframe Modernization para la implementación. Es compatible con Amazon EC2, Amazon EKS y ECS gestionados por Fargate para el despliegue de contenedores. El sistema puede utilizar AWS Identity and Access Management para la autenticación Amazon Simple Storage Service para el almacenamiento y admite la integración con otros Servicios de AWS a través de conectores de configuración y servicio.

6. ¿Cómo garantiza la aplicación refactorizada que se cumplan los requisitos de escalabilidad?

La solución garantiza la escalabilidad al transformar las aplicaciones en arquitecturas nativas de la nube que pueden utilizar la infraestructura elástica. AWS Implementa patrones de diseño modernos y mejores prácticas que permiten el escalado horizontal y vertical. El enfoque orientado a los servicios permite un escalado independiente de los componentes. Las aplicaciones modernizadas pueden aprovechar las características de escalabilidad inherentes a los servicios en la nube.

7. ¿Qué ocurre una vez finalizada la refactorización del código fuente?

   Tras la refactorización del código fuente, se producen dos pasos principales. En primer lugar, se crea la aplicación refactorizada. En segundo lugar, la aplicación se implementa y supervisa en AWS Mainframe AWS Modernization Blu Age Runtime. El despliegue se puede realizar en un entorno AWS gestionado (tiempo de ejecución gestionado por AWS Mainframe Modernization), en el que la infraestructura se gestiona de forma automatizada, o en su entorno Cuenta de AWS (entorno de ejecución no gestionado por AWS Mainframe Modernization AWS Blu Age), en el que los clientes gestionan su propia infraestructura. La opción no administrada se puede implementar en varias plataformas, incluidas Amazon EC2, ECS en Fargate EC2 o EKS en. EC2

8. ¿Cómo puedo implementar y ejecutar una aplicación modernizada con AWS Blu Age en una AMI Amazon Linux personalizada sin usar el servicio gestionado AWS Mainframe Modernization (M2)?

   Esto se puede lograr desplegando la aplicación mediante AWS Blu Age Runtime (no gestionado) en Amazon EC2. El proceso implica crear una aplicación Java/Spring que dependa de la biblioteca AWS Blu Age Runtime y desplegarla en una AMI de Amazon Linux personalizada. Para obtener instrucciones detalladas sobre este enfoque, consulte Configurar AWS Blu Age Runtime (no administrado) en Amazon EC2.

9. ¿Existe una imagen de máquina de Amazon (AMI) disponible? ¿Hay alguna imagen de Docker disponible?

   - AMI: No, debido a que los clientes necesitan personalizar y configurar su entorno como prefieran, no hay ninguna AMI disponible. Los clientes pueden recuperar los artefactos de la Era AWS Azul y configurar su instancia según sus necesidades.
   - Imagen de Docker: No, no hay ninguna imagen de docker disponible lista para usar, pero la página AWS Configurar Blu Age Runtime en un contenedor explica cómo crear e implementar su propia imagen de docker basada en binarios de Blu Age Runtime, AWS en un sistema de administración de contenedores adecuado.

10. ¿El cliente puede empaquetar y ejecutar una aplicación de AWS Blu Age como un contenedor de Docker?

No es posible para M2 Managed Runtime, pero sí para un entorno definido por el cliente basado en una AMI de Amazon Linux y para proveedores locales u otros proveedores de nube.

11 ¿Cómo puedo saber el ARN del recurso de la política de SQS necesario para ejecutar AWS Blu Age de forma no administrada si quiero reducirlo?

Para determinar el ARN del recurso de políticas de SQS específico para AWS ejecutar Blu Age de forma no administrada con una política limitada, consulte al equipo de entrega o al administrador técnico de cuentas (TAM). Pueden proporcionar orientación específica para cada cuenta. Para obtener información general sobre las políticas de SQS, consulte la documentación de políticas de AWS SQS.

12 ¿Cómo funciona la programación de trabajos con lotes?

Está integrado con la sucursal Control-M/Stone o con cualquier otro programador distribuido.

Recreación de plataformas de aplicaciones con Rocket Software (anteriormente Micro Focus)

Esta guía describe el end-to-end proceso de cambio de plataforma de las aplicaciones de mainframe utilizando soluciones de modernización de AWS mainframe en. AWS Describe todas las tareas e incluye información sobre la configuración y el funcionamiento del entorno de ejecución de modernización de AWS mainframe en Amazon, EC2 desde la configuración y el análisis iniciales hasta la creación, las pruebas y el despliegue de las aplicaciones modernizadas. AWS También se tratan temas avanzados, como el trabajo con estructuras de datos heredadas, el uso de plantillas y proyectos predefinidos, y la configuración de la automatización para las sesiones de streaming.

Temas

- Configurar Rocket Software (anteriormente Micro Focus) (en Amazon EC2)
- Configure la automatización para las sesiones de streaming de Rocket Enterprise Analyzer (anteriormente Micro Focus) y Rocket Enterprise Developer
- Vea los conjuntos de datos como tablas y columnas en Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)
- Edite conjuntos de datos con las herramientas de archivos de datos de Rocket Software (anteriormente Micro Focus) en Enterprise Developer
- Tutoriales para Rocket Software (anteriormente Micro Focus)
- Utilidades por lotes disponibles en la modernización de AWS mainframe

Configurar Rocket Software (anteriormente Micro Focus) (en Amazon EC2)

AWS Mainframe Modernization proporciona varios Amazon Machine Images (AMIs) que incluyen productos con licencia de Rocket Software (anteriormente Micro Focus). Esto AMIs le permite aprovisionar rápidamente instancias de Amazon Elastic Compute Cloud (Amazon EC2) para que sean compatibles con los entornos de Rocket Software que usted controla y administra. En este tema se proporcionan los pasos necesarios para acceder a ellos y lanzarlos AMIs. Su uso AMIs es totalmente opcional y no es obligatorio para completar los tutoriales de esta guía del usuario.

Temas

- Requisitos previos para configurar Rocket Software (anteriormente Micro Focus) (en Amazon EC2)
- Creación de puntos de conexión de VPC de Amazon para Amazon S3
- Solicitud de actualización de la lista de permitidos de la cuenta
- Crea el AWS Identity and Access Management rol
- Concesión de permisos necesarios con License Manager
- Suscríbase a Amazon Machine Images
- Lance una instancia de AWS Mainframe Modernization Rocket Software (anteriormente Micro Focus)
- Subred o VPC sin acceso a Internet

Requisitos previos para configurar Rocket Software (anteriormente Micro Focus) (en Amazon EC2)

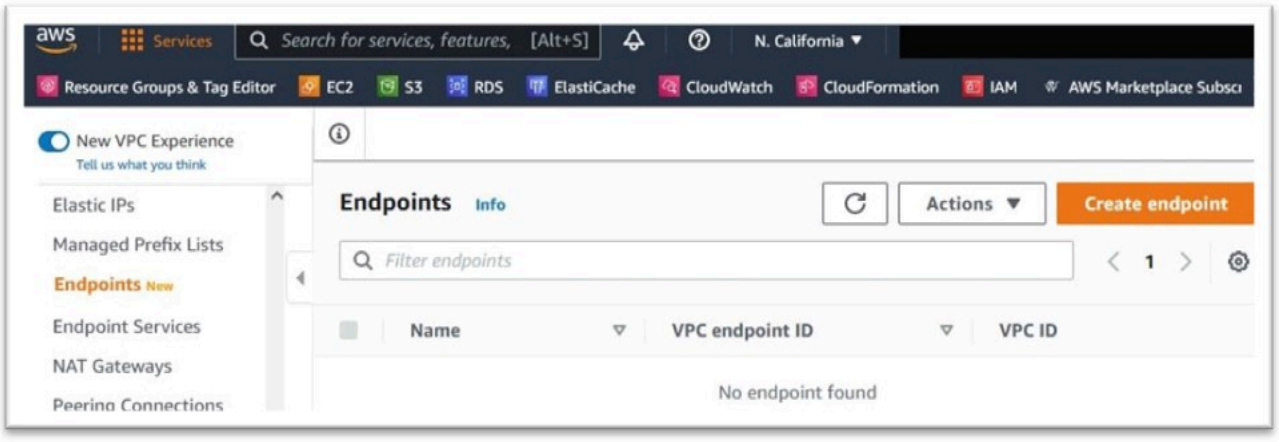
Al configurar Rocket Software (en Amazon EC2), asegúrate de cumplir los siguientes requisitos previos.

- Acceso de administrador a la cuenta en la que se crearán EC2 las instancias de Amazon.
- Identifica el Región de AWS lugar donde se crearán EC2 las instancias de Amazon y verifica que el AWS Mainframe Modernization servicio esté disponible. Consulte Servicios de AWS por región. Asegúrese de elegir una región en la que el servicio esté disponible.
- Identifique la Amazon Virtual Private Cloud (Amazon VPC) en la que se crearán EC2 las instancias de Amazon.

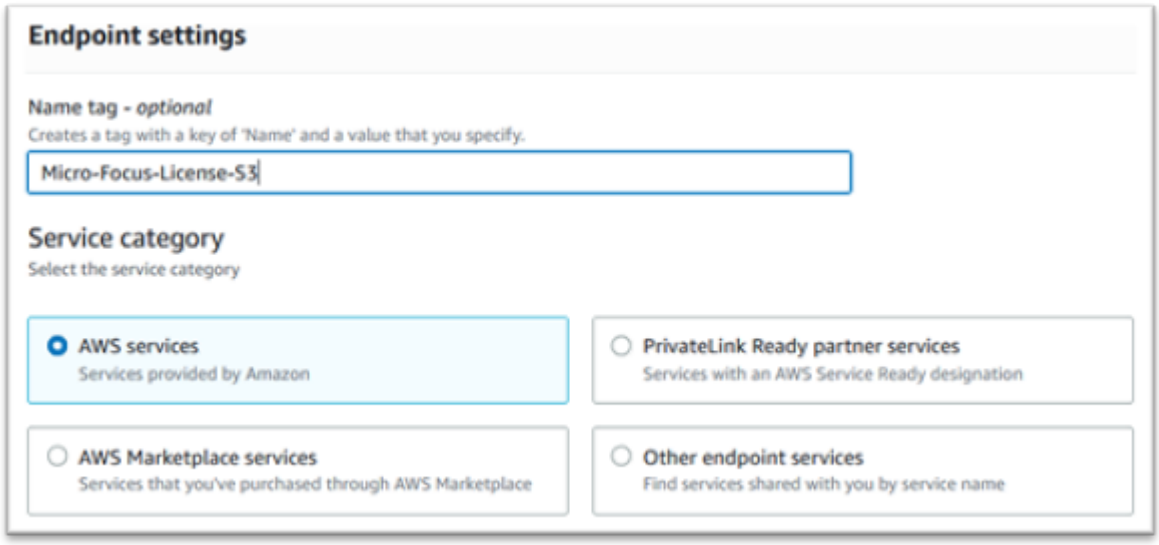
Creación de puntos de conexión de VPC de Amazon para Amazon S3

En esta sección, se crea un punto de conexión de Amazon VPC para Amazon S3. La configuración de este punto de conexión le ayudará más adelante a la hora de configurar el acceso a Internet para VPC.

1. Vaya a Amazon VPC en la AWS Management Console.
2. En el panel de navegación, elija Puntos de conexión.
3. Elija Crear punto de conexión.

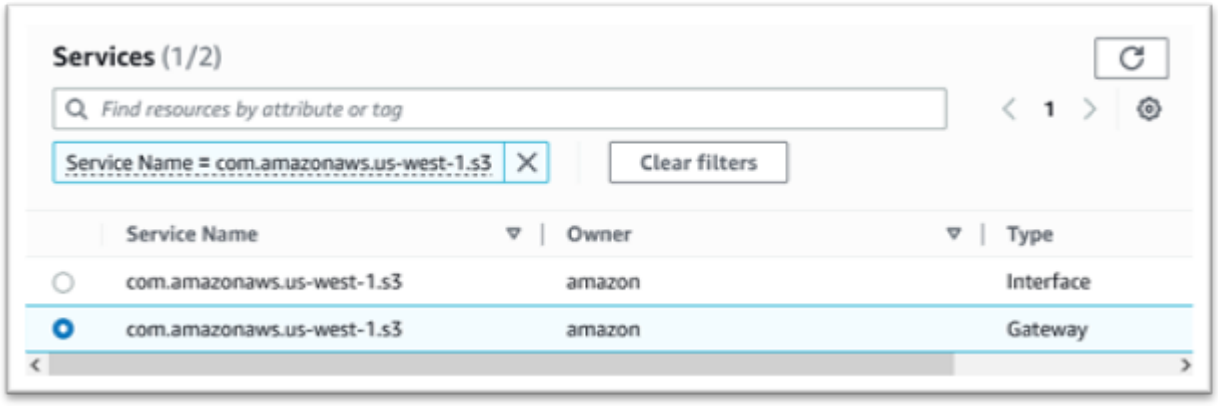


4. Introduzca una etiqueta con un nombre significativo, por ejemplo: “Micro-Focus-License-S3”.
5. Elija Servicios de AWS en Categoría de servicio.

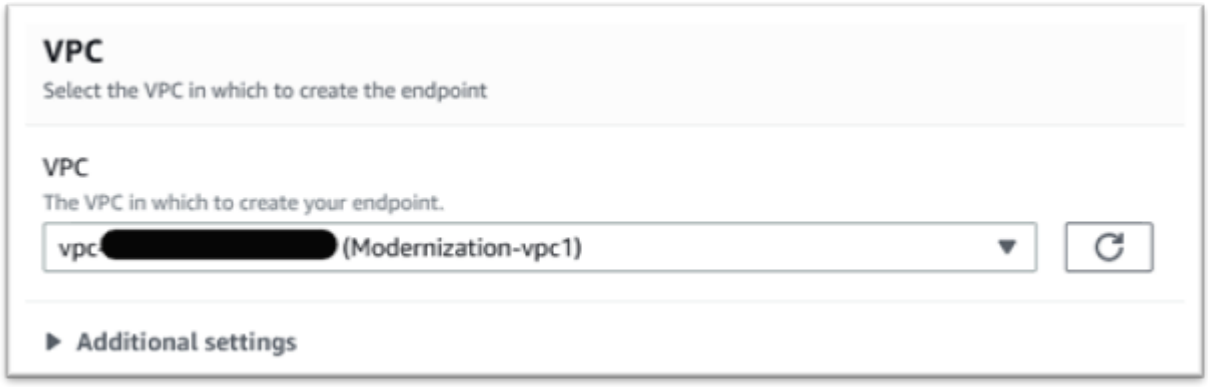


6. En Servicios, busque el servicio de puerta de enlace de Amazon S3: com.amazonaws. [región] .s3.

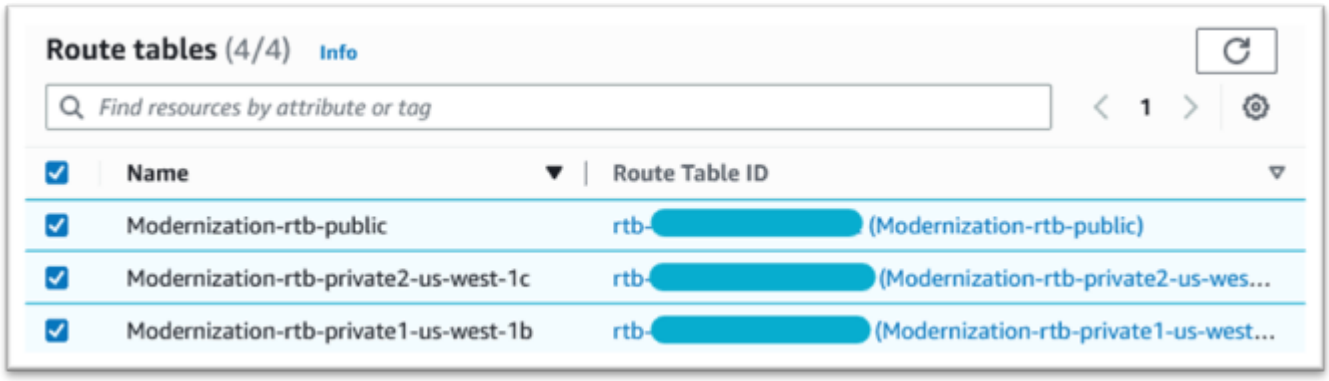
   Para `us-west-1` sería: `com.amazonaws.us-west-1.s3`.
7. Elija el servicio Puerta de enlace.



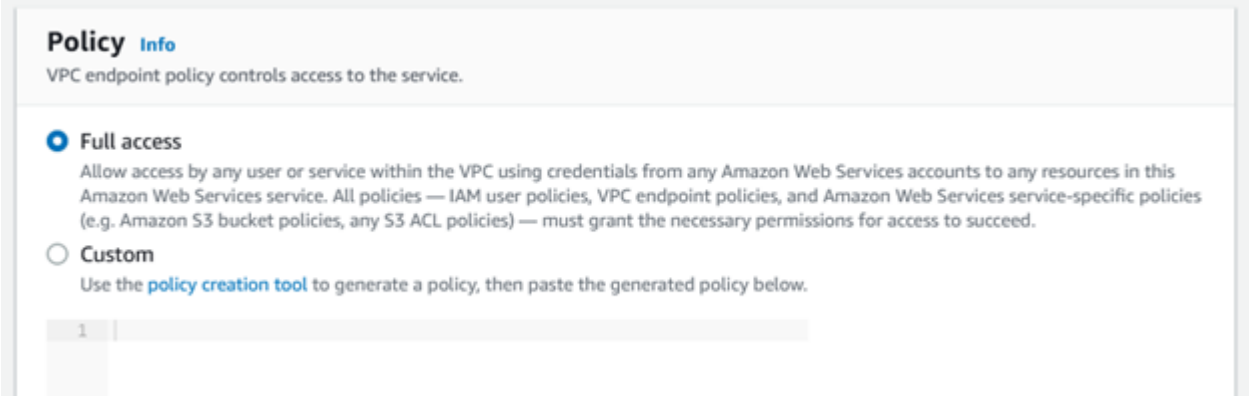
8. Como VPC, elija la VPC que vaya a utilizar.



9. Elija todas las tablas de enrutamiento de la VPC.



10. En Política, seleccione Full Acces.



> **Note**
>
> Si decide crear una política personalizada, asegúrese de que tiene acceso al bucket de Amazon S3 `s3://aws-supernova-marketplace-<region>-prod`.

11. Elija Crear punto de conexión.

Solicitud de actualización de la lista de permitidos de la cuenta

Trabaje con su AWS representante para incluir su cuenta en la lista de permitidos para. AWS Mainframe Modernization AMIs Proporcione la siguiente información:

- La Cuenta de AWS identificación.
- El Región de AWS lugar donde se creó el punto final de Amazon VPC.
- El ID del punto de conexión de Amazon VPC para Amazon S3 que se creó en Creación de puntos de conexión de VPC de Amazon para Amazon S3. Este es el identificador `vpce-xxxxxxxxxxxxxxxxx` del punto de conexión com.amazonaws.[region].s3 Gateway
- El número de licencias necesarias en todas las EC2 instancias de Amazon (AMI) de Rocket Software Enterprise Suite.

  Se requiere una licencia por núcleo de CPU (por cada 2 v CPUs para la mayoría de las EC2 instancias de Amazon).

  Para obtener más información, consulte Optimización de las opciones de CPU.

  El número solicitado se puede ajustar en el futuro mediante AWS.

> **Note**
>
> Ponte en contacto con tu AWS representante o con la AWS Support persona que abra el formulario de asistencia para solicitar la lista de permitidos en tu nombre. No puedes solicitarlo directamente y la solicitud puede tardar varios días en completarse.

Crea el AWS Identity and Access Management rol

Crea una AWS Identity and Access Management política y un rol para que los usen las EC2 instancias de AWS Mainframe Modernization Amazon. Al crear el rol a través de la consola de IAM, se creará un perfil de instancia asociado con el mismo nombre. Al asignar este perfil de instancia a las EC2 instancias de Amazon, se pueden asignar licencias de software de Rocket. Para obtener más información sobre los perfiles de instancia, consulta Cómo usar un rol de IAM para conceder permisos a las aplicaciones que se ejecutan en EC2 instancias de Amazon.

Creación de una política de IAM

Primero se crea una política de IAM y después se asocia al rol.

1. Navegue hasta AWS Identity and Access Management . AWS Management Console
2. Elija Políticas y, a continuación, Crear política.

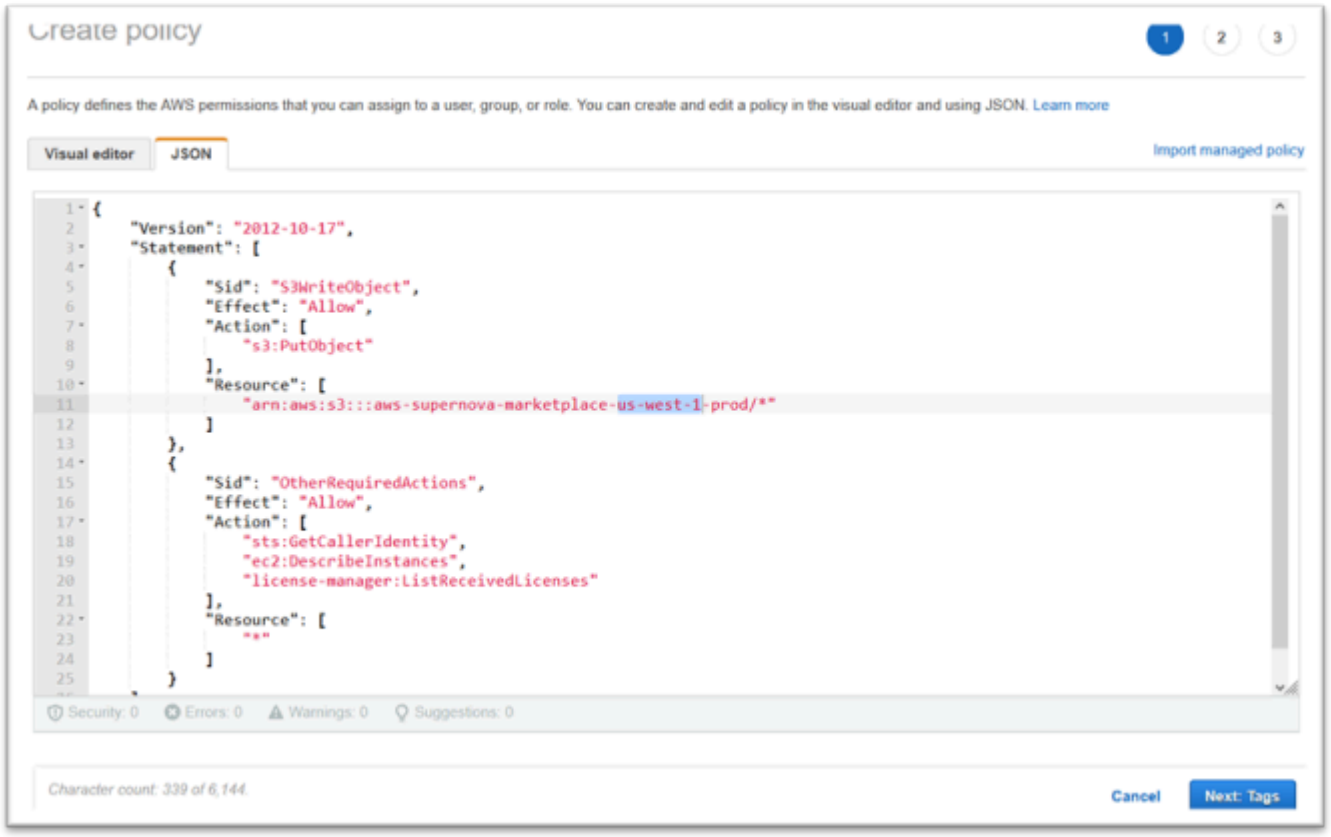
3. Seleccione la pestaña JSON.

4. Sustituya `us-west-1` el siguiente JSON por el Región de AWS lugar donde se definió el punto de conexión de Amazon S3 y, a continuación, copie y pegue el JSON en el editor de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "S3WriteObject",
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::aws-supernova-marketplace-us-west-1-prod/*"
            ]
        },
        {
            "Sid": "OtherRequiredActions",
            "Effect": "Allow",
            "Action": [
                "sts:GetCallerIdentity",
                "ec2:DescribeInstances",
                "license-manager:ListReceivedLicenses"
            ],
            "Resource": [
                "*"
            ]
        }
    ]
```

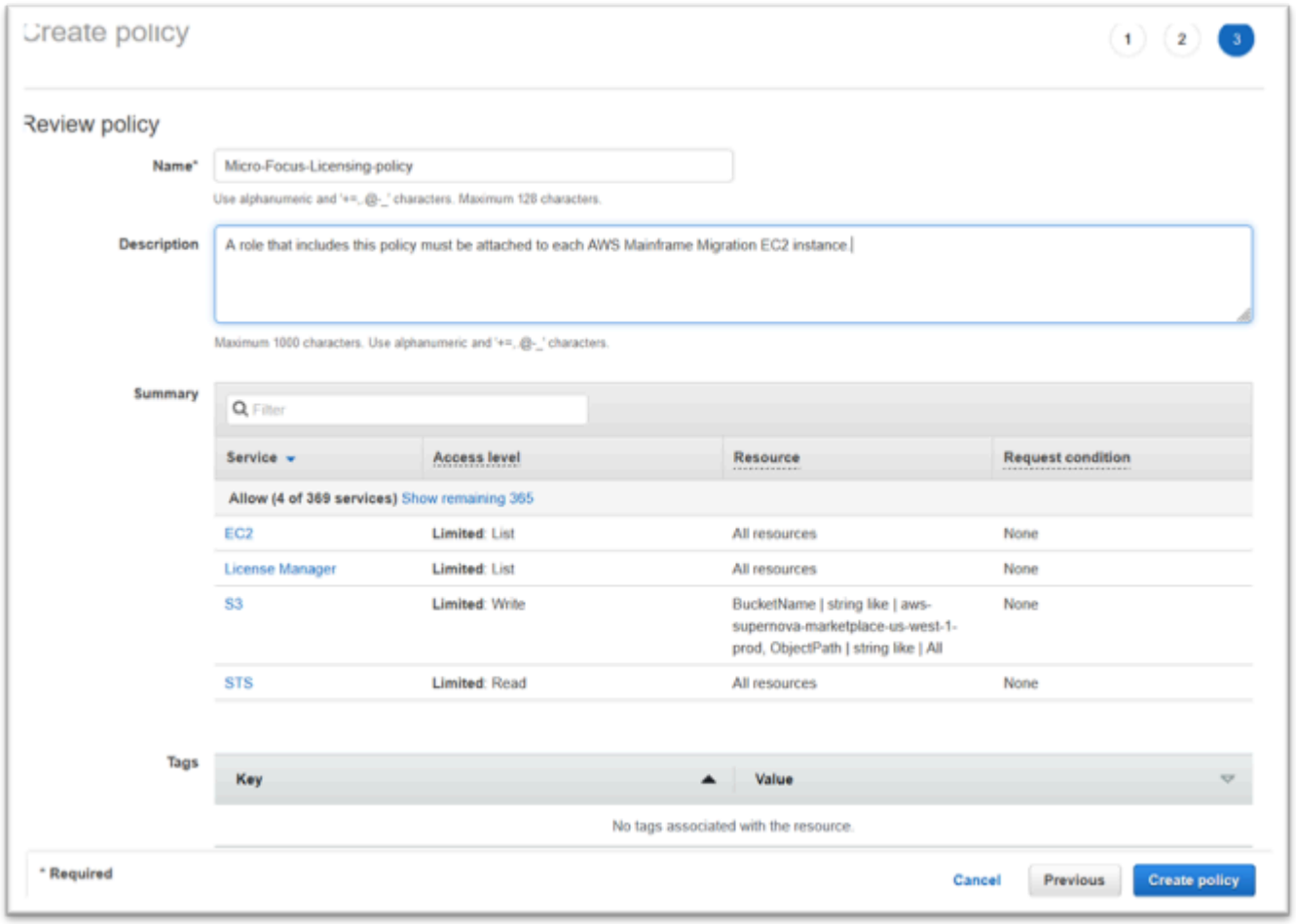
```
}
```

> **Note**
>
> Las acciones bajo el Sid `OtherRequiredActions` no admiten permisos de recursos y deben especificar * en el elemento de recurso.



5. Elija Siguiente: etiquetas.

6. Si lo desea, introduzca cualquier etiqueta y, a continuación, seleccione Siguiente: Revisar.
7. Introduzca un nombre para la política, por ejemplo, “Microfocus Licensing-Policy”. Si lo desea, introduzca una descripción, por ejemplo: «Debe adjuntarse un rol que incluya esta política a cada EC2 instancia de AWS Mainframe Modernization Amazon».

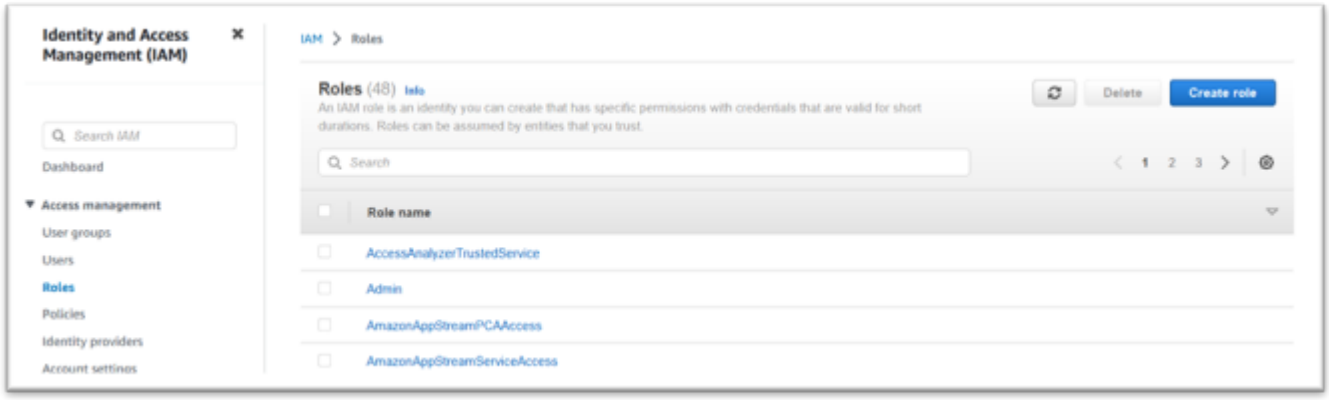


8. Elija Crear política.

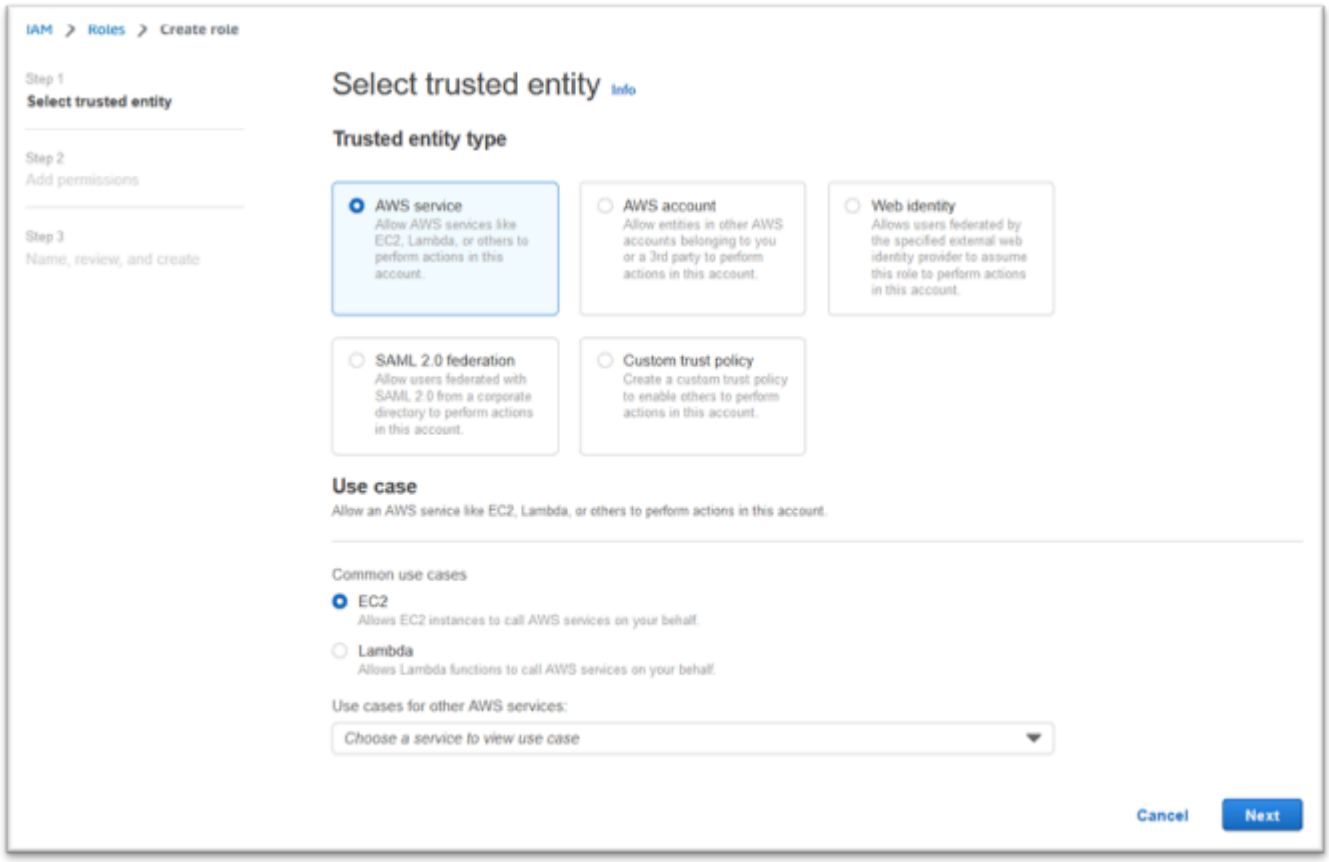
Creación del rol de IAM

Después de crear una política de IAM, cree un rol de IAM y asócielo a la política.

1. En la AWS Management Console, vaya a IAM.
2. Elija Roles y, a continuación, Crear rol.

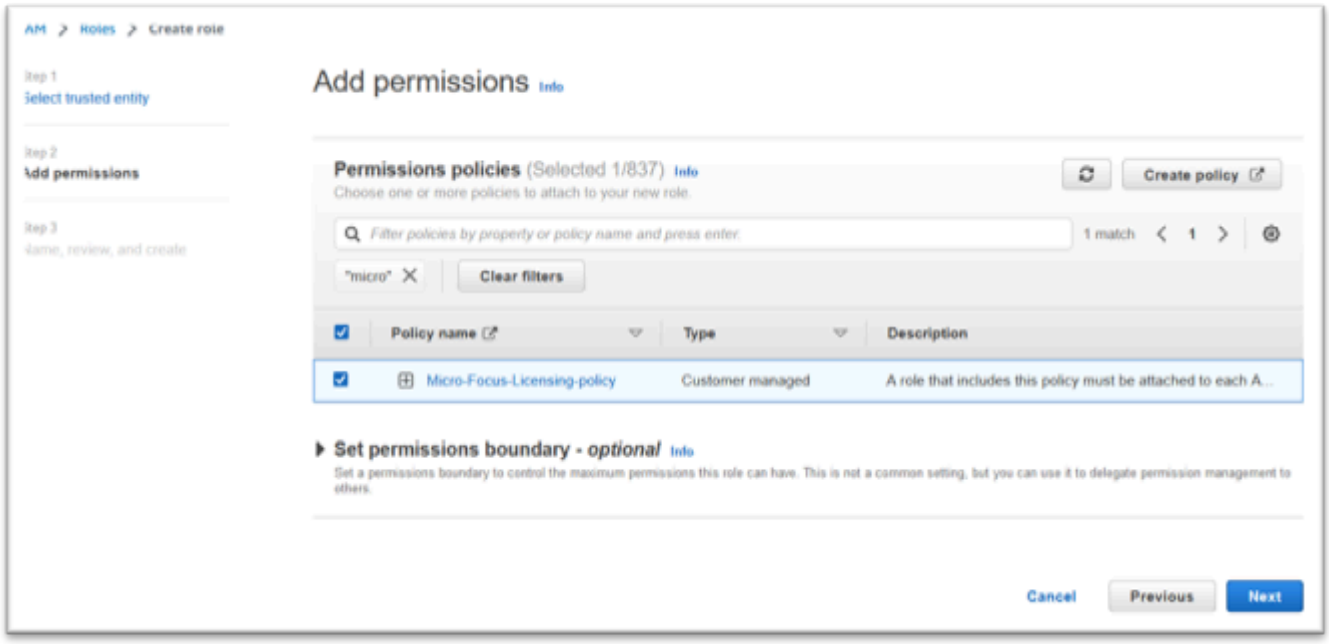


3. Deje el tipo de entidad de confianza como AWS servicio y elija el caso de uso EC2habitual.

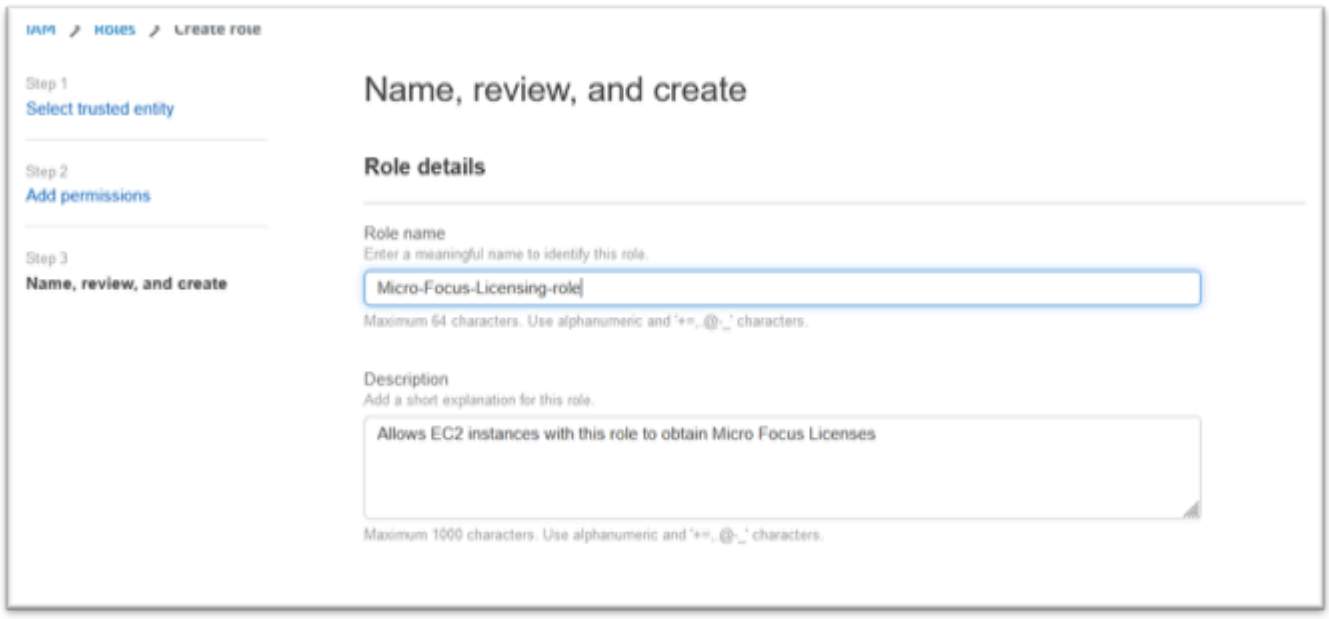


4. Elija Siguiente.

5. Introduzca “Micro” en el filtro y pulse Intro para aplicar el filtro.
6. Elija la política que se acaba de crear, por ejemplo, la “Micro-Focus-Licensing-policy”.
7. Elija Siguiente.



8. Introduzca el nombre del rol, por ejemplo, “Micro-focus-Licensing-Role”.
9. Sustituya la descripción por una propia, por ejemplo, «Permite que EC2 las instancias de Amazon con este rol obtengan licencias de Micro Focus».

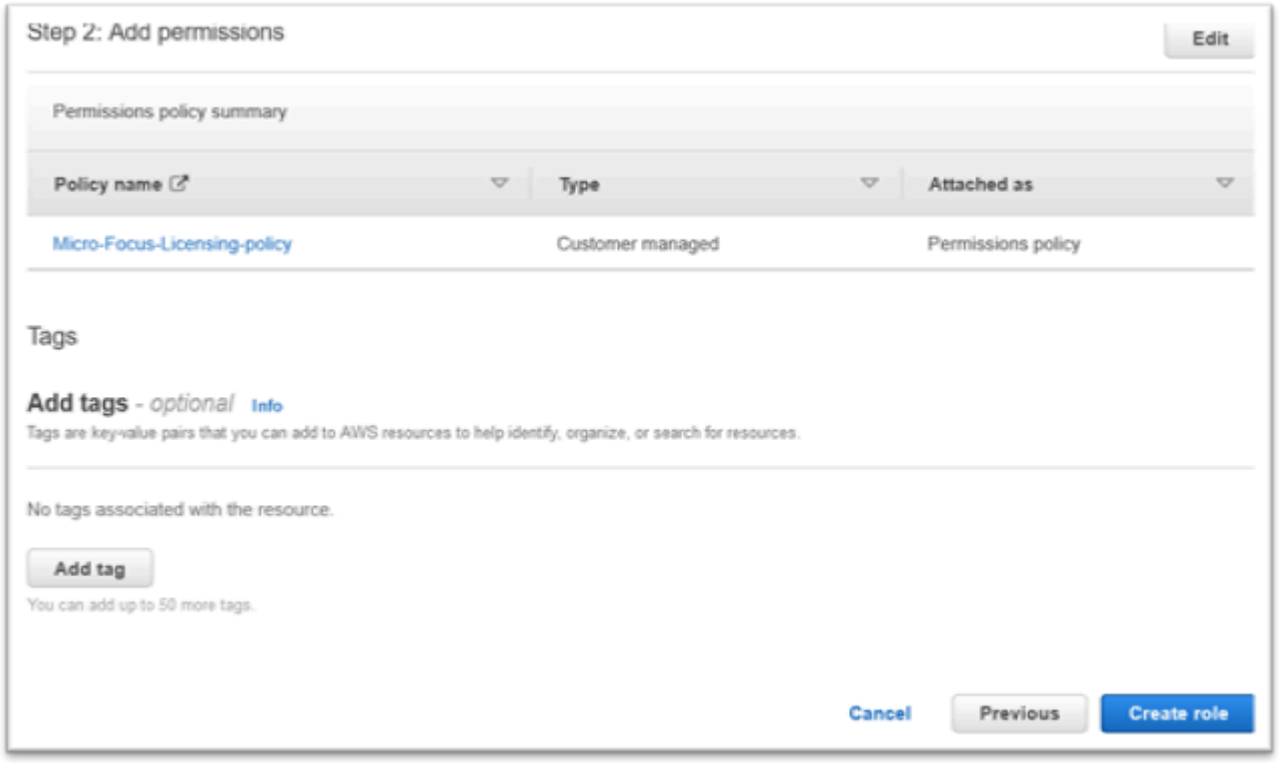


10. En el Paso 1: Seleccionar entidades de confianza, revise el JSON y confirme que tiene los siguientes valores:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sts:AssumeRole"
            ],
            "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            }
        }
    ]
}
```

> **Note**
>
> El orden del efecto, la acción y el principio no son significativos.

11. Confirme que el Paso 2: Agregar permisos muestre su política de licencias.



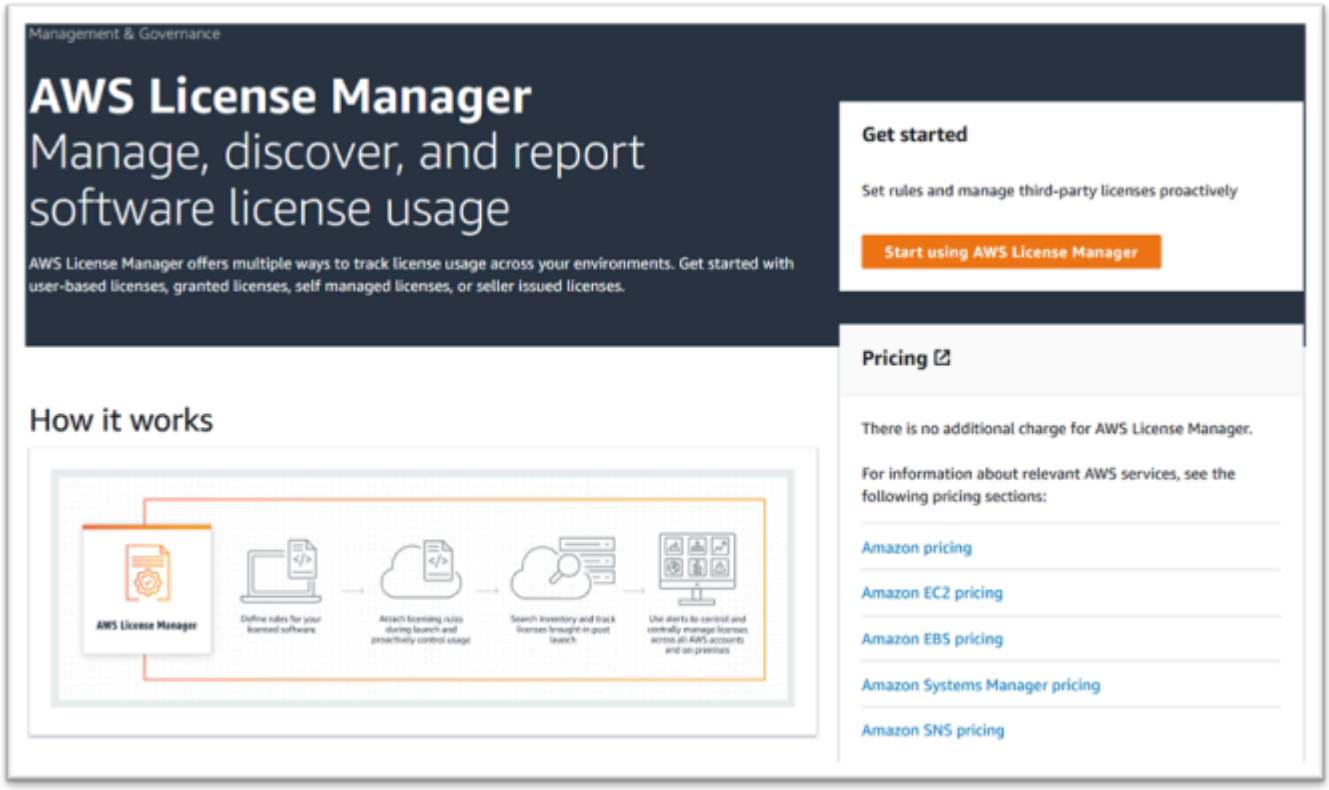
12. Elija Create role.

Una vez que se haya completado la solicitud de lista de permitidos, continúe con los siguientes pasos.

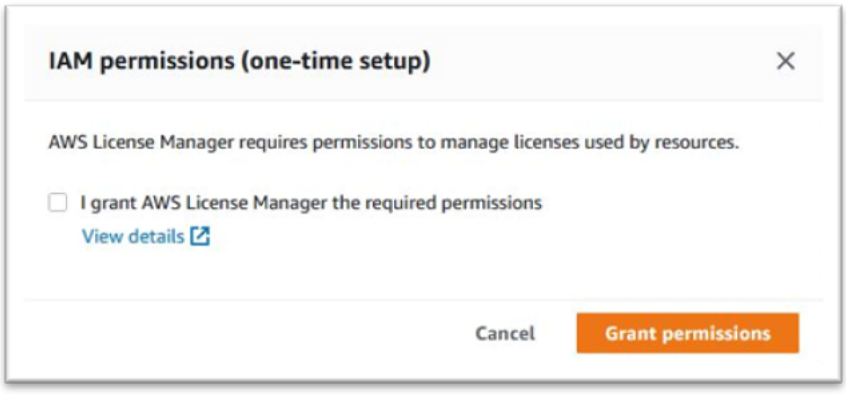
Concesión de permisos necesarios con License Manager

Debes concederte permisos AWS License Manager para configurar el motor de ejecución de Rocket Software (en Amazon EC2).

1. Navegue hasta AWS License Manager AWS Management Console.



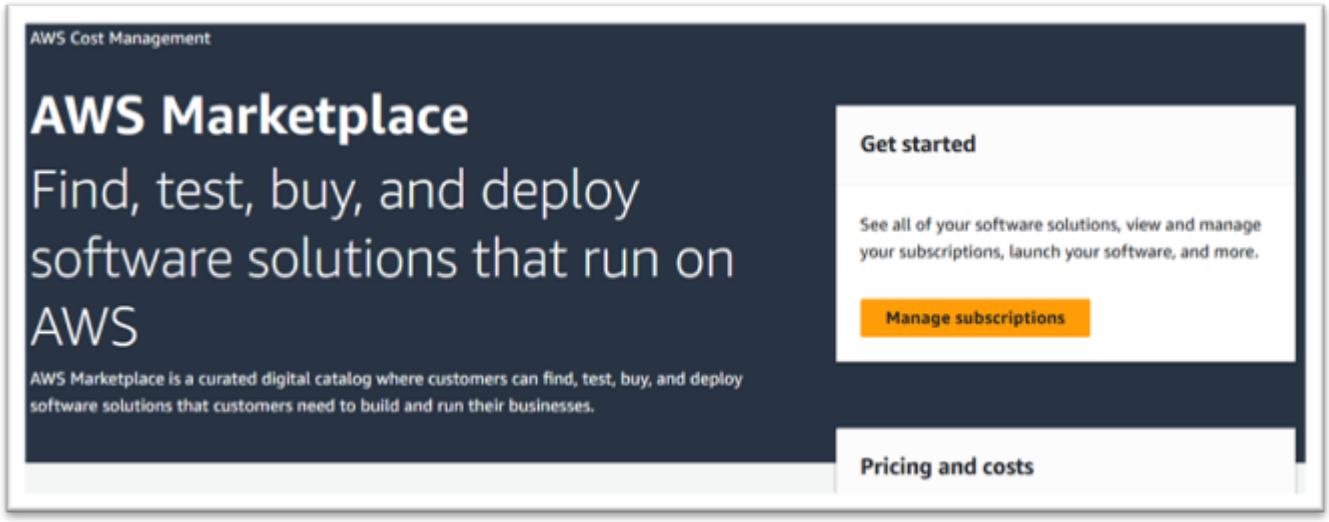
2. Elija Comenzar a usar AWS License Manager.
3. Si ve la siguiente ventana emergente, consulte los detalles, seleccione la casilla de verificación y pulse Conceder permisos.



Suscríbase a Amazon Machine Images

Tras suscribirse a un AWS Marketplace producto, puede lanzar una instancia desde la AMI del producto. También puedes gestionar tus suscripciones AMIs al configurar el motor de ejecución de Rocket Software (anteriormente Micro Focus) (en Amazon EC2).

1. Ve a AWS Marketplace Suscripciones en. AWS Management Console
2. Elija Administrar suscripciones.



3. Copie y pegue uno de los siguientes enlaces en la barra de direcciones del navegador.

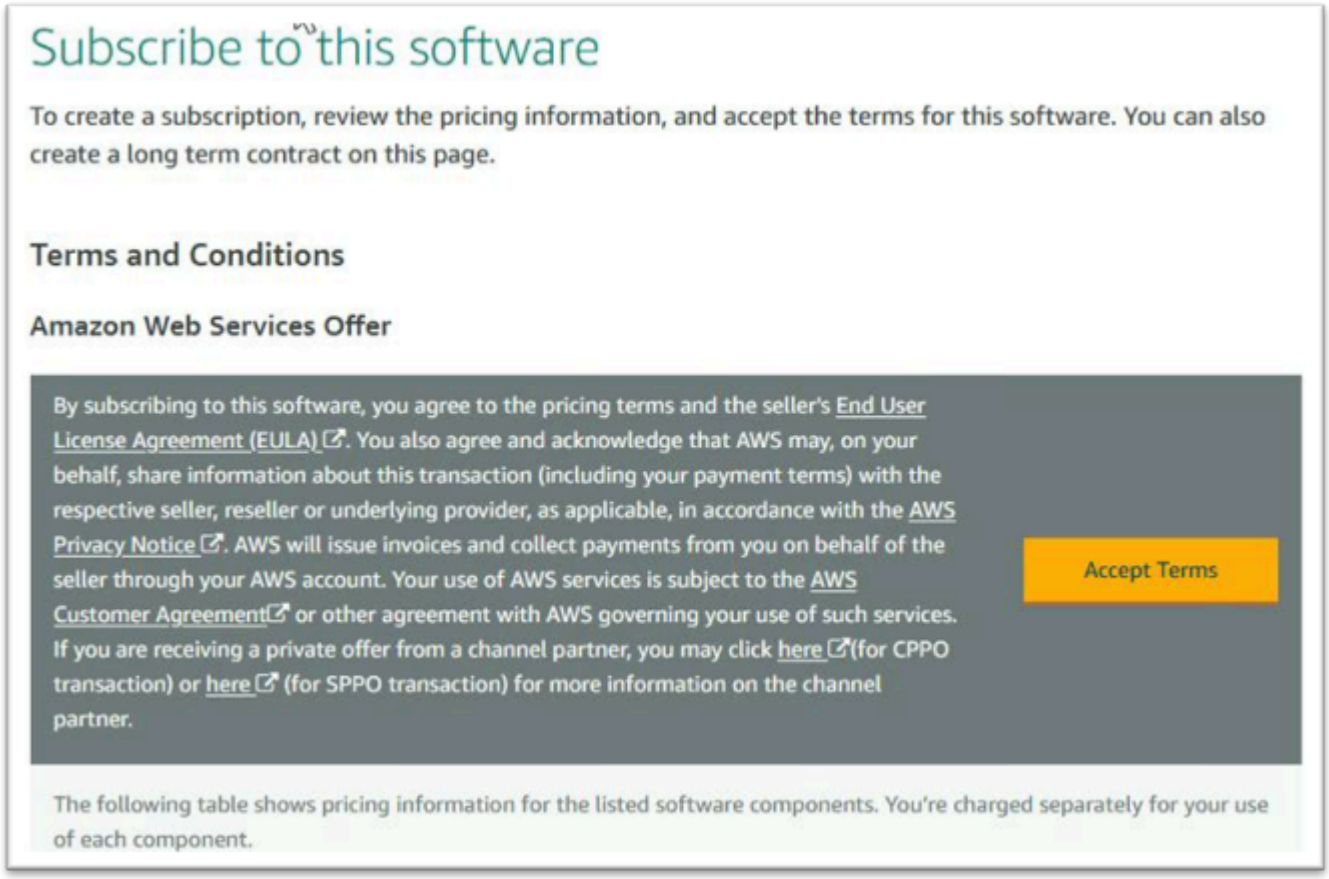
> **Note**
>
> 1. Elija únicamente un enlace para uno de los productos para los que esté autorizado a utilizar.
> 2. Asegúrese de que su cuenta esté incluida en la lista de permitidos siguiendo la Solicitud de actualización de la lista de permitidos de la cuenta página para utilizar estos enlaces.

- Servidor empresarial: https://aws.amazon.com/marketplace/ pp/prodview-g5emev63l7blc
- Servidor empresarial para Windows: pp/prodview-lwybsiyikbhc2 https://aws.amazon.com/marketplace/
- Desarrollador empresarial: pp/prodview-77qmpr42yzxwk https://aws.amazon.com/marketplace/
- Desarrollador empresarial con Visual Studio 2022: pp/prodview-m4l3lqiszo6cm https://aws.amazon.com/marketplace/
- Analizador empresarial: pp/prodview-tttheylcmcihm https://aws.amazon.com/marketplace/
- Herramientas de creación empresarial para Windows: pp/prodview-2rw35bbt6uozi https://aws.amazon.com/marketplace/

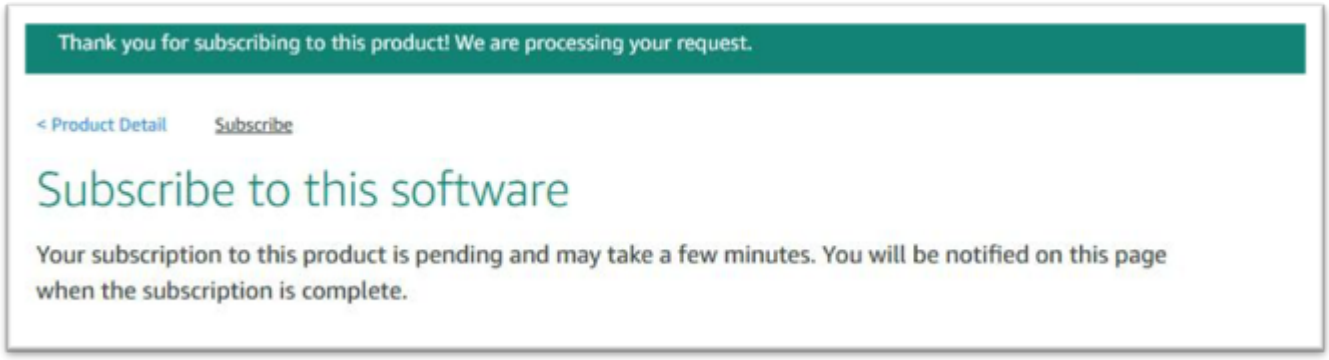
- Procedimientos almacenados empresariales: pp/prodview-zoeyqnsdsj6ha https://aws.amazon.com/marketplace/
- Procedimientos almacenados empresariales con SQL Server 2019: pp/prodview-ynfklquwubnz4 https://aws.amazon.com/marketplace/

4. Elija Continuar con la suscripción.

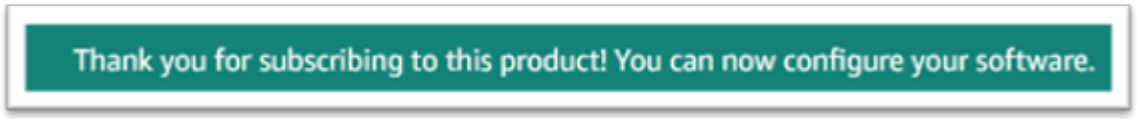
5. Si los términos y condiciones son aceptables, seleccione Aceptar términos.



6. El proceso de suscripción puede tardar unos minutos.

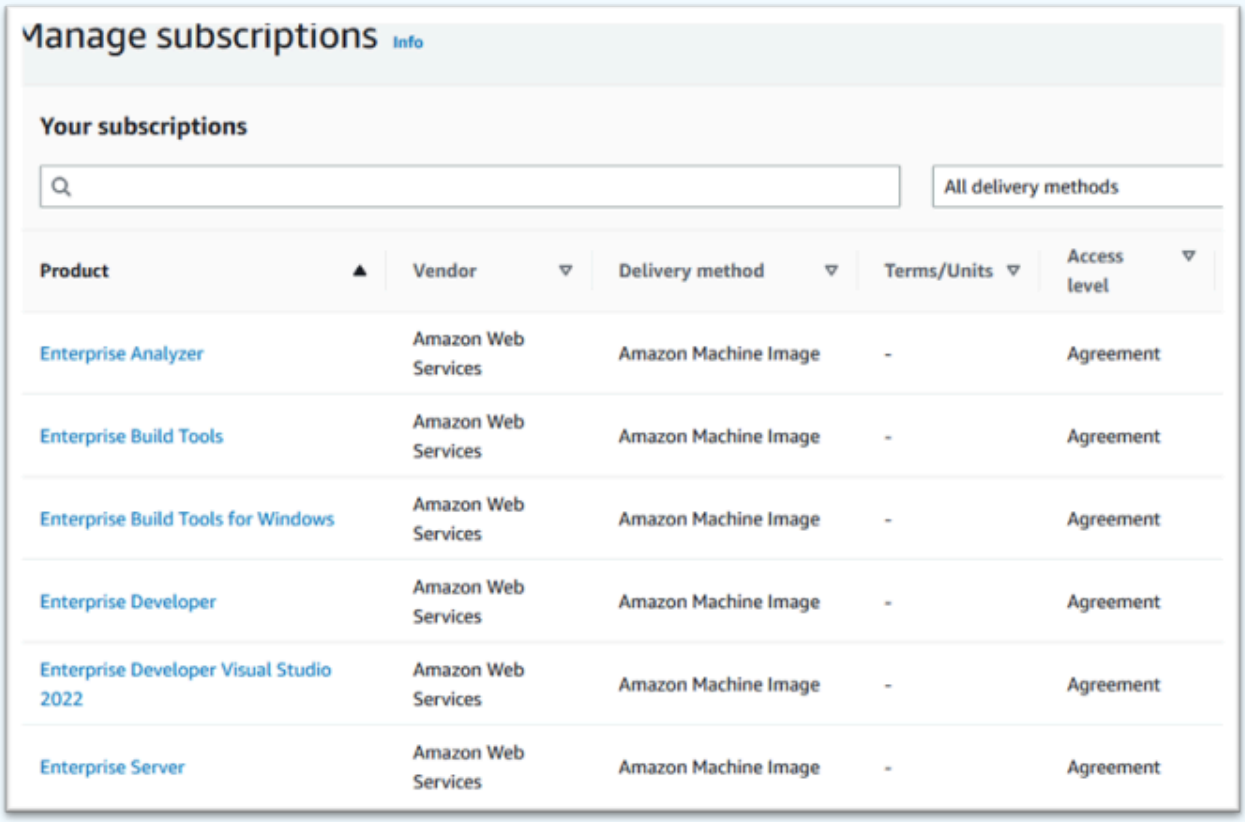


7. Cuando aparezca el mensaje de agradecimiento, copie y pegue el siguiente enlace del paso 3 para seguir añadiendo suscripciones.



8. Detener cuando la opción Administrar suscripciones muestre todos los suscriptores. AMIs

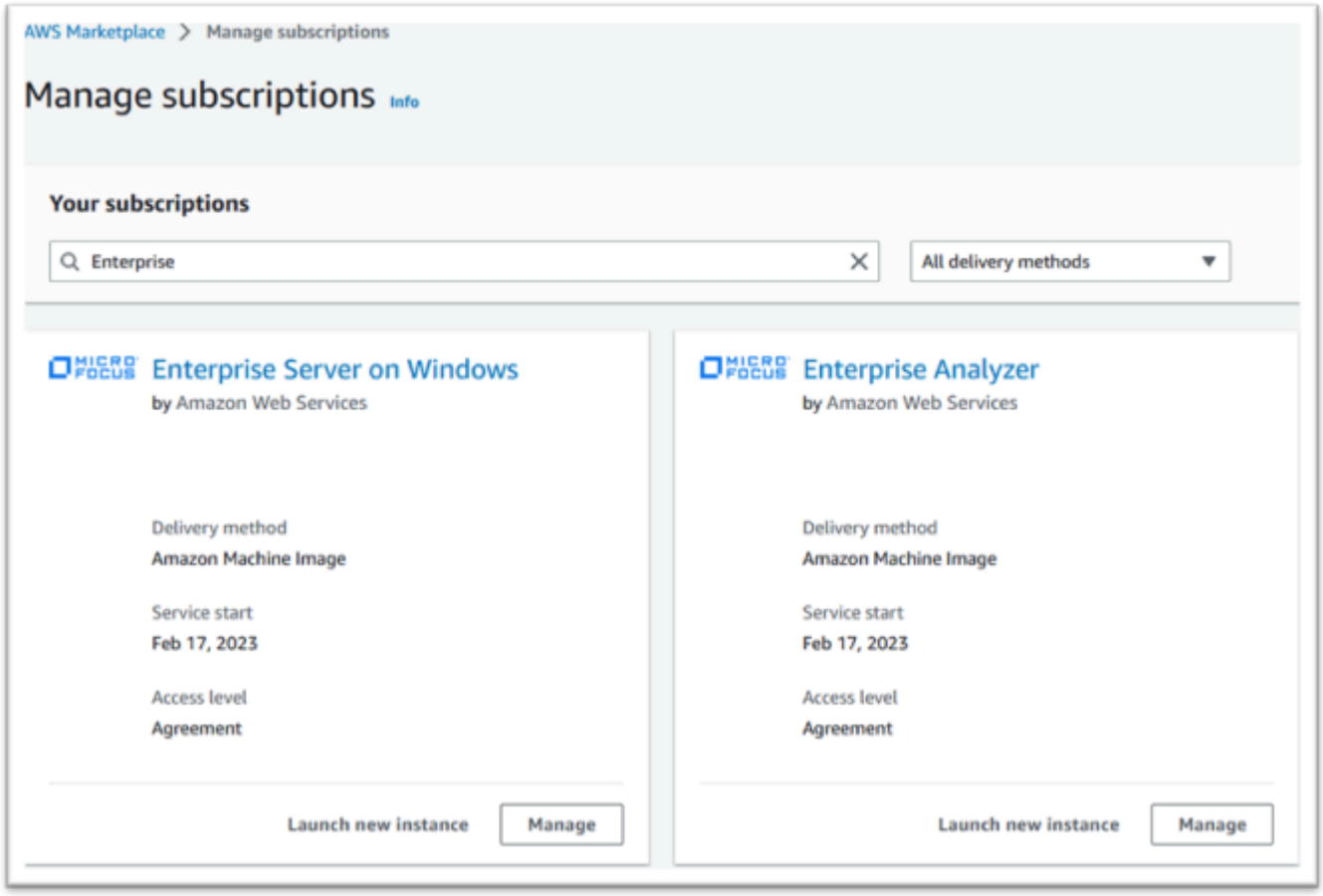
> **Note**
>
> Las preferencias del panel (icono con forma de engranaje) están configuradas para mostrar la vista en forma de tabla.



Lance una instancia de AWS Mainframe Modernization Rocket Software (anteriormente Micro Focus)

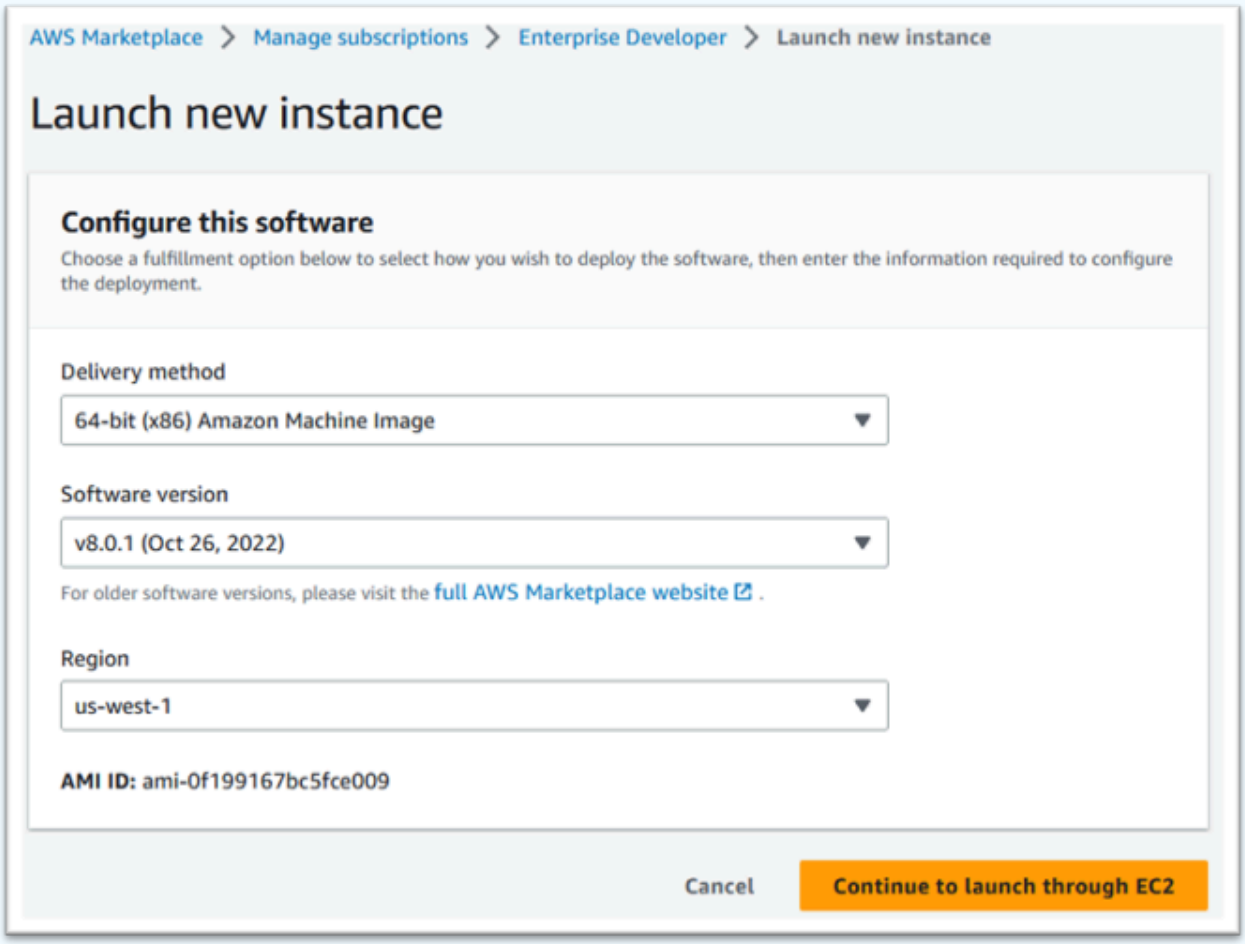
Tras crear los puntos de enlace, la política de IAM, la función de IAM y suscribirse a ellos AMIs, estará listo para lanzar una instancia de AWS Mainframe Modernization Rocket Software (Micro Focus) en la. AWS Management Console

1. Navegue hasta Suscripciones en. AWS Marketplace AWS Management Console
2. Localice la AMI que se va a lanzar y elija Lanzamiento de una instancia nueva.



3. En el cuadro de diálogo de lanzamiento de una nueva instancia, asegúrese de seleccionar la región de la lista de permitidos.
4. Presiona Continuar para iniciar EC2.

> **Note**
>
> El siguiente ejemplo muestra el lanzamiento de una AMI para desarrolladores empresariales, pero el proceso es el mismo para todas las AWS Mainframe Modernization AMIs.

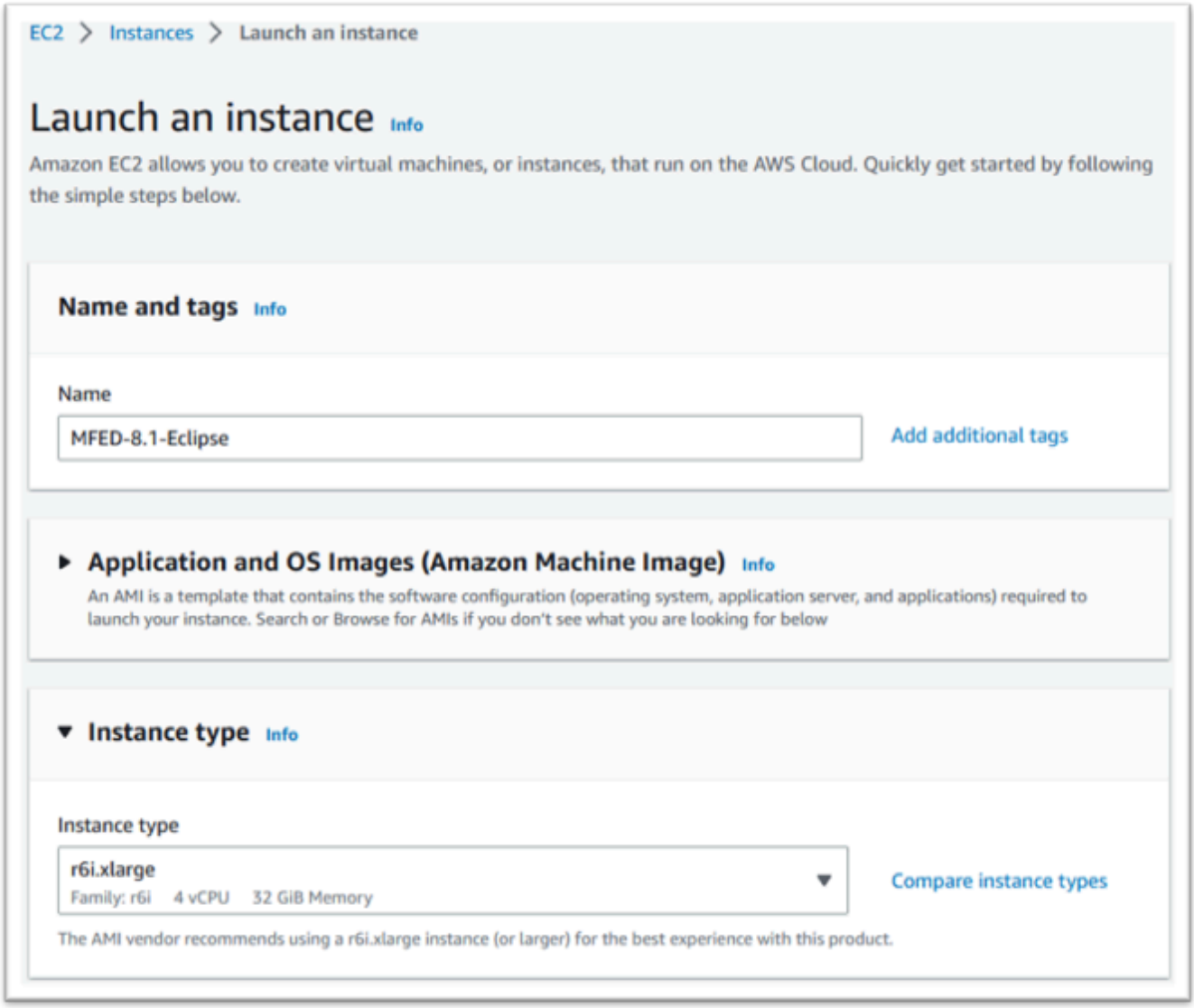


5. Escriba un nombre del servidor.
6. Elija un tipo de instancia.

   El tipo de instancia seleccionado debe determinarse en función de los requisitos de rendimiento y coste del proyecto. Se sugieren los siguientes puntos de partida:

   - Para Enterprise Analyzer, un r6i.xlarge
   - Para Enterprise Developer, un r6i.large
   - Para una instancia independiente de Enterprise Server, un r6i.xlarge
   - En el caso de Rocket Software Performance Availability Cluster (PAC) con escalabilidad horizontal, un r6i.large

   > **Note**
   >
   > La sección Application and OS Images se ha reducido para la captura de pantalla.



7. Elija o cree (y guarde) un par de claves (no se muestran).

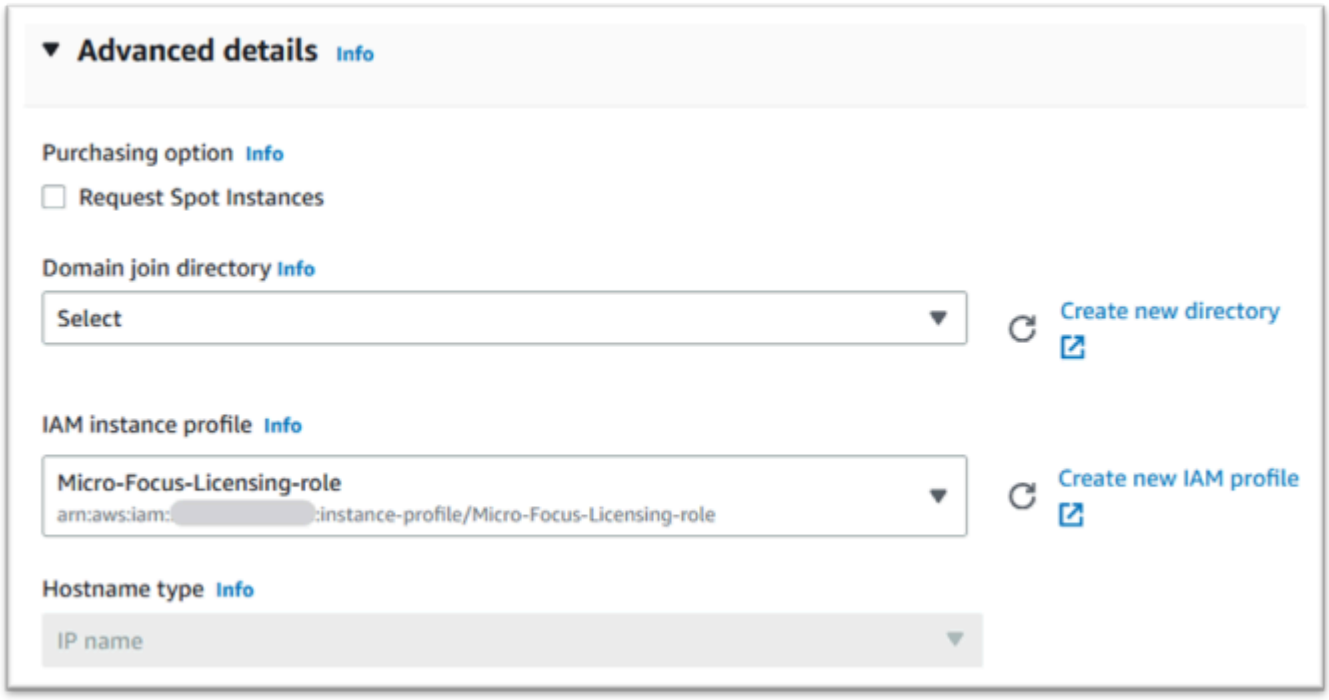
   Para obtener más información sobre los pares de claves para instancias de Linux, consulta Pares de EC2 claves e instancias de Linux de Amazon.

   Para obtener más información sobre los pares de claves para instancias de Windows, consulta Pares de EC2 claves e instancias de Windows de Amazon.

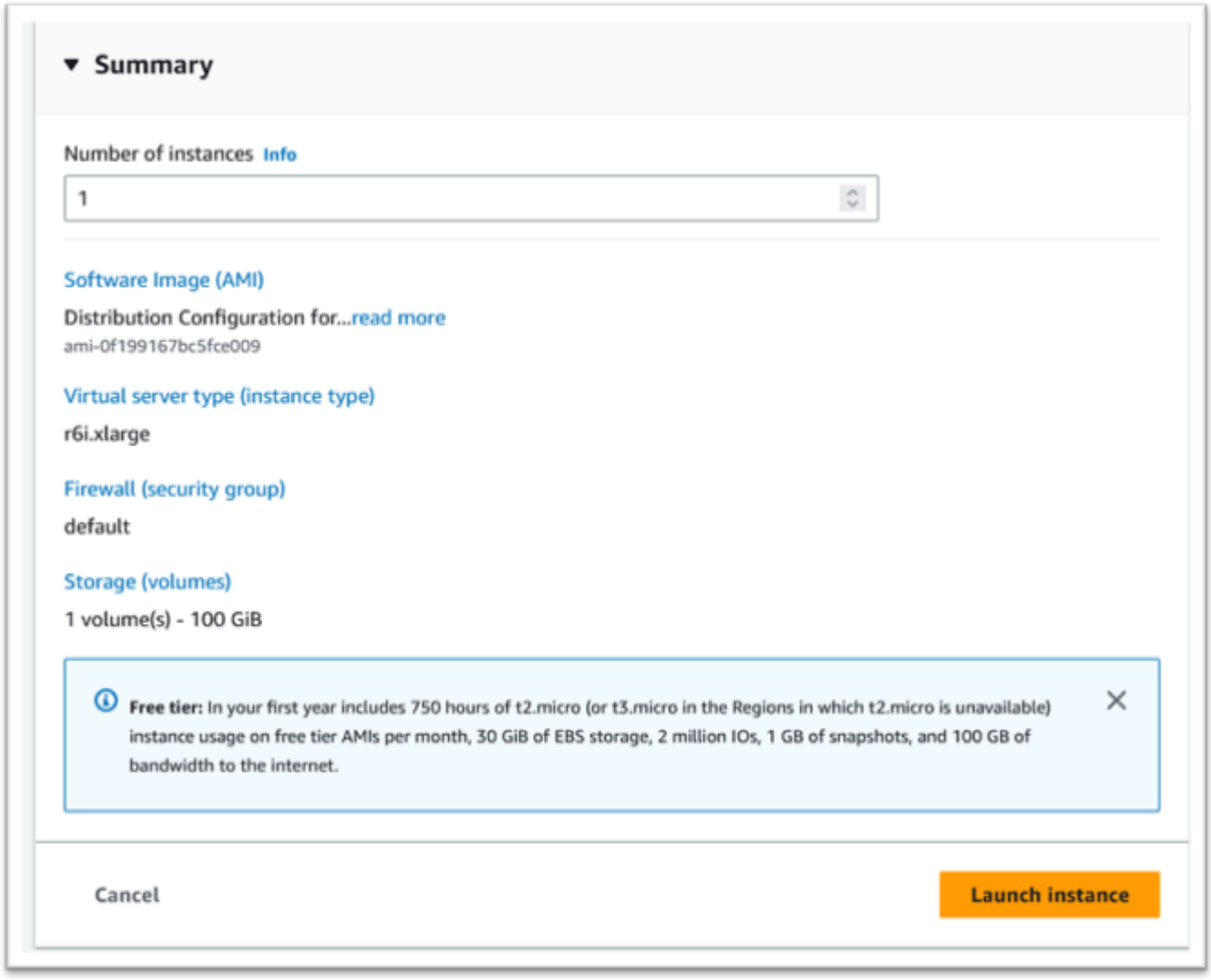
8. Edite la configuración de red y elija la VPC de la lista de permitidos y la subred adecuada.
9. Elija o cree un nuevo grupo de seguridad. Si se trata de una EC2 instancia de Enterprise Server, lo habitual es permitir que el tráfico TCP llegue a los puertos 86 y 10086 para administrar la configuración de Rocket Software.

10. Si lo desea, configure el almacenamiento de la EC2 instancia de Amazon.
11. Importante: amplíe los detalles avanzados y, en el perfil de instancia de IAM, elija la función de licencia creada anteriormente, por ejemplo, “Micro-focus-Licensing-role”.

> **Note**
>
> Si omite este paso, una vez creada la instancia, puede modificar la función de IAM desde la opción Seguridad del menú de acciones de la EC2 instancia.

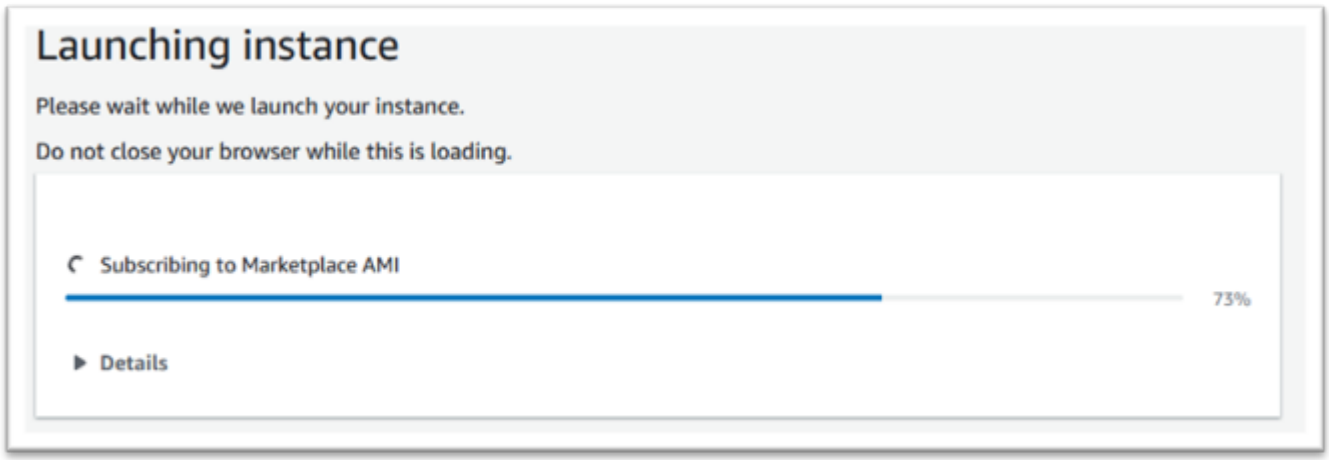


12. Revise el resumen y pulse Lanzar instancia.

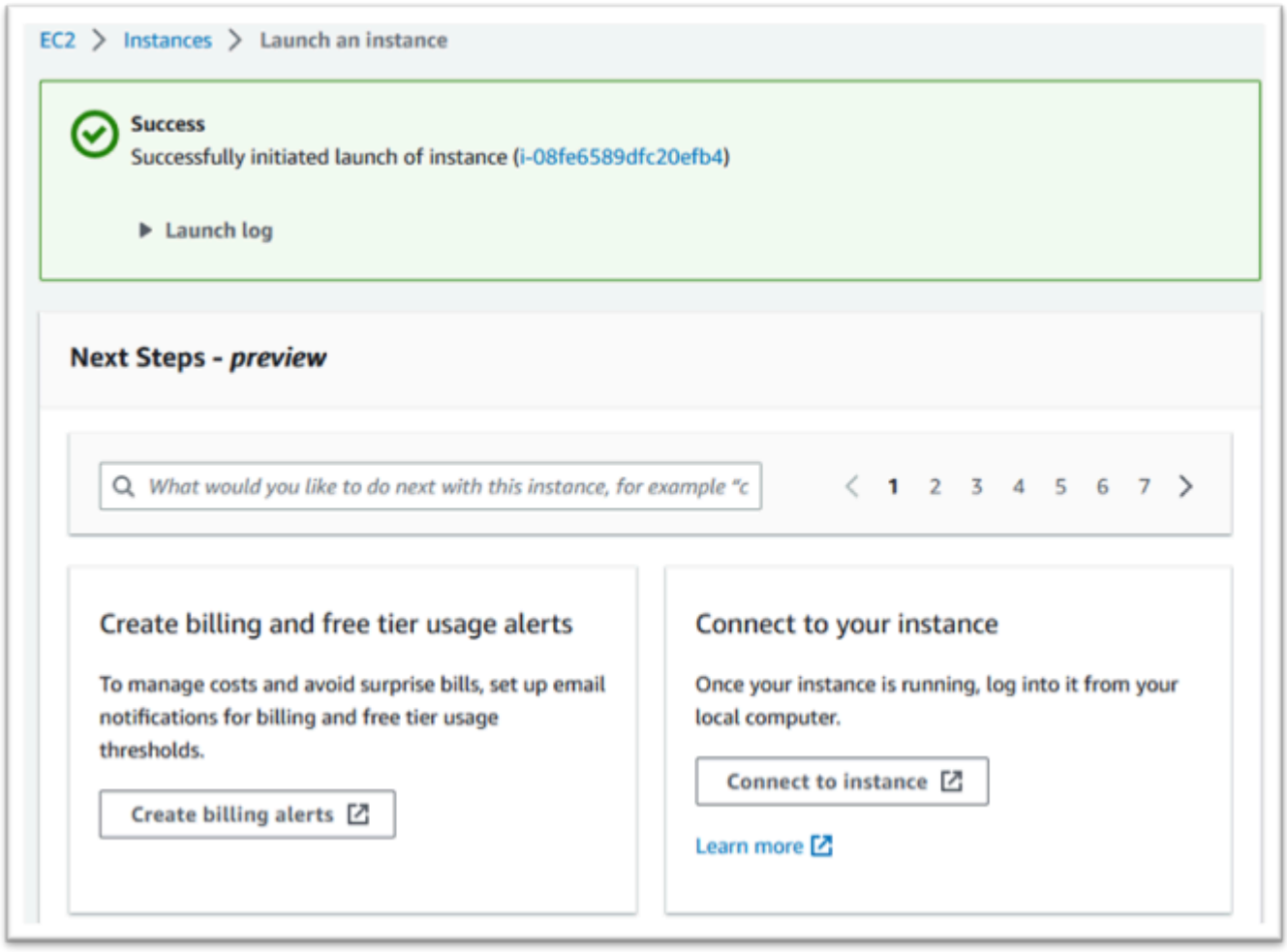


13. El lanzamiento de la instancia fallará si se elige un tipo de servidor virtual no válido.

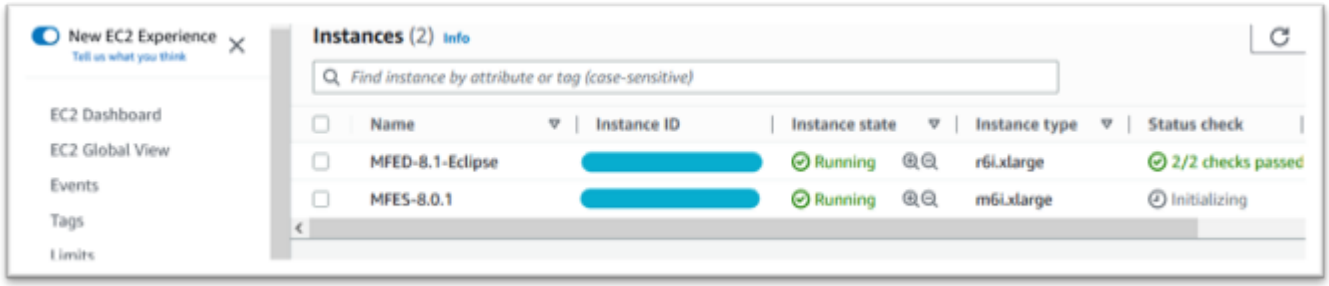
    Si esto ocurre, seleccione Editar configuración de instancias y cambie el tipo de instancia.



14. Cuando aparezca el mensaje “Correcto”, seleccione Conectarse a la instancia para obtener los detalles de la conexión.



15. Como alternativa, navegue hasta EC2. AWS Management Console
16. Elija Instancias para ver el estado de la nueva instancia.



Subred o VPC sin acceso a Internet

Realice estos cambios adicionales si la subred o la VPC no tienen acceso a Internet de salida.

El administrador de licencias requiere acceso a los siguientes servicios de AWS:

- com.amazonaws. *region*.s3
- com.amazonaws. *region*.ec2
- com.amazonaws. *region*.administrador de licencias
- com.amazonaws. *region*.sts

Los pasos anteriores definieron el com.amazonaws. *region*El servicio.s3 como punto final de puerta de enlace. Este punto de conexión necesita una entrada en la tabla de enrutamiento para cualquier subred sin acceso a Internet.

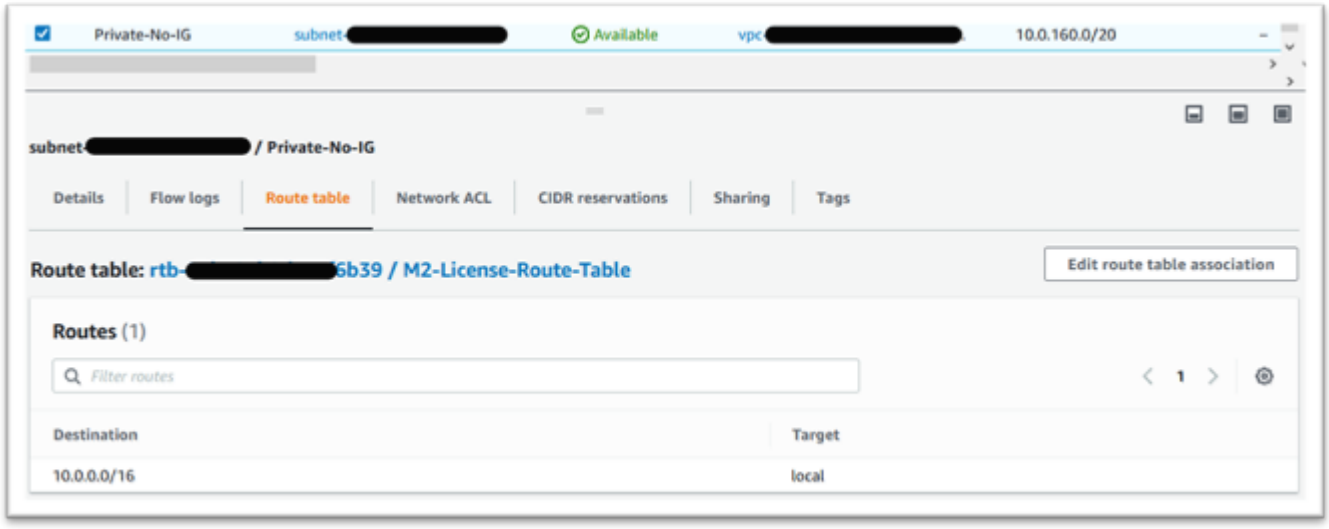
Los tres servicios adicionales se definirán como puntos de conexión de interfaz.

Temas

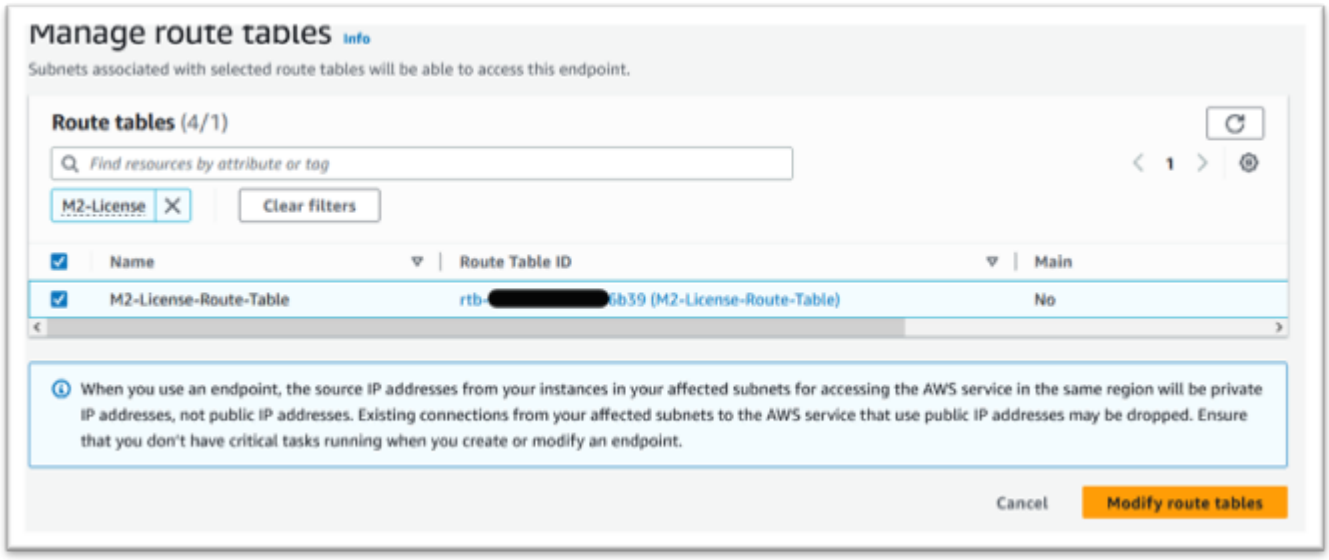
- Añadir la entrada de la tabla de enrutamiento para el punto de conexión Amazon S3
- Definición del grupo de seguridad necesario
- Creación de puntos de conexión del servicio

Añadir la entrada de la tabla de enrutamiento para el punto de conexión Amazon S3

1. Navegue hasta VPC en AWS Management Console y elija Subredes.
2. Elija la subred en la que se crearán EC2 las instancias de Amazon y elija la pestaña Tabla de rutas.
3. Anote algunos dígitos finales del identificador de la tabla de enrutamiento. Por ejemplo, 6b39 de la imagen de abajo.



4. En el panel de navegación, elija Puntos de conexión.
5. Elija el punto de conexión creado anteriormente y, a continuación, Administrar tablas de enrutamiento en la pestaña Tablas de enrutamiento del punto de conexión o en el menú desplegable Acciones.
6. Elija la tabla de enrutamiento con los dígitos identificados anteriormente y pulse Modificar tablas de enrutamiento.

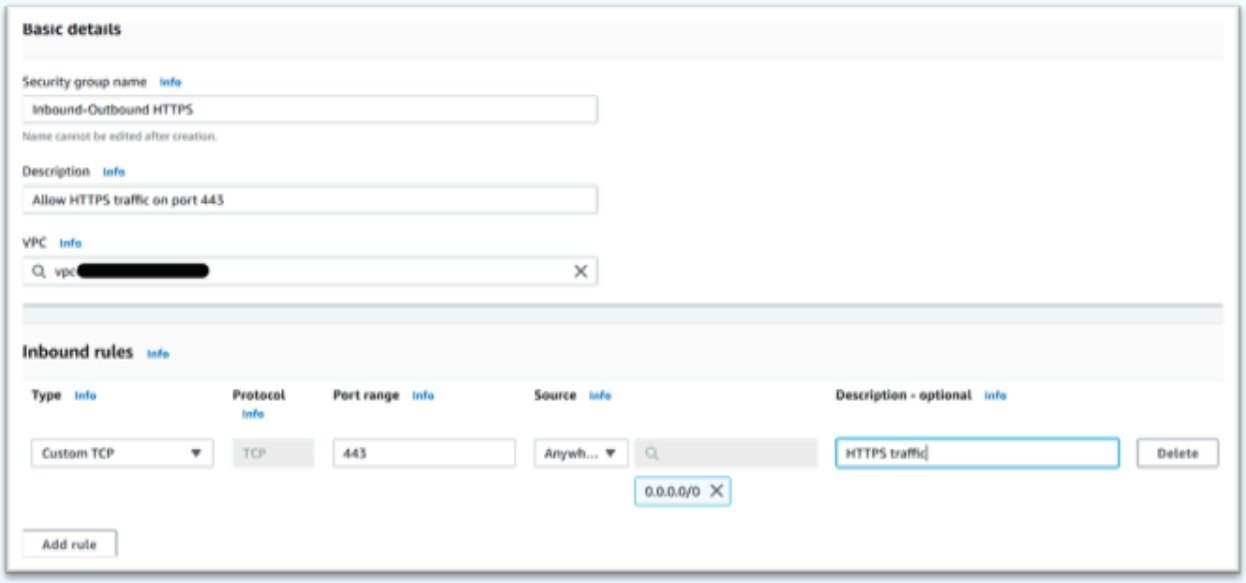


Definición del grupo de seguridad necesario

Los servicios Amazon EC2 y License Manager se comunican a través de HTTPS a través del puerto 443. AWS STS Esta comunicación es bidireccional y requiere reglas de entrada y salida para permitir que la instancia se comunique con los servicios.

1. Vaya a Amazon VPC en la AWS Management Console.
2. En el panel de navegación, en Grupos de seguridad, elija Crear grupo de seguridad.
3. Introduzca el nombre y la descripción de un grupo de seguridad, por ejemplo, “Inbound-Outbound HTTPS”.
4. Pulse la X en el área de selección de VPC para eliminar la VPC predeterminada y elija la VPC que contenga el punto de conexión de S3.
5. Agregue una regla de entrada que permita el tráfico TCP en el puerto 443 desde cualquier lugar.

> **Note**
>
> Las reglas de entrada (y de salida) se pueden restringir aún más limitando el origen. Para obtener más información, consulte Controle el tráfico a sus AWS recursos mediante grupos de seguridad en la Guía del usuario de Amazon VPC.



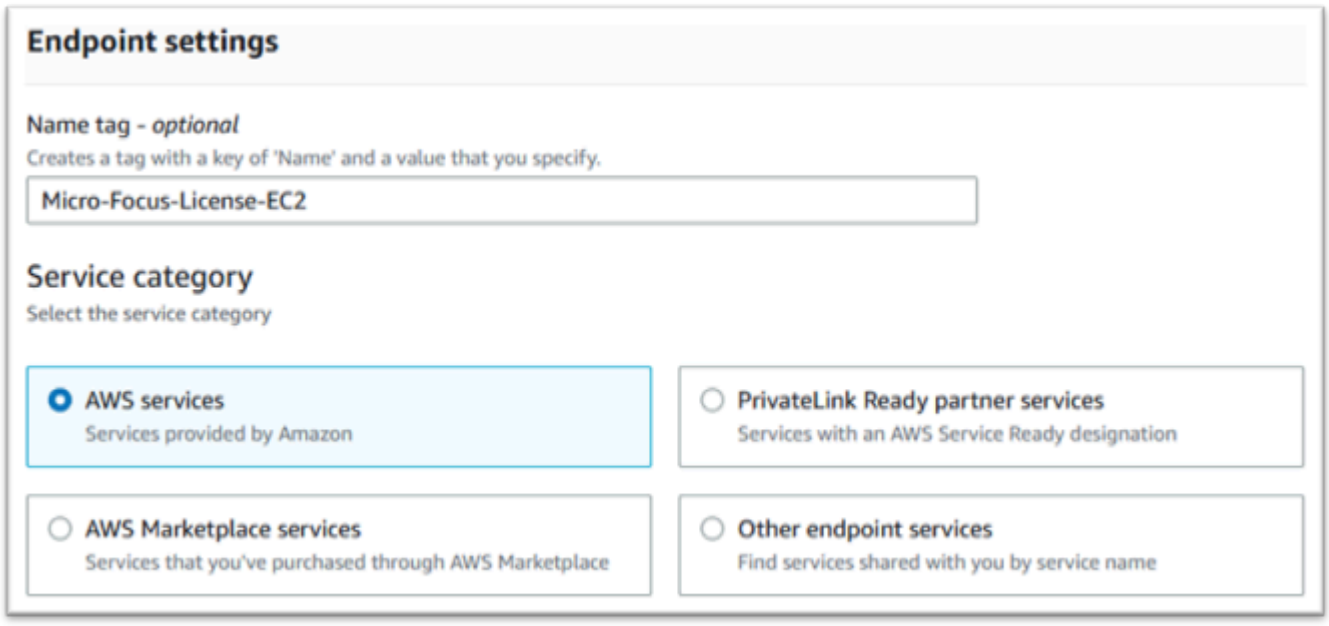
6. Pulse Crear grupo de seguridad.

Creación de puntos de conexión del servicio

Repita este proceso tres veces, una para cada servicio.

1. Diríjase a Amazon VPC en AWS Management Console y seleccione Endpoints.
2. Pulse Crear punto de conexión.

3. Introduzca un nombre, por ejemplo, «Micro-Focus-License-EC2», «Micro-Focus-License-STS» o «Micro-Focus-License-Manager».
4. Elija Servicios de AWS en Categoría de servicio.



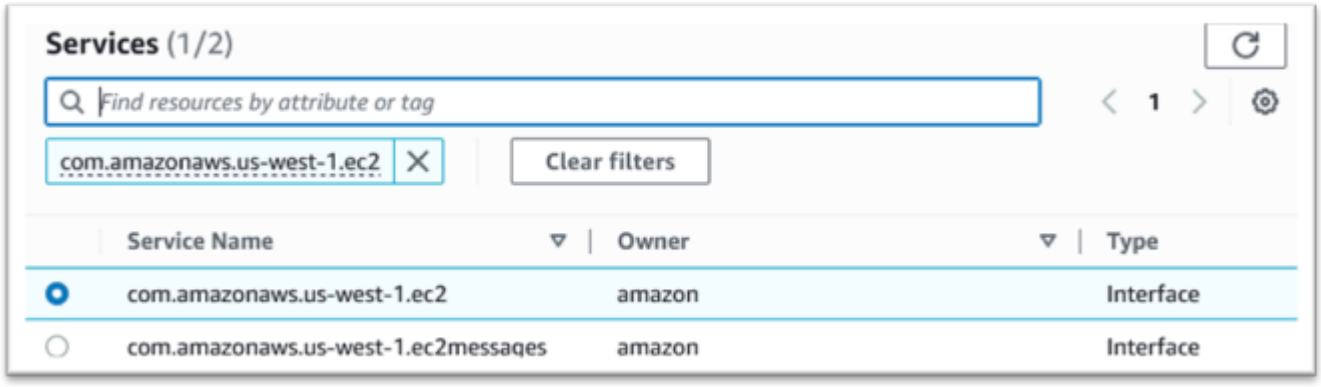
5. En Servicios, busque el servicio de interfaz correspondiente, que sea uno de los siguientes:

   - «com.amazonaws. *region*.ec2»
   - «com.amazonaws. *region*.sts»
   - «com.amazonaws. *region*.administrador de licencias»

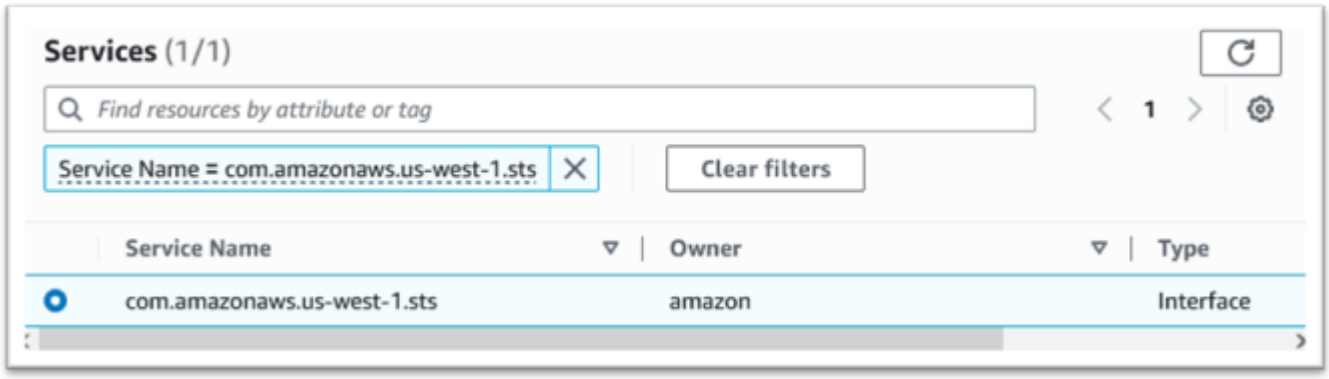
   Por ejemplo:

   - “com.amazonaws.us-west-1.ec2”
   - “com.amazonaws.us-west-1.sts”
   - “com.amazonaws.us-west-1.license-manager”
6. Elija el servicio de interfaz correspondiente.

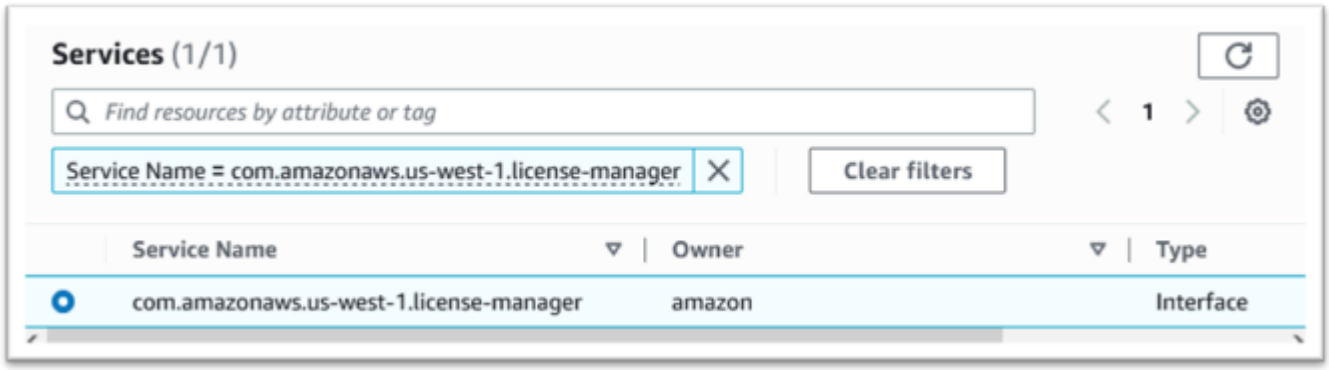
   com.amazonaws. ***region***.ec2:



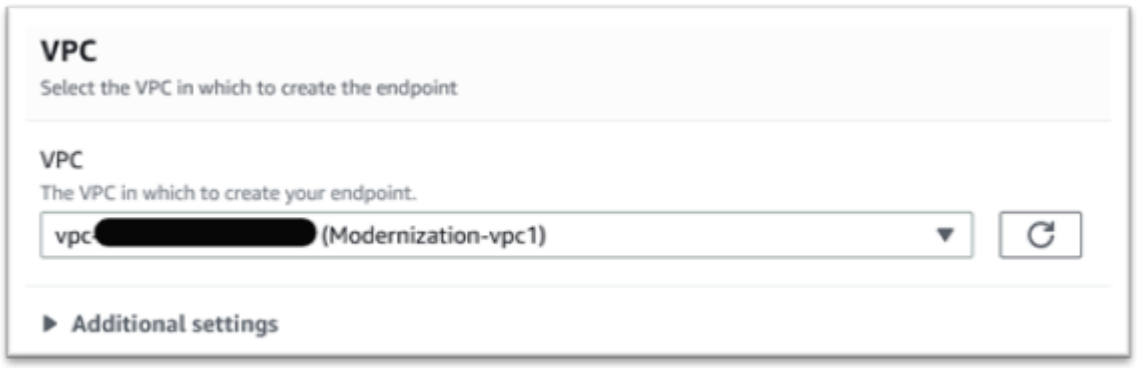
com.amazonaws. ***region***.sts:



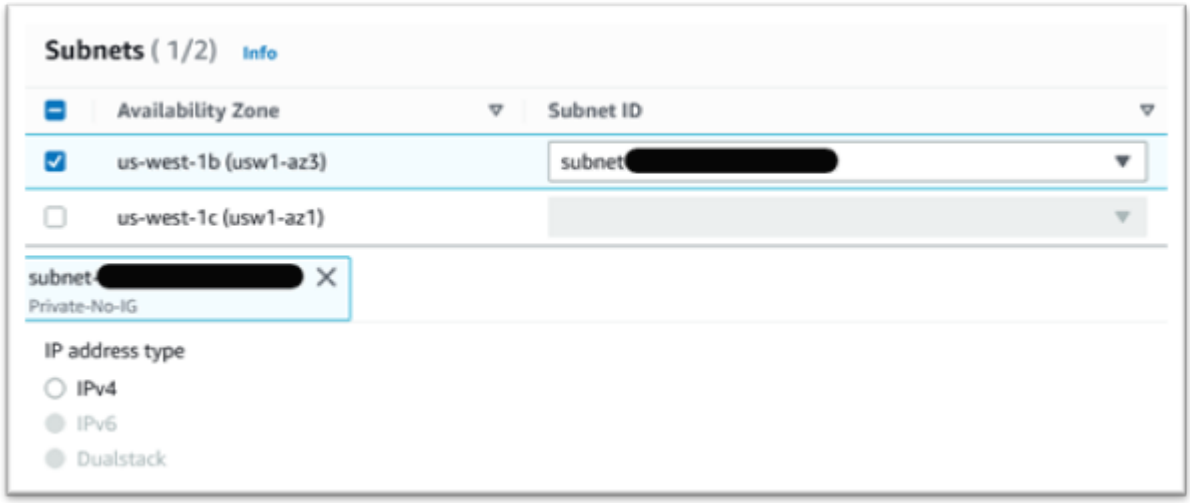
com.amazonaws. ***region***.administrador de licencias:



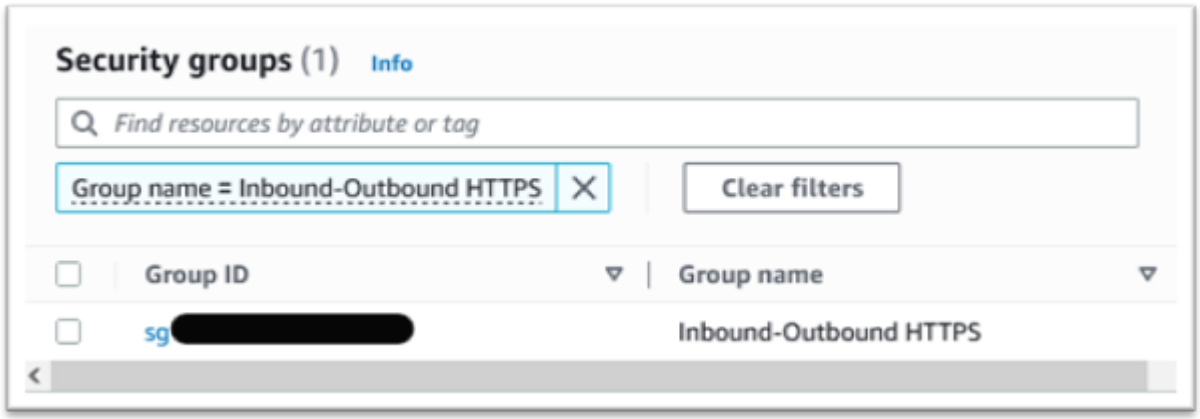
7. En VPC, elija la VPC para su instancia.



8. Elija la Zona de disponibilidad y las Subredes de la VPC.



9. Elija el Grupo de seguridad creado anteriormente.



10. En Política, seleccione Full Access.

11. Elija Crear punto de conexión.
12. Repita este proceso para el resto de las interfaces.

Configure la automatización para las sesiones de streaming de Rocket Enterprise Analyzer (anteriormente Micro Focus) y Rocket Enterprise Developer

Puede ejecutar automáticamente un script al inicio y al final de la sesión para permitir una automatización específica del contexto de su cliente. Para obtener más información sobre esta función AppStream 2.0, consulte Uso de scripts de sesión para gestionar la experiencia de streaming de los usuarios de la AppStream versión 2.0 en la Guía de administración de Amazon AppStream 2.0.

Esta característica requiere que tenga al menos las siguientes versiones de las imágenes de Enterprise Analyzer y Enterprise Developer:

- `m2-enterprise-analyzer-v8.0.4.R1`
- `m2-enterprise-developer-v8.0.4.R1`

Temas

- Configuración de la automatización al inicio de la sesión
- Configuración de la automatización al final de la sesión

Configuración de la automatización al inicio de la sesión

Si desea ejecutar un script de automatización cuando los usuarios se conecten a la AppStream versión 2.0, cree el suyo y asígnele un nombre`m2-user-setup.cmd`. Guarde el script en la carpeta principal de la AppStream versión 2.0 para el usuario. Las imágenes AppStream 2.0 que proporciona AWS Mainframe Modernization buscan un script con ese nombre en esa ubicación y lo ejecutan si existe.

> **Note**
>
> La duración del guion no puede superar el límite establecido en AppStream 2.0, que actualmente es de 60 segundos. Para obtener más información, consulte Ejecutar scripts antes de que comiencen las sesiones de streaming en la Guía de administración de Amazon AppStream 2.0.

Configuración de la automatización al final de la sesión

Si desea ejecutar un script de automatización cuando los usuarios se desconecten de la AppStream versión 2.0, cree el suyo y asígnele un nombre`m2-user-teardown.cmd`. Guarde el script en la carpeta principal de la AppStream versión 2.0 para el usuario. Las imágenes AppStream 2.0 que proporciona AWS Mainframe Modernization buscan un script con ese nombre en esa ubicación y lo ejecutan si existe.

> **Note**
>
> La duración del guion no puede superar el límite establecido en AppStream 2.0, que actualmente es de 60 segundos. Para obtener más información, consulte Ejecutar scripts tras finalizar las sesiones de streaming en la Guía de administración de Amazon AppStream 2.0.

Vea los conjuntos de datos como tablas y columnas en Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)

Puede acceder a los conjuntos de datos de mainframe que se implementan en AWS Mainframe Modernization mediante el entorno de ejecución de Rocket Software (anteriormente Micro Focus).

Puede ver los conjuntos de datos migrados como tablas y columnas desde una instancia de Rocket Enterprise Developer. Ver los conjuntos de datos de esta manera le permite:

- Realizar operaciones `SQL SELECT` en los archivos de datos migrados.
- Exponer los datos fuera de la aplicación de unidad central migrada sin cambiar la aplicación.
- Filtrar fácilmente los datos y guardarlos como CSV u otros formatos de archivo.

> **Note**
>
> Los pasos 1 y 2 son actividades que se realizan una sola vez. Repita los pasos 3 y 4 para en conjunto de datos para crear las vistas de base de datos.

Temas

- Requisitos previos
- Paso 1: Configurar la conexión ODBC al almacén de datos de Rocket Software (base de datos Amazon RDS)
- Paso 2: Crear el archivo MFDBFH.cfg
- Paso 3: Crear un archivo de estructura (STR) para el diseño del cuaderno
- Paso 4: Crear una vista de base de datos con el archivo de estructura (STR)
- Paso 5: Vea los conjuntos de datos de Rocket Software (anteriormente Micro Focus) en forma de tablas y columnas

Requisitos previos

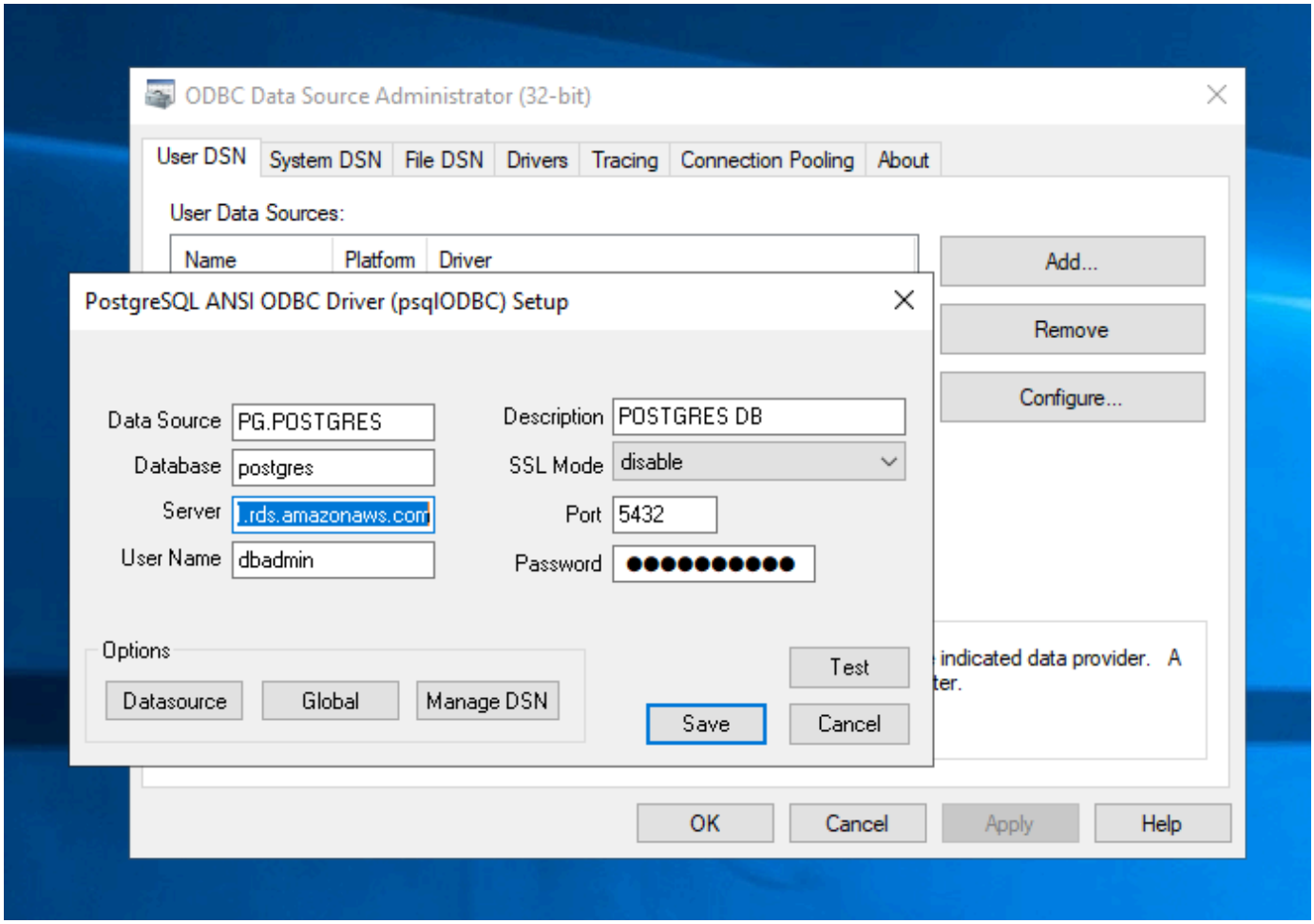
- Debe tener acceso a Rocket Enterprise Developer Desktop a través de la AppStream versión 2.0.
- Debe tener una aplicación implementada y ejecutada en el marco de AWS Mainframe Modernization mediante el motor de ejecución de Rocket Software.
- Va a almacenar los datos de la aplicación en la edición de PostgreSQL compatible con Aurora.

Paso 1: Configurar la conexión ODBC al almacén de datos de Rocket Software (base de datos Amazon RDS)

En este paso, configurará una conexión ODBC a la base de datos que contiene los datos que desee ver en forma de tablas y columnas. Solo tendrá que hacerlo una vez.

1. Inicie sesión en Rocket Enterprise Developer Desktop mediante AppStream la URL de streaming 2.0.
2. Abra el Administrador de orígenes de datos ODBC, elija DSN de usuario y, a continuación, seleccione Agregar.
3. En Crear nuevo origen de datos, elija PostgreSQL ANSI y, a continuación, Finalizar.
4. Cree un origen de datos para `PG.POSTGRES` proporcionando la información de base de datos necesaria, de la siguiente manera:

```
Data Source : PG.POSTGRES
Database    : postgres
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
User Name   : user_name
Password    : user_password
```



5. Elija Probar para asegurarse de que la conexión funciona. Si la prueba se realiza correctamente, aparecerá el mensaje `Connection successful`.

   Si la prueba no tiene éxito, lea la siguiente información.

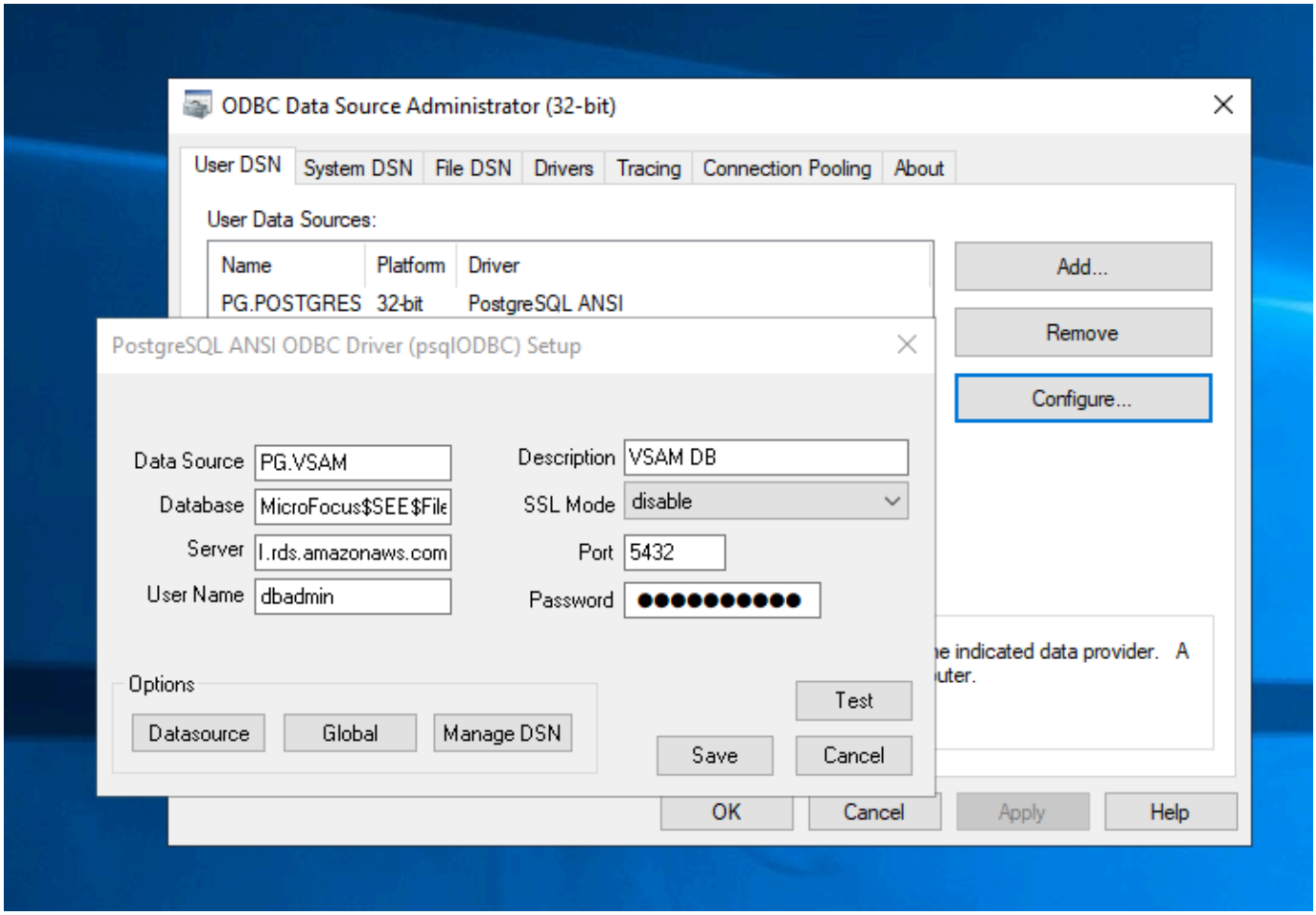
   - Solución de problemas de Amazon RDS
   - ¿Cómo resuelvo los problemas al conectarme a mi instancia de base de datos de Amazon RDS?

6. Guarde el origen de datos.

7. Cree un origen de datos para `PG.VSAM`, pruebe la conexión y guarde el origen de datos. Proporcione la siguiente información de la base de datos:

```
Data Source : PG.VSAM
Database    : MicroFocus$SEE$Files$VSAM
Server      : rds_endpoint.rds.amazonaws.com
Port        : 5432
```

```
User Name    : user_name
Password     : user_password
```



Paso 2: Crear el archivo MFDBFH.cfg

En este paso, creará un archivo de configuración que describa el almacén de datos de Micro Focus. Este paso de configuración solo se tiene que realizar una vez.

1. En su carpeta de inicio, por ejemplo, en `D:\PhotonUser\My Files\Home Folder\MFED \cfg\MFDBFH.cfg`, cree el archivo MFDBFH.cfg con el siguiente contenido.

```
<datastores>
       <server name="ESPACDatabase" type="postgresql" access="odbc">
        <dsn name="PG.POSTGRES" type="database" dbname="postgres"/>
        <dsn name="PG.VSAM" type="datastore" dsname="VSAM"/>
       </server>
```

```
      </datastores>
```

2. Compruebe la configuración del MFDBFH ejecutando los siguientes comandos para consultar el almacén de datos de Micro Focus:

```
*###*
*## Test the connection by running the following commands*
*###*

set MFDBFH_CONFIG="D:\PhotonUser\My Files\Home Folder\MFED\cfg\MFDBFH.cfg"

dbfhdeploy list sql://ESPACDatabase/VSAM?folder=/DATA
```

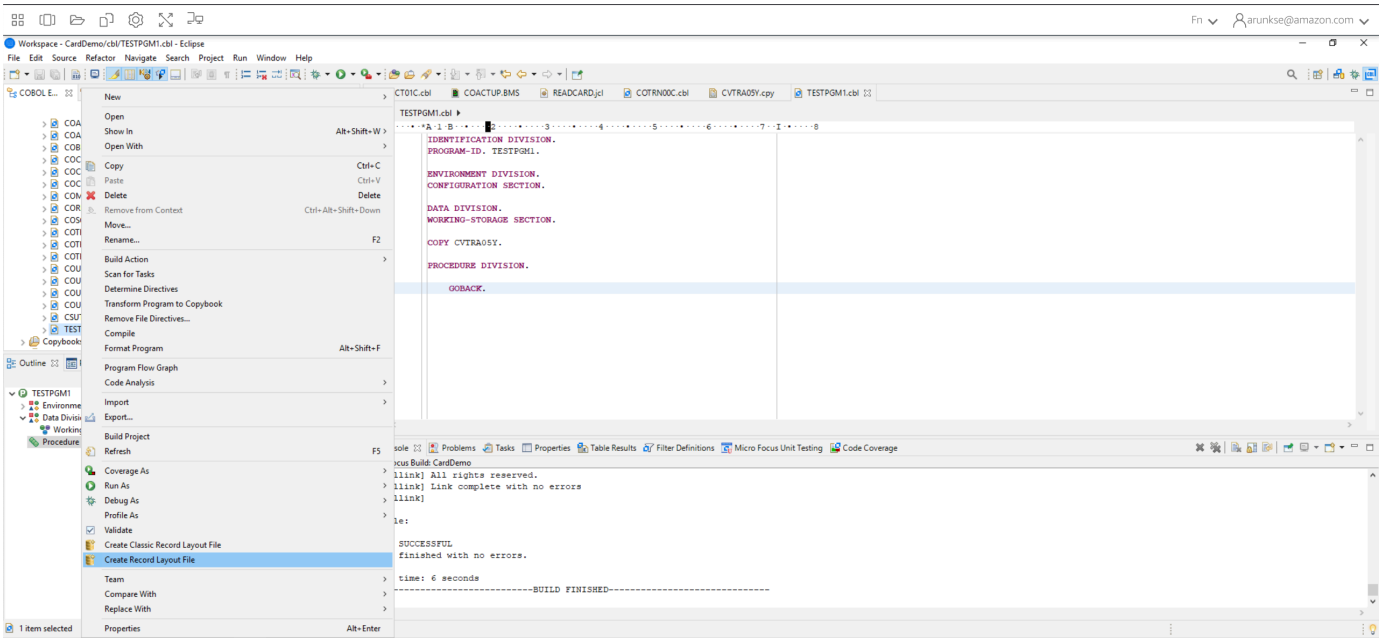
Paso 3: Crear un archivo de estructura (STR) para el diseño del cuaderno

En este paso, creará un archivo de estructura para el diseño del cuaderno de texto que utilizará más adelante para crear vistas de bases de datos a partir de los conjuntos de datos.

1. Compile el programa asociado al cuaderno. Si ningún programa utiliza el cuaderno, cree y compile un programa simple como el siguiente con una instrucción COPY para el cuaderno.

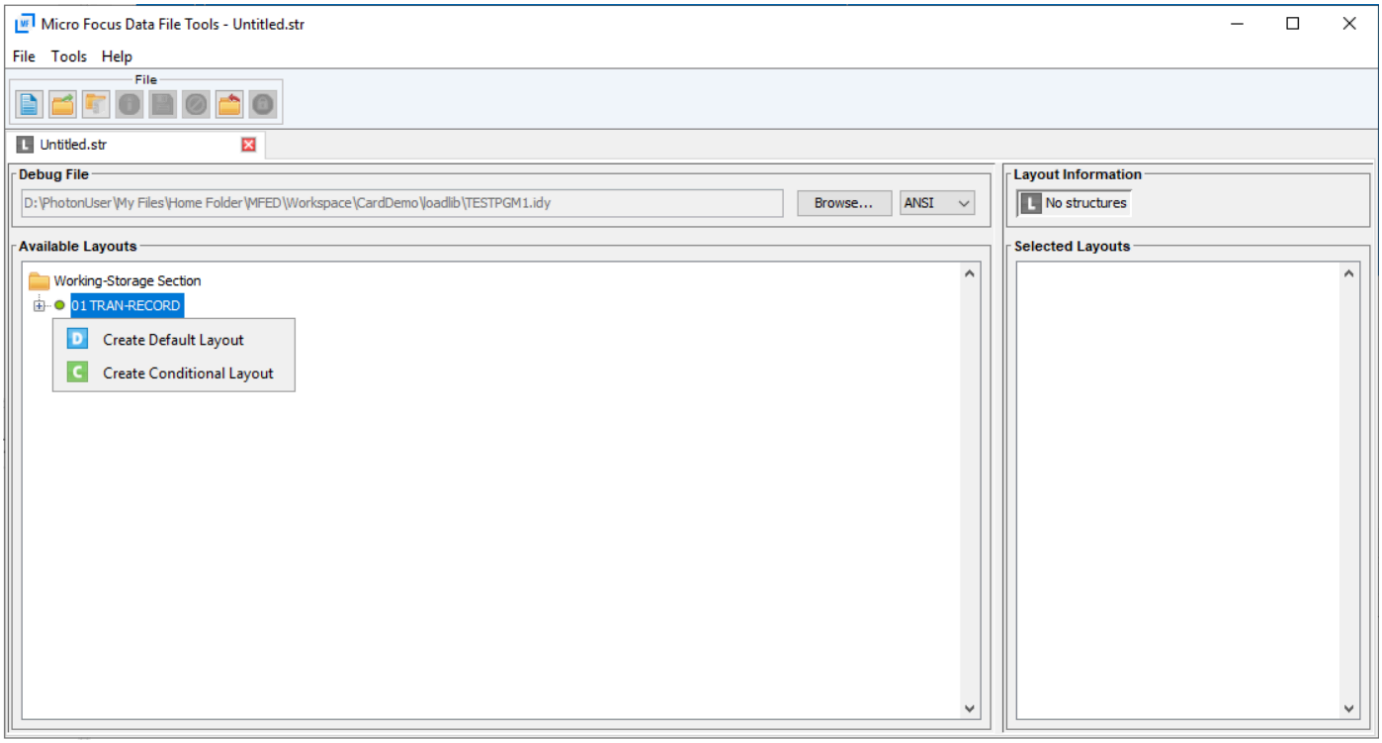
```
IDENTIFICATION DIVISION.
      PROGRAM-ID. TESTPGM1.

      ENVIRONMENT DIVISION.
      CONFIGURATION SECTION.

      DATA DIVISION.
      WORKING-STORAGE SECTION.

      COPY CVTRA05Y.

      PROCEDURE DIVISION.

      GOBACK.
```

2. Después de una compilación correcta, haga clic con el botón derecho en el programa y elija Crear archivo de diseño de registro. Se abrirán las herramientas de archivos de datos de Micro Focus que utilizan el archivo .idy generado durante la compilación.



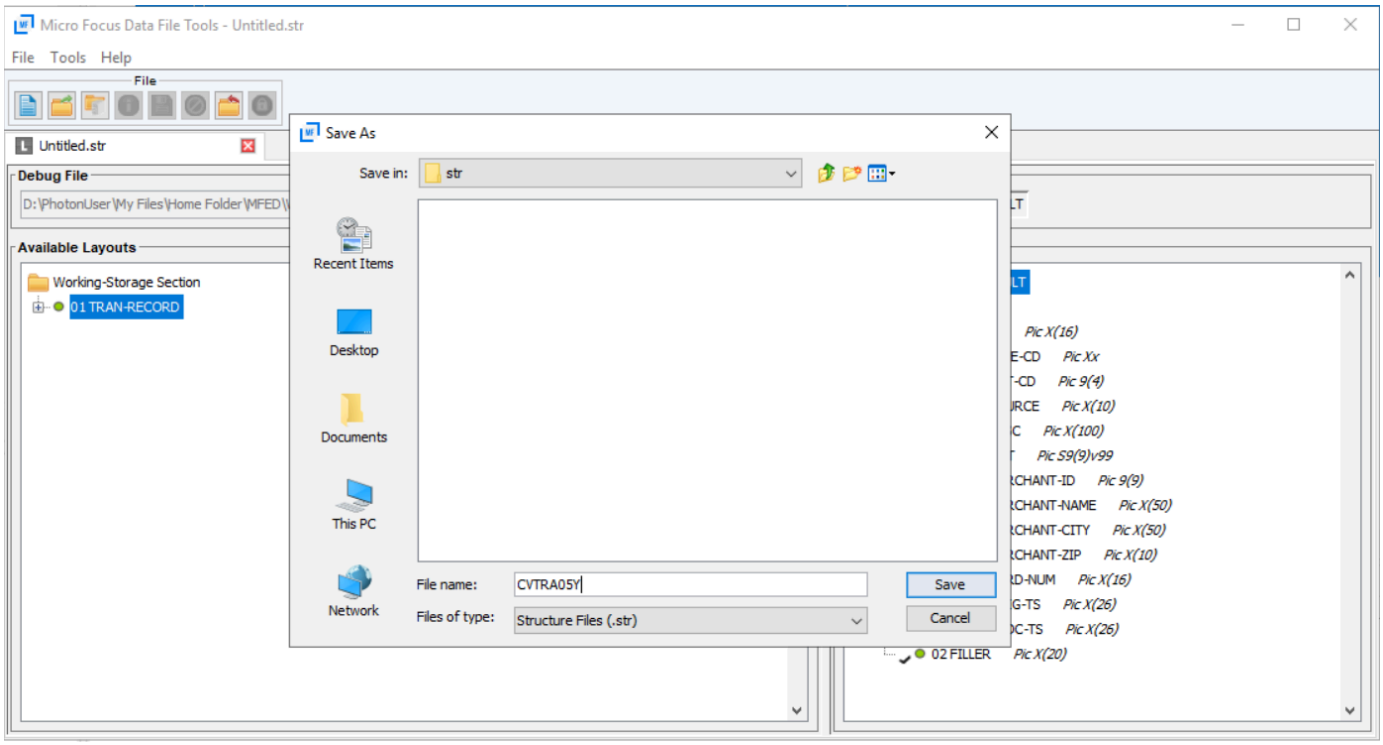
3. Haga clic con el botón derecho en la estructura del registro y elija Crear diseño predeterminado (estructura única) o Crear diseño condicional (estructura múltiple) según el diseño.

   Para obtener más información, consulte Creating Structure Files and Layouts en la documentación de Micro Focus.



4. Tras crear el diseño, seleccione Archivo en el menú y, a continuación, elija Guardar como. Busque y guarde el archivo en su carpeta principal con el mismo nombre de archivo que su

cuaderno. Puede optar por crear una carpeta llamada `str` y guardar allí todos sus archivos de estructura.



Paso 4: Crear una vista de base de datos con el archivo de estructura (STR)

En este paso, utilizará el archivo de estructura creado anteriormente para crear una vista de base de datos para un conjunto de datos.

- Utilice el comando `dbfhview` para crear una vista de base de datos para un conjunto de datos que ya se encuentra en el almacén de datos de Micro Focus, como se muestra en el siguiente ejemplo.

```
##
      ## The below command creates database view for VSAM file
 AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS
      ## using the STR file CVTRA05Y.str
      ##

      dbfhview -create -struct:"D:\PhotonUser\My Files\Home Folder\MFED\str
\CVTRA05Y.str" -name:V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT -file:sql://
ESPACDatabase/VSAM/AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT?folder=/DATA
```

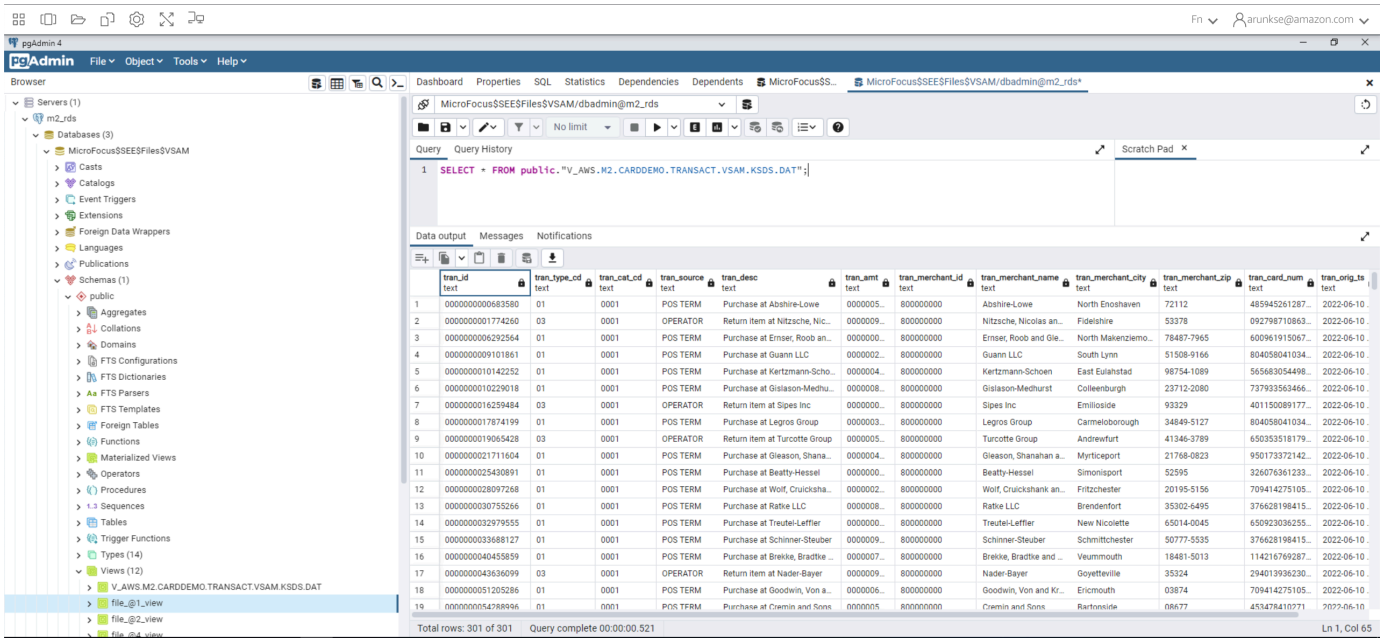
```
      ##
      ## Output:
      ##

      Micro Focus Database File Handler - View Generation Tool Version 8.0.00
      Copyright (C) 1984-2022 Micro Focus. All rights reserved.

      VGN0017I Using structure definition 'TRAN-RECORD-DEFAULT'
      VGN0022I View 'V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT' installed in
datastore 'sql://espacdatabase/VSAM'
      VGN0002I The operation completed successfully
```

Paso 5: Vea los conjuntos de datos de Rocket Software (anteriormente Micro Focus) en forma de tablas y columnas

En este paso, se conectará a la base de datos utilizando pgAdmin para poder ejecutar consultas y ver los conjuntos de datos, como tablas y columnas.

- Conéctese a la base de datos `MicroFocus$SEE$Files$VSAM` mediante pgAdmin y consulte la vista de base de datos que ha creado en el paso 4.

```
SELECT * FROM public."V_AWS.M2.CARDDEMO.TRANSACT.VSAM.KSDS.DAT";
```



Edite conjuntos de datos con las herramientas de archivos de datos de Rocket Software (anteriormente Micro Focus) en Enterprise Developer

Puede ver y editar conjuntos de datos en AWS Mainframe Modernization utilizando el motor de ejecución de Rocket Software para cualquier conjunto de datos migrado.
Los pasos de este documento lo guiarán a través del proceso de acceso a los conjuntos de datos mediante las herramientas de archivos de datos.
Esto le permite ver y editar los conjuntos de datos migrados según sea necesario.

Temas

- Requisitos previos
- Inicie las herramientas de archivos de datos de Rocket Software (anteriormente Micro Focus)
- Edite los conjuntos de datos de VSAM almacenados en la base de datos MFDBFH
- Edite conjuntos de datos que no sean de VSAM almacenados en la base de datos MFDBFH
- Edite conjuntos de datos VSAM y no VSAM almacenados en el sistema de archivos (EFS/FSx)

Requisitos previos

Antes de empezar, debe tener implementada una aplicación con los conjuntos de datos importado mediante el servicio de modernización de AWS mainframe mediante el motor de software Rocket.

Para continuar con la edición de los conjuntos de datos, debe completar el paso 1, paso 2, y (opcionalmente) el paso 3 de la the section called “Ver conjuntos de datos como tablas en Enterprise Developer” página para configurar la conexión ODBC y el almacén de datos de Micro Focus (es decir,). `MFDBFH`

> **Important**
>
> En esta guía se asume que utiliza Amazon Aurora Postgres como almacén de datos de Micro Focus () `MFDBFH` para almacenar los datos de su aplicación.

Inicie las herramientas de archivos de datos de Rocket Software (anteriormente Micro Focus)

Tras cumplir los requisitos previos, inicie las herramientas de archivos de datos de Micro Focus configurando la variable de `MFDBFH_CONFIG` entorno para acceder a los conjuntos de datos almacenados en la base de datos (`MFDBFH`).

Para ello,

1. Inicie sesión en el escritorio de Micro Focus Enterprise Developer e inicie la línea de comandos de Enterprise Developer (64 bits) desde el menú Inicio.
2. Defina la variable de `MFDBFH_CONFIG` entorno con la ruta completa `MFDBCH.cfg` del archivo.

```
set MFDBFH_CONFIG="C:\MicroFocus\config\MFDBFH.cfg"
```

3. Inicie Micro Focus Data File Tools desde la línea de comandos de Enterprise Developer mediante el siguiente comando.

```
mfdatatools2
```

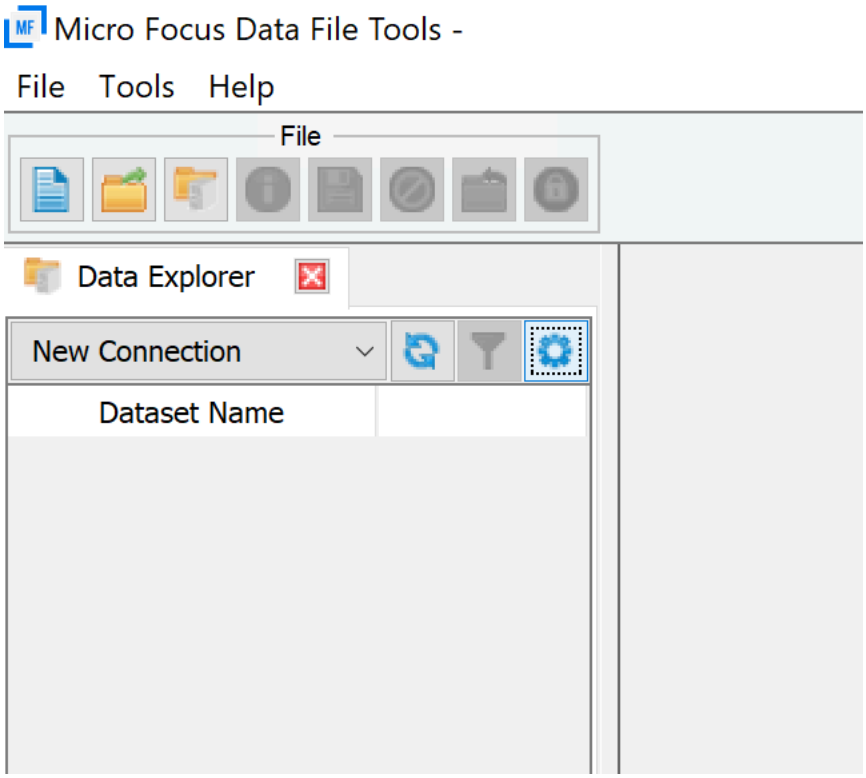
Se abren las herramientas de archivos de datos de Micro Focus en una ventana independiente.

Edite los conjuntos de datos de VSAM almacenados en la base de datos MFDBFH

Una vez que inicie las herramientas de archivos de datos de Micro Focus, abra un conjunto de datos de VSAM que está almacenado en el almacén de datos de Micro Focus.

Para ello,

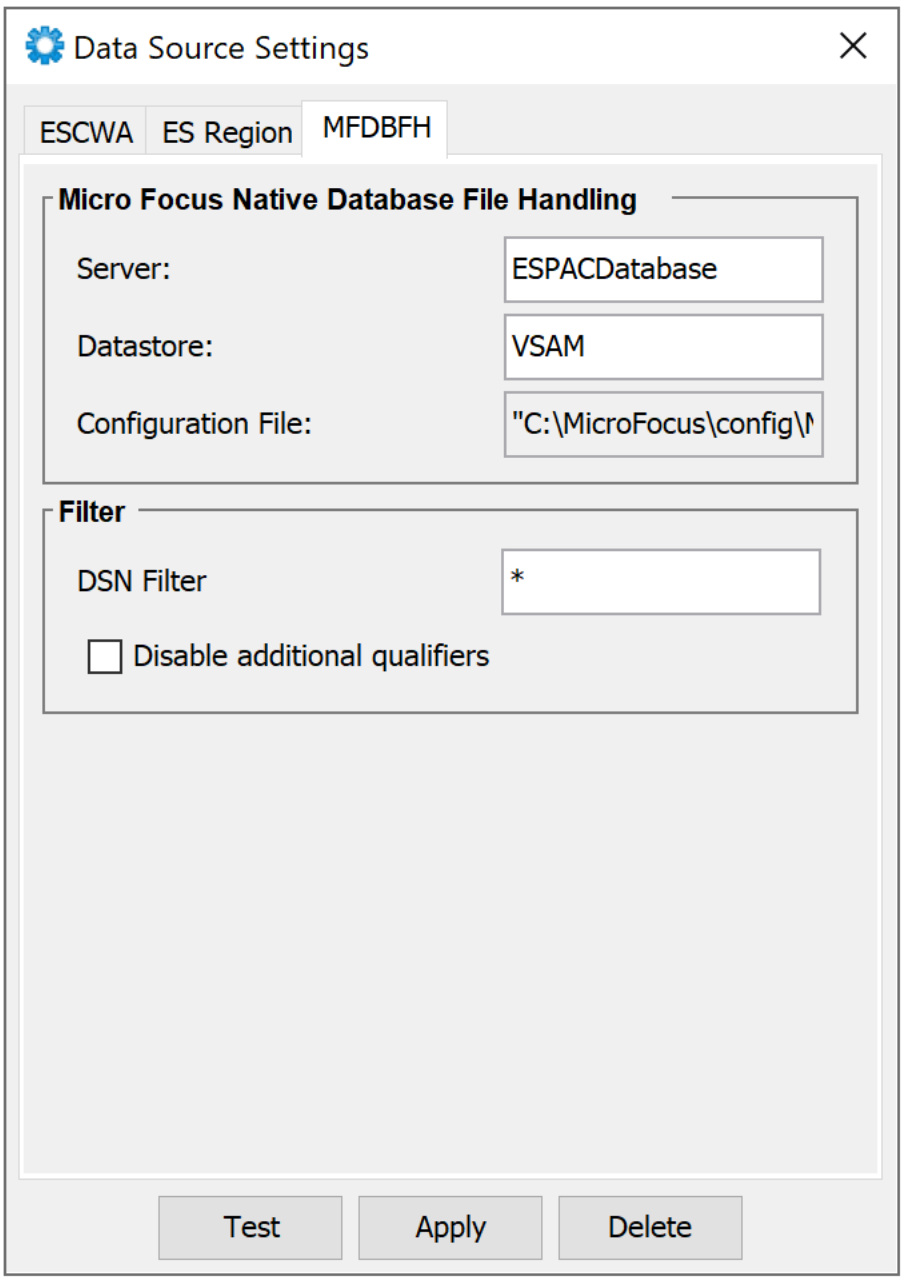
1. En el menú Archivo de la ventana Herramientas de archivos de datos de Micro Focus, seleccione el Explorador de datos.
2. En la sección Explorador de datos, seleccione Configuración (icono de engranaje) para configurar una nueva conexión. Se abrirá una ventana de configuración de la fuente de datos.



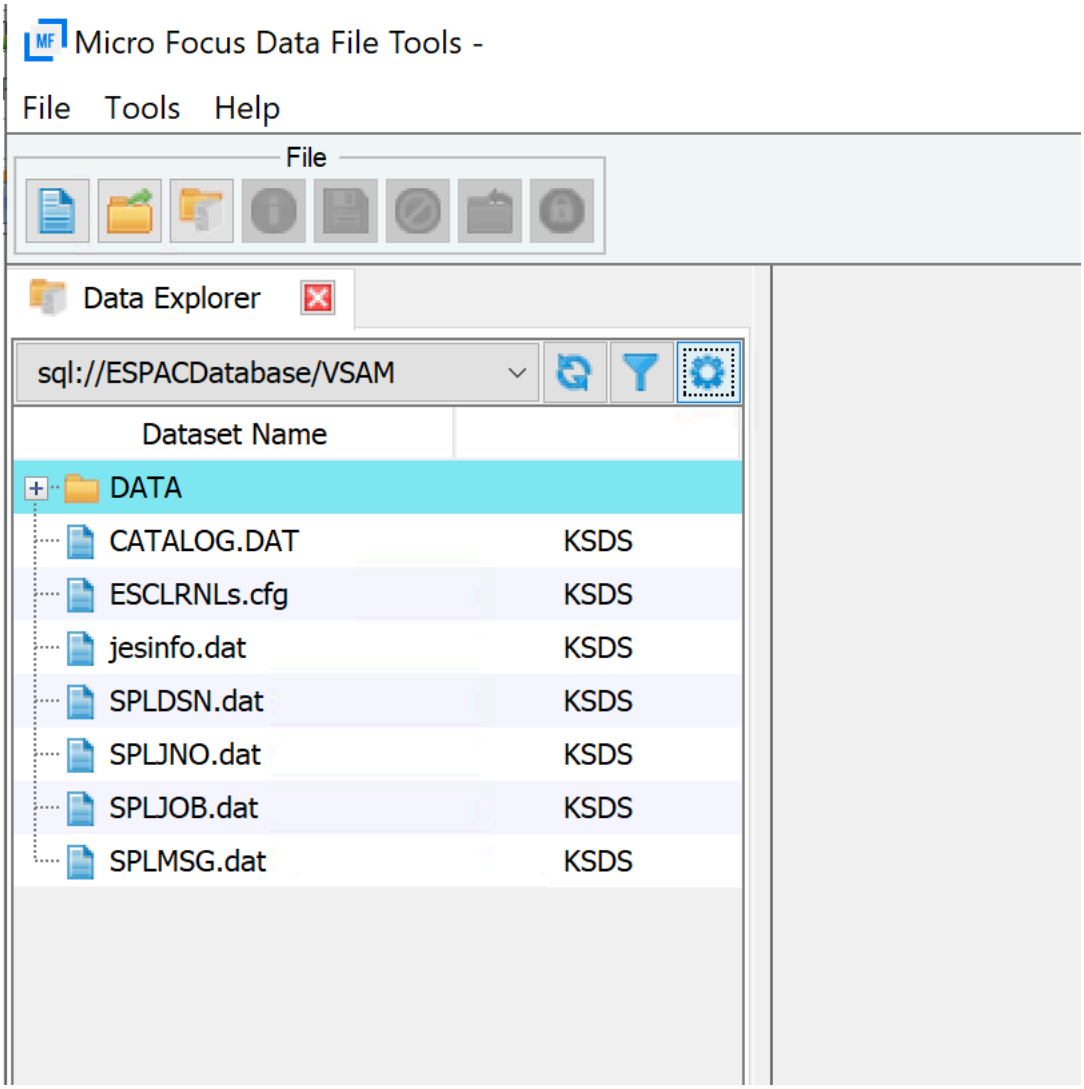
3. En la ventana de configuración de la fuente de datos, seleccione la pestaña MFDBFH e introduzca los siguientes valores:

   - Servidor: `ESPACDatabase`
   - Almacén de datos: `VSAM`

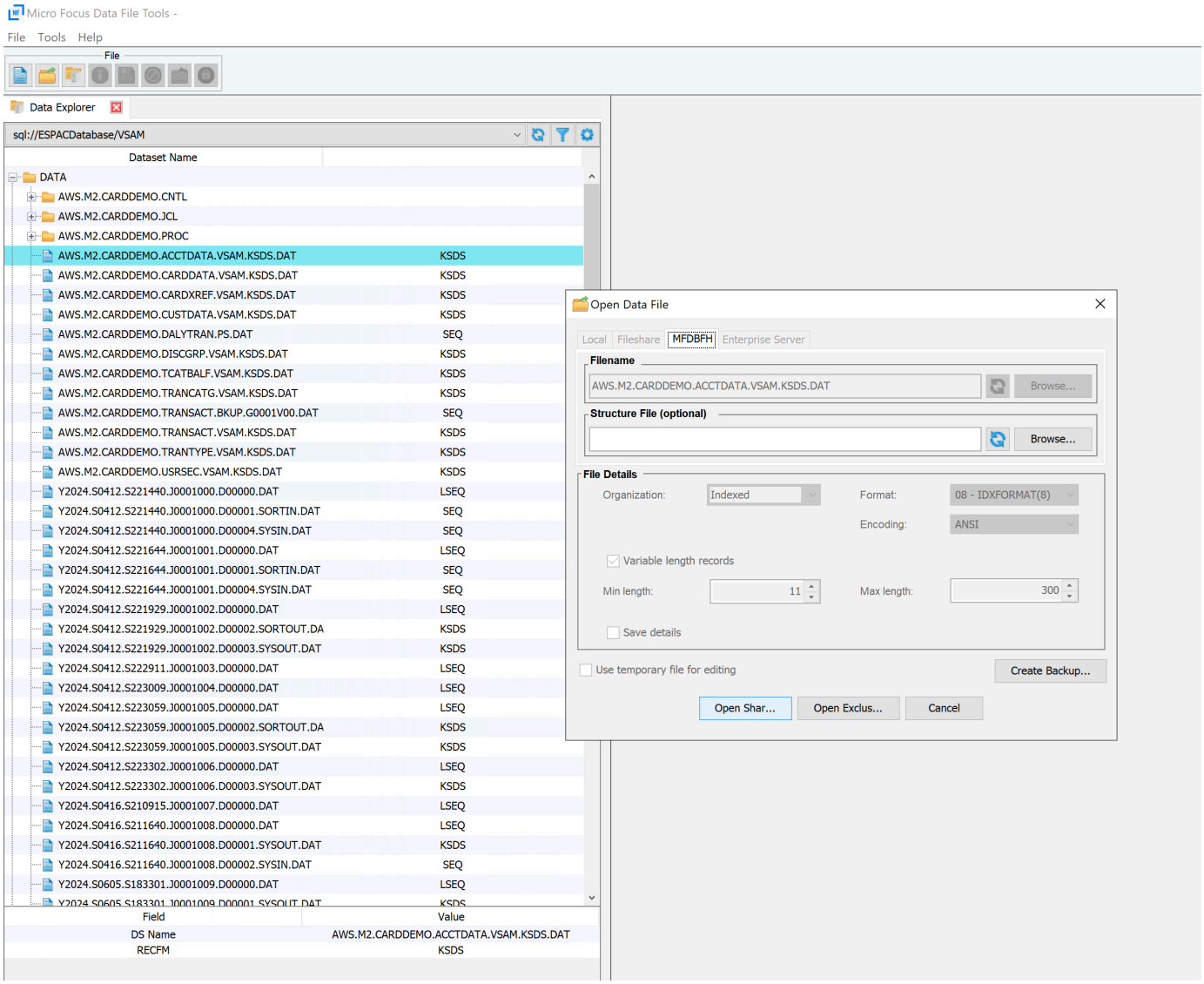
   Seleccione Aplicar para guardar la configuración.



El explorador de datos muestra ahora todos los conjuntos de datos que están almacenados. `MFDBFH`



4. Amplíe la ruta relativa `DATA` y haga doble clic en el conjunto de datos de VSAM que desee abrir.
5. En la ventana Abrir archivo de datos, elija Abrir compartido o Abrir en modo exclusivo para abrir el conjunto de datos.



Ahora puede ver o editar el conjunto de datos abierto.

Edite conjuntos de datos que no sean de VSAM almacenados en la base de datos MFDBFH

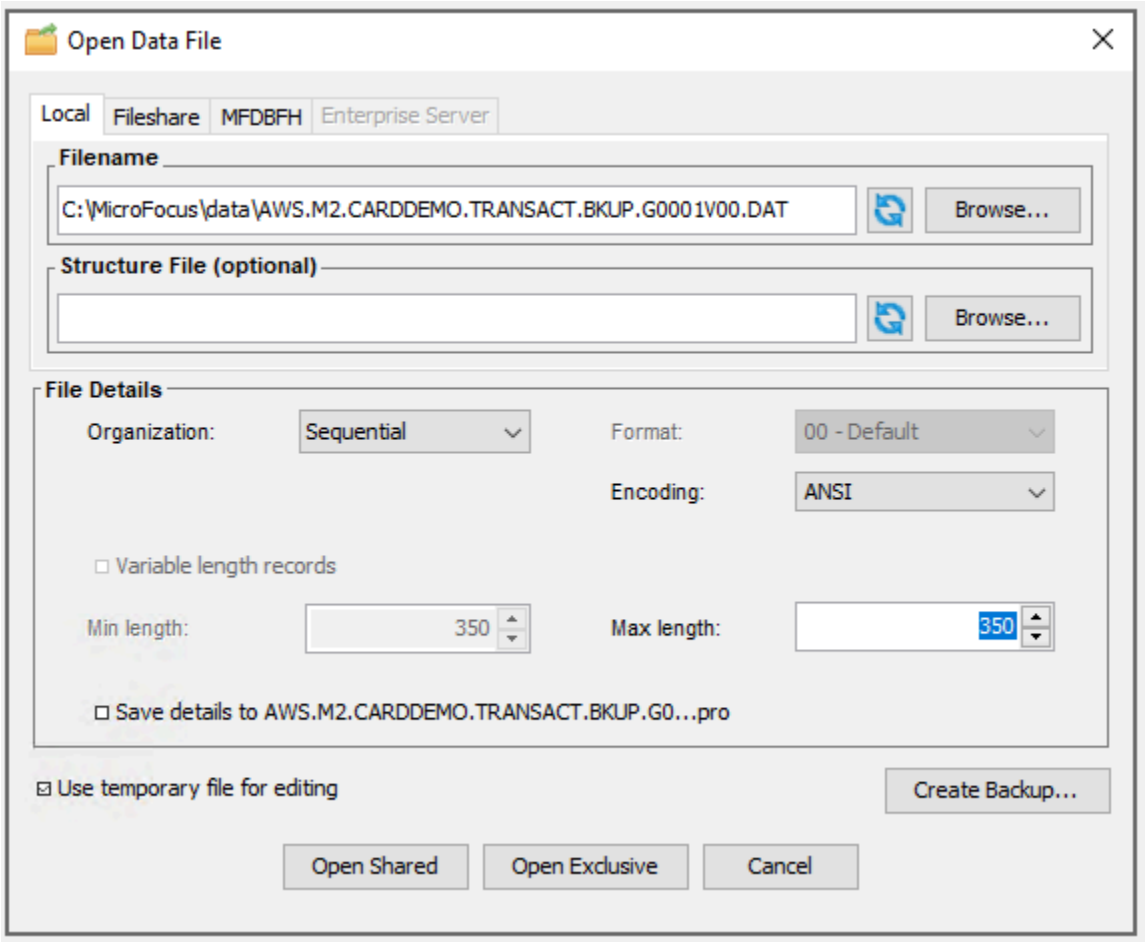
Si desea editar conjuntos de datos que no son de VSAM, abra un conjunto de datos que no sea de VSAM y que esté almacenado en el almacén de datos de Micro Focus.

Para ello,

1. Desde la línea de comandos de Enterprise Developer (64 bits), ejecute el `dbfhdeploy data extract` comando para descargar el conjunto de datos que no es de VSAM al sistema de archivos local.

   > **Note**
   >
   > Antes de ejecutar este comando, asegúrese de haber establecido la variable de `MFDBFH_CONFIG` entorno con la ruta completa al archivo`MFDBFH.cfg`.
   >
   > ```
   > dbfhdeploy data extract sql://ESPACDatabase/VSAM/
   > AWS.M2.CARDDEMO.TRANSACT.BKUP.G0001V00.DAT?folder=/DATA C:\MicroFocus\data
   > \AWS.M2.CARDDEMO.TRANSACT.BKUP.G0001V00.DAT
   > ```

2. Inicie las herramientas de archivos de datos de Micro Focus desde el menú Inicio.
3. En el menú Archivo de Micro Focus Data File Tools, seleccione Abrir y, a continuación, elija Archivo de datos.
4. En la ventana Abrir archivo de datos, busque el conjunto de datos descargado en su sistema de archivos local. Edite los detalles del archivo según sea necesario. A continuación, seleccione Abrir en modo compartido o Abrir en modo exclusivo para abrir el conjunto de datos.



Ahora puede ver o editar el conjunto de datos abierto.

Los conjuntos de datos editados o actualizados se pueden volver a importar al almacén de datos de Micro Focus siguiendo los pasos de the section called “Importación de conjuntos de datos para aplicaciones de ” o utilizando la utilidad de línea de comandos dbfhdeploy.

Edite conjuntos de datos VSAM y no VSAM almacenados en el sistema de archivos (EFS/FSx)

También puede abrir un conjunto de datos almacenado en un sistema de archivos.

Para ello,

1. Monte el EFS/FSx el sistema de archivos en la EC2 instancia de Enterprise Developer.
2. Utilice las herramientas de archivos de datos de Micro Focus para buscar y abrir los conjuntos de datos del sistema de archivos.

Tutoriales para Rocket Software (anteriormente Micro Focus)

Los tutoriales de esta sección le ayudan a empezar a configurar diversas tareas en el motor de ejecución de Rocket Software para el servicio de modernización de AWS mainframe. Estos tutoriales sirven para configurar una aplicación de muestra, CI/CD canalizaciones, uso de plantillas con Rocket Enterprise Developer y configuración de Enterprise Analyzer.

Temas

- Tutorial: Configuración de la versión de Rocket Software (anteriormente Micro Focus) para la aplicación BankDemo de muestra
- Tutorial: Configuración de un CI/CD canalización para su uso con Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)
- Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer
- Tutorial: Utilice plantillas con Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)
- Tutorial: Configurar Enterprise Analyzer en 2.0 AppStream
- Tutorial: Configurar Rocket Enterprise Developer en AppStream 2.0

Tutorial: Configuración de la versión de Rocket Software (anteriormente Micro Focus) para la aplicación BankDemo de muestra

AWS La modernización del mainframe le permite configurar compilaciones (y canalizacionesintegration/continuous delivery (CI/CD) continuas para sus aplicaciones migradas. Estas compilaciones y canalizaciones utilizan AWS CodeBuild y proporcionan estas AWS CodePipeline capacidades. AWS CodeCommit CodeBuild es un servicio de compilación totalmente gestionado que compila el código fuente, ejecuta pruebas unitarias y produce artefactos listos para su despliegue. CodeCommit es un servicio de control de versiones que te permite almacenar y gestionar de forma privada los repositorios de Git en la nube. AWS CodePipeline es un servicio de entrega continua que te permite modelar, visualizar y automatizar los pasos necesarios para lanzar tu software.

En este tutorial se muestra cómo AWS CodeBuild compilar el código fuente de la aplicación de BankDemo muestra desde Amazon S3 y, a continuación, exportar el código compilado a Amazon S3.

AWS CodeBuild es un servicio de integración continua totalmente gestionado que compila el código fuente, ejecuta pruebas y produce paquetes de software listos para su implementación. Con CodeBuild, puede usar entornos de compilación preempaquetados o puede crear entornos de compilación personalizados que usen sus propias herramientas de compilación. En esta situación de demostración se utiliza la segunda opción. Consiste en un entorno de CodeBuild compilación que utiliza una imagen de Docker preempaquetada.

> **Important**
>
> Antes de iniciar su proyecto de modernización de unidad central, le recomendamos que conozca el Programa de aceleración de la migración de AWS o que se ponga en contacto con especialistas en unidades centrales de AWS para obtener información sobre los pasos necesarios para modernizar una aplicación de unidad central.

Temas

- Requisitos previos
- Paso 1: compartición de los activos de compilación con la cuenta de AWS
- Paso 2: creación de buckets de Amazon S3
- Paso 3: creación del archivo de especificaciones de compilación
- Paso 4: carga de los archivos de origen
- Paso 5: creación de políticas de IAM
- Paso 6: creación de un rol de IAM
- Paso 7: asociación de las políticas de IAM al rol de IAM
- Paso 8: Crea el CodeBuild proyecto
- Paso 9: inicio de la compilación
- Paso 10: descarga de los artefactos de salida
- Eliminar recursos

Requisitos previos

Antes de comenzar este tutorial, complete los siguientes requisitos previos:

- Descarga la aplicación de BankDemo muestra y descomprímela en una carpeta. La carpeta de origen contiene programas COBOL y cuadernos, así como definiciones. También contiene una

carpeta JCL como referencia, aunque no es necesario compilar JCL. La carpeta también contiene los metaarchivos necesarios para la compilación.

- En la consola de modernización AWS del mainframe, elija Herramientas. En Análisis, desarrollo y compilación de activos, seleccione Compartir activos con mi cuenta de AWS.

Paso 1: compartición de los activos de compilación con la cuenta de AWS

En este paso, asegúrese de compartir los activos de compilación con su AWS cuenta, especialmente en la región en la que se utilizan los activos.

1. Abra la consola de modernización AWS del mainframe en https://console.aws.amazon.com/m2/.
2. En el panel de navegación izquierdo, seleccione Herramientas.
3. En Análisis, desarrollo y creación de activos, seleccione Compartir activos con mi AWS cuenta.

> **Important**
>
> Debes realizar este paso una vez en cada AWS región en la que vayas a realizar las compilaciones.

Paso 2: creación de buckets de Amazon S3

En este paso, creará dos buckets de Amazon S3. El primero es un bucket de entrada para almacenar el código fuente y el otro es un depósito de salida para almacenar la salida de la compilación. Para obtener más información, consulte Creación, configuración y trabajo con buckets de Amazon S3 en la Guía del usuario de Amazon S3.

1. Para crear el bucket de entrada, inicie sesión en la consola de Amazon S3 y elija Crear bucket.
2. En la configuración general, proporciona un nombre para el depósito y especifica el Región de AWS lugar en el que quieres crearlo. Un ejemplo Región de AWS de nombre `regionId` es `codebuild-regionId-accountId-input-bucket` dónde está el depósito y `accountId` es su Cuenta de AWS ID.

> **Note**
>
> Si va a crear el depósito en un lugar diferente al Región de AWS de EE. UU. Este (Virginia del Norte), especifique el `LocationConstraint` parámetro. Para obtener más información, consulte CreateBucket en Amazon Simple Storage Service API Referenc.

3. Conserve todos los demás valores y elija Crear bucket.
4. Repita los pasos 1 a 3 para crear el bucket de salida. Un ejemplo de nombre es `codebuild-regionId-accountId-output-bucket` dónde `regionId` está Región de AWS el depósito y `accountId` es su Cuenta de AWS ID.

   Utilice los mismos nombres que elija para los buckets en todo este tutorial.

Paso 3: creación del archivo de especificaciones de compilación

En este paso, creará un archivo de especificaciones de compilación. Este archivo proporciona los comandos de compilación y la configuración relacionada, en formato YAML, CodeBuild para ejecutar la compilación. Para obtener más información, consulta la referencia sobre las especificaciones de compilación CodeBuild en la Guía del AWS CodeBuild usuario.

1. Cree un archivo con el nombre `buildspec.yml` en el directorio que haya utilizado para descomprimir como se indicaba en los requisitos previos.
2. Añada el siguiente contenido al archivo y guarde. No se requieren cambios en este archivo.

```
version: 0.2
env:
  exported-variables:
    - CODEBUILD_BUILD_ID
    - CODEBUILD_BUILD_ARN
phases:
  install:
    runtime-versions:
      python: 3.7
  pre_build:
    commands:
      - echo Installing source dependencies...
      - ls -lR $CODEBUILD_SRC_DIR/source
  build:
    commands:
```

```
      - echo Build started on `date`
      - /start-build.sh -Dbasedir=$CODEBUILD_SRC_DIR/source -Dloaddir=
$CODEBUILD_SRC_DIR/target
  post_build:
    commands:
      - ls -lR $CODEBUILD_SRC_DIR/target
      - echo Build completed on `date`
artifacts:
  files:
    - $CODEBUILD_SRC_DIR/target/**
```

AquíCODEBUILD_BUILD_ID, CODEBUILD_BUILD_ARN$CODEBUILD_SRC_DIR/source, y $CODEBUILD_SRC_DIR/target están disponibles las variables de entorno CodeBuild. Para obtener más información, consulte Variables de entorno en los entornos de compilación.

En este punto, la estructura de directorios debería ser similar a la siguiente.

```
(root directory name)
    |-- build.xml
    |-- buildspec.yml
    |-- LICENSE.txt
    |-- source
         |... etc.
```

3. Comprima el contenido de la carpeta en un archivo con el nombre BankDemo.zip. En este tutorial, no puede comprimir la carpeta. Así es que, comprima el contenido de la carpeta en un archivo con el nombre BankDemo.zip.

Paso 4: carga de los archivos de origen

En este paso, debe cargar el código fuente de la aplicación de BankDemo muestra en su bucket de entrada de Amazon S3.

1. En la consola de Amazon SNS, en el panel de navegación izquierdo, seleccione Buckets. A continuación, elija el bucket de entrada que ha creado anteriormente.
2. En Objetos, selecciona Cargar.
3. En la sección Archivos y carpetas, elija Añadir archivos.
4. Vaya al archivo BankDemo.zip y elíjalo.
5. Seleccione Cargar.

Paso 5: creación de políticas de IAM

En este paso, creará dos políticas de IAM. Una política otorga permisos para que AWS Mainframe Modernization pueda acceder a la imagen de Docker que contiene las herramientas de compilación de Rocket Software y utilizarla. Esta política no está personalizada para los clientes. La otra política otorga permisos para que AWS Mainframe Modernization interactúe con los buckets de entrada y salida y con los CloudWatch registros de Amazon que se CodeBuild generan.

Para obtener más información sobre la creación de una política de IAM, consulte Crear políticas de IAM en la Guía del usuario de IAM.

Para crear una política de acceso a las imágenes de Docker

1. En la consola de IAM, copie el siguiente documento de política y péguelo en el editor de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:GetDownloadUrlForLayer",
                "ecr:BatchGetImage"
            ],
            "Resource": "arn:aws:ecr:*:673918848628:repository/m2-enterprise-build-
tools"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::aws-m2-repo-*-<region>-prod"
```

```
        }
    ]
}
```

2. Escriba un nombre para la política, por ejemplo, `m2CodeBuildPolicy`.

Crear una política que permita a la modernización del AWS mainframe interactuar con los depósitos y los registros

1. En la consola de IAM, copie el siguiente documento de política y péguelo en el editor de políticas. Asegúrese de actualizar `regionId` a Región de AWS, y `accountId` a su. Cuenta de AWS

```
{
      "Version": "2012-10-17",
      "Statement": [
          {
              "Action": [
                  "logs:CreateLogGroup",
                  "logs:CreateLogStream",
                  "logs:PutLogEvents"
              ],
              "Resource": [
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project",
                  "arn:aws:logs:regionId:accountId:log-group:/aws/codebuild/
codebuild-bankdemo-project:*"
              ],
              "Effect": "Allow"
          },
          {
              "Action": [
                  "s3:PutObject",
                  "s3:GetObject",
                  "s3:GetObjectVersion",
                  "s3:GetBucketAcl",
                  "s3:GetBucketLocation",
                  "s3:List*"
              ],
              "Resource": [
                  "arn:aws:s3:::codebuild-regionId-accountId-input-bucket",
                  "arn:aws:s3:::codebuild-regionId-accountId-input-bucket/*",
```

```
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-output-bucket/*"
            ],
            "Effect": "Allow"
        }
    ]
}
```

2. Escriba un nombre para la política, por ejemplo, `BankdemoCodeBuildRolePolicy`.

Paso 6: creación de un rol de IAM

En este paso, creará una nueva función de IAM que le CodeBuild permita interactuar con AWS los recursos, tras asociar las políticas de IAM que creó anteriormente a esta nueva función de IAM.

Para obtener información sobre la creación de un rol de servicio, consulte Creación de un rol para delegar permisos a un AWS servicio en la Guía del usuario de IAM,.

1. Inicie sesión en la consola de IAM; y elija Roles en el panel de navegación de la izquierda.
2. Elija Crear rol.
3. En Tipo de entidad de confianza, seleccione Servicio de AWS.
4. En Casos de uso de otros servicios de AWS, elija y CodeBuild, a continuación, CodeBuildvuelva a elegir.
5. Elija Siguiente.
6. Elija Siguiente en la página Agregar permisos. Posteriormente, asignará una política al rol.
7. En Detalles del rol, proporcione un nombre para el rol, por ejemplo, `BankdemoCodeBuildServiceRole`.
8. En Seleccionar entidades de confianza, compruebe que el documento de política tenga el siguiente aspecto:

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
```

```
            "Action": "sts:AssumeRole"
          }
        ]
      }
```

9. Seleccione Crear rol.

Paso 7: asociación de las políticas de IAM al rol de IAM

En este paso, asociará las políticas de IAM que ha creado antes al rol de IAM `BankdemoCodeBuildServiceRole`.

1. Inicie sesión en la consola de IAM; y elija Roles en el panel de navegación de la izquierda.
2. En Roles, elija el rol que ha creado anteriormente, por ejemplo, `BankdemoCodeBuildServiceRole`.
3. En la sección Políticas de permisos, elija Agregar permisos y, después, Asociar políticas.
4. En Otras políticas de permisos, elija las políticas que ha creado anteriormente, por ejemplo, `m2CodeBuildPolicy` y`BankdemoCodeBuildRolePolicy`.
5. Seleccione Asociar políticas.

Paso 8: Crea el CodeBuild proyecto

En este paso, crearás el CodeBuild proyecto.

1. Inicie sesión en la CodeBuild consola y elija Crear proyecto de compilación.
2. En la sección Configuración del proyecto, proporciona un nombre para el proyecto, por ejemplo, `codebuild-bankdemo-project`.
3. En la sección Origen, en Proveedor del origen, elija Amazon S3 y, a continuación, elija el bucket de entrada que ha creado anteriormente, por ejemplo, `codebuild-regionId-accountId-input-bucket`.
4. En el campo Clave de objeto de S3 o carpeta de S3, introduzca el nombre del archivo zip que ha cargado en el bucket de S3. En este caso, el nombre de archivo es `bankdemo.zip`.
5. En la sección Entorno, seleccione Imagen personalizada.
6. En el campo Tipo de entorno, seleccione Linux.
7. En Registro de imágenes, seleccione Otro registro.

8. En el campo URL del registro externo,

   - Para Rocket Software v9: Introduce`673918848628.dkr.ecr.us-west-1.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1`. Si está utilizando una AWS región diferente con Rocket Software v9, también puede especificar `673918848628.dkr.ecr.<m2-region>.amazonaws.com/m2-enterprise-build-tools:9.0.7.R1` dónde <m2-region>hay una AWS región en la que esté disponible el servicio de modernización de AWS mainframe (por ejemplo,). `eu-west-3`
   - Para Rocket Software v8: Introduce `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:8.0.9.R1`
   - Para Rocket Software v7: Introduzca `673918848628.dkr.ecr.us-west-2.amazonaws.com/m2-enterprise-build-tools:7.0.R10`
9. En Rol de servicio, elija Rol de servicio existente y, en el campo ARN de rol, elija el rol de servicio que ha creado anteriormente; por ejemplo, `BankdemoCodeBuildServiceRole`.
10. En Especificaciones de la compilación, elija Utilizar un archivo de especificación de compilación.
11. En la sección Artefactos, en Tipo, elija Amazon S3 y, a continuación, elija su bucket de salida, por ejemplo, `codebuild-regionId-accountId-output-bucket`.
12. En el campo Nombre, introduzca el nombre de una carpeta del bucket en la que desee que se incluyan los artefactos de salida de la compilación, por ejemplo, `bankdemo-output.zip`.
13. En Empaquetado de artefactos, elija Zip.
14. Elija Crear el proyecto de compilación.

Paso 9: inicio de la compilación

En este paso, iniciará la compilación.

1. Inicie sesión en la CodeBuild consola.
2. En el panel de navegación, elija Proyectos de compilación.
3. Elija el proyecto de compilación que ha creado antes, por ejemplo,`codebuild-bankdemo-project`.
4. Seleccione Iniciar la compilación.

Este comando inicia la compilación. La compilación se ejecuta de forma asincrónica. La salida del comando es un JSON que incluye el ID del atributo. Este identificador de atributo es una referencia

al identificador de CodeBuild compilación de la compilación que acabas de iniciar. Puedes ver el estado de la compilación en la CodeBuild consola. También puede ver los registros detallados sobre la ejecución de la compilación en la consola. Para obtener más información, consulte View detailed build information en AWS CodeBuild User Guide.

Cuando se complete la fase actual, significa que la compilación ha finalizado correctamente y que los artefactos compilados están listos en Amazon S3.

Paso 10: descarga de los artefactos de salida

En este paso, descargará los artefactos de salida de Amazon S3. La herramienta de compilación de Rocket Software puede crear varios tipos de ejecutables diferentes. En este tutorial, generará objetos compartidos.

1. Inicie sesión en la consola de Amazon S3.
2. En la sección Buckets role="bold">, elija el nombre del bucket de salida, por ejemplo, `codebuild-regionId-accountId-output-bucket`.
3. Seleccione Descargar role="bold">.
4. Descomprima el archivo descargado. Navegue hasta la carpeta de destino para ver los artefactos de la compilación. Estos incluyen los objetos compartidos de Linux `.so`.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para evitar cargos adicionales. Para ello, complete los siguientes pasos.

- Elimine los depósitos de S3 que ha creado para este tutorial. Para obtener más información, consulte Eliminar un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Elija las políticas que ha creado para este tutorial. Para obtener más información, consulte Eliminación de políticas de IAM en la Guía del usuario de IAM.
- Elija el rol de IAM que ha creado para este tutorial. Para obtener más información, consulte Eliminación de roles o perfiles de instancia en la Guía del usuario de IAM.
- Elimine el CodeBuild proyecto que creó para este tutorial. Para obtener más información, consulte Eliminar un proyecto de compilación CodeBuild en la Guía del AWS CodeBuild usuario.

Tutorial: Configuración de un CI/CD canalización para su uso con Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)

En este tutorial se muestra cómo importar, editar, compilar y ejecutar la aplicación de BankDemo ejemplo en Rocket Enterprise Developer y, a continuación, cómo confirmar los cambios para activar un CI/CD canalización.

Contenido

- Requisitos previos
- Creación CI/CD infraestructura básica de canalización
- Cree un AWS CodeCommit repositorio y CI/CD canalización
  - Ejemplo de archivo de activación de YAML: config_git.yml
- Creación de Enterprise Developer AppStream 2.0
- Configuración y prueba de Enterprise Developer
  - Clona el BankDemo CodeCommit repositorio en Enterprise Developer
  - Cree un proyecto COBOL para BankDemo mainframe y cree una aplicación
  - Cree un BankDemo CICS local y un entorno por lotes para las pruebas
  - Inicie el servidor de BANKDEMO desde Enterprise Developer
  - Inicie el terminal Rumba 3270
  - Ejecute una BankDemo transacción
  - Detenga el servidor de BANKDEMO desde Enterprise Developer
- Ejercicio 1: Mejorar el cálculo de los préstamos en la aplicación de BANKDEMO
  - Agregue la regla de análisis de préstamos a Enterprise Developer Code Analysis
  - Paso 1: Realizar un análisis de código para calcular el préstamo
  - Paso 2: Modificar el mapa CICS BMS y el programa COBOL y pruébelo
  - Paso 3: Agregar el cálculo del importe total en el programa COBOL
  - Paso 4: Confirmar los cambios y ejecutar la canalización de CI/CD
- Ejercicio 2: Extraiga el cálculo del préstamo en BankDemo la solicitud
  - Paso 1: Refactorizar la rutina de cálculo de préstamos en una sección de COBOL
  - Paso 2: Extraer la rutina de cálculo de préstamos a un programa COBOL independiente
  - Paso 3: Confirma los cambios y ejecuta el CI/CD canalización
- Limpieza de recursos

Requisitos previos

Descargue los siguientes archivos:

- `basic-infra.yaml`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).
- `pipeline.yaml`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).
- `m2-code-sync-function.zip`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).
- `config_git.yml`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).
- `BANKDEMO-source.zip`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).
- `BANKDEMO-exercise.zip`
  - Descarga desde la región Europa (Fráncfort).
  - Descarga desde la región Este de EE. UU. (Norte de Virginia).

El propósito de cada archivo es el siguiente:

`basic-infra.yaml`

Esta AWS CloudFormation plantilla crea la infraestructura básica necesaria para CI/CD canalización: VPC, buckets de Amazon S3, etc.

`pipeline.yaml`

Una función Lambda utiliza esta AWS CloudFormation plantilla para lanzar la pila de canalización. Asegúrese de que la plantilla se encuentre en un bucket de Amazon S3 de acceso público. Añada

el enlace a este bucket como valor predeterminado para el parámetro `PipelineTemplateURL` de la plantilla `basic-infra.yaml`.

`m2-code-sync-function.zip`

Esta función Lambda crea el CodeCommit repositorio, la estructura de directorios basada en`config_git.yaml`, y lanza la pila de canalizaciones mediante. `pipeline.yaml` Asegúrese de que este archivo zip esté disponible en un bucket de Amazon S3 de acceso público en todos los Regiones de AWS lugares en los que se admita la modernización de AWS mainframe. Le recomendamos que almacene el archivo en un depósito en uno Región de AWS y lo replique en todos los depósitos. Regiones de AWS Utilice una convención de nomenclatura para el depósito con un sufijo que identifique lo específico Región de AWS (por ejemplo`m2-cicd-deployment-source-eu-west-1`) y añada el prefijo `m2-cicd-deployment-source` como valor predeterminado para el parámetro `DeploymentSourceBucket` y forme el depósito completo mediante la función de AWS CloudFormation sustitución `!Sub {DeploymentSourceBucket}-${AWS::Region}` mientras hace referencia a ese depósito en la `basic-infra.yaml` plantilla como recurso. `SourceSyncLambdaFunction`

`config_git.yml`

CodeCommit definición de la estructura de directorios. Para obtener más información, consulte Ejemplo de archivo de activación de YAML: config_git.yml.

`BANKDEMO-source.zip`.

BankDemo código fuente y archivo de configuración creados a partir del CodeCommit repositorio.

`BANKDEMO-exercise.zip`.

BankDemo fuente de los ejercicios tutoriales creados a partir del CodeCommit repositorio.

Creación CI/CD infraestructura básica de canalización

Utilice la AWS CloudFormation plantilla `basic-infra.yaml` para crear la pila de infraestructura básica de la canalización de CI/CD a través de la AWS CloudFormation consola. Esta pila crea buckets de Amazon S3 en los que puede cargar el código y los datos de su aplicación, y una AWS Lambda función de apoyo para crear otros recursos necesarios, como un AWS CodeCommit repositorio y una AWS CodePipeline canalización.

> **Note**
>
> Para lanzar esta pila, necesita permisos para administrar IAM, Amazon S3, Lambda AWS CloudFormation y permisos de uso. AWS KMS

1. Inicie sesión en AWS Management Console /cloudformation y abra la AWS CloudFormation consola. https://console.aws.amazon.com
2. Cree una nueva pila utilizando una de las siguientes opciones:
   - Elija Crear pila. Esta es la única opción si tiene una pila actualmente en activo.
   - En la página Pilas, elija Crea pila. Esta opción sólo es visible si no hay pilas en ejecución.
3. En la página Especificar la plantilla:
   - En Preparar la plantilla, elija La plantilla está lista.
   - En Especificar plantilla, elija la URL de Amazon S3 como fuente de la plantilla e introduzca una de las siguientes opciones en URLs función de su perfil Región de AWS.
     - `https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/basic-infra.yaml`
     - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/basic-infra.yaml`
   - Para aceptar la configuración elija Siguiente.

   Se abre la página Crear pila.

Specify stack details

Stack name

Stack name

m2-cicd-pipeline-foundation

Stack name can include letters (A-Z and a-z), numbers (0-9), and dashes (-).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Networking Configuration

Do you want to use an existing VPC in your account?

If you select 'Yes', then you must provide the VPC ID and the Subnet IDs.

No (Create one)

Which VPC ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID should be used?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Which private subnet ID in a different AZ should be used for HA?

If you selected 'Yes' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

Enter the CIDR block that should be used for the new VPC

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

11.11.0.0/16

CIDR bits for creating subnets. Choose 5 for /27, 6 for /26, 7 for /25, 8 for /24 range

If you selected 'No (Create one)' for UseExistingVPC, this parameter is required. Otherwise, this value will be ignored.

5

Deployment Configuration

Name of the S3 bucket which contains the source files for this stack deployment

Don't change unless you know what you are doing.

m2

Name of the source package file for the infrastructure Lambda function

Don't change unless you know what you are doing.

cicd/mf/m2-code-sync-function.zip

Full URL of the pipeline CloudFormation template file

Don't change unless you know what you are doing.

https://m2-us-east-1.s3.us-east-1.amazonaws.com/cicd/mf/pipeline.yaml

What name prefix to use for the new S3 buckets?

A name prefix for the S3 buckets that will be created by this stack.

mf

Cancel Previous Next

Realice los siguientes cambios:

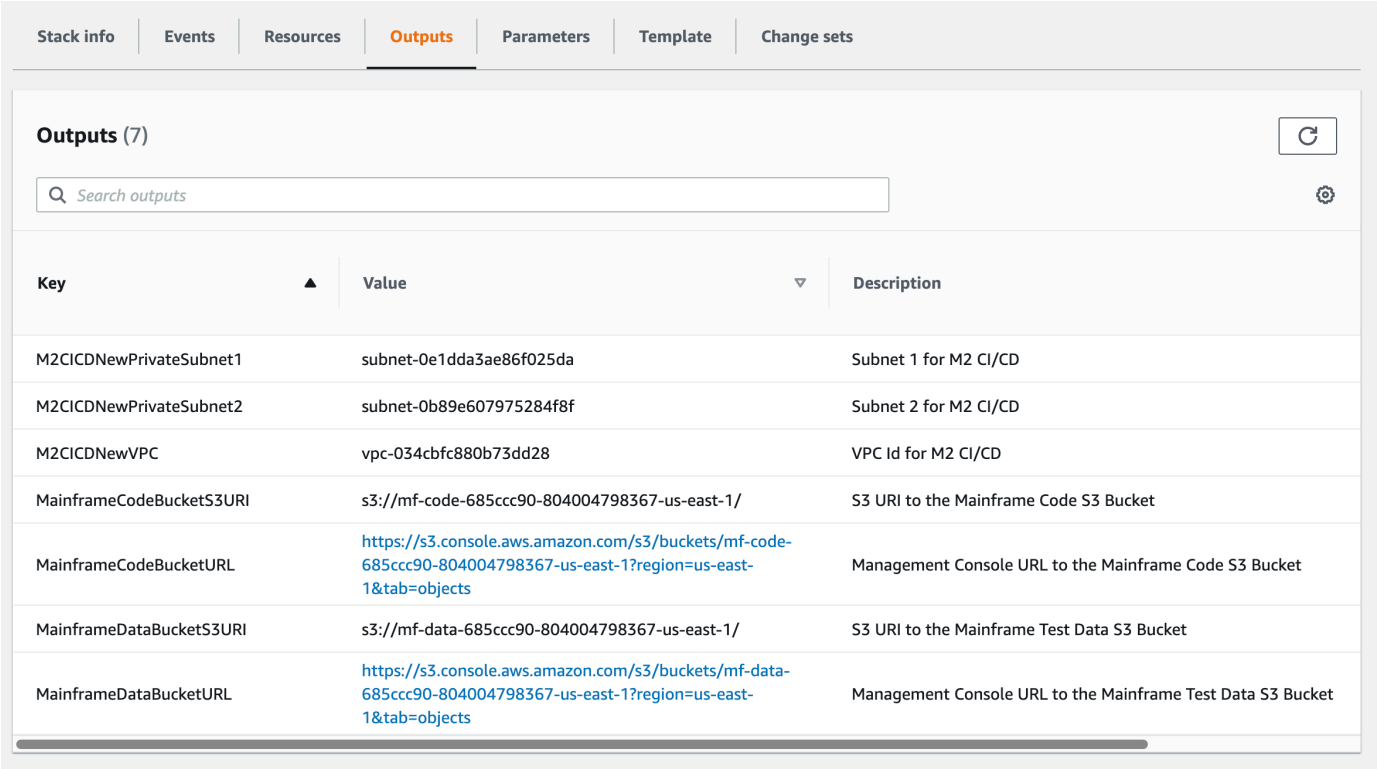
- Proporcione los valores adecuados para el nombre y los parámetros de la pila para la configuración de red.
- La mayoría de los parámetros de Configuraciones de la implementación se rellenan previamente de forma adecuada, por lo que no es necesario modificarlos. En función de su Región de AWS ubicación, cambie la AWS CloudFormation plantilla de canalización por una de las siguientes opciones de Amazon S3 URLs.
  - `https://m2-us-east-1.s3.amazonaws.com/cicd/mf/pipeline.yaml`
  - `https://m2-eu-central-1.s3.eu-central-1.amazonaws.com/cicd/mf/pipeline.yaml`
- Elija Siguiente.

> **Note**
>
> No cambie los valores de los parámetros predeterminados a menos que haya modificado la AWS CloudFormation plantilla usted mismo.

4. En Configurar opciones de pila, elija Siguiente.
5. En Capacidades, elija Acepto que AWS CloudFormation podría crear recursos de IAM AWS CloudFormation para permitir la creación de un rol de IAM en su nombre. Seleccione Creación de pila.

> **Note**
>
> El aprovisionamiento de la pila puede tardar entre 3 y 5 minutos.

6. Una vez que la pila se haya creado correctamente, diríjase a la sección Salidas de la pila recién aprovisionada. Allí encontrará el bucket de Amazon S3 en el que debe cargar el código de la unidad central y los archivos dependientes.



Cree un AWS CodeCommit repositorio y CI/CD canalización

En este paso, crea un CodeCommit repositorio y aprovisiona un CI/CD pila de canalización llamando a una función Lambda que llama AWS CloudFormation para crear la pila de canalización.

1. Descargue la aplicación BankDemo de muestra en su máquina local.
2. Cargue `bankdemo.zip` desde el equipo local al bucket de Amazon S3 creado en Creación CI/CD infraestructura básica de canalización.
3. Descargue `config_git.yml`.
4. Modifique el `config_git.yml` si es necesario, de la siguiente manera:

   - Añada su propio nombre de repositorio de destino, rama de destino y mensaje de confirmación.

   ```
   repository-config:
     target-repository: bankdemo-repo
     target-branch: main
     commit-message: Initial commit for bankdemo-repo main branch
   ```

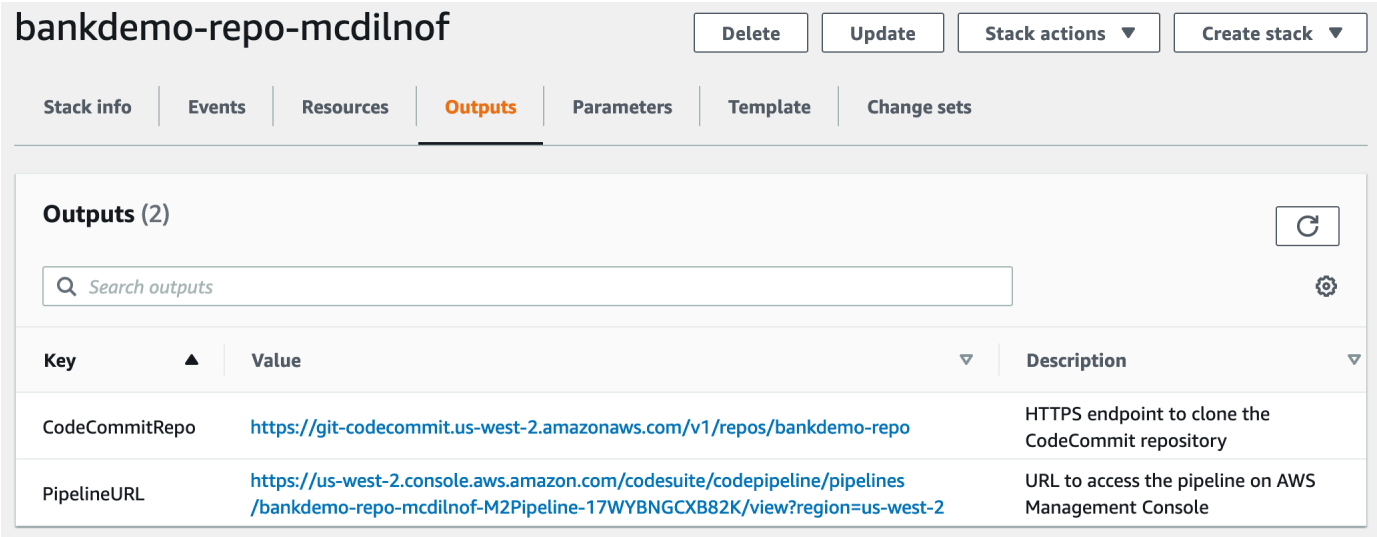
   - Escriba la dirección de correo electrónico que desea utilizar para las notificaciones.

```
pipeline-config:
  # Send pipeline failure notifications to these email addresses
  alert-notifications:
    - myname@mycompany.com
  # Send notifications for manual approval before production deployment to these
 email addresses
  approval-notifications:
    - myname@mycompany.com
```

5. Cargue el `config_git.yml` archivo que contiene la definición de la estructura de carpetas del CodeCommit repositorio en el bucket de Amazon S3 creado enCreación CI/CD infraestructura básica de canalización. Esto invocará la función de Lambda que aprovisionará automáticamente el repositorio y la canalización.

   Esto creará un CodeCommit repositorio con el nombre indicado en `target-repository` el `config_git.yml` archivo; por ejemplo,`bankdemo-repo`.

   La función Lambda también creará el CI/CD pila de tuberías a través AWS CloudFormation. La pila AWS CloudFormation tendrá el mismo prefijo que el nombre de `target-repository` proporcionado seguido de una cadena aleatoria (por ejemplo, `bankdemo-repo-`*`01234567`*). Puede encontrar la URL del CodeCommit repositorio y la URL para acceder a la canalización creada en la consola AWS de administración.



6. Si se ha completado la creación del CodeCommit repositorio, el CI/CD la canalización se activará inmediatamente para realizar una operación completa CI/CD.

7. Una vez insertado el archivo, se activará automáticamente la canalización, que se construirá, se desplegará en el entorno provisional, realizará algunas pruebas y esperará a la aprobación manual antes de implementarlo todo en el entorno de producción.

Ejemplo de archivo de activación de YAML: config_git.yml

```
repository-config:
  target-repository: bankdemo-repo
  target-branch: main
  commit-message: Initial commit for bankdemo-repo main branch
  directory-structure:
    - '/':
        files:
        - build.xml
        - '*.yaml'
        - '*.yml'
        - '*.xml'
        - 'LICENSE.txt'
        readme: |
          # Root Folder
          - 'build.xml' : Build configuration for the application
    - tests:
        files:
        - '*.py'
        readme: |
          # Test Folder
          - '*.py' : Test scripts
    - config:
        files:
        - 'BANKDEMO.csd'
        - 'BANKDEMO.json'
        - 'BANKDEMO_ED.json'
        - 'dfhdrdat'
        - 'ESPGSQLXA.dll'
        - 'ESPGSQLXA64.so'
        - 'ESPGSQLXA64_S.so'
        - 'EXTFH.cfg'
        - 'm2-2021-04-28.normal.json'
        - 'MFDBFH.cfg'
        - 'application-definition-template-config.json'
        readme: |
          # Config Folder
```

```
          This folder contains the application configuration files.
          - 'BANKDEMO.csd'     : CICS Resource definitions export file
          - 'BANKDEMO.json'    : Enterprise Server configuration
          - 'BANKDEMO_ED.json' : Enterprise Server configuration for ED
          - 'dfhdrdat'         : CICS resource definition file
          - 'ESPGSQLXA.dll'    : XA switch module Windows
          - 'ESPGSQLXA64.so'   : XA switch module Linux
          - 'ESPGSQLXA64_S.so' : XA switch module Linux
          - 'EXTFH.cfg'        : Micro Focus File Handler configuration
          - 'm2-2021-04-28.normal.json' : M2 request document
          - 'MFDBFH.cfg'       : Micro Focus Database File Handler
          - 'application-definition-template-config.json' : Application definition for
 M2
    - source:
        subdirs:
        - .settings:
            files:
              - '.bms.mfdirset'
              - '.cbl.mfdirset'
        - copybook:
            files:
              - '*.cpy'
              - '*.inc'
            readme: |
              # Copy folder
              This folder contains the source for COBOL copy books, PLI includes, ...
              - .cpy COBOL copybooks
              - .inc PLI includes
#        - ctlcards:
#            files:
#              - '*.ctl'
#              - 'KBNKSRT1.txt'
#            readme: |
#              # Control Card folder
#              This folder contains the source for Batch Control Cards
#              - .ctl Control Cards
        - ims:
            files:
              - '*.dbd'
              - '*.psb'
            readme: |
              # ims folder
              This folder contains the IMS DB source files with the extensions
              - .dbd for IMS DBD source
```

```
              - .psb for IMS PSB source
        - jcl:
            files:
              - '*.jcl'
              - '*.ctl'
              - 'KBNKSRT1.txt'
              - '*.prc'
            readme: |
              # jcl folder
              This folder contains the JCL source files with the extensions
              - .jcl
#        - proclib:
#            files:
#              - '*.prc'
#            readme: |
#              # proclib folder
#              This folder contains the JCL procedures referenced via PROCLIB
 statements in the JCL with extensions
#              - .prc
        - rdbms:
            files:
              - '*.sql'
            readme: |
              # rdbms folder
              This folder contains any DB2 related source files with extensions
              - .sql for any kind of SQL source
        - screens:
            files:
            - '*.bms'
            - '*.mfs'
            readme: |
              # screens folder
              This folder contains the screens source files with the extensions
              - .bms for CICS BMS screens
              - .mfs for IMS MFS screens
            subdirs:
            - .settings:
                files:
                    - '*.bms.mfdirset'
        - cobol:
            files:
              - '*.cbl'
              - '*.pli'
            readme: |
```

```
              # source folder
              This folder contains the program source files with the extensions
              - .cbl for COBOL source
              - .pli for PLI source
            subdirs:
            - .settings:
                files:
                   - '*.cbl.mfdirset'
    - tests:
        files:
        - 'test_script.py'
        readme: |
          # tests Folder
          This folder contains the application test scripts
pipeline-config:
  alert-notifications:
    - myname@mycompany.com
  approval-notifications:
    - myname@mycompany.com
```

Creación de Enterprise Developer AppStream 2.0

Para configurar Rocket Enterprise Developer en la AppStream versión 2.0, consulteTutorial: Configurar Rocket Enterprise Developer en AppStream 2.0.

Para conectar el CodeCommit repositorio a Enterprise Developer, utilice el nombre especificado `target-repository` enEjemplo de archivo de activación de YAML: config_git.yml.

Configuración y prueba de Enterprise Developer

Temas

- Clona el BankDemo CodeCommit repositorio en Enterprise Developer
- Cree un proyecto COBOL para BankDemo mainframe y cree una aplicación
- Cree un BankDemo CICS local y un entorno por lotes para las pruebas
- Inicie el servidor de BANKDEMO desde Enterprise Developer
- Inicie el terminal Rumba 3270
- Ejecute una BankDemo transacción
- Detenga el servidor de BANKDEMO desde Enterprise Developer

Conéctese a la instancia de Enterprise Developer AppStream 2.0 en la que creóCreación de Enterprise Developer AppStream 2.0.

1. Inicie Enterprise Developer desde el menú de incio de Windows. Elija Micro Focus Enterprise Developer y, a continuación, Enterprise Developer for Eclipse. Si es la primera vez que lo inicia, es posible que tarde algún tiempo.
2. En el lanzador de Eclipse, en el espacio de trabajo: introduzca `C:\Users\<username>\workspace` y seleccione Lanzar.

   > **Note**
   >
   > Asegúrese de elegir la misma ubicación después de volver a conectarse a la instancia AppStream 2.0. La selección del espacio de trabajo no es persistente.

3. En Welcome, elija Open COBOL Perspective. Esto solo se mostrará la primera vez en un espacio de trabajo nuevo.

Clona el BankDemo CodeCommit repositorio en Enterprise Developer

1. Elija Window / Perspective / Open Perspective / Other ... / Git.
2. Elija Clone a Git repository.
3. En Clone Git Repository, introduzca la siguiente información:

   - En el URI de ubicación, introduzca la URL HTTPS del CodeCommit repositorio.

     > **Note**
     >
     > Copie la URL HTTPS del clon del CodeCommit repositorio en la consola AWS de administración y péguela aquí. La URI se dividirá en las rutas de Host y Repository.

   - El usuario CodeCommit deposita las credenciales en Authentication User and Password y selecciona Guardar en almacenamiento seguro.
4. En Branch Selection, elija Main branch y, a continuación, Next .
5. En Destino local, en Directorio, ingresa `C:\Users\<username>\workspace` y selecciona Finalizar.

El proceso de clonación finaliza cuando `BANKDEMO [main]` se muestra en la vista Repositorios de Git.

Cree un proyecto COBOL para BankDemo mainframe y cree una aplicación

1. Cambie a COBOL Perspective.
2. En Project, desactive Build Automatically.
3. En File, seleccione New y, a continuación, Mainframe COBOL Project.
4. En New Mainframe COBOL Project, introduzca la siguiente información:
   - En Project name, introduzca `BankDemo`.
   - Elija la Micro Focus template [64 bit]
   - Elija Finish.
5. En COBOL Explorer, amplíe el nuevo proyecto. BankDemo

   > **Note**
   >
   > `[BANKDEMO main]`entre corchetes indica que el proyecto está conectado con el BankDemo CodeCommit repositorio local.

6. Si la vista en árbol no muestra las entradas de los programas COBOL, los libros de texto, el código fuente de BMS y los archivos JCL, seleccione Actualizar en el menú contextual del BankDemo proyecto.
7. BankDemo En el menú contextual, seleccione Propiedades/Micro Focus/Configuración del proyecto/COBOL:
   - Seleccione Character Set - ASCII.
   - Elija Apply y, después, Close.
8. Si la compilación del origen BMS y COBOL no se inicia inmediatamente, compruebe en el menú Project que la opción Build Automatically esté habilitada.

   La salida de la compilación se mostrará en la vista de la consola y lo normal es que se complete después de unos minutos con los mensajes `BUILD SUCCESSFUL` y `Build finished with no errors`.

   La BankDemo aplicación debería estar ahora compilada y lista para su ejecución local.

Cree un BankDemo CICS local y un entorno por lotes para las pruebas

1. En COBOL Explorer, expanda `BANKDEMO / config`.
2. En el editor, abra `BANKDEMO_ED.json`.
3. Busque la cadena `ED_Home=` y cambie la ruta para que apunte al proyecto de Enterprise Developer, de la siguiente manera: `D:\\<username>\\workspace\\BANKDEMO`. Observe el uso de barras dobles (`\\`) en la definición de la ruta.
4. Guarde y cierre el archivo.
5. Elija Server Explorer.
6. En el menú contextual predeterminado, elija Open Administration Page. La página de administración del servidor Micro Focus Enterprise se abre en el navegador predeterminado.
7. Solo para las sesiones AppStream 2.0, realice los siguientes cambios para poder conservar su región local de Enterprise Server para las pruebas locales:

   - En Directory Server / Default, elija PROPERTIES / Configuration.
   - Sustituya la Repository Location por `D:\<username>\My Files\Home Folder\MFDS`.

   > **Note**
   >
   > Debe completar los pasos 5 a 8 después de cada nueva conexión a una instancia AppStream 2.0.

8. En Directory Server / Default, elija Import y, a continuación, complete los siguientes pasos:

   - En Step 1: Import Type, elija JSON y Next.
   - En Step 2: Upload , haga clic para cargar el archivo del cuadrado azul.
   - En Choose File to Upload, introduzca:
     - File name: `D:\<username>\workspace\BANKDEMO\config\BANKDEMO_ED.json`.
     - Elija Open.
   - Elija Next.
   - En Step 3: Regions elimine Clear Ports from Endpoints.
   - Elija Siguiente.
   - En Step 4: Import, elija Import.
   - Seleccione Finalizar.

La lista mostrará ahora un nuevo nombre de servidor `BANKDEMO`.

Inicie el servidor de BANKDEMO desde Enterprise Developer

1. Elija Enterprise Developer.
2. En el explorador de servidores, elija Default y, a continuación, elija Refresh en el menú contextual.

   La lista de servidores ahora también debería mostrar BANKDEMO.
3. Elija BANKDEMO.
4. En el menú contextual, elija Associate with project y, a continuación, elija BANKDEMO.
5. En el menú contextual, elija Start.

   La vista de la consola debería mostrar el registro del inicio del servidor.

   Si aparece el mensaje `BANKDEMO CASSI5030I PLTPI Phase 2 List(PI) Processing Completed`, el servidor está listo para probar la aplicación BANKDEMO de CICS.

Inicie el terminal Rumba 3270

1. Desde el menú de inicio de Windows, inicie Micro Focus Rumba+ Desktop o Rumba+ Desktop.
2. En Welcome, elija CREATE NEW SESSION / Mainframe Display.
3. En Mainframe Display, elija Connection / Configure.
4. En Configuración de sesión, elija Connection/ TN3270.
5. En Host Name / Address, elija Insert and enter IP address `127.0.0.1`.
6. En Telnet Port, introduzca el puerto `6000`.
7. Seleccione Apply.
8. Elija Connect.

   La pantalla de bienvenida del CICS muestra una pantalla con el mensaje de la fila 1: `This is the Micro Focus MFE CICS region BANKDEMO`.
9. Pulse Ctrl+Shift+z para borrar la pantalla.

Ejecute una BankDemo transacción

1. En una pantalla vacía, introduzca `BANK`.
2. En la pantalla BANK10, en el campo de entrada del seudónimo... :, introduzca `guest` y pulse Entrar.
3. En la pantalla BANK20, en el campo de entrada situado antes de Calcular el coste de un préstamo, introduzca `/` (barra inclinada) y pulse Entrar.
4. En la pantalla BANK70:

   - En The amount you would like to borrow...:, introduzca `10000`.
   - En At an interest rate of.....:, introduzca `5.0`.
   - En For how many months.....:, introduzca `10`.
   - Pulse Intro.

   Se debe mostrar el siguiente resultado:

   ```
   Resulting monthly payment.............:   $1023.06
   ```

   Esto completa la configuración de la aplicación de BANKDEMO en Enterprise Developer.

Detenga el servidor de BANKDEMO desde Enterprise Developer

1. En el explorador de servidores, elija Default y, a continuación, elija Refresh en el menú contextual.
2. Elija BANKDEMO.
3. En el menú contextual, elija Stop.

   La vista de la consola debería mostrar el registro de la detención del servidor.

   Si se muestra el mensaje `Server: BANKDEMO stopped successfully`, el servidor se ha apagado correctamente.

Ejercicio 1: Mejorar el cálculo de los préstamos en la aplicación de BANKDEMO

Temas

- Agregue la regla de análisis de préstamos a Enterprise Developer Code Analysis
- Paso 1: Realizar un análisis de código para calcular el préstamo
- Paso 2: Modificar el mapa CICS BMS y el programa COBOL y pruébelo
- Paso 3: Agregar el cálculo del importe total en el programa COBOL
- Paso 4: Confirmar los cambios y ejecutar la canalización de CI/CD

En este escenario, explicará el proceso de realizar un ejemplo de cambio en el código, implementarlo y probarlo.

El departamento de préstamos quiere que aparezca un nuevo campo en la pantalla BANK7 0 de cálculo del préstamo para mostrar el importe total del préstamo. Para ello, es necesario cambiar la pantalla MBANK7 0.CBL del BMS y añadir un nuevo campo y el correspondiente programa de gestión de pantalla SBANK7 0P.CBL con los cuadernos correspondientes. Además, la rutina de cálculo de préstamos de BBANK7 0P.CBL debe ampliarse con la fórmula adicional.

Para completar este ejercicio, asegúrese de completar los siguientes requisitos previos.

- Descargue BANKDEMO-exercise.zip en `D:\PhotonUser\My Files\Home Folder`.
- Descomprima el archivo .zip en `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise`.
- Crea la carpeta `D:\PhotonUser\My Files\Home Folder\AnalysisRules`.
- Copie el archivo de reglas `Loan+Calculation+Update.General-1.xml` de la carpeta `BANKDEMO-exercise` a `D:\PhotonUser\My Files\Home Folder\AnalysisRules`.

> **Note**
>
> Para este ejercicio, los cambios de código en *.CBL y *.CPY se marcan con EXER01 en las columnas 1 a 6.

Agregue la regla de análisis de préstamos a Enterprise Developer Code Analysis

Las reglas de análisis definidas en Rocket Enterprise Analyzer se pueden exportar desde Enterprise Analyzer e importar a Enterprise Developer para ejecutar las mismas reglas de análisis en todas las fuentes del proyecto Enterprise Developer.

1. Abra `Window/Preferences/Micro Focus/COBOL/Code Analysis/Rules`.
2. Elija Edit... e introduzca el nombre de la carpeta `D:\PhotonUser\My Files\Home Folder\AnalysisRules` que contiene el archivo de reglas `Loan+Calculation +Update.General-1.xml`.
3. Seleccione Finalizar.
4. Seleccione Apply y, a continuación, seleccione Close.
5. En el menú contextual del proyecto de BANKDEMO, seleccione Code Analysis.

   Debería ver una entrada para Loan Calculation Update.

Paso 1: Realizar un análisis de código para calcular el préstamo

Con la nueva regla de análisis, queremos identificar los programas COBOL y las líneas de código que contienen que coinciden con los patrones de búsqueda `*PAYMENT*` `*LOAN*` y `*RATE*` en las expresiones, instrucciones y variables. Esto nos ayudará a navegar por el código e identificar los cambios de código necesarios.

1. En el menú contextual del proyecto de BANKDEMO, seleccione Code Analysis/Loan Calculation Update.

   Esto ejecutará la regla de búsqueda y mostrará los resultados en una nueva pestaña llamada Code Analysis. La ejecución del análisis finaliza cuando desaparece la barra de progreso verde situada en la parte inferior derecha.

   La pestaña Code Analysis debería mostrar una lista expandida de `BBANK20P.CBL`, `BBANK70P.CBL` y `SBANK70P.CBL`, cada uno con un listado de las instrucciones, expresiones y variables que coincidan con los patrones de búsqueda.

   Al observar el resultado de `BBANK20P.CBL`, solo se mueven los literales que coinciden con el patrón de búsqueda. Por lo tanto, este programa se puede ignorar.
2. En la barra de menú de pestañas, elija - Icon para colapsar todo.
3. Expanda `SBANK70P.CBL` y seleccione las líneas en cualquier orden haciendo doble clic para ver cómo se abrirá el origen y se resaltará la línea seleccionada en el código fuente. También verá que todas las líneas fuente identificadas están marcadas.

Paso 2: Modificar el mapa CICS BMS y el programa COBOL y pruébelo

Primero cambiaremos el mapa del BMS `MBANK70.BMS`, el programa de gestión de pantalla `SBANK70P.CBL` y el cuaderno `CBANKDAT.CPY` para mostrar el nuevo campo. Para evitar codificación innecesaria en este ejercicio, los módulos de origen modificados están disponibles en la carpeta `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01`. Normalmente, un desarrollador utilizaría los resultados del análisis de código para navegar y modificar los orígenes. Si tiene tiempo y desea realizar los cambios manuales, hágalo con la información proporcionada en *Cambio manual en MBANK7 0.BMS y SBANK7 0P.CBL (opcional) *.

Para realizar cambios rápidos, copie los siguientes archivos:

1. `..\BANKDEMO-exercise\Exercis01\screens\MBANK70.BMS` a `D:\PhotonUser\workspace\bankdemo\source\screens`.
2. `.\BANKDEMO-exercise\Exercis01\cobol\SBANK70P.CBL` a `D:\PhotonUser\workspace\bankdemo\source\cobol`.
3. `..\BANKDEMO-exercise\Exercis01\copybook\CBANKDAT.CPY` a `D:\PhotonUser\workspace\bankdemo\source\copybook`.
4. Para asegurarse de que se compilan todos los programas afectados por los cambios, elija todos los proyectos. Project/Clean.../Clean

Para realizar cambios manuales en `MBANK70.BMS` y`SBANK70P.CBL`, complete los siguientes pasos:

- Para cambiar manualmente el origen `MBANK70.BMS` del BMS, añada después del campo `PAYMENT`:
  - TXT09 con los mismos atributos que el TXT08 y el valor INITIAL “Total Loan Amount”
  - TOTAL con los mismos atributos que PAYMENT

Pruebe los cambios

Para probar los cambios, repita los pasos de las siguientes secciones:

1. Inicie el servidor de BANKDEMO desde Enterprise Developer
2. Inicie el terminal Rumba 3270
3. Ejecute una BankDemo transacción

Además, ahora también debería ver el texto `Total Loan Amount.....................:`.

4. Detenga el servidor de BANKDEMO desde Enterprise Developer

Paso 3: Agregar el cálculo del importe total en el programa COBOL

En el segundo paso, cambiaremos `BBANK70P.CBL` y sumaremos el cálculo del importe total del préstamo. El origen preparado con los cambios necesarios está disponible en la carpeta `D:\PhotonUser\My Files\Home Folder\BANKDEMO-exercise\Exercise01`. Si tiene tiempo y desea realizar los cambios manuales, hágalo con la información proporcionada en *Cambio manual en BBANK7 0P.CBL (opcional) *.

Para realizar cambios rápidos, copie el siguiente archivo:

- `..\BANKDEMO-exercise\Exercis01\source\cobol\BBANK70P.CBL` a `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Para realizar un cambio manual en `BBANK70P.CBL`, complete los siguientes pasos:

- Utilice el resultado del análisis de código para identificar los cambios necesarios.

Pruebe los cambios

Para probar los cambios, repita los pasos de las siguientes secciones:

1. Inicie el servidor de BANKDEMO desde Enterprise Developer
2. Inicie el terminal Rumba 3270
3. Ejecute una BankDemo transacción

   Además, ahora también debería ver el texto `Total Loan Amount.....................: $10230.60`.
4. Detenga el servidor de BANKDEMO desde Enterprise Developer

Paso 4: Confirmar los cambios y ejecutar la canalización de CI/CD

Guarde los cambios en el repositorio central CodeCommit y active el CI/CD canalización para crear, probar e implementar los cambios.

1. En el menú contextual del proyecto de BANKDEMO, seleccione Team/Commit.
2. En la pestaña Git Staging, introduzca el siguiente mensaje de confirmación: `Added Total Amount Calculation`.
3. Elija Commit and Push....
4. Abre la CodePipeline consola y comprueba el estado de la ejecución de la canalización.

> **Note**
>
> En caso de que tenga algún problema con la característica Commit o Push de Enterprise Developer o Teams, utilice la interfaz de línea de comandos de Git Bash.

Ejercicio 2: Extraiga el cálculo del préstamo en BankDemo la solicitud

Temas

- Paso 1: Refactorizar la rutina de cálculo de préstamos en una sección de COBOL
- Paso 2: Extraer la rutina de cálculo de préstamos a un programa COBOL independiente
- Paso 3: Confirma los cambios y ejecuta el CI/CD canalización

En el siguiente ejercicio, analizará otro ejemplo de solicitud de cambio. En este escenario, el departamento de préstamos quiere reutilizar la rutina de cálculo de préstamos de forma independiente. WebService La rutina debería permanecer en COBOL y también debería poder invocarse desde el programa CICS COBOL existente `BBANK70P.CBL`.

Paso 1: Refactorizar la rutina de cálculo de préstamos en una sección de COBOL

En el primer paso, extraemos la rutina de cálculo del préstamo en una sección de COBOL. Este paso es necesario para extraer el código en un programa COBOL independiente en el siguiente paso.

1. Abra `BBANK70P.CBL` en el editor de COBOL.
2. En el menú contextual del proyecto de BANKDEMO, seleccione Code Analysis/Loan Calculation Update. Esto solo escaneará el origen actual en busca de los patrones definidos en la regla de análisis.
3. En el resultado, en la pestaña Code Analysis, busque la primera instrucción aritmética `DIVIDE WS-LOAN-INTEREST BY 12`.

4. Haga doble clic en la instrucción para ir a la línea de origen en el editor. Esta es la primera instrucción de la rutina de cálculo de préstamos.
5. Marque el siguiente bloque de código para extraer la rutina de cálculo de préstamos en una sección.

```
DIVIDE WS-LOAN-INTEREST BY 12
             GIVING WS-LOAN-INTEREST ROUNDED.
           COMPUTE WS-LOAN-MONTHLY-PAYMENT ROUNDED =
             ((WS-LOAN-INTEREST * ((1 + WS-LOAN-INTEREST)
                 ** WS-LOAN-TERM)) /
             (((1 + WS-LOAN-INTEREST) * WS-LOAN-TERM) - 1 ))
                 * WS-LOAN-PRINCIPAL.
EXER01     COMPUTE WS-LOAN-TOTAL-PAYMENT =
EXER01               (WS-LOAN-MONTHLY-PAYMENT * WS-LOAN-TERM).
```

6. En el menú contextual del editor, seleccione Refactor/Extract to Section....
7. Introduzca el New section name: LOAN-CALCULATION.
8. Seleccione Aceptar.

   El bloque de código marcado se ha extraído ahora a la nueva sección `LOAN-CALCULATION` y el bloque de código se ha sustituido por la instrucción `PERFROM LOAN-CALCULATION`.

Pruebe los cambios

Para probar los cambios, repita los pasos de las siguientes secciones:

1. Inicie el servidor de BANKDEMO desde Enterprise Developer
2. Inicie el terminal Rumba 3270
3. Ejecute una BankDemo transacción

   Además, ahora también debería ver el texto `Total Loan Amount.....: $10230.60`.
4. Detenga el servidor de BANKDEMO desde Enterprise Developer

> **Note**
>
> Si quiere evitar los pasos anteriores para extraer el bloque de código a una sección, puede copiar el origen modificado del paso 1 de `..\BANKDEMO-exercise`

\Exercis02\Step1\cobol\BBANK70P.CBL a D:\PhotonUser\workspace \bankdemo\source\cobol.

Paso 2: Extraer la rutina de cálculo de préstamos a un programa COBOL independiente

En el paso 2, el bloque de código de la sección LOAN-CALCULATION se extraerá a un programa independiente y el código original se sustituirá por el código para llamar al nuevo subprograma.

1. Abra BBANK70P.CBL en el editor y busque la nueva instrucción PERFORM LOAN-CALCULATION creada en el paso 1.
2. Coloque el cursor dentro del nombre de la sección. Se marcará en gris.
3. En el menú contextual, seleccione Refactor->Extract Section/Paragraph to Program....
4. En Extract Section/Paragraph to Program, introduzca New file name: LOANCALC.CBL.
5. Seleccione OK.

   El nuevo programa LOANCALC.CBL se abrirá en el editor.
6. Desplácese hacia abajo y revise el código que se está extrayendo y generando para la interfaz de llamada.
7. Seleccione el editor con BBANK70P.CBL y vaya a LOAN-CALCULATION SECTION. Revise el código que se está generando para llamar al nuevo subprograma LOANCALC.CBL.

> **Note**
>
> La instrucción CALL utiliza DFHEIBLK y DFHCOMMAREA para llamar a LOANCALC con bloques de control de CICS. Como queremos denominar al nuevo subprograma LOANCALC.CBL como programa ajeno al CICS, tenemos que eliminar DFHEIBLK y DFHCOMMAREA de la llamada comentando o borrando.

Pruebe los cambios

Para probar los cambios, repita los pasos de las siguientes secciones:

1. Inicie el servidor de BANKDEMO desde Enterprise Developer
2. Inicie el terminal Rumba 3270
3. Ejecute una BankDemo transacción

Además, ahora también debería ver el texto `Total Loan Amount.....................: $10230.60`.

4. Detenga el servidor de BANKDEMO desde Enterprise Developer

> **Note**
>
> Si quiere evitar los pasos anteriores para extraer el bloque de código a una sección, puede copiar el origen modificado del paso 1 de `..\BANKDEMO-exercise\Exercis02\Step2\cobol\BBANK70P.CBL` y `LOANCALC.CBL` a `D:\PhotonUser\workspace\bankdemo\source\cobol`.

Paso 3: Confirma los cambios y ejecuta el CI/CD canalización

Confirme los cambios en el CodeCommit repositorio central y active el CI/CD canalización para crear, probar e implementar los cambios.

1. En el menú contextual del proyecto de BANKDEMO, seleccione Team/Commit.
2. En la pestaña Git Staging
   - Añada Unstaged Stages LOANCALC.CBL y LOANCALC.CBL.mfdirset.
   - Introduzca un mensaje de confirmación: `Added Total Amount Calculation`.
3. Elija Commit and Push....
4. Abre la CodePipeline consola y comprueba el estado de la ejecución de la canalización.

> **Note**
>
> En caso de que tenga algún problema con la característica Commit o Push de Enterprise Developer o Teams, utilice la interfaz de línea de comandos de Git Bash.

Limpieza de recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para que no se le siga cobrando por ellos. Realice los siguientes pasos:

- Elimine la CodePipeline canalización. Para obtener más información, consulte Eliminar una canalización CodePipeline en la Guía del AWS CodePipeline usuario.
- Elimine el CodeCommit repositorio. Para obtener más información, consulte Eliminar un CodeCommit repositorio en la Guía del AWS CodeCommit usuario.
- Elimine el bucket de S3. Para obtener más información, consulte Eliminar un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Elimine la AWS CloudFormation pila. Para obtener más información, consulte Eliminar una pila de la AWS CloudFormation consola en la Guía del AWS CloudFormation usuario.

Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer

AWS La modernización del mainframe proporciona varias herramientas a través de Amazon AppStream 2.0. AppStream 2.0 es un servicio de streaming de aplicaciones seguro y totalmente gestionado que permite transmitir aplicaciones de escritorio a los usuarios sin tener que volver a escribirlas. AppStream La versión 2.0 proporciona a los usuarios acceso instantáneo a las aplicaciones que necesitan, con una experiencia de usuario fluida y con gran capacidad de respuesta en el dispositivo que elijan. El uso de la AppStream versión 2.0 para alojar herramientas específicas del motor de ejecución ofrece a los equipos de aplicaciones de los clientes la posibilidad de utilizar las herramientas directamente desde sus navegadores web e interactuar con los archivos de las aplicaciones almacenados en depósitos o repositorios de Amazon S3. CodeCommit

Para obtener información sobre la compatibilidad de los navegadores en la AppStream versión 2.0, consulte System Requirements and Feature Support (navegador web) en la Guía de administración de Amazon AppStream 2.0. Si tiene problemas al utilizar la AppStream versión 2.0, consulte Solución de problemas de usuario de la AppStream versión 2.0 en la Guía de administración de Amazon AppStream 2.0.

Este documento está dirigido a los miembros del equipo de operaciones del cliente. Describe cómo configurar las flotas y pilas de Amazon AppStream 2.0 para alojar las herramientas Rocket Enterprise Analyzer y Rocket Enterprise Developer que se utilizan con AWS Mainframe Modernization. Rocket Enterprise Analyzer se suele utilizar durante la fase de evaluación y Rocket Enterprise Developer se suele utilizar durante la fase de migración y modernización del enfoque de modernización del mainframe. AWS Si piensa utilizar Enterprise Analyzer y Enterprise Developer, debe crear flotas y pilas independientes para cada herramienta. Cada herramienta requiere su propia flota y su propia pila, ya que las condiciones de licencia son diferentes.

> **Important**
>
> Los pasos de este tutorial se basan en la AWS CloudFormation plantilla descargable cfn-m2-.yml. appstream-fleet-ea-ed

Temas

- Requisitos previos
- Paso 1: Obtenga las imágenes AppStream 2.0
- Paso 2: Crea la pila con la AWS CloudFormation plantilla
- Paso 3: Crea un usuario en la AppStream versión 2.0
- Paso 4: Inicie sesión en AppStream 2.0
- Paso 5: Verificar los buckets de Amazon S3 (opcional)
- Pasos a seguir a continuación
- Eliminar recursos

Requisitos previos

- Descargue la plantilla: cfn-m2-.yml. appstream-fleet-ea-ed
- Obtenga el ID de la VPC y el grupo de seguridad predeterminados. Para obtener más información sobre la VPC predeterminada, consulte Predeterminado VPCs en la Guía del usuario de Amazon VPC. Para obtener más información sobre el grupo de seguridad predeterminado, consulte Grupos de seguridad predeterminados y personalizados en la Guía del EC2 usuario de Amazon.
- Asegúrese de que tiene permisos para:
  - cree pilas, flotas y usuarios en AppStream la versión 2.0.
  - cree pilas AWS CloudFormation utilizando una plantilla.
  - Crear buckets y cargar archivos a buckets de Amazon S3.
  - Descargar credenciales (`access_key_id` y `secret_access_key`) de IAM.

Paso 1: Obtenga las imágenes AppStream 2.0

En este paso, comparte las imágenes AppStream 2.0 de Enterprise Analyzer y Enterprise Developer con su AWS cuenta.

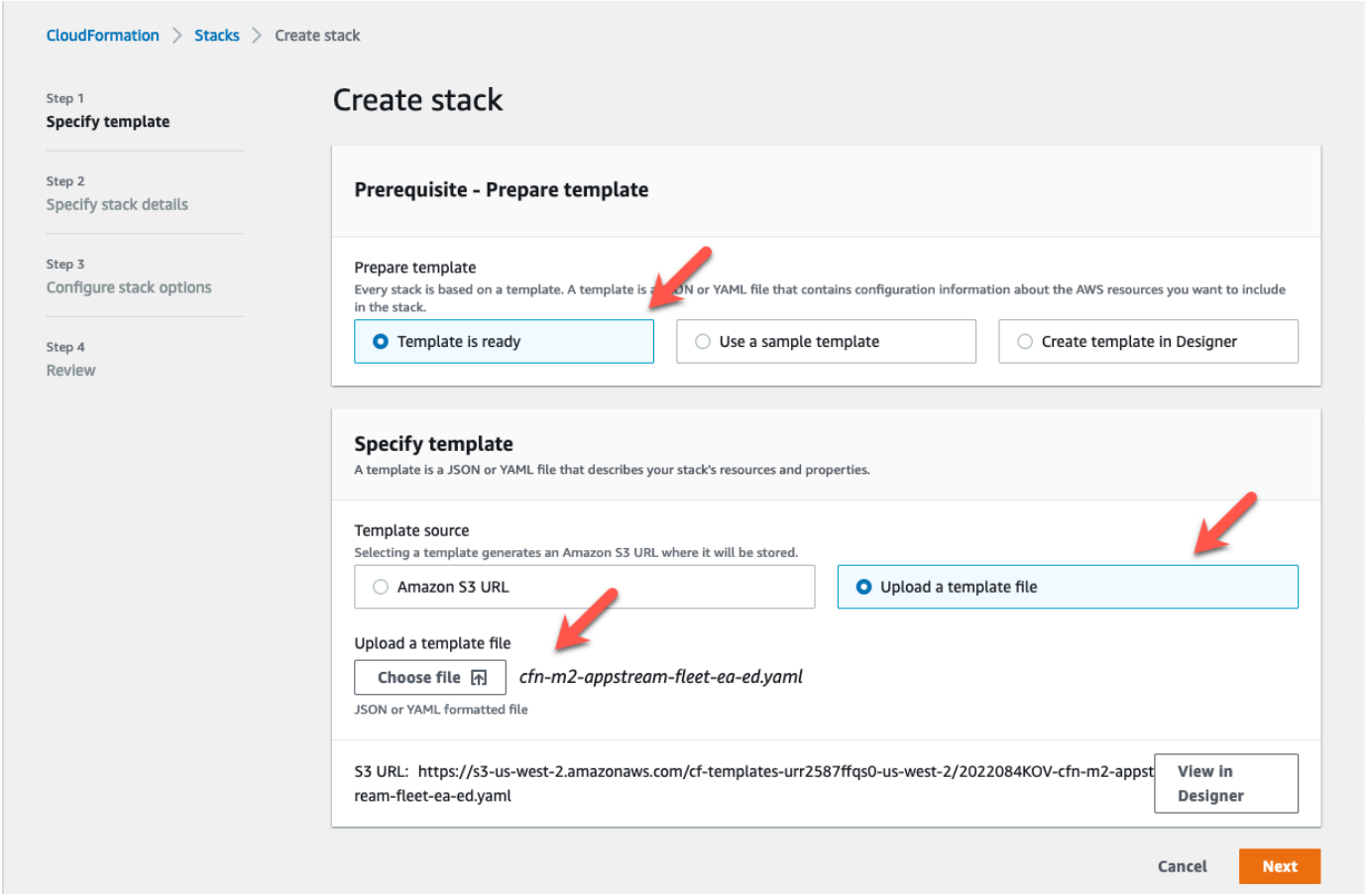
1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el panel de navegación izquierdo, seleccione Herramientas.
3. En Análisis, desarrollo y creación de activos, seleccione Compartir activos con mi AWS cuenta.

Paso 2: Crea la pila con la AWS CloudFormation plantilla

En este paso, utilizarás la AWS CloudFormation plantilla descargada para crear una pila AppStream 2.0 y una flota para ejecutar Rocket Enterprise Analyzer. Puedes repetir este paso más adelante para crear otra pila y flota AppStream 2.0 para ejecutar Rocket Enterprise Developer, ya que cada herramienta requiere su propia flota y pila en la AppStream versión 2.0. Para obtener más información sobre las AWS CloudFormation pilas, consulta Cómo trabajar con pilas en la Guía del AWS CloudFormation usuario.

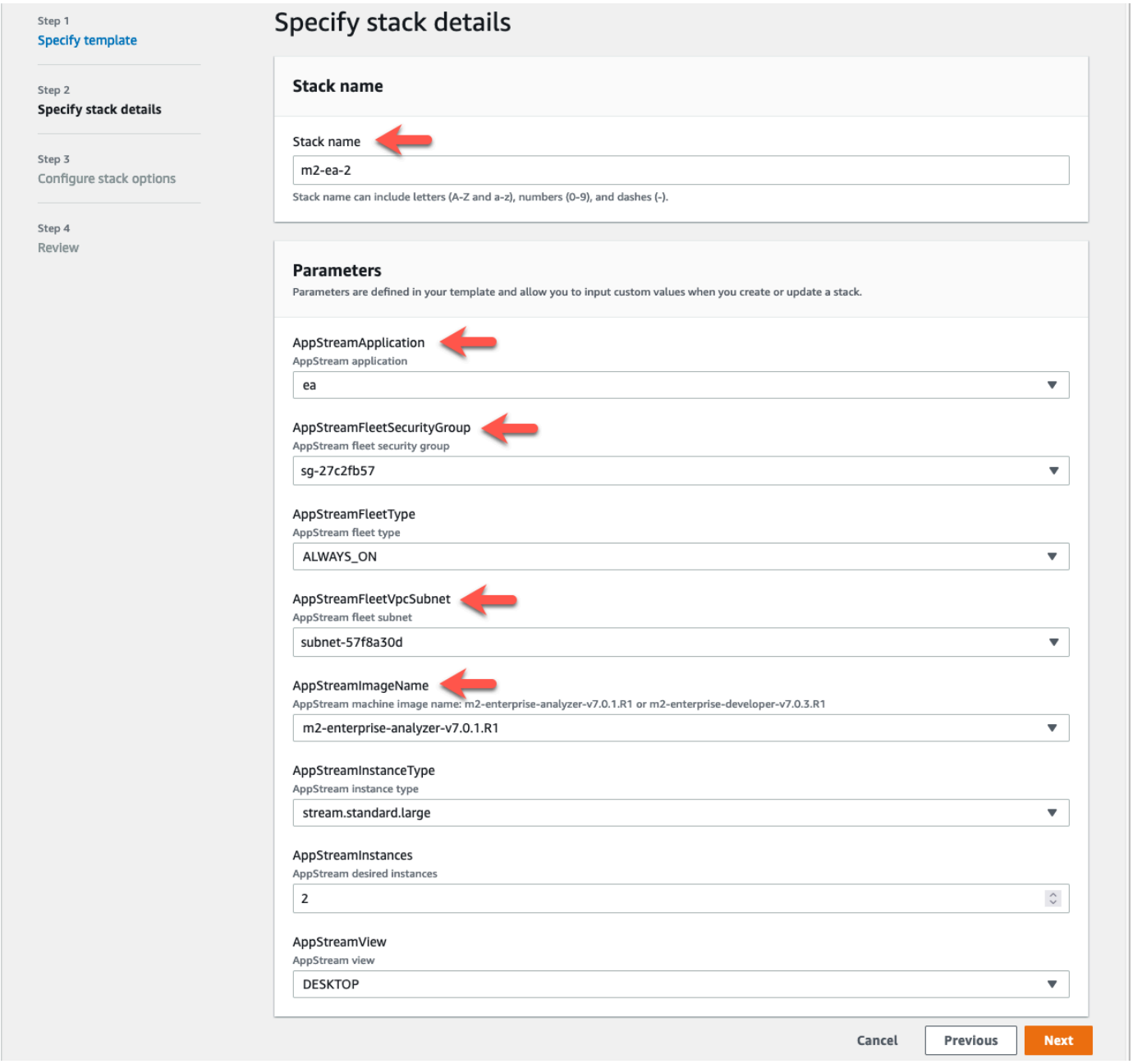
> **Note**
>
> AWS La modernización de mainframe añade una tarifa adicional al precio estándar de la AppStream versión 2.0 por el uso de Enterprise Analyzer y Enterprise Developer. Para obtener más información, consulte AWS Mainframe Modernization Pricing.

1. Descargue la plantilla appstream-fleet-ea-edcfn-m2-.yml, si es necesario.
2. Abre la AWS CloudFormation consola y selecciona Crear pila y con recursos nuevos (estándar).
3. En Requisito previo: preparar la plantilla, elija La plantilla está lista.
4. En Especificar plantilla, elija Cargar un archivo de plantilla.
5. En Cargar un archivo de plantilla, selecciona Elegir archivo y carga la plantilla appstream-fleet-ea-edcfn-m2-.yml.
6. Elija Siguiente.



7. En Especificar los detalles de la pila, introduzca la siguiente información:

   - En Nombre de pila, introduzca un nombre de su elección. Por ejemplo, **`m2-ea`**.
   - En AppStreamApplication, elige ea.
   - En AppStreamFleetSecurityGroup, elige el grupo de seguridad predeterminado de tu VPC predeterminada.
   - En AppStreamFleetVpcSubnet, elige una subred dentro de tu VPC predeterminada.
   - En AppStreamImageName, elige la imagen que empiece por. `m2-enterprise-analyzer` Esta imagen contiene la versión actualmente compatible de la herramienta Rocket Enterprise Analyzer.
   - Acepte los valores predeterminados para el resto de los campos y después elija Siguiente.



8. Acepte los valores predeterminados y, a continuación, elija Siguiente.
9. En Revisar, asegúrese de que todos los parámetros son los que desea.
10. Desplázate hasta el final, selecciona Acepto que AWS CloudFormation podría crear recursos de IAM con nombres personalizados y selecciona Create Stack.

La pila y la flota tardan entre 20 y 30 minutos en crearse. Puede elegir Actualizar para ver los AWS CloudFormation eventos a medida que se producen.

Paso 3: Crea un usuario en la AppStream versión 2.0

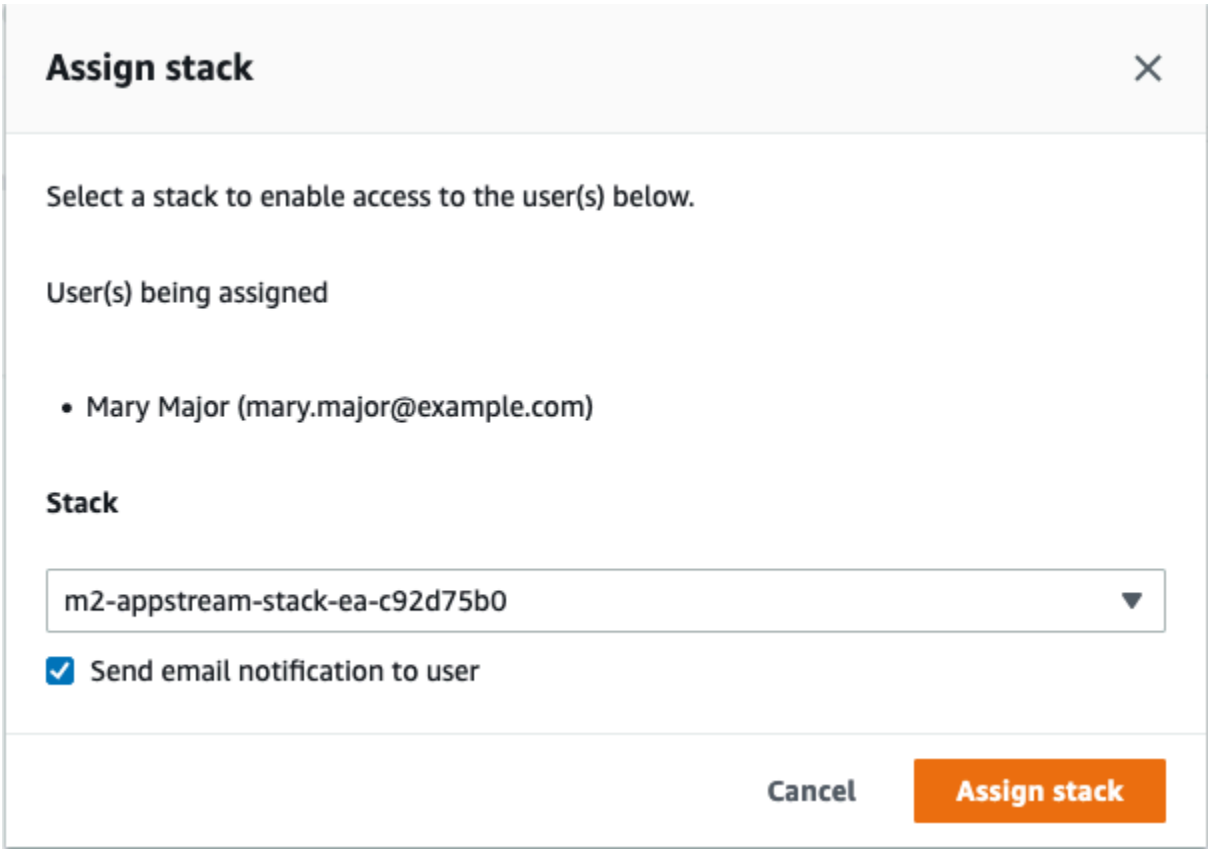
Mientras esperas AWS CloudFormation a terminar de crear la pila, puedes crear uno o más usuarios en la AppStream versión 2.0. Estos usuarios son los que utilizarán Enterprise Analyzer en la AppStream versión 2.0. Deberá especificar una dirección de correo electrónico para cada usuario y asegurarse de que cada uno tenga los permisos suficientes para crear buckets en Amazon S3, cargar archivos en un bucket y vincular a un bucket para asignar su contenido.

1. Abra la consola AppStream 2.0.
2. En el panel de navegación izquierdo, elija Grupo de usuarios.
3. Seleccione la opción Crear usuario.
4. Proporcione una dirección de correo electrónico en la que el usuario pueda recibir una invitación por correo electrónico para usar la AppStream versión 2.0, un nombre y apellidos, y seleccione Crear usuario.
5. Repita este procedimiento si es necesario para crear más usuarios. La dirección de correo electrónico de cada usuario debe ser única.

Para obtener más información sobre la creación de usuarios AppStream 2.0, consulte Grupos de usuarios AppStream 2.0 en la Guía de administración de Amazon AppStream 2.0.

Cuando AWS CloudFormation termine de crear la pila, puede asignar el usuario que creó a la pila de la siguiente manera:

1. Abre la consola AppStream 2.0.
2. Elija el nombre de usuario.
3. Elija Acciones y luego Asignar pila.
4. En Asignar pila, elija la pila que empieza por `m2-appstream-stack-ea`.
5. Elija Asignar pila.



Al asignar un usuario a una pila, AppStream 2.0 envía un correo electrónico al usuario a la dirección que ha proporcionado. Este correo electrónico contiene un enlace a la página de inicio de sesión de AppStream 2.0.

Paso 4: Inicie sesión en AppStream 2.0

En este paso, inicia sesión en AppStream 2.0 mediante el enlace del correo electrónico enviado por AppStream 2.0 al usuario que creóPaso 3: Crea un usuario en la AppStream versión 2.0.

1. Inicie sesión en AppStream 2.0 mediante el enlace que se proporciona en el correo electrónico enviado por AppStream 2.0.
2. Cambie la contraseña si se le solicita. La pantalla AppStream 2.0 que aparece es similar a la siguiente:

3. Seleccione Escritorio.
4. En la barra de tareas, seleccione Bucar e introduzca **`D:`** para navegar hasta la carpeta de inicio.

> **Note**
>
> Si omite este paso, es posible que aparezca el error `Device not ready` al intentar acceder a la carpeta de inicio.

En cualquier momento, si tienes problemas para iniciar sesión en la AppStream versión 2.0, puedes reiniciar tu flota de versiones AppStream 2.0 e intentar iniciar sesión de nuevo siguiendo estos pasos.

1. Abre la consola AppStream 2.0.
2. En el panel de navegación izquierdo, elija Flotas.
3. Elija la flota que está intentando utilizar.
4. Seleccione Acción y, a continuación, Detener.
5. Espere a que la flota se detenga.
6. Seleccione Acción y, a continuación, seleccione Inciar.

Este proceso puede tardar unos 10 minutos.

Paso 5: Verificar los buckets de Amazon S3 (opcional)

Una de las tareas que llevó a cabo la AWS CloudFormation plantilla que utilizó para crear la pila consistía en crear dos depósitos en Amazon S3, que son necesarios para guardar y restaurar los datos de los usuarios y la configuración de la aplicación en las sesiones de trabajo. Estos buckets son los siguientes:

- Su nombre comienza por `appstream2-`. Este bucket asigna los datos a su carpeta de inicio en la AppStream versión 2.0 (`D:\PhotonUser\My Files\Home Folder`).

  > **Note**
  >
  > La carpeta de inicio es única para una dirección de correo electrónico determinada y se comparte entre todas las flotas y pilas de una cuenta determinada AWS . El nombre de la carpeta principal es un SHA256 hash de la dirección de correo electrónico del usuario y se almacena en una ruta basada en ese hash.

- Su nombre comienza por `appstream-app-settings-`. Este depósito contiene información sobre las sesiones de usuario de la AppStream versión 2.0 e incluye ajustes como los favoritos del navegador, los perfiles de conexión del IDE y de las aplicaciones y las personalizaciones de la interfaz de usuario. Para obtener más información, consulte Cómo funciona la persistencia de la configuración de aplicaciones en la Guía de administración de Amazon AppStream 2.0.

Para comprobar que se han creado los buckets, siga estos pasos:

1. Abra la consola de Amazon S3.
2. En el panel de navegación izquierdo, elija Buckets.
3. En Buscar buckets por nombre, introduzca **`appstream`** para filtrar la lista.

Si ve los buckets, no es necesario realizar ninguna otra acción. Solo es para saber que los buckets existen. Si no ve los cubos, significa que la AWS CloudFormation plantilla no ha terminado de ejecutarse o que se ha producido un error. Ve a la AWS CloudFormation consola y revisa los mensajes de creación de la pila.

Pasos a seguir a continuación

Ahora que la infraestructura AppStream 2.0 está configurada, puede configurar y empezar a utilizar Enterprise Analyzer. Para obtener más información, consulte Tutorial: Configurar Enterprise Analyzer

en 2.0 AppStream . También puede configurar Enterprise Developer. Para obtener más información, consulte Tutorial: Configurar Rocket Enterprise Developer en AppStream 2.0.

Eliminar recursos

El procedimiento para limpiar la pila y las flotas creadas se describe en Crear una flota y una pila AppStream 2.0.

Cuando se hayan eliminado los objetos AppStream 2.0, el administrador de la cuenta también puede, si procede, limpiar los depósitos de Amazon S3 para la configuración de la aplicación y las carpetas de inicio.

> **Note**
>
> La carpeta de inicio de un usuario determinado es única en todas las flotas, por lo que es posible que tengas que conservarla si hay otras pilas AppStream 2.0 activas en la misma cuenta.

Por último, la AppStream versión 2.0 actualmente no permite eliminar usuarios mediante la consola. Para ello, debe usar la API de servicio con la CLI. Para obtener más información, consulte Administración de grupos de usuarios en la Guía de administración de Amazon AppStream 2.0.

Tutorial: Utilice plantillas con Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer)

En este tutorial se describe cómo utilizar plantillas y proyectos predefinidos con Rocket Enterprise Developer. Incluye tres casos de uso. Todos los casos de uso utilizan el código de ejemplo que se proporciona en el BankDemo ejemplo. Para descargar el ejemplo, elija `bankdemo.zip`.

> **Important**
>
> Si utiliza la versión de Enterprise Developer para Windows, los binarios generados por el compilador solo se pueden ejecutar en el Enterprise Server suministrado con Enterprise Developer. No puede ejecutarlos en el entorno de ejecución de modernización del AWS mainframe, que está basado en Linux.

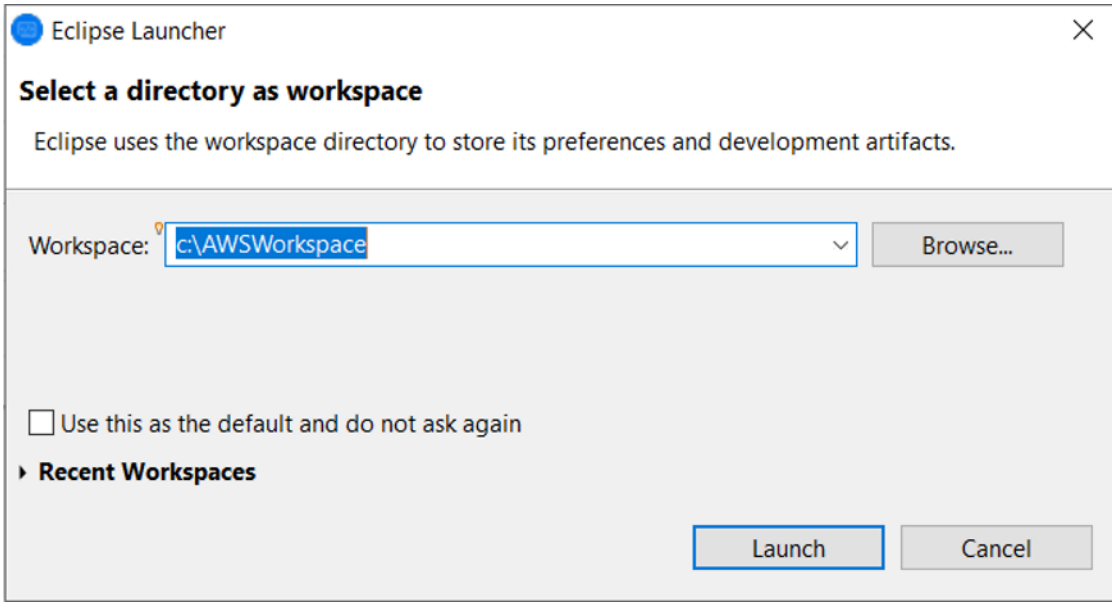
Temas

- Caso de uso 1: Uso de la plantilla de proyecto de COBOL que contiene los componentes de origen
- Caso de uso 2: Uso de la plantilla de proyecto de COBOL sin los componentes de origen
- Caso de uso 3: Uso del proyecto de COBOL predefinido que enlaza con las carpetas de origen
- Uso de la plantilla JSON de definición de región

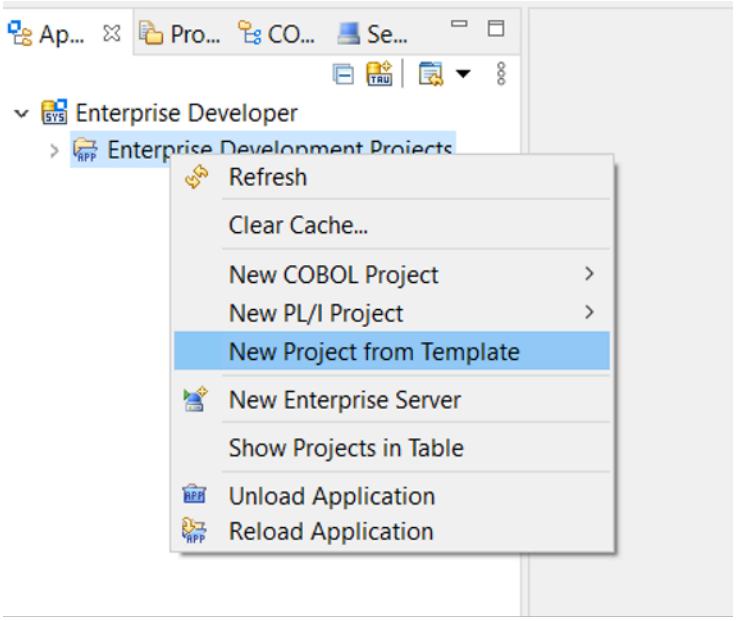
Caso de uso 1: Uso de la plantilla de proyecto de COBOL que contiene los componentes de origen

Este caso de uso requiere que copie los componentes de origen en la estructura de directorios de la plantilla como parte de los pasos de preconfiguración de la demostración. En el `bankdemo.zip`, se ha cambiado con respecto al del `AWSTemplates.zip` original para evitar tener dos copias del origen.

1. Inicie Enterprise Developer y especifique el espacio de trabajo elegido.

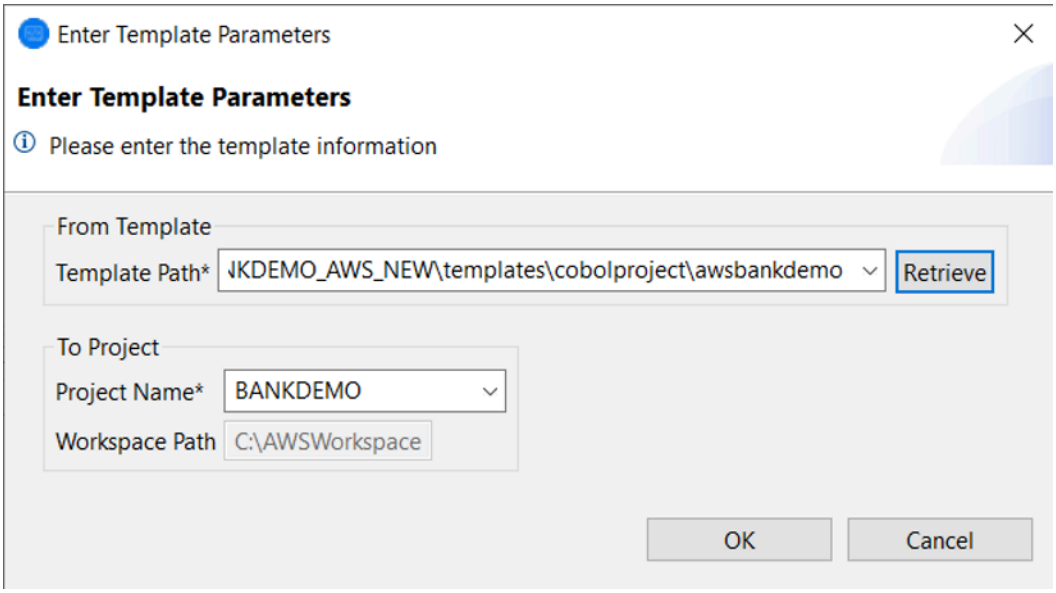


2. En la vista del explorador de aplicaciones, en el elemento de vista en árbol Enterprise Development Project, seleccione New Project from Template en el menú contextual.

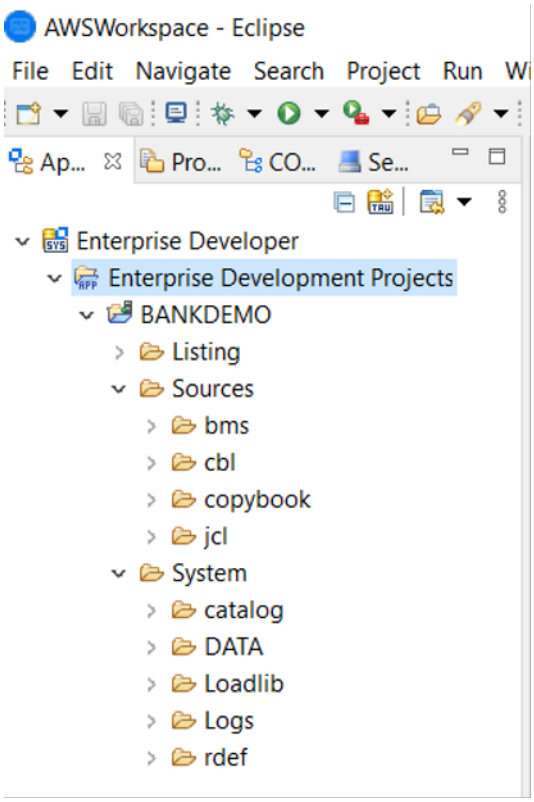


3. Introduzca los parámetros de la plantilla como se muestra.

> **Note**
>
> La ruta de la plantilla hará referencia al lugar donde se haya extraído el ZIP.



4. Si selecciona OK, se creará un proyecto de Eclipse de desarrollo local basado en la plantilla proporcionada, con una estructura completa de origen y entorno de ejecución.



La estructura de `System` contiene un archivo completo de definición de recursos con las entradas requeridas para BANKDEMO, el catálogo requerido con las entradas añadidas y los archivos de datos ASCII correspondientes.

Como la estructura de la plantilla de origen contiene todos los elementos de origen, estos archivos se copian en el proyecto local y, por lo tanto, se incorporan automáticamente a Enterprise Developer.

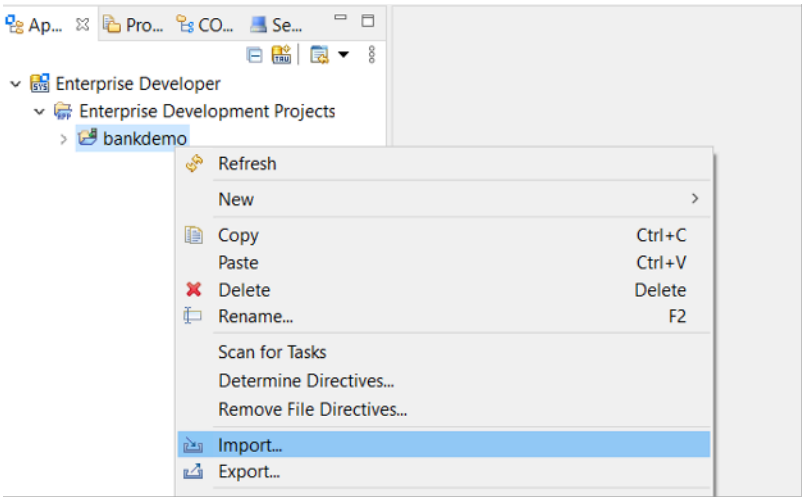
Caso de uso 2: Uso de la plantilla de proyecto de COBOL sin los componentes de origen

Los pasos 1 a 3 son idénticos al Caso de uso 1: Uso de la plantilla de proyecto de COBOL que contiene los componentes de origen.

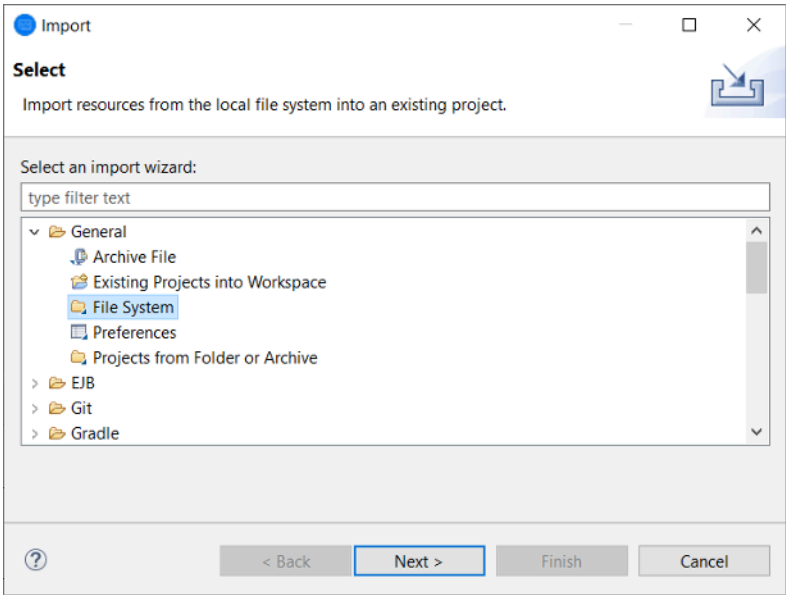
En este caso de uso, la `System` estructura también contiene un archivo completo de definición de recursos con las entradas necesarias BankDemo, el catálogo necesario con las entradas añadidas y los archivos de datos ASCII correspondientes.

Sin embargo, la estructura de origen de la plantilla no contiene ningún componente. Debe importarlos al proyecto desde el repositorio de origen que utilice.

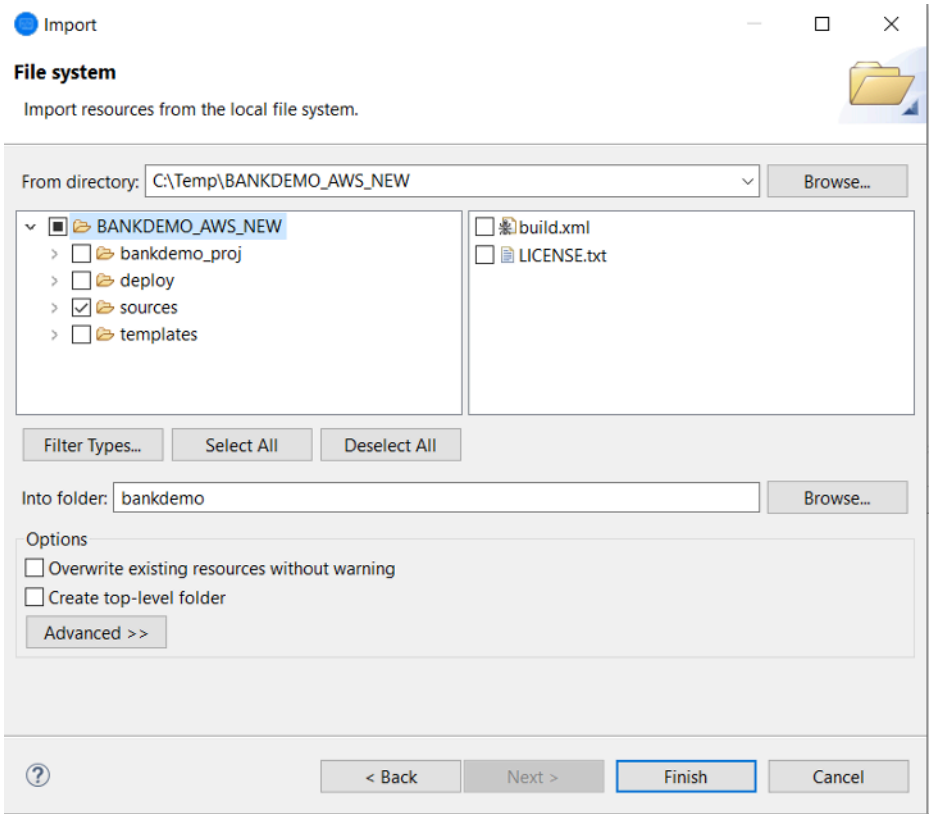
1. Elija el nombre del proyecto. En el menú contextual relacionado, seleccione Import.



2. En el cuadro de diálogo resultante, en la sección General, elija File System y, a continuación, elija Next.



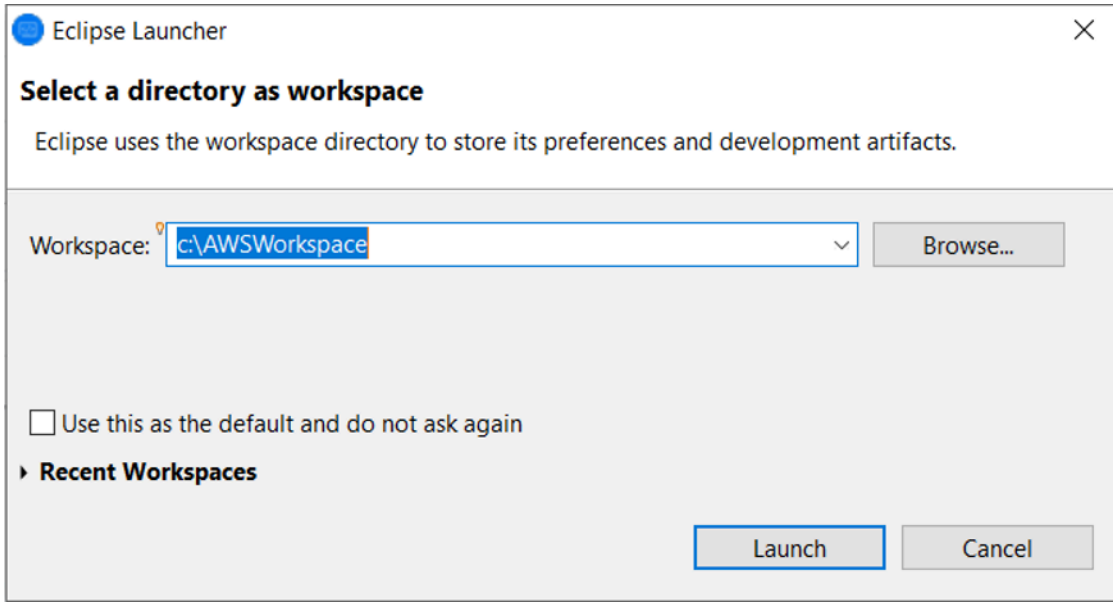
3. Rellene el campo From directory navegando por el sistema de archivos hasta que señale a la carpeta del repositorio. Seleccione todas las carpetas que desee importar, por ejemplo `sources`. El campo `Into folder` se rellenará automáticamente. Elija Finish.



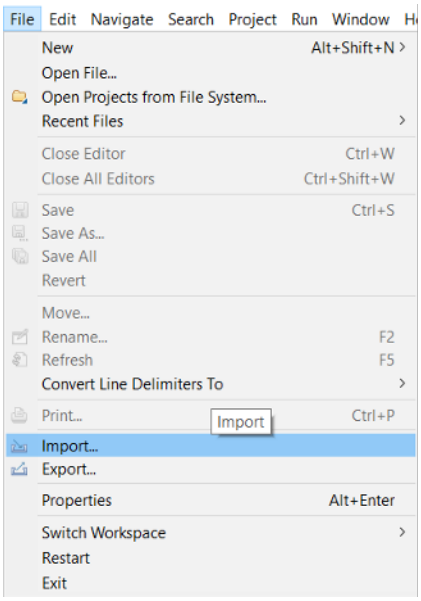
Una vez que la estructura de la plantilla de origen contiene todos los elementos de origen, estos se crean automáticamente en Enterprise Developer.

Caso de uso 3: Uso del proyecto de COBOL predefinido que enlaza con las carpetas de origen

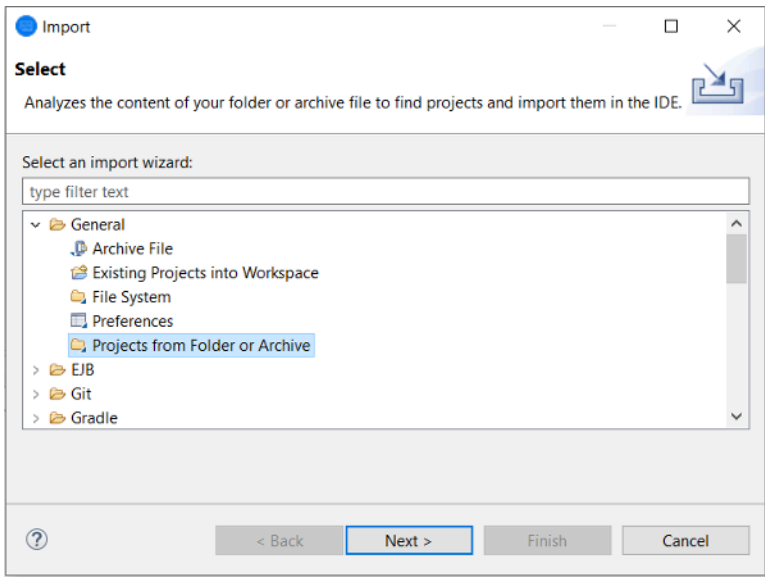
1. Inicie Enterprise Developer y especifique el espacio de trabajo elegido.



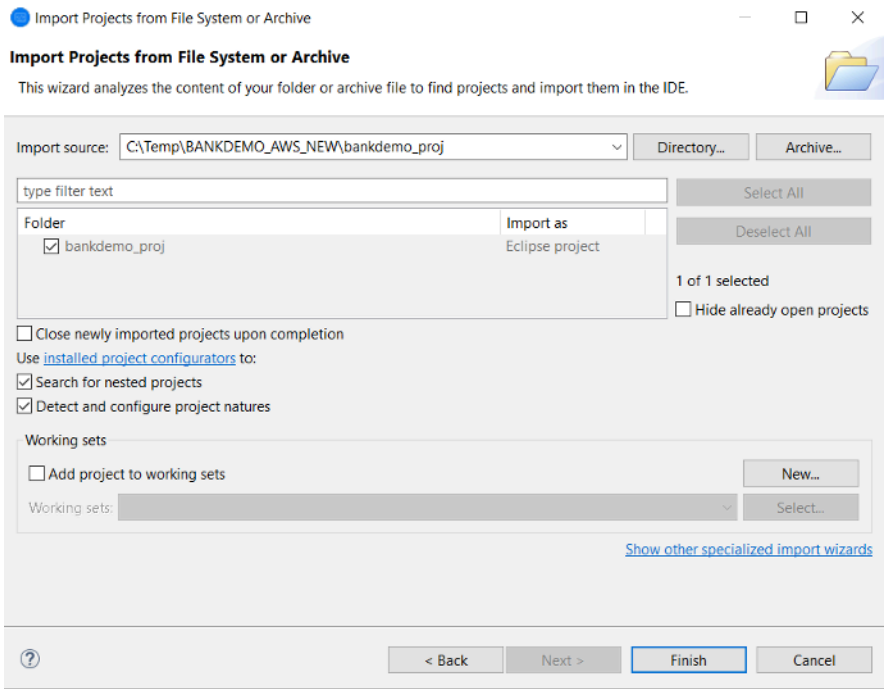
2. En el menú File, elija Import.



3. En el cuadro de diálogo resultante, en General, seleccione Projects from Folder or Archive y pulse Next.



4. Introduzca el Import source, elija el Directory y navegue por el sistema de archivos para seleccionar la carpeta de proyecto predefinida. El proyecto que contiene tiene enlaces a las carpetas de origen del mismo repositorio.

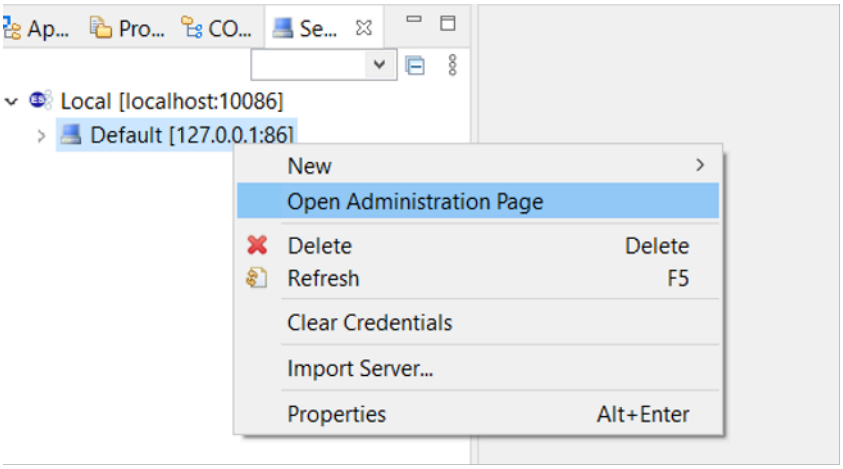


Seleccione Finalizar.

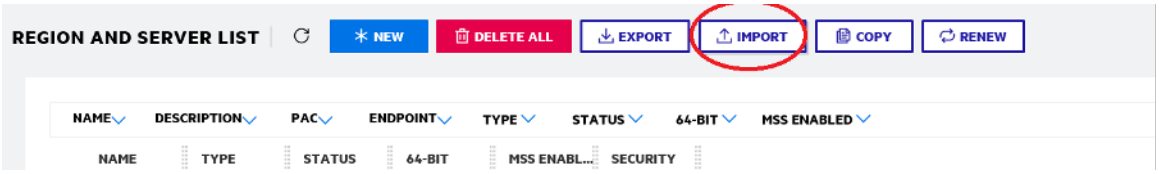
Como el proyecto se rellena con los enlaces a la carpeta de origen, el código se genera automáticamente.

Uso de la plantilla JSON de definición de región

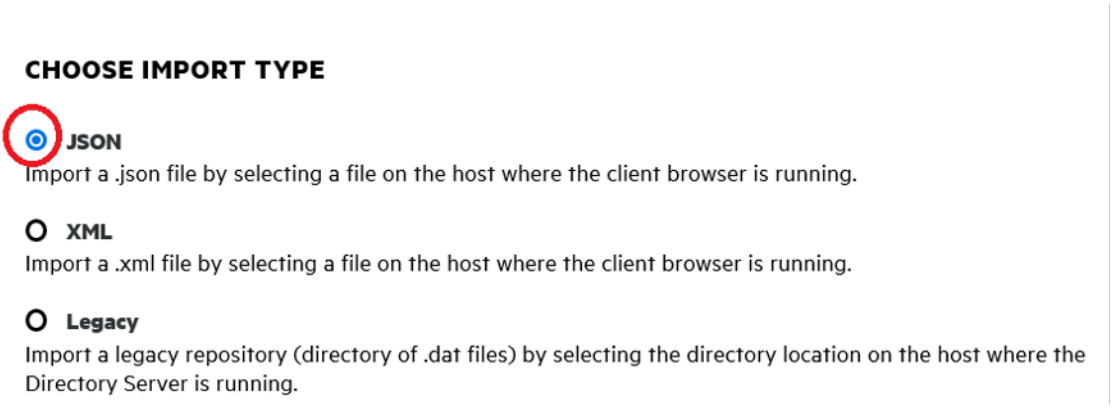
1. Cambie a la vista del explorador de servidores. En el menú contextual correspondiente, seleccione Open Administration Page, que iniciará el navegador predeterminado.



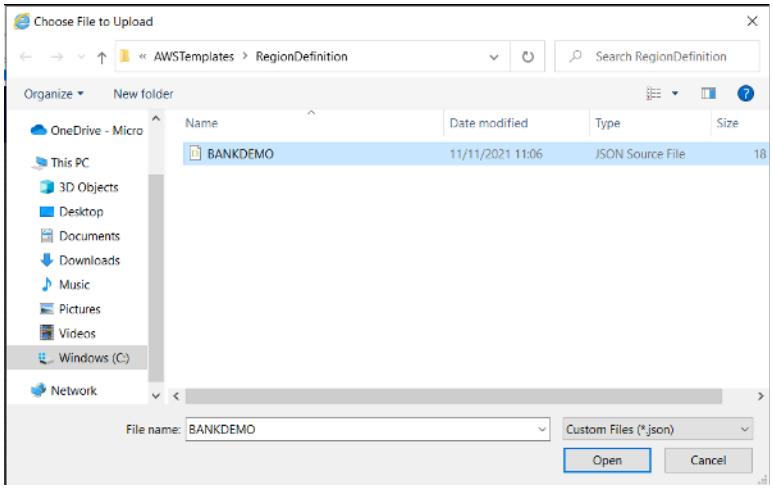
2. En la pantalla resultante de Enterprise Server Common Web Administration (ESCWA), seleccione Import.



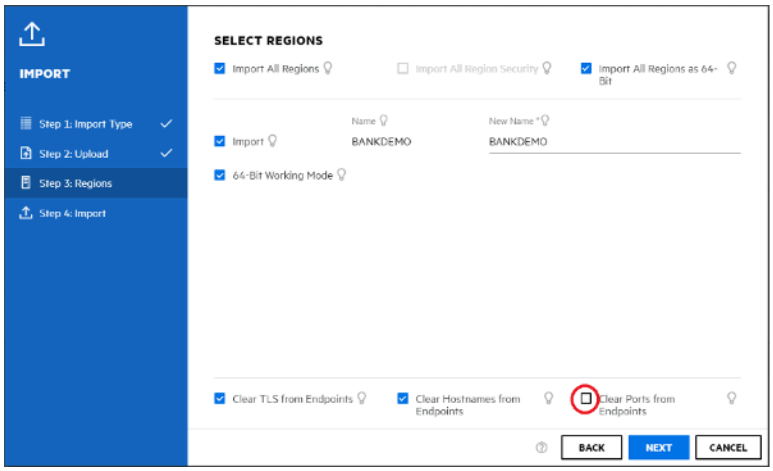
3. Seleccione el tipo de importación JSON y, después, Next.



4. Cargue el archivo `BANKDEMO.JSON` suministrado.

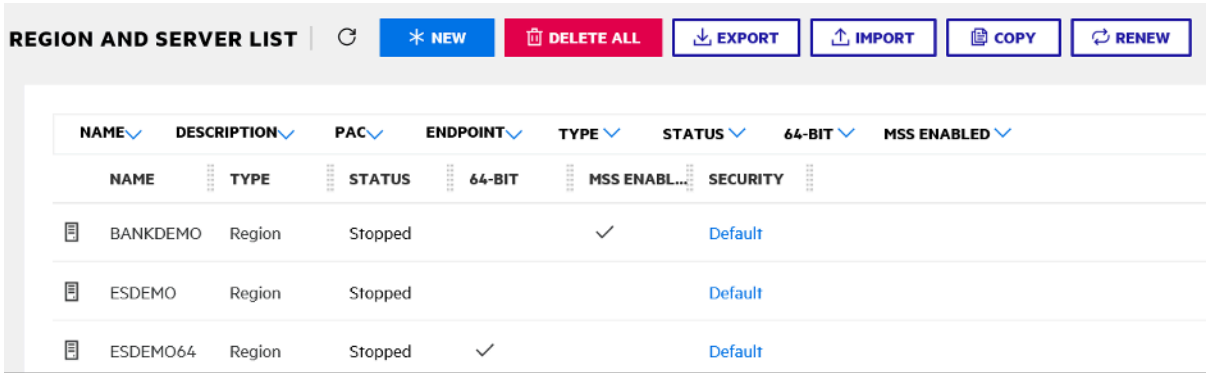


Una vez seleccionado, elija Next.

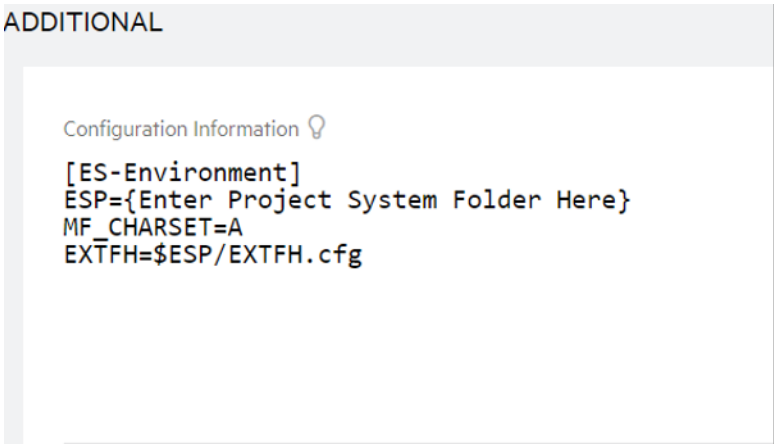


En el panel Selecr Regions, asegúrese de que la opción Clear Ports from Endpoints no esté seleccionada y, a continuación, siga seleccionando Next en los paneles hasta que aparezca el panel Perform Import. A continuación, elija Importar en el panel de navegación izquierdo.

Por último, haga clic en Finish. La región BANKDEMO se añadirá a la lista de servidores.



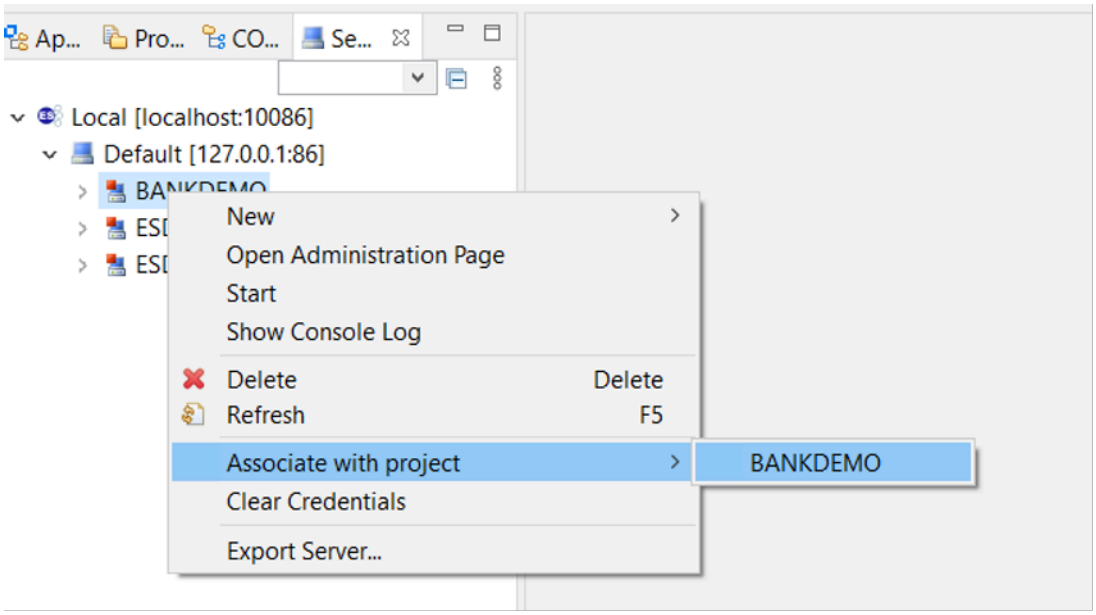
5. Navegue hasta las General Properties de la región BANKDEMO.
6. Desplácese a la sección Configuration.
7. La variable de entorno ESP debe configurarse en la carpeta `System` correspondiente al proyecto de Eclipse creado en los pasos anteriores. El valor debería ser `workspacefolder/projectname/System`.



8. Haga clic en Apply.

   La región ahora está completamente configurada para ejecutarse junto con el proyecto COBOL de Eclipse.

9. Por último, vuelva a Enterprise Developer y asocie la región importada al proyecto.



   El entorno Enterprise Developer ya está listo para su uso, con una versión funcional completa de BankDemo. Puede editar, compilar y depurar el código de la región.

> **Important**
>
> Si utiliza la versión de Enterprise Developer para Windows, los binarios generados por el compilador solo se pueden ejecutar en el Enterprise Server suministrado con Enterprise Developer. No se pueden ejecutar en el entorno de ejecución AWS Mainframe Modernization, que está basado en Linux.

Tutorial: Configurar Enterprise Analyzer en 2.0 AppStream

Este tutorial describe cómo configurar Rocket Enterprise Analyzer (anteriormente Micro Focus Enterprise Analyzer) para analizar una o más aplicaciones de mainframe. La herramienta Enterprise Analyzer proporciona varios informes basados en su análisis del código fuente de la aplicación y las definiciones del sistema.

Esta configuración está diseñada para fomentar la colaboración en equipo. La instalación utiliza un bucket de Amazon S3 para compartir el código fuente con los discos virtuales. Para ello, se utiliza Rclone) en el equipo de Windows. Con una instancia de Amazon RDS común que ejecute PostgreSQL, cualquier miembro del equipo puede acceder a todos los informes solicitados.

Los miembros del equipo también pueden montar el disco virtual respaldado por Amazon S3 en sus equipos personales y actualizar el bucket de origen desde sus estaciones de trabajo. También pueden usar scripts o cualquier otra forma de automatización en sus equipos si están conectados a otros sistemas internos en las instalaciones.

La configuración se basa en las imágenes de Windows AppStream 2.0 que AWS Mainframe Modernization comparte con el cliente. La configuración también se basa en la creación de flotas y pilas AppStream 2.0, tal como se describe en. Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer

> **Important**
>
> En los pasos de este tutorial se supone que se ha configurado la AppStream versión 2.0 con la AWS CloudFormation plantilla descargable cfn-m2-.yml. appstream-fleet-ea-ed Para obtener más información, consulte Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer.

Para realizar los pasos en este tutorial, debe haber configurado la flota y la pila de Enterprise Analyzer y deben estar en funcionamiento.

Para obtener una descripción completa de las funciones y prestaciones de Enterprise Analyzer, consulte la documentación de Enterprise Analyzer en el sitio web de Rocket Software (anteriormente Micro Focus).

Contenido de la imagen

Además de la propia aplicación Enterprise Analyzer, la imagen contiene las siguientes herramientas y bibliotecas.

Herramientas de terceros

- Python
- Rclone
- pgAdmin
- git-scm
- Controlador ODBC para PostgreSQL

Bibliotecas en `C:\Users\Public`

- BankDemo código fuente y definición del proyecto para Enterprise Developer:. `m2-bankdemo-template.zip`
- Paquete de instalación de MFA para el sistema central: `mfa.zip`. Para obtener más información, consulte Mainframe Access Overview en la documentación de Micro Focus Enterprise Developer.
- Archivos de comandos y configuración para Rclone (instrucciones para su uso en los tutoriales): `m2-rclone.cmd` y `m2-rclone.conf`.

Temas

- Requisitos previos
- Paso 1: Configuración
- Paso 2: Crear la carpeta virtual basada en Amazon S3 en Windows
- Paso 3: Crear un origen de ODBC para la instancia de Amazon RDS

- Sesiones posteriores
- Solución de problemas de conexión de espacio de trabajo
- Eliminar recursos

Requisitos previos

- Cargue el código fuente y las definiciones del sistema de la aplicación del cliente que desee analizar en un bucket de S3. Las definiciones del sistema incluyen CICS CSD, definiciones de DB2 objetos, etc. Puede crear una estructura de carpetas dentro del bucket que se adapte a la forma en que desea organizar los artefactos de la aplicación. Por ejemplo, al descomprimir la BankDemo muestra, tiene la siguiente estructura:

```
demo
     |--> jcl
     |--> RDEF
     |--> transaction
     |--> xa
```

- Cree una instancia de Amazon RDS que ejecute PostgreSQL e iníciela. Esta instancia almacenará los datos y los resultados que produzca Enterprise Analyzer. Puede compartir esta instancia con todos los miembros del equipo de la aplicación. Además, cree un esquema vacío llamado `m2_ea` (o cualquier otro nombre adecuado) en la base de datos. Defina las credenciales para los usuarios autorizados que les permitan crear, insertar, actualizar y eliminar elementos de este esquema. Puede obtener el nombre de la base de datos, la URL del punto de conexión del servidor y el puerto TCP desde la consola de Amazon RDS o a través del administrador de la cuenta.
- Asegúrese de haber configurado el acceso programático a su. Cuenta de AWS Para obtener más información, consulte Acceso programático en Referencia general de Amazon Web Services.

Paso 1: Configuración

1. Inicie una sesión con AppStream 2.0 con la URL que recibió en el mensaje de correo electrónico de bienvenida de AppStream 2.0.
2. Utilice su correo electrónico como ID de usuario y defina su contraseña permanente.
3. Seleccione la pila de Enterprise Analyzer.
4. En la página del menú AppStream 2.0, selecciona Escritorio para acceder al escritorio de Windows que está transmitiendo la flota.

Paso 2: Crear la carpeta virtual basada en Amazon S3 en Windows

> **Note**
>
> Si ya utilizó Rclone durante la versión preliminar de AWS Mainframe Modernization, debe actualizar `m2-rclone.cmd` a la versión más reciente que se encuentra en. `C:\Users\Public`

1. Copie los archivos `m2-rclone.conf` y `m2-rclone.cmd` proporcionados en `C:\Users\Public` a su carpeta de inicio `C:\Users\PhotonUser\My Files\Home Folder` mediante el Explorador de archivos.
2. Actualice los parámetros de `m2-rclone.conf` configuración con su clave de AWS acceso y el secreto correspondiente, así como con su. Región de AWS

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. En `m2-rclone.cmd`, realice los siguientes cambios:

   - Cambie `amzn-s3-demo-bucket` por el nombre de su bucket de Amazon S3. Por ejemplo, `m2-s3-mybucket`.
   - Cambie `your-s3-folder-key` por el nombre de su clave de bucket de Amazon S3. Por ejemplo, `myProject`.
   - Cambie `your-local-folder-path` por la ruta del directorio en el que desee sincronizar los archivos de la aplicación desde el bucket de Amazon S3 que los contiene. Por ejemplo, `D:\PhotonUser\My Files\Home Folder\m2-new`. Este directorio sincronizado debe ser un subdirectorio de la carpeta principal para que la AppStream versión 2.0 pueda realizar copias de seguridad y restaurarlo correctamente al iniciar y finalizar la sesión.

```
:loop
timeout /T 10
```

```
"C:\Program Files\rclone\rclone.exe" sync m2-s3:amzn-s3-demo-bucket/your-s3-
folder-key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Abra una línea de comandos de Windows, ejecute un cd a `C:\Users\PhotonUser\My Files\Home Folder` si es necesario y ejecute `m2-rclone.cmd`. Este script de comandos ejecuta un bucle continuo y sincroniza el bucket y la clave de Amazon S3 con la carpeta local cada 10 segundos. Puede ajustar el tiempo de espera según sea necesario. Debería ver el código fuente de la aplicación ubicado en el bucket de Amazon S3 en el Explorador de archivos de Windows.

Para añadir nuevos archivos al conjunto en el que está trabajando o actualizar los existentes, cargue los archivos en el bucket de Amazon S3 y se sincronizarán con su directorio en la siguiente iteración definida en `m2-rclone.cmd`. Del mismo modo, si desea eliminar algunos archivos, elimínelos del bucket de Amazon S3. La siguiente operación de sincronización los eliminará de su directorio local.

Paso 3: Crear un origen de ODBC para la instancia de Amazon RDS

1. Para iniciar la herramienta EA_Admin, navegue hasta el menú de selección de aplicaciones situado en la esquina superior izquierda de la ventana del navegador y seleccione MF EA_Admin.
2. En el menú Administrar, seleccione Orígenes de datos de ODBC y, en la pestaña DSN de usuario, seleccione Agregar.
3. En el cuadro de diálogo Crear nuevo origen de datos, elija el controlador PostgreSQL Unicode y, a continuación, elija Finalizar.
4. En el cuadro de diálogo Configuración de controlador ODBC Unicode PostgreSQL (psqlODBC), defina y anote el nombre del origen de datos que desee. Complete los siguientes parámetros con los valores de la instancia de RDS que ha creado anteriormente:

   Descripción

   Descripción opcional para ayudarle a identificar rápidamente esta conexión a la base de datos.

   Database

   La base de datos de Amazon RDS que ha creado anteriormente.

Server

El punto de conexión de Amazon RDS.

Puerto

El puerto de Amazon RDS.

User Name

El que se ha definido en la instancia de Amazon RDS.

Password

La que se ha definido en la instancia de Amazon RDS.

5. Elija Prueba para validar que la conexión a Amazon RDS se ha realizado correctamente y, a continuación, seleccione Guardar para guardar su nuevo DSN de usuario.
6. Espere a que aparezca el mensaje que confirme la creación del espacio de trabajo adecuado y, a continuación, pulse Aceptar para terminar con los orígenes de datos de ODBC y cerrar la herramienta EA_Admin.
7. Vuelva al menú de selección de aplicaciones y elija Enterprise Analyzer para iniciar la herramienta. Elija Crear nuevo.
8. En la ventana de configuración del espacio de trabajo, introduzca el nombre del espacio de trabajo y defina su ubicación. El espacio de trabajo puede ser el disco basado en Amazon S3 si trabaja con esta configuración, o su carpeta de inicio si lo prefiere.
9. Seleccione Elegir otra base de datos para conectarse a su instancia de Amazon RDS.
10. Elija el icono de Postgre de entre las opciones y, a continuación, pulse Aceptar.
11. Para la configuración de Windows, en Opciones: Definir parámetros de conexión, introduzca el nombre del origen de datos que ha creado. Introduzca también el nombre de la base de datos, el nombre de esquema, el nombre de usuario y la contraseña. Seleccione Aceptar.
12. Espere a que Enterprise Analyzer cree todas las tablas, índices, etc. que necesite para almacenar los resultados. Es posible que tarde un par de minutos. Enterprise Analyzer confirma cuándo la base de datos y el espacio de trabajo están listos para su uso.
13. Vuelva al menú de selección de aplicaciones y elija Enterprise Analyzer para iniciar la herramienta.
14. La ventana de inicio de Enterprise Analyzer aparece en la nueva ubicación del espacio de trabajo seleccionada. Seleccione Aceptar.

15. Navegue hasta el repositorio en el panel izquierdo, seleccione el nombre del repositorio y elija Añadir archivos/carpetas al espacio de trabajo. Seleccione la carpeta en la que está almacenado el código de la aplicación para añadirlo al espacio de trabajo. Si lo desea, puede usar el código del BankDemo ejemplo anterior. Cuando Enterprise Analyzer le pida que verifique esos archivos, elija Verificar para iniciar el informe de verificación inicial de Enterprise Analyzer. Este proceso puede tardar unos minutos, dependiendo del tamaño de la aplicación.
16. Amplíe su espacio de trabajo para ver los archivos y carpetas que ha agregado al espacio de trabajo. Los tipos de objetos y los informes de complejidad ciclomática también están visibles en el cuadrante superior del panel Visor de gráficos.

Ahora puede usar Enterprise Analyzer para todas las tareas necesarias.

Sesiones posteriores

1. Inicie una sesión con AppStream 2.0 con la URL que recibió en el mensaje de correo electrónico de bienvenida de AppStream 2.0.
2. Inicie sesión con su correo electrónico y contraseña permanente.
3. Seleccione la pila de Enterprise Analyzer.
4. Inicie `Rclone` para conectarse al disco respaldado por Amazon S3 si utiliza esta opción para compartir los archivos del espacio de trabajo.
5. Inicie Enterprise Analyzer para hacer sus tareas.

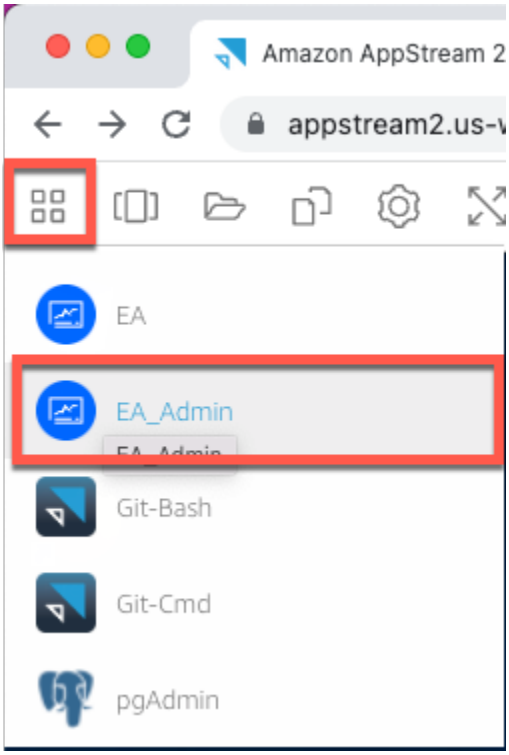
Solución de problemas de conexión de espacio de trabajo

Cuando intente volver a conectarse a su espacio de trabajo de Enterprise Analyzer, es posible que aparezca un error como el siguiente:

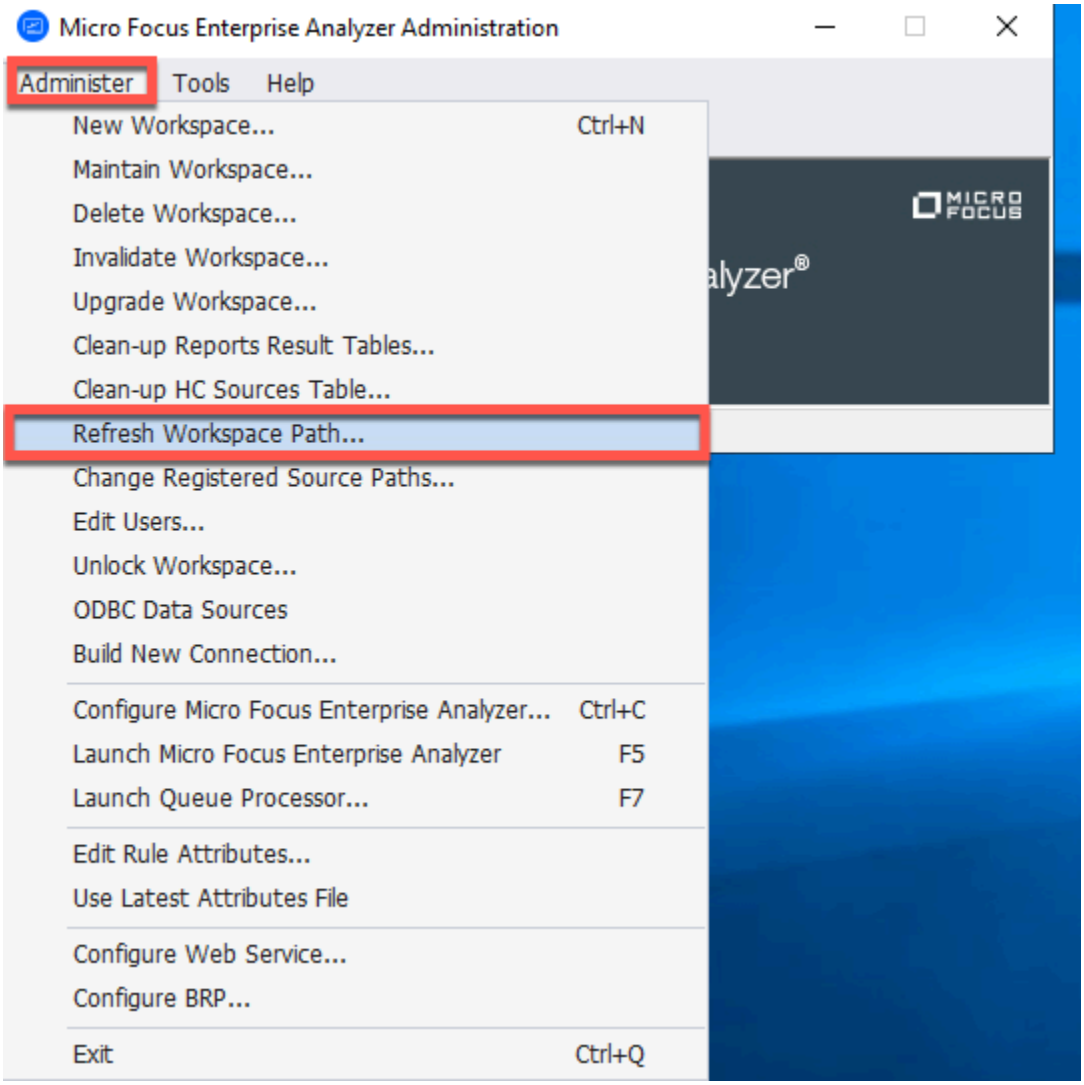
```
Cannot access the workspace directory D:\PhotonUser\My Files\Home Folder\EA_BankDemo.
 The workspace has been created on a non-shared disk of the EC2AMAZ-E6LC33H computer.
 Would you like to correct the workspace directory location?
```

Para resolver este problema, pulse Aceptar para borrar el mensaje y, a continuación, complete los pasos siguientes.

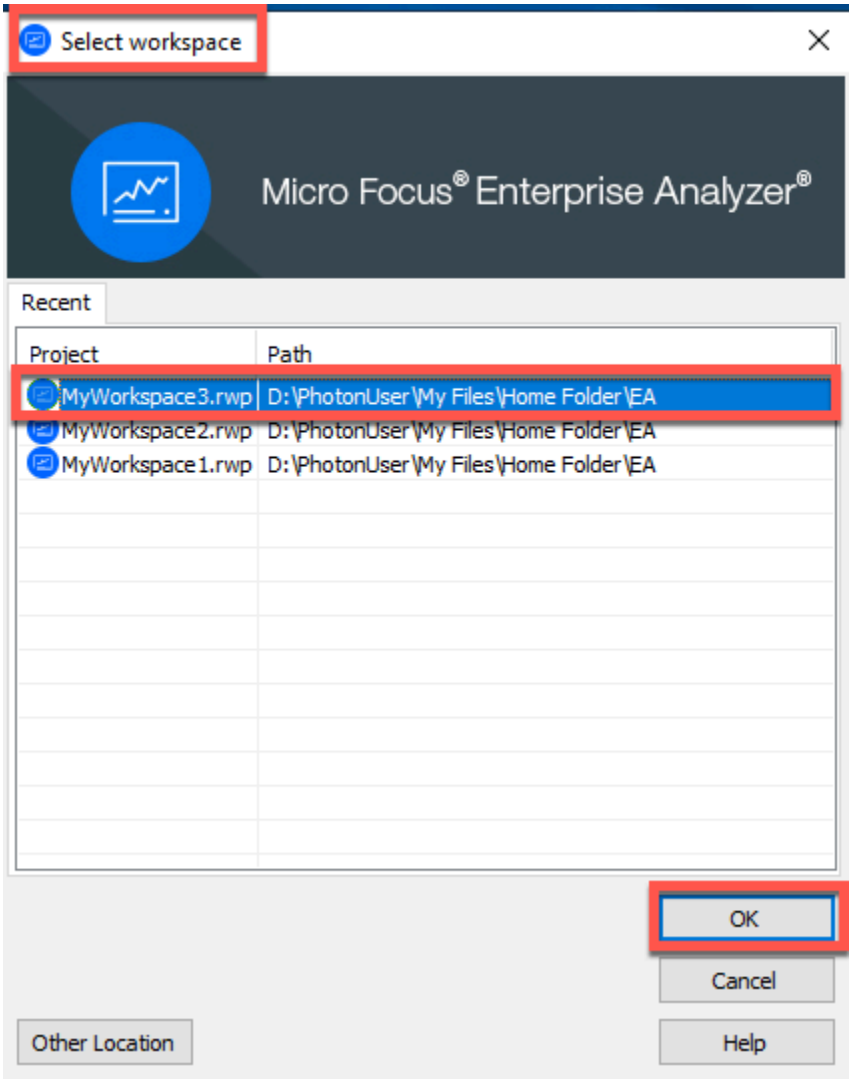
1. En la AppStream versión 2.0, seleccione el icono Iniciar aplicación en la barra de herramientas y, a continuación, elija EA_Admin para iniciar la herramienta de administración de Enterprise Analyzer.



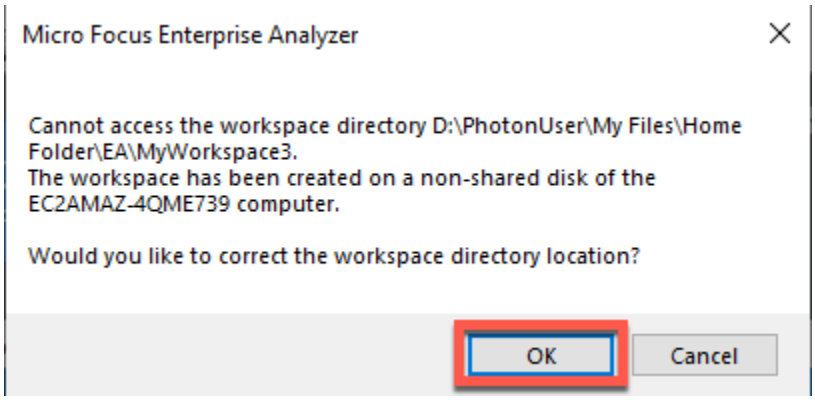
2. En el menú Administrar, seleccione Actualizar ruta de espacio de trabajo....



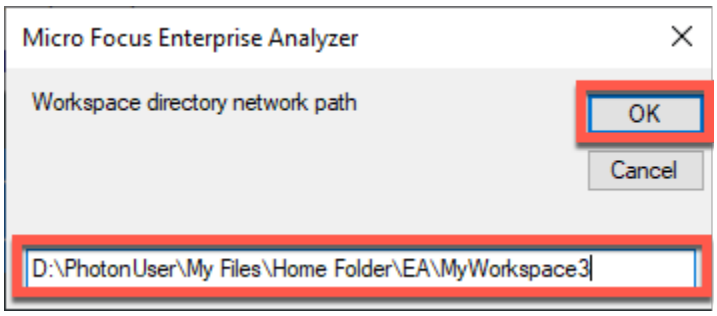
3. En Seleccionar espacio de trabajo, elija el espacio de trabajo que desee y, a continuación, pulse Aceptar.



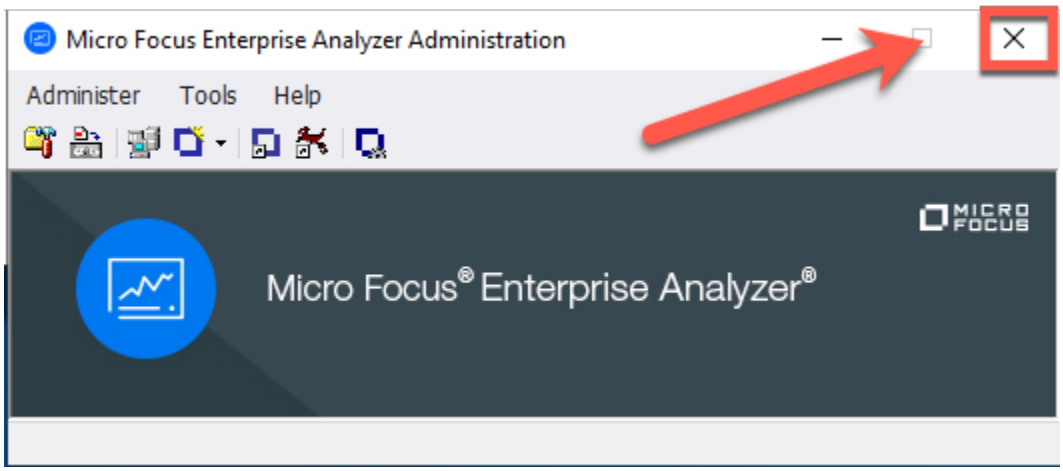
4. Pulse Aceptar para confirmar el mensaje de error.



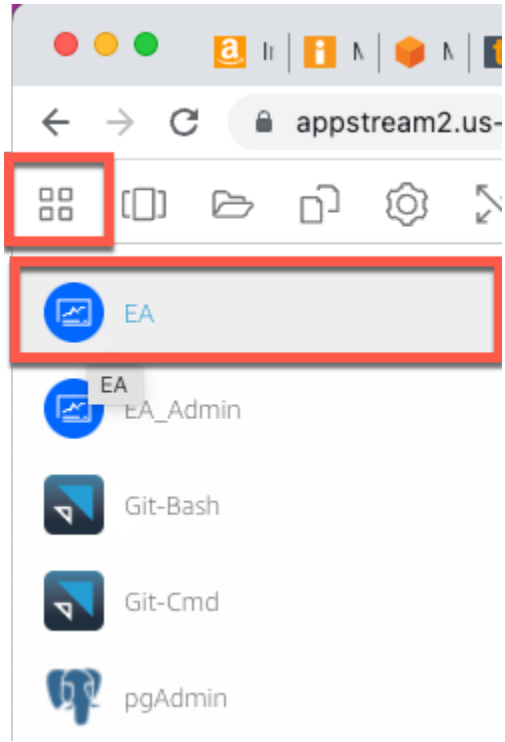
5. En Ruta de red del directorio de espacio de trabajo, introduzca la ruta correcta a su espacio de trabajo, por ejemplo `D:\PhotonUser\My Files\Home Folder\EA\MyWorkspace3`.



6. Cierre la herramienta Micro Focus Enterprise Analyzer Administration.



7. En la AppStream versión 2.0, seleccione el icono Iniciar aplicación en la barra de herramientas y, a continuación, elija EA para iniciar Micro Focus Enterprise Analyzer.



8. Repita los pasos 3 a 5.

Ahora Micro Focus Enterprise Analyzer debería abrirse con el espacio de trabajo existente.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para que no se le cobre por ellos. Realice los siguientes pasos:

- Use la herramienta EA_Admin para eliminar el espacio de trabajo.
- Elimine los buckets de S3 que ha creado para este tutorial. Para obtener más información, consulte Eliminar un bucket en la Guía del usuario de Amazon S3.
- Elija la base de datos que ha creado para este tutorial. Para obtener más información, consulte Eliminación de una instancia de base de datos.

Tutorial: Configurar Rocket Enterprise Developer en AppStream 2.0

Este tutorial describe cómo configurar Rocket Enterprise Developer (anteriormente Micro Focus Enterprise Developer) para una o más aplicaciones de mainframe con el fin de mantenerlas, compilarlas y probarlas mediante las funciones de Enterprise Developer. La configuración se basa en las imágenes de Windows AppStream 2.0 que AWS Mainframe Modernization comparte con el cliente y en la creación de flotas y pilas AppStream 2.0, tal como se describe en. Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer

> **Important**
>
> En los pasos de este tutorial se parte del supuesto de que se ha configurado la AppStream versión 2.0 mediante la AWS CloudFormation plantilla descargable cfn-m2 - .yaml. appstream-fleet-ea-ed Para obtener más información, consulte Tutorial: Configurar AppStream 2.0 para su uso con Rocket Enterprise Analyzer y Rocket Enterprise Developer. Debe realizar los pasos de esta configuración cuando la flota y la pila de Enterprise Developer estén en ejecución.

Para obtener una descripción completa de las funciones y prestaciones de Enterprise Developer v7, consulte su documentación up-to-date en línea (versión 7.0) en el sitio de Rocket Software (anteriormente Micro Focus).

Contenido de la imagen

Además de Enterprise Developer en sí, la imagen contiene la imagen que contiene Rumba (un TN327 emulador 0). También contiene las siguientes herramientas y bibliotecas.

Herramientas de terceros

- Python
- Rclone
- pgAdmin
- git-scm
- Controlador ODBC para PostgreSQL

Bibliotecas en `C:\Users\Public`

- BankDemo código fuente y definición de proyecto para Enterprise Developer:`m2-bankdemo-template.zip`.
- Paquete de instalación de MFA para el sistema central: `mfa.zip`. Para obtener más información, consulte Mainframe Access Overview en la documentación de Micro Focus Enterprise Developer.
- Archivos de comandos y configuración para Rclone (instrucciones para su uso en los tutoriales): `m2-rclone.cmd` y `m2-rclone.conf`.

Si necesita acceder a un código fuente que aún no está cargado en CodeCommit los repositorios, pero que está disponible en un bucket de Amazon S3, por ejemplo, para realizar la carga inicial del código fuente en git, siga el procedimiento para crear un disco virtual de Windows tal y como se describe enTutorial: Configurar Enterprise Analyzer en 2.0 AppStream .

Temas

- Requisitos previos
- Paso 1: Configuración por parte de usuarios individuales de Enterprise Developer
- Paso 2: Crear la carpeta virtual basada en Amazon S3 en Windows (opcional)
- Paso 3: Clonar el repositorio
- Sesiones posteriores
- Eliminar recursos

Requisitos previos

- Uno o más CodeCommit repositorios cargados con el código fuente de la aplicación que se va a mantener. La configuración del repositorio debe cumplir con los requisitos del CI/CD canalice lo anterior para crear sinergias mediante la combinación de ambas herramientas.
- Cada usuario debe tener credenciales para el CodeCommit repositorio o los repositorios definidos por el administrador de la cuenta de acuerdo con la información de Autenticación y control de acceso para AWS CodeCommit. La estructura de esas credenciales se analiza en Autenticación y control de acceso para AWS CodeCommit y la referencia completa para las autorizaciones de IAM CodeCommit se encuentra en la referencia de CodeCommit permisos: el administrador puede definir políticas de IAM distintas para funciones distintas con credenciales específicas para la función de cada repositorio y limitar las autorizaciones del usuario al conjunto específico de tareas que debe realizar en un repositorio determinado. Por lo tanto, para cada responsable del CodeCommit repositorio, el administrador de la cuenta generará un usuario principal y le concederá permisos para acceder al repositorio o repositorios necesarios seleccionando la política o políticas de CodeCommit acceso de IAM adecuadas.

Paso 1: Configuración por parte de usuarios individuales de Enterprise Developer

1. Obtenga sus credenciales de IAM:
    1. Conéctese a la AWS consola en https://console.aws.amazon.com/iam/.
    2. Siga el procedimiento descrito en el paso 3 de Setup for HTTPS users using Git credentials en la guía del usuario de AWS CodeCommit .
    3. Copie las credenciales CodeCommit de inicio de sesión específicas que IAM generó para usted, ya sea mostrando, copiando y pegando esta información en un archivo seguro de su ordenador local o seleccionando Descargar credenciales para descargar esta información como un archivo.CSV. Necesitará esta información para conectarse a CodeCommit.
2. Inicie una sesión con la AppStream versión 2.0 en función de la URL recibida en el correo electrónico de bienvenida. Utilice su correo electrónico como nombre de usuario y cree su contraseña.
3. Seleccione su pila de Enterprise Developer.
4. En la página de menú, elija Escritorio para acceder al escritorio de Windows que está transmitiendo la flota.

Paso 2: Crear la carpeta virtual basada en Amazon S3 en Windows (opcional)

Si es necesario utilizar Rclone (véase más arriba), cree la carpeta virtual basada en Amazon S3 en Windows: (opcional si todos los artefactos de la aplicación provienen exclusivamente de Access). CodeCommit

> **Note**
>
> Si ya utilizó Rclone durante la versión preliminar de AWS Mainframe Modernization, debe actualizar `m2-rclone.cmd` a la versión más reciente que se encuentra en. `C:\Users\Public`

1. Copie los archivos `m2-rclone.conf` y `m2-rclone.cmd` proporcionados en `C:\Users\Public` a su carpeta de inicio `C:\Users\PhotonUser\My Files\Home Folder` mediante el Explorador de archivos.
2. Actualice los parámetros de `m2-rclone.conf` configuración con su clave de AWS acceso y el secreto correspondiente, así como con su. Región de AWS

```
[m2-s3]
type = s3
provider = AWS
access_key_id = YOUR-ACCESS-KEY
secret_access_key = YOUR-SECRET-KEY
region = YOUR-REGION
acl = private
server_side_encryption = AES256
```

3. En `m2-rclone.cmd`, realice los siguientes cambios:

   - Cambie `amzn-s3-demo-bucket` por el nombre de su bucket de Amazon S3. Por ejemplo, `m2-s3-mybucket`.
   - Cambie `your-s3-folder-key` por el nombre de su clave de bucket de Amazon S3. Por ejemplo, `myProject`.
   - Cambie `your-local-folder-path` por la ruta del directorio en el que desee sincronizar los archivos de la aplicación desde el bucket de Amazon S3 que los contiene. Por ejemplo, `D:\PhotonUser\My Files\Home Folder\m2-new`. Este directorio sincronizado debe ser un subdirectorio de la carpeta principal para que la AppStream versión 2.0 pueda realizar copias de seguridad y restaurarlo correctamente al iniciar y finalizar la sesión.

```
:loop
timeout /T 10
"C:\Program Files\rclone\rclone.exe" sync m2-s3:amzn-s3-demo-bucket/your-s3-
folder-key "D:\PhotonUser\My Files\Home Folder\your-local-folder-path" --config "D:
\PhotonUser\My Files\Home Folder\m2-rclone.conf"
goto :loop
```

4. Abra una línea de comandos de Windows, ejecute un cd a `C:\Users\PhotonUser\My Files\Home Folder` si es necesario y ejecute `m2-rclone.cmd`. Este script de comandos ejecuta un bucle continuo y sincroniza el bucket y la clave de Amazon S3 con la carpeta local cada 10 segundos. Puede ajustar el tiempo de espera según sea necesario. Debería ver el código fuente de la aplicación ubicado en el bucket de Amazon S3 en el Explorador de archivos de Windows.

Para añadir nuevos archivos al conjunto en el que está trabajando o actualizar los existentes, cargue los archivos en el bucket de Amazon S3 y se sincronizarán con su directorio en la siguiente iteración definida en `m2-rclone.cmd`. Del mismo modo, si desea eliminar algunos archivos, elimínelos del bucket de Amazon S3. La siguiente operación de sincronización los eliminará de su directorio local.

Paso 3: Clonar el repositorio

1. Navegue hasta el menú selector de aplicaciones en la esquina superior izquierda de la ventana del navegador y seleccione Enterprise Developer.
2. Complete la creación del espacio de trabajo requerida por Enterprise Developer en su carpeta de inicio eligiendo `C:\Users\PhotonUser\My Files\Home Folder` (también conocida como `D: \PhotonUser\My Files\Home Folder`) como ubicación del espacio de trabajo.
3. En Enterprise Developer, clona tu CodeCommit repositorio yendo al Explorador de proyectos, haz clic con el botón derecho y selecciona Importar, Importar..., Git, Projects from Git Clone URI. A continuación, introduce tus credenciales CodeCommit de inicio de sesión específicas y completa el cuadro de diálogo de Eclipse para importar el código.

El repositorio de CodeCommit git ahora está clonado en tu espacio de trabajo local.

Su espacio de trabajo de Enterprise Developer ya está listo para iniciar los trabajos de mantenimiento de su aplicación. En concreto, puedes usar la instancia local de Enterprise Server (ES) integrada con Enterprise Developer para depurar y ejecutar tu aplicación de forma interactiva y validar los cambios a nivel local.

> **Note**
>
> El entorno de Enterprise Developer local, incluida la instancia local de Enterprise Server, se ejecuta en Windows, mientras que AWS Mainframe Modernization se ejecuta en Linux. Le recomendamos que ejecute pruebas complementarias en el entorno Linux proporcionado por AWS Mainframe Modernization después de instalar la nueva aplicación CodeCommit y reconstruirla para este objetivo y antes de lanzar la nueva aplicación a producción.

Sesiones posteriores

Al seleccionar una carpeta que esté gestionada por la AppStream versión 2.0, como la carpeta principal, para clonar el CodeCommit repositorio, se guardará y restaurará de forma transparente en todas las sesiones. Complete los siguientes pasos la próxima vez que necesite trabajar con la aplicación:

1. Inicie una sesión con la AppStream versión 2.0 en función de la URL recibida en el correo electrónico de bienvenida.
2. Inicie sesión con su correo electrónico y contraseña permanente.
3. Seleccione la pila de Enterprise Developer.
4. Inicie `Rclone` para conectarse (véase anteriormente) al disco respaldado por Amazon S3 si utiliza esta opción para compartir los archivos del espacio de trabajo.
5. Inicie Enterprise Developer para hacer su trabajo.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para que no se le siga cobrando por ellos. Realice los siguientes pasos:

- Elimine el CodeCommit repositorio que creó para este tutorial. Para obtener más información, consulte Eliminar un CodeCommit repositorio en la Guía del AWS CodeCommit usuario.
- Elimine la base de datos que ha creado para este tutorial. Para obtener más información, consulte Eliminación de una instancia de base de datos.

Utilidades por lotes disponibles en la modernización de AWS mainframe

Las aplicaciones de mainframe suelen utilizar programas de utilidad por lotes para realizar funciones específicas, como ordenar datos, transferir archivos mediante FTP, cargar datos en bases de datos DB2, descargar datos de bases de datos, etc.

Al migrar sus aplicaciones a la modernización del AWS mainframe, necesitará utilidades de reemplazo funcionalmente equivalentes que puedan realizar las mismas tareas que las que utilizaba en el mainframe. Es posible que algunas de estas utilidades ya estén disponibles como parte de los motores de ejecución de AWS Mainframe Modernization, pero ofrecemos las siguientes utilidades de reemplazo:

- M2SFTP: permite la transferencia segura de archivos mediante el protocolo SFTP.
- M2WAIT: espera un período de tiempo específico antes de continuar con el siguiente paso de un trabajo por lotes.
- TXT2PDF: convierte archivos de texto a formato PDF.
- M2DFUTIL: proporciona funciones de copia de seguridad, restauración, eliminación y copia en conjuntos de datos, de forma similar al soporte que proporciona la utilidad ADRDSSU del mainframe.
- M2RUNCMD: permite ejecutar comandos, scripts y llamadas al sistema de Rocket Software (anteriormente Micro Focus) directamente desde JCL.

Desarrollamos estas utilidades por lotes basándonos en los comentarios de los clientes y las diseñamos para que proporcionaran la misma funcionalidad que las utilidades de unidad central. El objetivo es hacer que la transición del mainframe a la modernización del mainframe sea lo más AWS fluida posible.

Temas

- Ubicación binaria
- Utilidad por lotes M2SFTP
- Utilidad por lotes M2WAIT
- TXT2Utilidad de lotes de PDF
- Utilidad por lotes M2DFUTIL

- Utilidad por lotes M2RUNCMD

Ubicación binaria

Estas utilidades vienen preinstaladas en los productos Rocket Enterprise Developer (ED) y Rocket Software (ES). Puede encontrarlas en la siguiente ubicación de todas las variantes de ED y ES:

- Linux: `/opt/aws/m2/microfocus/utilities/64bit`
- Windows de 32 bits: `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows de 64 bits: `C:\AWS\M2\MicroFocus\Utilities\64bit`

Utilidad por lotes M2SFTP

M2SFTP es un programa de utilidades de JCL diseñado para realizar transferencias de archivos seguras entre sistemas que utilizan el protocolo de transferencia segura de archivos (SFTP). El programa utiliza el cliente SFTP de Putty, `psftp`, para realizar las transferencias reales de archivos. El programa funciona de manera similar a un programa de utilidades de FTP para unidades centrales y utiliza la autenticación de usuario y contraseña.

> **Note**
>
> No se admite la autenticación de clave pública.

Para convertir el FTP de su ordenador central JCLs a SFTP, cambie `PGM=FTP` a. `PGM=M2SFTP`

Temas

- Plataformas admitidas
- Instalación de dependencias
- Configure M2SFTP para gestionar la modernización del AWS mainframe
- Configurar M2SFTP para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la versión 2.0) AppStream
- Ejemplo JCLs
- Referencia de comandos del cliente SFTP de Putty (PSFTP)
- Pasos a seguir a continuación

Plataformas admitidas

Puede utilizar M2SFTP en cualquiera de las siguientes plataformas:

- AWS Modernización del mainframe gestionada por Rocket Software (anteriormente Micro Focus)
- Rocket Software Runtime (en Amazon EC2)
- Todas las variantes de los productos Rocket Software Enterprise Developer (ED) y Rocket Software Enterprise Server (ES).

Instalación de dependencias

Para instalar el cliente SFTP de Putty en Windows

- Descargue el cliente SFTP de PuTTY e instálelo.

Para instalar el cliente SFTP de Putty en Linux:

- Ejecute el siguiente comando para instalar el cliente SFTP de Putty.

```
sudo yum -y install putty
```

Configure M2SFTP para gestionar la modernización del AWS mainframe

Si sus aplicaciones migradas se ejecutan en AWS Mainframe Modernization Managed, tendrá que configurar M2SFTP de la siguiente manera.

- Establezca las variables de entorno de Rocket Enterprise Server adecuadas para el MFFTP. A continuación se muestran algunos ejemplos:
  - `MFFTP_TEMP_DIR`
  - `MFFTP_SENDEOL`
  - `MFFTP_TIME`
  - `MFFTP_ABEND`

Puede configurar tantas variables como desee. Puede configurarlas en su JCL mediante la instrucción `ENVAR DD`. Para obtener más información sobre estas variables, consulte MFFTP Control Variables en la documentación de Micro Focus.

Para probar la configuración, consulte Ejemplo JCLs.

Configurar M2SFTP para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la versión 2.0) AppStream

Si las aplicaciones migradas se ejecutan en el entorno de ejecución de modernización de AWS mainframe en Amazon EC2, configure M2SFTP de la siguiente manera.

1. Cambie la ruta del programa JES de Micro Focus para incluir la ubicación binaria de las utilidades por lotes. Si necesita especificar varias rutas, utilice dos puntos (`:`) para separarlas en Linux y punto y coma (`;`) en Windows.

   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

2. Establezca las variables de entorno de Rocket Enterprise Server adecuadas para el MFFTP. A continuación se muestran algunos ejemplos:

   - `MFFTP_TEMP_DIR`
   - `MFFTP_SENDEOL`
   - `MFFTP_TIME`
   - MFFTP_ABEND

   Puede configurar tantas variables como desee. Puede configurarlas en su JCL mediante la instrucción `ENVAR DD`. Para obtener más información sobre estas variables, consulte MFFTP Control Variables en la documentación de Micro Focus.

Para probar la configuración, consulte Ejemplo JCLs.

Ejemplo JCLs

Para probar la instalación, puede utilizar uno de los siguientes archivos JCL de ejemplo.

M2 SFTP1 .jcl

Este JCL muestra cómo llamar a M2SFTP para enviar un archivo a un servidor SFTP remoto. Observe las variables de entorno que se establecen en la instrucción `ENVVAR DD`.

```
//M2SFTP1 JOB 'M2SFTP1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* Sample SFTP JCL step to send a file to SFTP server*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP,
//            PARM='127.0.0.1 (EXIT=99 TIMEOUT 300'
//*
//SYSFTPD  DD  *
RECFM FB
LRECL 80
SBSENDEOL CRLF
MBSENDEOL CRLF
TRAILINGBLANKS FALSE
/*
//NETRC    DD  *
machine 127.0.0.1 login sftpuser password sftppass
/*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
type a
locsite notrailingblanks
cd files
put 'AWS.M2.TXT2PDF1.PDF' AWS.M2.TXT2PDF1.pdf
put 'AWS.M2.CARDDEMO.CARDDATA.PS' AWS.M2.CARDDEMO.CARDDATA.PS1.txt
quit
/*
//ENVVAR   DD *
```

```
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

M2 .jcl SFTP2

Este JCL muestra cómo llamar a M2SFTP para recibir un archivo de un servidor SFTP remoto. Observe las variables de entorno establecidas en la instrucción `ENVVAR DD`.

```
//M2SFTP2 JOB 'M2SFTP2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* Sample SFTP JCL step to receive a file from SFTP server*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2SFTP
//*
//SYSPRINT DD  SYSOUT=*
//OUTPUT   DD  SYSOUT=*
//STDOUT   DD  SYSOUT=*
//INPUT    DD  *
open 127.0.0.1
sftpuser
sftppass
cd files
locsite recfm=fb lrecl=150
get AWS.M2.CARDDEMO.CARDDATA.PS.txt +
'AWS.M2.CARDDEMO.CARDDATA.PS2' (replace
quit
/*
//ENVVAR   DD *
MFFTP_VERBOSE_OUTPUT=ON
MFFTP_KEEP=N
/*
//*
//
```

> **Note**
>
> Recomendamos encarecidamente almacenar las credenciales de FTP en un archivo NETRC y restringir el acceso únicamente a los usuarios autorizados.

Referencia de comandos del cliente SFTP de Putty (PSFTP)

El cliente PSFTP no admite todos los comandos de FTP. La siguiente lista muestra todos los comandos compatibles con PSFTP.

Comando	Descripción
!	Ejecuta un comando local
bye	Finaliza su sesión de SFTP
cd	Cambia su directorio de trabajo remoto
chmod	Cambia los permisos y modos de los archivos
close	Finaliza la sesión de SFTP pero no cierra PSFTP
del	Elimina archivos en el servidor remoto
dir	Lista los archivos remotos
exit	Finaliza su sesión de SFTP
introducción	Descarga el archivo del servidor en el equipo local.
help	Brinda ayuda
lcd	Cambia el directorio de trabajo local
lpwd	Imprime el directorio de trabajo local
ls	Lista los archivos remotos

Comando	Descripción
mget	Descarga varios archivos a la vez
mkdir	Crea directorios en el servidor remoto
mput	Carga varios archivos a la vez
mv	Mueve o cambia el nombre de los archivos en el servidor remoto
open	Conectarse a un host
put	Carga un archivo del equipo local al servidor.
pwd	Imprime su directorio de trabajo remoto
quit	Finaliza su sesión de SFTP
reget	Continúa descargando archivos
ren	Mueve los archivos en el servidor remoto o les cambia el nombre
reput	Continúa descargando archivos
rm	Elimina archivos en el servidor remoto
rmdir	Elimina directorios en el servidor remoto

Pasos a seguir a continuación

Para cargar y descargar archivos en Amazon Simple Storage Service mediante SFTP, puede utilizar M2SFTP junto con el AWS Transfer Family, tal y como se describe en las siguientes entradas del blog.

- Uso de directorios lógicos de AWS SFTP para crear un servicio de distribución de datos sencillo
- Habilite la autenticación por contraseña para su uso AWS Transfer for SFTPAWS Secrets Manager

Utilidad por lotes M2WAIT

M2WAIT es un programa de utilidad para unidades centrales que le permite introducir un periodo de espera en los scripts de JCL especificando una duración en segundos, minutos u horas. Puede llamar a M2WAIT directamente desde JCL pasando el tiempo de espera como parámetro de entrada. Internamente, el programa M2WAIT llama al módulo suministrado por Rocket Software (anteriormente Micro Focus) `C$SLEEP` para que espere un tiempo específico.

> **Note**
>
> Puede utilizar los alias de Micro Focus para reemplazar los que tiene en sus scripts de JCL. Para obtener más información, consulte JES Alias en la documentación de Micro Focus.

Temas

- Plataformas admitidas
- Configure M2WAIT para gestionar la modernización AWS del mainframe
- Configurar M2WAIT para el tiempo de ejecución de modernización AWS del mainframe en Amazon EC2 (incluida AppStream la versión 2.0)
- JCL de ejemplo

Plataformas admitidas

Puede utilizar M2WAIT en cualquiera de las siguientes plataformas:

- AWS Rocket Software (anteriormente Micro Focus) gestionó la modernización de la unidad central
- Rocket Software Runtime (en Amazon EC2)
- Todas las variantes de los productos Rocket Software Enterprise Developer (ED) y Rocket Software Enterprise Server (ES).

Configure M2WAIT para gestionar la modernización AWS del mainframe

Si sus aplicaciones migradas se ejecutan en AWS Mainframe Modernization Managed, tendrá que configurar M2WAIT de la siguiente manera.

- Utilice el programa M2WAIT en su JCL pasando el parámetro de entrada como se muestra en JCL de ejemplo.

Configurar M2WAIT para el tiempo de ejecución de modernización AWS del mainframe en Amazon EC2 (incluida AppStream la versión 2.0)

Si las aplicaciones migradas se ejecutan en el entorno de ejecución de modernización de AWS mainframe en Amazon EC2, configure M2WAIT de la siguiente manera.

1. Cambie la ruta del programa JES de Micro Focus para incluir la ubicación binaria de las utilidades por lotes. Si necesita especificar varias rutas, utilice dos puntos (`:`) para separarlas en Linux y punto y coma (`;`) en Windows.
   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`
2. Utilice el programa M2WAIT en su JCL pasando el parámetro de entrada como se muestra en JCL de ejemplo.

JCL de ejemplo

Para probar la instalación, puede utilizar el programa `M2WAIT1.jcl`.

Este ejemplo de JCL muestra cómo llamar a M2WAIT y pasarle varias duraciones diferentes.

```
//M2WAIT1 JOB 'M2WAIT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*-------------------------------------------------------------------**
//* Wait for 12 Seconds*
//*-------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=M2WAIT,PARM='S012'
//SYSOUT DD SYSOUT=*
//*
//*-------------------------------------------------------------------**
//* Wait for 0 Seconds (defaulted to 10 Seconds)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=M2WAIT,PARM='S000'
//SYSOUT DD SYSOUT=*
```

```
//*
//*-------------------------------------------------------------------**
//* Wait for 1 Minute*
//*-------------------------------------------------------------------**
//*
//STEP03 EXEC PGM=M2WAIT,PARM='M001'
//SYSOUT DD SYSOUT=*
//*
//
```

TXT2Utilidad de lotes de PDF

TXT2PDF es un programa de utilidad para ordenadores centrales que se utiliza habitualmente para convertir un archivo de texto en un archivo PDF. Esta utilidad utiliza el mismo código fuente para el TXT2 PDF (z/OS freeware). Lo modificamos para que se ejecute en el entorno de ejecución del software AWS Mainframe Modernization Rocket (anteriormente Micro Focus).

Temas

- Plataformas admitidas
- Configure el TXT2 PDF para gestionar la modernización AWS del mainframe
- Configurar TXT2 PDF para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la AppStream versión 2.0)
- ACL de ejemplo
- Modificaciones
- Referencias

Plataformas admitidas

Puede utilizar TXT2 PDF en cualquiera de las siguientes plataformas:

- AWS Modernización del mainframe gestionada por Rocket Software
- Rocket Software Runtime (en Amazon EC2)
- Todas las variantes de los productos Rocket Enterprise Developer (ED) y Rocket Enterprise Server (ES).

Configure el TXT2 PDF para gestionar la modernización AWS del mainframe

Si las aplicaciones migradas se ejecutan en AWS Mainframe Modernization Managed, configure el TXT2 PDF de la siguiente manera.

- Cree una biblioteca REXX EXEC llamada `AWS.M2.REXX.EXEC`. Descargue estos módulos REXX y cópielos en la biblioteca.
  - `TXT2PDF.rex`- PDF TXT2 z/OS freeware (modificado)
  - `TXT2PDFD.rex`- PDF TXT2 z/OS freeware (sin modificar)
  - `TXT2PDFX.rex`- PDF TXT2 z/OS freeware (modificado)
  - `M2GETOS.rex`- Para comprobar el tipo de sistema operativo (Windows o Linux)

Para probar la configuración, consulte ACL de ejemplo.

Configurar TXT2 PDF para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la AppStream versión 2.0)

Si las aplicaciones migradas se ejecutan en el entorno de ejecución de AWS Mainframe Modernization de Amazon EC2, configure el TXT2 PDF de la siguiente manera.

1. Establezca la variable `MFREXX_CHARSET` de entorno de Rocket Software en el valor adecuado, como «`A`" para los datos ASCII.

   > **Important**
   >
   > Si introduce un valor incorrecto, podrían producirse problemas de conversión de datos (de EBCDIC a ASCII), lo que haría que el PDF resultante fuera ilegible o inoperable. Recomendamos configurar `MFREXX_CHARSET` para que coincida con `MF_CHARSET`.

2. Cambie la ruta del programa JES de Micro Focus para que incluya la ubicación binaria de las utilidades por lotes. Si necesita especificar varias rutas, utilice dos puntos (`:`) para separarlas en Linux y punto y coma (`;`) en Windows.
   - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
   - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
   - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

3. Cree una biblioteca REXX EXEC llamada `AWS.M2.REXX.EXEC`. Descargue estos módulos REXX y cópielos en la biblioteca.

   - `TXT2PDF.rex`- PDF TXT2 z/OS freeware (modificado)
   - `TXT2PDFD.rex`- PDF TXT2 z/OS freeware (sin modificar)
   - `TXT2PDFX.rex`- PDF TXT2 z/OS freeware (modificado)
   - `M2GETOS.rex`- Para comprobar el tipo de sistema operativo (Windows o Linux)

Para probar la configuración, consulte ACL de ejemplo.

ACL de ejemplo

Para probar la instalación, puede utilizar uno de los siguientes archivos JCL de ejemplo.

TXT2PDF1.jcl

Este archivo JCL de ejemplo utiliza un nombre DD para la TXT2 conversión a PDF.

```
//TXT2PDF1 JOB 'TXT2PDF1',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
```

```
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//OUTDD    DD DSN=AWS.M2.TXT2PDF1.PDF.VB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=VB,BLKSIZE=0)
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT DD:OUTDD +
CC YES
/*
//*
//*--------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*--------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF1.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF1.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

TXT2PDF2.jcl

En este ejemplo de JCL se utiliza un nombre DSN para la conversión a PDF. TXT2

```
//TXT2PDF2 JOB 'TXT2PDF2',CLASS=A,MSGCLASS=X,TIME=1440
//*
//* Copyright Amazon.com, Inc. or its affiliates.*
//* All Rights Reserved.*
//*
//*--------------------------------------------------------------------**
//* PRE DELETE*
//*--------------------------------------------------------------------**
//*
//PREDEL  EXEC PGM=IEFBR14
//*
//DD01     DD DSN=AWS.M2.TXT2PDF2.PDF.VB,
//            DISP=(MOD,DELETE,DELETE)
//*
//DD02     DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(MOD,DELETE,DELETE)
//*
//*--------------------------------------------------------------------**
//* CALL TXT2PDF TO CONVERT FROM TEXT TO PDF (VB)*
//*--------------------------------------------------------------------**
//*
//STEP01 EXEC PGM=IKJEFT1B
//*
//SYSEXEC  DD DISP=SHR,DSN=AWS.M2.REXX.EXEC
//*
//INDD     DD *
1THIS IS THE FIRST LINE ON THE PAGE 1
0THIS IS THE THIRD LINE ON THE PAGE 1
-THIS IS THE   6TH LINE ON THE PAGE 1
THIS IS THE   7TH LINE ON THE PAGE 1
+_____________________________________ - OVERSTRIKE 7TH LINE
1THIS IS THE FIRST LINE ON THE PAGE 2
0THIS IS THE THIRD LINE ON THE PAGE 2
-THIS IS THE   6TH LINE ON THE PAGE 2
THIS IS THE   7TH LINE ON THE PAGE 2
+_____________________________________ - OVERSTRIKE 7TH LINE
/*
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN  DD DDNAME=SYSIN
```

```
//*
//SYSIN    DD *
%TXT2PDF BROWSE Y IN DD:INDD +
OUT 'AWS.M2.TXT2PDF2.PDF.VB' +
CC YES
/*
//*
//*-------------------------------------------------------------------**
//* CONVERT PDF (VB) TO PDF (LSEQ - BYTE STREAM)*
//*-------------------------------------------------------------------**
//*
//STEP02 EXEC PGM=VB2LSEQ
//*
//INFILE   DD DSN=AWS.M2.TXT2PDF2.PDF.VB,DISP=SHR
//*
//OUTFILE  DD DSN=AWS.M2.TXT2PDF2.PDF,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(LRECL=256,DSORG=PS,RECFM=LSEQ,BLKSIZE=0)
//*
//SYSOUT   DD SYSOUT=*
//*
//
```

Modificaciones

Para que el programa TXT2 PDF se ejecute en el entorno de ejecución de AWS Mainframe Modernization Rocket Software, realizamos los siguientes cambios:

- Cambios en el código fuente para garantizar la compatibilidad con el entorno de ejecución REXX de Rocket Software
- Cambios para garantizar que el programa pueda ejecutarse en los sistemas operativos Windows y Linux
- Modificaciones para admitir el tiempo de ejecución en EBCDIC y ASCII

Referencias

TXT2Referencias en PDF y código fuente:

- Convertidor de texto a PDF
- z/OS Herramientas gratuitas de correo y TCP/IP

- TXT2Guía de referencia para el usuario en PDF

Utilidad por lotes M2DFUTIL

M2DFUTIL es un programa de utilidad de JCL que proporciona funciones de copia de seguridad, restauración, eliminación y copia en conjuntos de datos, de forma similar al soporte que proporciona la utilidad ADRDSSU del mainframe. Este programa conserva muchos de los parámetros SYSIN de ADRDSSU, lo que agiliza el proceso de migración a esta nueva utilidad.

Temas

- Plataformas admitidas
- Requisitos de la plataforma
- Compatibilidad futura planificada
- Ubicaciones de los activos
- Configurar el tiempo de ejecución de M2DFUTIL o AWS Mainframe Modernization en Amazon EC2 (incluida la versión 2.0) AppStream
- Sintaxis general
- Ejemplo JCLs

Plataformas admitidas

Puede utilizar M2DFUTIL en cualquiera de las siguientes plataformas:

- Rocket Software (anteriormente Micro Focus) ES en Windows (64 y 32 bits)
- Rocket Software ES en Linux (64 bits)

Requisitos de la plataforma

M2DFUTIL depende de llamar a un script para realizar una prueba de expresión regular. En Windows, debe instalar los Windows Services for Linux (WSL) para que se ejecute este script.

Compatibilidad futura planificada

Entre las funciones que actualmente no están disponibles en la utilidad ADRDSSU del mainframe, pero que estarán disponibles en el futuro, se incluyen:

- M2 administrado
- VSAM
- Compatibilidad con COPY para cambiar el nombre de los archivos
- Compatibilidad con RENAME para RESTORE
- Múltiples INCLUDE y EXCLUDE
- Cláusula BY para la subselección mediante DSORG, CREDT, EXPDT
- Cláusula MWAIT para reintentar errores en la cola
- Compatibilidad con el almacenamiento de S3 para DUMP/RESTORE

Ubicaciones de los activos

El módulo de carga de esta utilidad se llama `M2DFUTIL.so` en Linux y `M2DFUTIL.dll` en Windows. Este módulo de carga se puede encontrar en las siguientes ubicaciones:

- Linux: `/opt/aws/m2/microfocus/utilities/64bit`
- Windows de 32 bits: `C:\AWS\M2\MicroFocus\Utilities\32bit`
- Windows de 64 bits: `C:\AWS\M2\MicroFocus\Utilities\64bit`

El script utilizado para las pruebas de expresiones regulares se llama `compare.sh`. Este script se puede encontrar en las siguientes ubicaciones:

- Linux: `/opt/aws/m2/microfocus/utilities/scripts`
- Windows de 32 bits: `C:\AWS\M2\MicroFocus\Utilities\scripts`

Configurar el tiempo de ejecución de M2DFUTIL o AWS Mainframe Modernization en Amazon EC2 (incluida la versión 2.0) AppStream

Configure su región de Enterprise Server con lo siguiente:

- Agregue las siguientes variables en [ES-Environment]
  - `M2DFUTILS_BASE_LOC`: la ubicación predeterminada para la salida de DUMP
  - `M2DFUTILS_SCRIPTPATH`: la ubicación del script `compare.sh` documentada en Ubicaciones de activos

- `M2DFUTILS_VERBOSE`: [DETALLADO o NORMAL]. Esto controla el nivel de detalle de la salida `SYSPRINT`

- Compruebe que la ruta del módulo de carga se haya agregado a la configuración de `JES \Configuration\JES Program Path`
- Compruebe que los scripts del directorio de utilidades tengan permisos de ejecución. Puede agregar un permiso de ejecución mediante el comando `chmod + x <script name>`, en el entorno de Linux

Sintaxis general

DUMP

Ofrece la posibilidad de copiar archivos desde la ubicación catalogada actual a una ubicación de copia de seguridad. Actualmente, esta ubicación debe ser un sistema de archivos.

Proceso

DUMP realizará las siguientes tareas:

1. Crear el directorio de ubicaciones de destino.
2. Catalogar el directorio de ubicaciones de destino como miembro del PDS.
3. Determinar los archivos que se van a incluir procesando el parámetro INCLUDE.
4. Anular la selección de los archivos incluidos procesando el parámetro EXCLUDE.
5. Determinar si los archivos que se van a volcar deben eliminarse con DELETED.
6. Colocar en la cola los archivos que se procesarán.
7. Copiar los archivos.
8. Exportar la información DCB catalogada de los archivos copiados a un archivo lateral en la ubicación de destino para facilitar las futuras operaciones de RESTORE.

Sintaxis

```
DUMP
TARGET ( TARGET LOCATION  )    -
INCLUDE ( DSN. )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
```

```
[ DELETE ]
```

Parámetros necesarios

A continuación se enumeran los parámetros necesarios para el DUMP:

- `SYSPRINT DD NAME`: para contener información de registro adicional
- `TARGET`: ubicación de destino. Puede ser alguno de los siguientes:
  - Ruta completa de la ubicación del volcado
  - Nombre del subdirectorio creado en la ubicación definida en la variable M2DFUTILS_BASE_LOC
- `INCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida
- `EXCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida

Parámetros opcionales

- CANCEL: se cancela si se produce algún error. Los archivos que se procesaron se conservarán
- (Predeterminado) IGNORE: ignora cualquier error y proceso hasta el final
- DELETE: si no se produce ningún error de ENQ, el archivo se elimina y no se cataloga

DELETE

Ofrece la posibilidad de eliminar y descatalogar archivos de forma masiva. No se hace una copia de seguridad de los archivos.

Proceso

DELETE realizará las siguientes tareas:

1. Determinar los archivos que se van a incluir procesando el parámetro INCLUDE.
2. Anular la selección de los archivos incluidos procesando el parámetro EXCLUDE.
3. Colocar en la cola los archivos que se procesarán. Establecer la disposición como OLD, DELETE, KEEP.

Sintaxis

```
DELETE
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ DELETE ]
```

Parámetros necesarios

A continuación aparecen los parámetros necesarios para DELETE:

- `SYSPRINT DD NAME`: para contener información de registro adicional
- `INCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida
- `EXCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida

Parámetros opcionales

- CANCEL: se cancela si se produce algún error. Los archivos que se procesen se conservarán
- (Predeterminado) IGNORE: ignora cualquier error y proceso hasta el final

RESTORE

Ofrece la posibilidad de restaurar archivos de los que se haya hecho una copia de seguridad previa mediante DUMP. Los archivos se restauran en la ubicación catalogada original, a menos que se utilice RENAME para modificar el DSNAME restaurado.

Proceso

RESTORE realizará las siguientes tareas:

1. Valide el directorio de ubicaciones de origen.
2. Determine los archivos que se van a incluir procesando el archivo de exportación del catálogo.
3. Anular la selección de los archivos incluidos procesando el parámetro EXCLUDE.
4. Colocar en la cola los archivos que se procesarán.

5. Archivos de catálogo que no están catalogados en función de su información de exportación.
6. Si un archivo ya está catalogado y la información del catálogo de exportación es la misma, RESTORE reemplazará el conjunto de datos catalogado si se ha activado la opción REPLACE.

Sintaxis

```
RESTORE
SOURCE ( TARGET LOCATION )
INCLUDE ( DSN )
[ EXCLUDE ( DSN ) ]
[ CANCEL | IGNORE ]
[ REPLACE]
```

Parámetros necesarios

Los siguientes son los parámetros necesarios para RESTORE:

- `SYSPRINT DD NAME`: para contener información de registro adicional
- `SOURCE`: ubicación del origen. Puede ser alguno de los siguientes:
  - Ruta completa de la ubicación del volcado
  - Nombre del subdirectorio creado en la ubicación definida en la variable M2DFUTILS_BASE_LOC
- `INCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida
- `EXCLUDE`: una cadena de búsqueda DSNAME única con nombre o una cadena de búsqueda DSN de mainframe válida

Parámetros opcionales

- CANCEL: se cancela si hay algún error. Archivos procesados retenidos
- (Predeterminado) IGNORE: ignora cualquier error y proceso hasta el final
- REPLACE: si el archivo que se está restaurando ya está catalogado y los registros del catálogo son los mismos, sustituye el archivo catalogado

Ejemplo JCLs

Trabajo DUMP

Este trabajo creará un subdirectorio llamado TESTDUMP. Esta es la ubicación de copia de seguridad predeterminada especificada por la variable M2DFUTILS_BASE_LOC. Creará una biblioteca PDS para esta copia de seguridad llamada M2DFUTILS.TESTDUMP. Los datos del catálogo exportados se almacenan en un archivo secuencial de líneas en el directorio de copias de seguridad denominado CATDUMP.DAT. Todos los archivos seleccionados se copiarán en este directorio de copia de seguridad.

```
//M2DFDMP JOB 'M2DFDMP',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDUMP.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSIN    DD *
DUMP TARGET(TESTDUMP)               -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
/*
//
```

Trabajo DELETE

Este trabajo eliminará todos los archivos del catálogo que coincidan con el parámetro INCLUDE.

```
/M2DFDEL JOB 'M2DFDEL',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
//SYSPRINT DD DSN=TESTDEL.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
  DELETE                            -
     INCLUDE(TEST.FB.FILE*.ABC)     -
 CANCEL
/*
 //
```

Trabajo RESTORE

Este trabajo restaurará los archivos que coincidan con el parámetro INCLUDE desde la ubicación de la copia de seguridad de TESTDUMP. Los archivos catalogados se reemplazarán si el archivo catalogado es el mismo que el de la exportación a CATDUMP y se especifica la opción REPLACE.

```
//M2DFREST JOB 'M2DFREST',CLASS=A,MSGCLASS=X
//STEP001  EXEC PGM=M2DFUTIL
////SYSPRINT DD DSN=TESTREST.SYSPRINT,
//        DISP=(NEW,CATLG,DELETE),
//        DCB=(RECFM=LSEQ,LRECL=256)
//SYSPRINT DD SYSOUT=A
//SYSIN    DD *
RESTORE SOURCE(TESTDUMP)               -
     INCLUDE(TEST.FB.FILE*.ABC)        -
 IGNORE
 REPLACE
/*
//
```

Utilidad por lotes M2RUNCMD

Puede utilizar M2RUNCMD, un programa de utilidad por lotes, para ejecutar comandos, scripts y llamadas al sistema de Rocket Software (anteriormente Micro Focus) directamente desde JCL, en lugar de ejecutarlos desde un terminal o una línea de comandos. El resultado de los comandos se registra en el registro spool del trabajo por lotes.

Temas

- Plataformas admitidas
- Configurar M2RUNCMD para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la versión 2.0) AppStream
- Ejemplo JCLs

Plataformas admitidas

Puede utilizar M2RUNCMD en las siguientes plataformas:

- Rocket Software Runtime (en Amazon EC2)
- Todas las variantes de los productos Rocket Software Enterprise Developer (ED) y Rocket Software Enterprise Server (ES).

Configurar M2RUNCMD para el tiempo de ejecución de modernización de AWS mainframe en Amazon EC2 (incluida la versión 2.0) AppStream

Si las aplicaciones migradas se ejecutan en el entorno de ejecución de modernización de AWS mainframe en Amazon EC2, configure M2RUNCMD de la siguiente manera.

- Cambie la ruta del programa JES de Micro Focus para incluir la ubicación binaria de las utilidades por lotes. Si necesita especificar varias rutas, utilice dos puntos (:) para separarlas en Linux y punto y coma (;) en Windows.
  - Linux: `/opt/aws/m2/microfocus/utilities/64bit`
  - Windows (32 bits): `C:\AWS\M2\MicroFocus\Utilities\32bit`
  - Windows (64 bits): `C:\AWS\M2\MicroFocus\Utilities\64bit`

Ejemplo JCLs

Para probar la instalación, puede utilizar cualquiera de los siguientes ejemplos JCLs.

RUNSCRL1.jcl

En este JCL de ejemplo se crea un script y se ejecuta. El primer paso crea un script denominado `/tmp/TEST_SCRIPT.sh` y con contenido a partir de datos de `SYSUT1` internos. El segundo paso establece el permiso de ejecución y ejecuta el script creado en el primer paso. También puedes elegir realizar solo el segundo paso para ejecutar el software Rocket y los comandos del sistema ya existentes.

```
//RUNSCRL1 JOB 'RUN SCRIPT',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*  CREATE SCRIPT (LINUX)
//*--------------------------------------------------------------------*
//*
//STEP0010 EXEC PGM=IEBGENER
//*
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//*
//SYSUT1   DD *
#!/bin/bash
```

```
set -x

## ECHO PATH ENVIRONMNET VARIABLE
echo $PATH

## CLOSE/DISABLE VSAM FILE
casfile -r$ES_SERVER -oc  -ed -dACCTFIL

## OPEN/ENABLE VSAM FILE
casfile -r$ES_SERVER -ooi -ee -dACCTFIL

exit $?
/*
//SYSUT2   DD DSN=&&TEMP,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=LSEQ,LRECL=300,DSORG=PS,BLKSIZE=0)
//*MFE: %PCDSN='/tmp/TEST_SCRIPT.sh'
//*
//*--------------------------------------------------------------------*
//*   RUN SCRIPT (LINUX)                                               *
//*--------------------------------------------------------------------*
//*
//STEP0020 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*RUN SCRIPT
 sh /tmp/TEST_SCRIPT.sh
/*
//
```

SYSOUT

El resultado del comando o script que se ejecuta se escribe en el registro SYS0UT. Para cada comando ejecutado, muestra el comando, la salida y el código de retorno.

```
************ CMD Start ************

CMD_STR: sh /tmp/TEST_SCRIPT.sh
```

```
CMD_OUT:


+ echo /opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
/opt/microfocus/EnterpriseServer/bin:/sbin:/bin:/usr/sbin:/usr/bin
+ casfile -rMYDEV -oc -ed -dACCTFIL

-Return Code:   0

Highest return code:    0

+ casfile -rMYDEV -ooi -ee -dACCTFIL

-Return Code:   8

Highest return code:    8

+ exit 8


CMD_RC=8


************  CMD End  ************
```

RUNCMDL1.jcl

Este ejemplo de JCL usa RUNCMD para ejecutar varios comandos.

```
//RUNCMDL1 JOB 'RUN CMD',CLASS=A,MSGCLASS=X,TIME=1440
//*
//*
//*--------------------------------------------------------------------*
//*   RUN SYSTEM COMMANDS                                              *
//*--------------------------------------------------------------------*
//*
//STEP0001 EXEC PGM=RUNCMD
//*
//SYSOUT  DD  SYSOUT=*
//*
//SYSIN   DD *
*LIST DIRECTORY
 ls
*ECHO PATH ENVIRONMNET VARIABLE
```

```
 echo $PATH
/*
//
```

Transferencia de archivos en la modernización del AWS mainframe

La Transferencia de archivos de AWS Mainframe Modernization le permite transferir y convertir conjuntos de datos de mainframe a Amazon S3 para casos de uso de modernización, migración y aumento de mainframes. Simplifica el proceso de la transferencia de conjuntos de datos del mainframe a la Nube de AWS. Entre las características clave se incluyen el descubrimiento de conjuntos de datos y artefactos de mainframe de origen y la escalabilidad y eficiencia para transferencias de datos más rápidas a Amazon S3. File Transfer admite varios tipos de conjuntos de datos de mainframe, como secuenciales, PDS, GDS, GDG y KSDS de VSAM. El servicio transfiere los conjuntos de datos a un bucket intermedio de Amazon S3, los convierte en la página de código de destino especificada y, a continuación, los mueve al bucket de S3 de destino que desee.

Temas

- ¿Qué es AWS Mainframe Modernization File Transfer?
- Instalación de un agente de File Transfer
- Configuración del agente de File Transfer
- Cree puntos de conexión de transferencia de datos para File Transfer
- Creación de tareas de transferencia en File Transfer
- Tutorial: Introducción a AWS Mainframe Modernization File Transfer
- Codificaciones de origen y destino admitidas en la Transferencia de archivos de AWS Mainframe Modernization

¿Qué es AWS Mainframe Modernization File Transfer?

Con AWS Mainframe Modernization File Transfer, puede transferir y convertir conjuntos de datos y archivos con un servicio totalmente gestionado para acelerar y simplificar los casos de uso de modernización, migración y aumento al servicio AWS Mainframe Modernization y Amazon S3.

Temas

- Ventajas de AWS Mainframe Modernization File Transfer
- Cómo funciona AWS Mainframe Modernization File Transfer

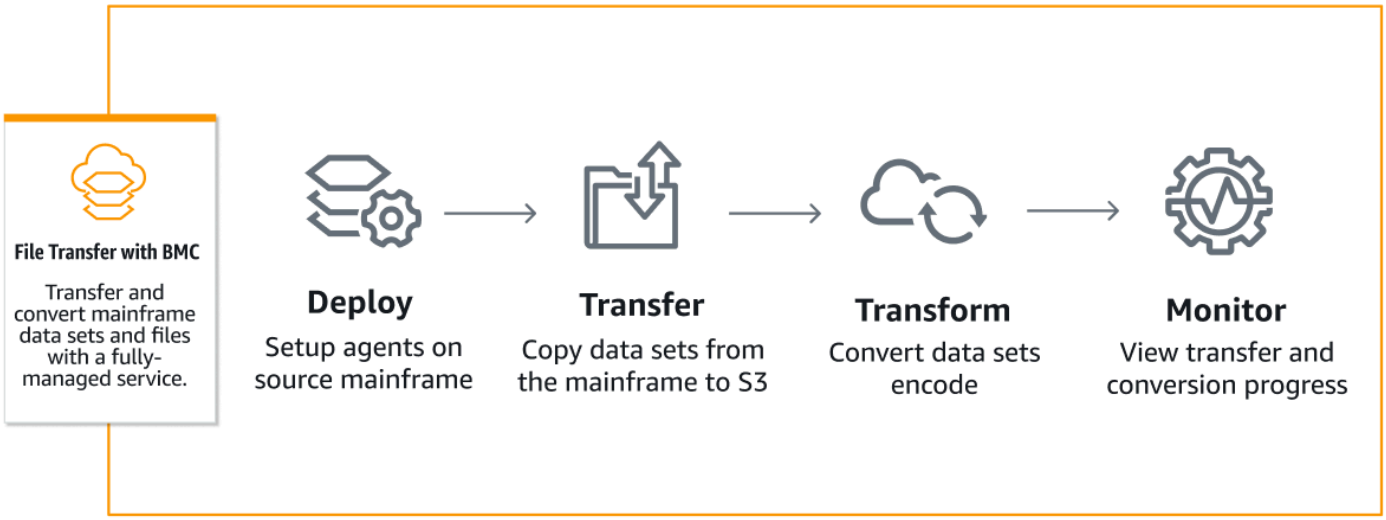
Ventajas de AWS Mainframe Modernization File Transfer

AWS Mainframe Modernization File Transfer le ayuda a transferir conjuntos de datos del mainframe a Amazon S3. Algunas de sus ventajas son:

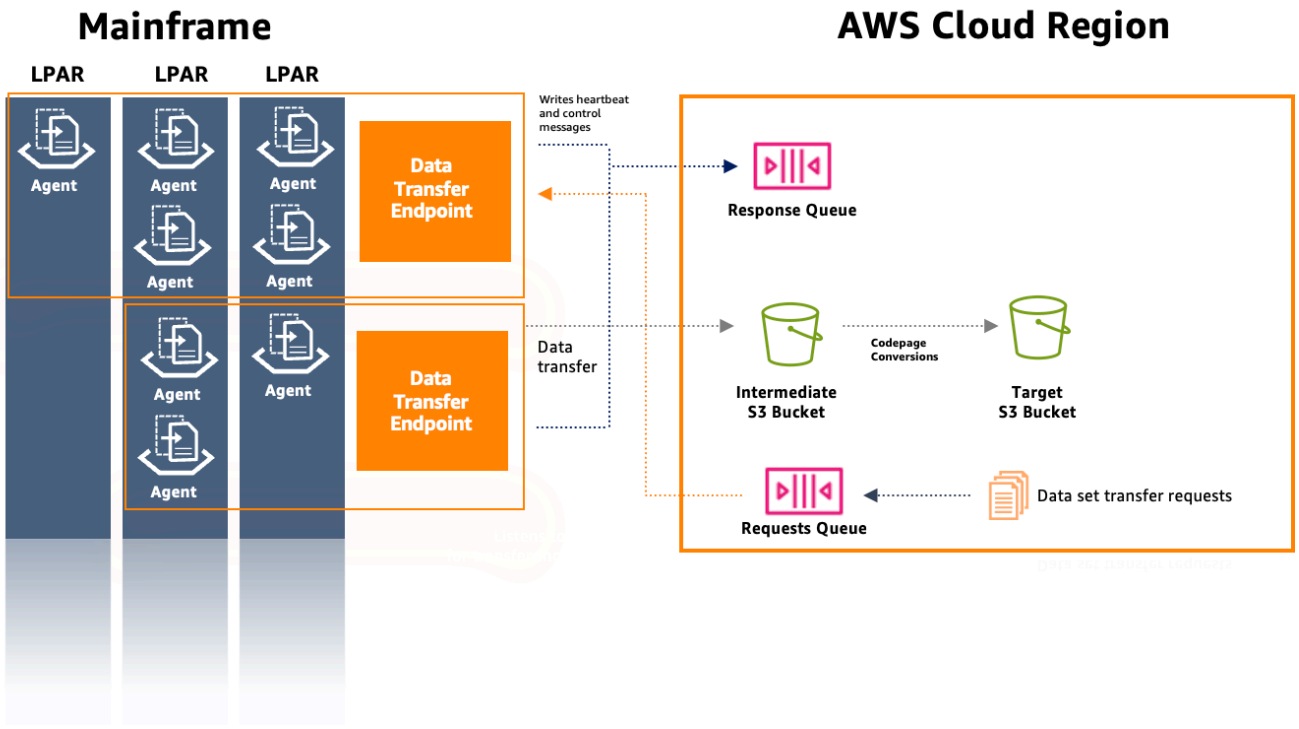
- Detección de conjuntos de datos y artefactos de mainframe de origen
- Transferencia y conversión de conjuntos de datos automatizada
- Escalabilidad, eficiencia y velocidad para lograr transferencias de conjuntos de datos a AWS más rápidas

Cómo funciona AWS Mainframe Modernization File Transfer

En la siguiente figura se muestra una descripción general de cómo funciona AWS Mainframe Modernization File Transfer en un nivel conceptual.



La siguiente figura es una descripción general de la arquitectura de la característica AWS Mainframe Modernization File Transfer.



Instalación de un agente de File Transfer

Puede utilizar este documento como step-by-step guía para instalar un agente en el mainframe de origen.

> **Note**
>
> Esta guía está destinada únicamente a los programadores de sistemas de mainframe.

Temas

- Paso 1: creación de un conjunto de datos de zFS para M2-agent
- Paso 2: formateo del conjunto de datos como zFS
- Paso 3: montaje del sistema de archivos
- Paso 4: verificación del montaje
- Paso 5: introducción de OMV
- Paso 6: establecimiento de la variable de entorno del directorio de instalación del agente
- Paso 7: establecimiento de la variable de entorno del directorio de trabajo

- Paso 8: creación del directorio de trabajo
- Paso 9: copia del archivo tar del agente y del directorio de trabajo
- Paso 10: Finalizar la instalación del agente

Paso 1: creación de un conjunto de datos de zFS para M2-agent

Cree un zFS para la instalación de M2-agent utilizando el JCL (Lenguaje de control de tareas) que se muestra a continuación:

```
DEFINE   EXEC   PGM=IDCAMS
SYSPRINT DD     SYSOUT=A
SYSIN    DD     *
 DEFINE CLUSTER (NAME(yourhlq.M2AGENT.ZFS) -
 VOLUMES(*) -
 LINEAR CYL(1000 200))
```

Paso 2: formateo del conjunto de datos como zFS

Después de crear el conjunto de datos, formatéelo como sistema de archivos zFS.

Una manera de hacerlo es mediante el siguiente JCL:

```
FORMAT  EXEC PGM=IOEAGFMT,
PARM=('-aggregate yourhlq.M2AGENT.ZFS -size 1200') ,
SYSPRINT DD  SYSOUT=*
```

Envíe este trabajo y compruebe si se ha completado correctamente.

Paso 3: montaje del sistema de archivos

Para montar el sistema de archivos, utilice el comando `MOUNT`. Puede montar el sistema de archivos en la línea de comandos, en la ISPF o por lotes.

Por ejemplo:

```
MOUNT FILESYSTEM('yourhlq.M2AGENT.ZFS') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/usr/lpp/aws/
m2-agent')
```

Utilizará este punto de montaje en el paso 6.

> **Note**
>
> La definición de la ruta de montaje es opcional y, para ello, debe utilizar un directorio existente.

Paso 4: verificación del montaje

Verifique que el sistema de archivos esté montado correctamente mediante un comando `D OMVS,F` o comprobándolo en el Servicio de Sistema Unix (USS).

Paso 5: introducción de OMV

Utilice el siguiente comando para introducir OMV:

```
TSO OMVS
```

Paso 6: establecimiento de la variable de entorno del directorio de instalación del agente

Utilice el siguiente comando para definir el entorno del directorio de instalación del agente:

```
export AGENT_DIR=/usr/lpp/aws/m2-agent
```

> **Note**
>
> El punto de montaje se define en el paso 3.

Paso 7: establecimiento de la variable de entorno del directorio de trabajo

Utilice el siguiente comando para definir la variable de entorno del directorio de trabajo:

```
export WORK_DIR=$AGENT_DIR/tmp
```

Paso 8: creación del directorio de trabajo

Utilice el siguiente comando para definir el entorno del directorio de trabajo:

```
mkdir -p $WORK_DIR
```

Paso 9: copia del archivo tar del agente y del directorio de trabajo

Descargue el archivo tar del agente de AWS mediante el enlace del agente M2.

El mecanismo de transferencia dependerá del entorno, pero asegúrese de que el archivo tar se transfiera en modo binario.

Paso 10: Finalizar la instalación del agente

Siga estos pasos para finalizar la instalación del agente:

1. Establezca la variable de entorno de la versión m2-agent en la versión que se esté instalando actualmente mediante el siguiente comando:

   ```
   export M2_AGENT_VERSION=1.0.0
   ```

2. Extraiga el paquete tar del agente con el siguiente comando:

   ```
   tar -xpf m2-agent-$M2_AGENT_VERSION.tar -C $AGENT_DIR
   ```

3. Cree un enlace simbólico `current-version` al directorio de instalación del agente actual con el siguiente comando:

   ```
   ln -s $AGENT_DIR/m2-agent-v$M2_AGENT_VERSION $AGENT_DIR/current-version
   ```

4. Actualice y envíe `CPY#PDS` para crear los conjuntos de datos del agente de File Transfer.

   > **Note**
   >
   > JCL usa `SYS2.AWS.M2 HLQ`.

   Para crear el agente de File Transfer, actualice las tres variables simbólicas HLQ (calificador de alto nivel), `VOLSER` y `AGNTPATH` para usarlas más adelante en el JCL:

   ```
   oedit $AGENT_DIR/current-version/installation/CPY#PDS
   ```

> **Note**
>
> Este JCL está diseñado para configurar ciertos aspectos de la instalación del agente en el mainframe. Asigna los conjuntos de datos necesarios y, a continuación, copia archivos específicos del sistema de archivos Unix a estos conjuntos de datos.

Configuración del agente de File Transfer

Una vez que haya instalado un agente de transferencia de archivos, siga estos pasos para configurarlo. Si necesita instalar un agente nuevo, siga las instrucciones de la página the section called “Instalación de un agente de File Transfer”.

Temas

- Paso 1: configuración de los permisos y control de tarea iniciada (STC)
- Paso 2: creación de buckets de Amazon S3
- Paso 3: Cree una clave de cifrado gestionada por el AWS KMS cliente
- Paso 4: Cree un AWS Secrets Manager secreto para las credenciales del mainframe
- Paso 5: creación de una política de IAM
- Paso 6: creación de un usuario de IAM con credenciales de acceso a largo plazo
- Paso 7: creación de un rol de IAM para que lo asuma el agente
- Paso 8: configuración del agente

Paso 1: configuración de los permisos y control de tarea iniciada (STC)

1. Actualice y envíe o bien `SYS2.AWS.M2.SAMPLIB(SEC#RACF)` (para configurar los permisos de RACF) o bien `SYS2.AWS.M2.SAMPLIB(SEC#TSS)` (para configurar los permisos de TSS) de acuerdo con sus instrucciones. Estos miembros se crearon en el paso `CPY#PDS` anterior.

   > **Note**
   >
   > `SYS2.AWS.M2`debe sustituirse por el calificador de alto nivel (HLQ) elegido durante la instalación.

2. Actualice la exportación de PWD en el `SYS2.AWS.M2.SAMPLIB(M2AGENT)` STC JCL si se ha cambiado la ruta de directorio predeterminada del agente de File Transfer (`/usr/lpp/aws/m2-agent`).
3. Actualice el PROC de acuerdo con los estándares de su sitio:
   a. Actualice la tarjeta PROC según sus requisitos de instalación.
   b. Actualice el STEPLIB con el. `M2 LOADLIB PDSE ALIAS`
   c. Edite el PWD para indicar la ruta de instalación del agente (solo se incluye esto).
   d. Actualice `JAVA_HOME` si es necesario.
4. Actualiza y copia el `SYS2.AWS.M2.SAMPLIB(M2AGENT)` JCL a tu `PROCLIB` concatenación `SYS1.PROCLIB` o a uno de ellos. PROCLIBs
5. Añada `SYS2.AWS.M2.LOADLIB` a la lista de APF utilizando el siguiente comando:

```
SETPROG APF ADD DSNAME(SYS2.AWS.M2.LOADLIB) SMS
```

6. Establezca el grupo y el propietario del agente en el agente (2GROUP). user/group (M2USER/M Utilice el siguiente comando en el OMVS:

```
chown -R M2USER:M2GROUP $AGENT_DIR/current-version
```

> **Note**
>
> Edite M2USER y M2GROUP con los nombres que utilizó en el trabajo de definiciones de seguridad.

Paso 2: creación de buckets de Amazon S3

La Transferencia de archivos de AWS Mainframe Modernization requiere un bucket intermedio de Amazon S3 como área de trabajo. Recomendamos crear un bucket específico para esto.

Si lo desea, cree un nuevo bucket de Amazon S3 de destino para los conjuntos de datos transferidos. De lo contrario, también puede usar su bucket de Amazon S3 existente. Para obtener información sobre la creación de un bucket de Amazon S3, consulte Crear un bucket.

Paso 3: Cree una clave de cifrado gestionada por el AWS KMS cliente

Para crear una clave gestionada por el cliente en AWS KMS

1. Abra la AWS KMS consola enhttps://console.aws.amazon.com/kms.
2. En el panel de navegación izquierdo, elija Claves administradas por el cliente.
3. Elija Crear clave.
4. En Configurar clave, seleccione Tipo de clave como simétrico y Uso de clave como Cifrar y descifrar. Utilice otras configuraciones predeterminadas.
5. Elija Siguiente.
6. En Agregar etiquetas, agrega el alias y la descripción de la clave.
7. Elija Siguiente.
8. En Definir permisos de administración de claves, elija al menos un usuario y un rol de IAM que administre esta clave.
9. Elija Siguiente.
10. Si lo desea, en Definir permisos administrativos clave, elija al menos un usuario y un rol de IAM que pueda usar esta clave.
11. Elija Siguiente.
12. En la sección Editar política clave, elija Editar y añada la siguiente sintaxis a la política clave. Esto permite que el servicio de modernización del AWS mainframe lea y utilice estas claves para el cifrado y el descifrado.

> **Important**
>
> Agregue la instrucción a las instrucciones existentes. No reemplace lo que ya está en la política.

```
{
    "Sid" : "Enable AWS M2 File Transfer Permissions",
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
    },
    "Action" : [
```

```
        "kms:Encrypt",
        "kms:Decrypt"
    ],
   "Resource" : "*"
},
```

13. Elija Siguiente.
14. En la página de revisión, compruebe todos los detalles y seleccione Finalizar.

Copie y guarde el ARN de la clave gestionada por el cliente abriendo la clave KMS recién creada. Se utilizará en la política más adelante.

Paso 4: Cree un AWS Secrets Manager secreto para las credenciales del mainframe

Las credenciales del mainframe son necesarias para acceder a los conjuntos de datos que se van a transferir y estas deben almacenarse en secreto. AWS Secrets Manager

Para crear un secreto AWS Secrets Manager

1. Abra la consola de Secrets Manager en https://console.aws.amazon.com/secretsmanager.
2. Elija Almacenar un secreto nuevo.
3. En Elija tipo de secreto, elija Otro tipo de secreto.
4. Utilice el valor clave `userId` para el UserID del mainframe que tiene acceso a los conjuntos de datos. Utilice el valor de clave `password` para el campo de contraseña.
5. Para la clave de cifrado, elija la clave gestionada por el AWS cliente que se creó anteriormente.
6. Elija Siguiente.
7. En la página Configurar regla, escriba un nombre y una descripción.
8. En la misma página, edite los permisos de los recursos y utilice la siguiente política de recursos para que el servicio de modernización del AWS mainframe pueda acceder a ellos.

```
{
  "Version" : "2012-10-17",
  "Statement" : [ {
    "Effect" : "Allow",
    "Principal" : {
        "Service" : "m2.amazonaws.com"
```

```
    },
    "Action" : [ "secretsmanager:GetSecretValue",
                 "secretsmanager:DescribeSecret" ],
    "Resource" : "*"
  } ]
}
```

9. Seleccione Guardar para guardar los permisos actualizados.
10. Elija Siguiente.
11. Vaya a la página Configurar rotaciones y seleccione Siguiente.
12. En la página Revisar, compruebe todas las configuraciones y elija Almacenar para guardar el secreto.

> **Important**
>
> Las claves secretas `userId` y `password` distinguen mayúsculas de minúsculas y se deben introducir tal y como se muestra.

Paso 5: creación de una política de IAM

Creación de una nueva política con los permisos necesarios para el agente

1. Abra la consola de IAM en https://console.aws.amazon.com/iam.
2. Elija Políticas en Administración de acceso.
3. Elija Crear política.
4. En la página Especificar permisos, en el editor de políticas, cambie del editor visual al editor JSON y sustituya el contenido por la siguiente plantilla:
5.

```
{
"Version": "2012-10-17",
    "Statement": [
        {
         "Sid": "FileTransferAgentSQSReceive",
            "Effect": "Allow",
            "Action": [
                "sqs:DeleteMessage",
                "sqs:ReceiveMessage"
```

```
            ],
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-request-queue.fifo"
        },
        {
         "Sid": "FileTransferAgentSQSSend",
            "Effect": "Allow",
            "Action": "sqs:SendMessage",
            "Resource": "arn:aws:sqs:*:111122223333:m2-*-response-queue.fifo"
        },
        {
         "Sid": "FileTransferWorkingS3",
            "Effect": "Allow",
            "Action": "s3:PutObject",
            "Resource": "<file-transfer-endpoint-intermediate-bucket-arn>/*"
        },
        {
         "Sid": "FileTransferAgentKMSDecrypt",
            "Effect": "Allow",
            "Action": "kms:Decrypt",
            "Resource": "<kms-key-arn>"
        }
    ]
}
```

6. Reemplace 111122223333 en los ARN de la cola de respuesta y la cola de solicitudes por su cuenta.

> **Note**
>
> Son ARN comodín que coinciden con las dos colas de Amazon SQS creadas durante la inicialización del punto de conexión de transferencia de datos. Tras crear un punto de conexión de File Transfer, si lo desea, reemplace estos ARN por los valores reales de Amazon SQS.

7. Reemplace `file-transfer-endpoint-intermediate-bucket-arn` por el ARN del bucket de transferencia creado anteriormente. Deje el comodín /* al final.
8. `kms-key-arn`Sustitúyala por el ARN de la AWS KMS clave creada anteriormente.
9. Elija Siguiente.
10. En la página Revisar y crear, añada el nombre y la descripción de la política.
11. Elija Crear política.

Paso 6: creación de un usuario de IAM con credenciales de acceso a largo plazo

Cree un usuario de IAM que permita que el agente de mainframe se conecte a su AWS cuenta. El agente se conectará con este usuario y, a continuación, asumirá un rol que se defina con permisos para usar las colas de respuestas y solicitudes de Amazon SQS y guardar los conjuntos de datos en los buckets de Amazon S3.

Creación de este usuario de IAM

1. Vaya a la consola de IAM en https://console.aws.amazon.com/iam.
2. Elija Usuarios en Administración de acceso.
3. Seleccione la opción Crear un usuario.
4. Agregue un nombre de usuario significativo en Detalles del usuario. Por ejemplo, `Configure-ft-agent`.
5. Elija Siguiente.
6. En Opciones de permisos, elija la opción Adjuntar políticas directamente, pero no asocie ninguna política de permisos. Estos permisos los administrará un rol que se asociará.
7. Elija Siguiente.
8. Revise los detalles y elija Crear usuario.
9. Una vez creado el usuario, elija el usuario y abra la pestaña Credenciales de seguridad.
10. En Clave de acceso, elija Crear clave de acceso.
11. A continuación, selecciona Otro cuando se te pida que escribas Caso de uso.
12. Elija Siguiente.
13. Si lo desea, puede establecer una etiqueta descriptiva como,`Access key for configuring file transfer agent`.
14. Elija Create access key (Crear clave de acceso).
15. Copie y guarde de forma segura la clave de acceso generada y la clave de acceso secreta. Se utilizarán más adelante.

A fin de obtener más información acerca de la creación de claves de acceso de IAM, consulte Administración de claves de acceso para usuarios de IAM.

> **Important**
>
> Guarde la Clave de acceso y la Clave de acceso secreta que aparecen en la última página del asistente de creación de claves de acceso antes de seleccionar Listo. Estas claves se utilizan para configurar el agente de mainframe y no se pueden recuperar más adelante.

> **Note**
>
> Guarde el ARN del usuario de IAM utilizado para configurar una relación de confianza con un rol de IAM.

Paso 7: creación de un rol de IAM para que lo asuma el agente

Creación de un nuevo rol de IAM para el agente

1. Elija Roles en la cola de IAM en https://console.aws.amazon.com/iam.
2. Seleccione Crear rol.
3. En la página Seleccionar entidad de confianza, elija Política de confianza personalizada para el Tipo de entidad de confianza.
4. Reemplace la política de confianza personalizada por la siguiente y sustituya `<iam-user-arn>` por el ARN del usuario que se ha creado anteriormente.

```
{
    "Version": "2012-10-17",
    "Statement": [ {
         "Sid": "FileTransferAgent",
         "Effect": "Allow",
         "Principal": {
            "AWS": "<IAM-User-arn>"
         },
         "Action": "sts:AssumeRole"
    } ]
}
```

5. Elija Siguiente.
6. En Agregar permisos, filtre por el Nombre de política que ha creado anteriormente y elíjalo.

7. Elija Siguiente.
8. Asigne al rol un nombre y elija Crear rol.

> **Note**
>
> Guarde el nombre de rol, que utilizará más adelante para configurar el agente de mainframe.

Paso 8: configuración del agente

Configuración del agente de File Transfer

1. Vaya a `$AGENT_DIR/current-version/config`.
2. Edite el archivo de configuración del agente `appication.properties` para agregar una configuración de entorno mediante el siguiente comando:

```
oedit $AGENT_DIR/current-version/config/application.properties
```

Por ejemplo:

```
agent.environments[0].account-id=<AWS_ACCOUNT_ID>
agent.environments[0].agent-role-name=<AWS_IAM_ROLE_NAME>
agent.environments[0].access-key-id=<AWS_IAM_ROLE_ACCESS_KEY>
agent.environments[0].secret-access-id=<AWS_IAM_ROLE_SECRET_KEY>
agent.environments[0].bucket-name=<AWS_S3_BUCKET_NAME>
agent.environments[0].environment-name=<AWS_REGION>
agent.environments[0].region=<AWS_REGION>
zos.complex-name=<File_Transfer_Endpoint_Name>
```

Donde:

- `AWS_ACCOUNT_ID`es el identificador de la AWS cuenta.
- `AWS_IAM_ROLE_NAME` es el nombre del rol de IAM creado en la the section called “Paso 7: creación de un rol de IAM para que lo asuma el agente”.
- `AWS_IAM_ROLE_ACCESS_KEY` es la clave de acceso del usuario de IAM creado en the section called “Paso 6: creación de un usuario de IAM con credenciales de acceso a largo plazo”.

- `AWS_IAM_ROLE_SECRET_KEY` es la clave secreta de acceso del usuario de IAM creado en the section called “Paso 6: creación de un usuario de IAM con credenciales de acceso a largo plazo”.
- `AWS_S3_BUCKET_NAME` es el nombre del bucket de transferencia creado con el punto de conexión de transferencia de datos.
- `AWS_REGION` es la región en la que se configura el agente de File Transfer.

> **Note**
>
> Puede hacer que el agente de transferencia de archivos se transfiera a varias regiones y cuentas AWS definiendo varios entornos.

- (Opcional). `zos.complex-name` es el nombre complejo que creó al crear un punto de conexión de File Transfer.

> **Note**
>
> Este campo solo es necesario si desea personalizar el nombre complejo (que de forma predeterminada es su nombre de sysplex), que es el mismo que definió al crear el punto de conexión de File Transfer. Para obtener más información, consulte the section called “Creación de puntos de conexión de transferencia de datos”.

> **Important**
>
> Puede haber varias secciones de este tipo, siempre que el índice entre corchetes, `[0]`, se incremente para cada una de ellas.

Debe reiniciar el agente para que los cambios surtan efecto.

Requisitos

1. Cuando se agrega o elimina un parámetro, hay que detener e iniciar el agente. Inicie el agente de File Transfer mediante el siguiente comando de la CLI:

```
/S M2AGENT
```

Para detener el agente de M2, ejecute el comando siguiente en la CLI:

```
/P M2AGENT
```

2. Puede configurar el agente de transferencia de archivos para transferir datos a varias regiones y cuentas AWS definiendo las entradas del entorno.

> **Note**
>
> Reemplace los valores por los valores de los parámetros que creó y configuró anteriormente.

```
#Region 1
agent.environments[0].account-id=AWS_ACCOUNT_ID
agent.environments[0].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[0].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[0].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[0].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[0].environment-name=AWS_REGION
agent.environments[0].region=AWS_REGION

#Region 2
agent.environments[1].account-id=AWS_ACCOUNT_ID
agent.environments[1].agent-role-name=AWS_IAM_ROLE_NAME
agent.environments[1].access-key-id=AWS_IAM_ROLE_ACCESS_KEY
agent.environments[1].secret-access-id=AWS_IAM_ROLE_SECRET_KEY
agent.environments[1].bucket-name=AWS_S3_BUCKET_NAME
agent.environments[1].environment-name=AWS_REGION
agent.environments[1].region=AWS_REGION
```

Cree puntos de conexión de transferencia de datos para File Transfer

Los puntos de conexión de transferencia de datos permiten la conectividad con el mainframe de origen y admiten alta disponibilidad, escalabilidad y una administración optimizada de los agentes. Los agentes individuales se instalan en el mainframe LPARs y se pueden agrupar en un punto final de transferencia de datos. Cuando se solicita la transferencia de un conjunto de datos, un agente del

punto de conexión de transferencia de datos se encargará de esa transferencia concreta. Para iniciar las transferencias de datos, al menos un agente del punto de conexión de transferencia de datos debe estar en línea.

Este procedimiento supone que ha completado los pasos de Configuración para la modernización del AWS mainframe y de Configurar el agente de File Transfer en el mainframe de origen.

Creación de puntos de conexión de transferencia de datos

Para crear puntos finales de transferencia de datos para la transferencia de archivos, siga estos pasos en la consola de modernización del AWS mainframe.

Para crear un punto de conexión de transferencia de datos

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región a la que quiere transferir los archivos del mainframe a un bucket de Amazon S3.
3. En la página Puntos de conexión de transferencia de datos, en File Transfer, elija Crear punto de conexión de transferencia de datos.
4. En la página Requisitos previos para el punto de conexión de transferencia de datos, lea todas las instrucciones para asegurarse de haber completado estos pasos en el mainframe de origen. Una vez confirmado, seleccione Siguiente.
5. En la página Configurar punto de conexión de transferencia de datos, agregue la información básica del punto de conexión de transferencia de datos.

    1. En la sección de información básica, introduzca el nombre del punto de conexión de transferencia de datos.

        > **Note**
        >
        > El nombre del punto de conexión de transferencia de datos debe coincidir con el nombre de Sysplex, a menos que especifique un nombre complejo en la configuración del agente.

    2. Una descripción opcional.
    3. La clave de KMS se utiliza para cifrar el secreto.

> **Note**
>
> Debe agregar la siguiente política basada en recursos para KMS para que el servicio de AWS Mainframe Modernization pueda leer y usar estas claves para el cifrado y el descifrado:

```
{
  "Sid" : "Enable AWS M2 Permissions",
  "Effect" : "Allow",
  "Principal" : {
      "Service" : "m2.amazonaws.com"
            },
   "Action" : [
       "kms:Encrypt",
       "kms:Decrypt"
              ],
   "Resource" : "*"
}
```

4. Especifique la Ubicación S3 para los datos intermedios, que es la ubicación de S3 intermedia donde se almacenan los conjuntos de datos transferidos desde el mainframe antes de convertirse y transferirse al bucket de Amazon S3 de destino.

   .

   > **Note**
   >
   > Se recomienda crear un nuevo bucket de Amazon S3 para las tareas de transferencia. Para obtener más información, consulte Creación de un bucket. También puede explorar sus buckets de Amazon S3 existentes seleccionando la opción Explorar S3.

5. Tras introducir los campos obligatorios, seleccione Siguiente.
6. En la página Revisar y crear un punto de conexión de transferencia de datos, compruebe si ha cumplido los requisitos previos y revise la información básica. Una vez confirmado, elija Crear punto de conexión de transferencia de datos.

Se le redirigirá a la página de Información general de los puntos finales de transferencia de datos, donde podrá ver la lista de todos los puntos de conexión de transferencia de datos. También podrá ver los puntos de conexión de transferencia de datos que están disponibles o que han fallado.

También puede buscar los puntos de conexión de transferencia de datos por nombre y acceder a información adicional sobre cada agente disponible.

Creación de tareas de transferencia en File Transfer

Las tareas de transferencia se utilizan para especificar los conjuntos de datos que se transferirán del mainframe a Amazon S3 y le permiten elegir las opciones de conversión de páginas de códigos.

En estas instrucciones se presupone que ha completado los pasos que se detallan en Configuración para la modernización del AWS mainframe y que ha creado the section called “Creación de puntos de conexión de transferencia de datos”.

Temas

- Creación de tareas de transferencia
- Visualización de las tareas de transferencia

Creación de tareas de transferencia

Para crear tareas de transferencia en File Transfer, siga estos pasos en la consola de modernización del AWS mainframe.

Para crear una tarea de transferencia

> **Important**
>
> Debe tener al menos un punto de conexión de transferencia de datos para crear nuevas tareas de transferencia.

1. Abra la consola de modernización AWS del mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región a la que desea transferir los archivos de su mainframe a un bucket de Amazon S3.
3. En la página Tareas de transferencia, puede elegir cualquier punto de conexión de transferencia de datos para crear tareas de transferencia.

4. En la página Crear tarea de transferencia, configure las propiedades de la tarea de transferencia. Si no ha creado ninguna tarea de transferencia anteriormente, puede crear la primera eligiendo la opción Crear tarea de transferencia.

   - En esta página, introduzca la información básica de la tarea de transferencia, como el nombre de la tarea de transferencia, la descripción y la clave secreta.

   > **Note**
   >
   > - Cifre el secreto con la clave KMS definida en el punto de conexión de transferencia de datos. El secreto debe contener las credenciales del mainframe necesarias para acceder a los conjuntos de datos del mainframe mediante las claves `userId` y `password`. Para obtener más información, consulte el secreto de AWS Secrets Manager.
   > - Debe configurar la clave secreta con la siguiente política basada en los recursos para que el servicio de modernización del AWS mainframe pueda acceder a ella para realizar la tarea de transferencia de datos.
   >
   > ```
   > {
   >   "Version" : "2012-10-17",
   >   "Statement" : [ {
   >     "Effect" : "Allow",
   >     "Principal" : {
   >         "Service" : "m2.amazonaws.com"
   >     },
   >     "Action" : [ "secretsmanager:GetSecretValue",
   >                  "secretsmanager:DescribeSecret" ],
   >     "Resource" : "*"
   >   } ]
   > }
   > ```

   > **Note**
   >
   > El tamaño máximo del conjunto de datos actualmente admitido para la transferencia es de 90 GB.

   - A continuación, seleccione la ubicación del bucket de Amazon S3 de destino a la que se transferirán los conjuntos de datos de destino del mainframe.

- Se seleccionará el punto de conexión para la transferencia que se ha elegido anteriormente. También puede seleccionar otro punto de conexión entre los puntos de conexión disponibles.

5. Elija Siguiente.
6. En la página Agregar los conjuntos de datos, en la sección de Configuración de la tarea de transferencia, puede elegir entre configurar la tarea de transferencia en modo binario o convertir y transferir los conjuntos de datos.

   - La opción de Transferir en modo binario le permite transferir conjuntos de datos omitiendo las conversiones de páginas de código y reteniendo sus bytes de palabra de descripción de registro (RDW).
   - La opción de Transferir y convertir conjuntos de datos le permite transferir conjuntos de datos estableciendo las páginas de códigos fuente y de destino de sus conjuntos de datos. Puede ver las páginas de código disponibles para File Transfer en la página the section called “Páginas de códigos fuente y de destino admitidas”.

7. Introduzca la consulta en la Buscar mainframe para conjuntos de datos para buscar en el mainframe para los conjuntos de datos que se incluirán en la tarea de transferencia. Elija Ver conjuntos de datos.

   Los siguientes símbolos de comodín se pueden utilizar como parte de los criterios de búsqueda de conjuntos de datos para mainframe:

   - Un solo asterisco (*) como calificador (entre puntos o después del punto final) coincide con un solo calificador en esa posición.
   - Un solo asterisco (*) dentro de un calificador coincide con cero o más caracteres en esa posición.
   - Un asterisco doble (**) como calificador (entre puntos o después del punto final) coincide con cero o más calificadores en esa posición.
   - Un asterisco doble (**) dentro de un calificador no es una consulta válida.
   - Un único signo de porcentaje (%) coincide con cualquier carácter alfanumérico o nacional que se encuentre en esa posición. Puede usar hasta un ocho por ciento en cada calificador.

   > **Note**
   >
   > Le sugerimos que termine siempre los criterios de búsqueda con un punto seguido de un asterisco doble (.**) y, si es necesario, restrinja aún más la búsqueda.

Para obtener más información sobre las reglas de caracteres comodín, consulte la sección Filtering data set names en la documentación de IBM.

8. Estos conjuntos de datos se cargarán en la sección Conjuntos de datos de mainframe, donde puede buscar o elegir uno o más conjuntos de datos para los que desee configurar las conversiones de páginas de códigos. Los conjuntos de datos elegidos se mostrarán en la sección Conjuntos de datos agregados. Si no se ha cargado ningún conjunto de datos, debe revisitar el paso 7.

> **Note**
>
> Puede seleccionar conjuntos de datos de varias consultas de búsqueda y agregarlos a la tarea de transferencia.

9. En la sección Conjuntos de datos agregados, verá el nombre, el tipo y el nombre del volumen de sus conjuntos de datos.

> **Important**
>
> Para la opción Transferir y convertir conjuntos de datos, introduzca manualmente la página de código fuente y la página de código de destino de cada conjunto de datos que elija. La Página de código fuente es el formato del conjunto de datos de origen y la Página de código de destino es el formato del conjunto de datos de destino que se utiliza para convertir los conjuntos de datos y almacenarlos en el bucket de Amazon S3 de destino.

10. Tras confirmar los conjuntos de datos en la sección Conjuntos de datos agregados (y las páginas de código fuente y destino para la opción Transferir y convertir conjuntos de datos), elija Siguiente.
11. En la página Revisar y crear, puede revisar o editar la información para la tarea de transferencia.
12. A continuación, elija Crear tarea de transferencia.

> **Important**
>
> Al pulsar el botón Crear tarea de transferencia, se iniciará la transferencia de datos, que se facturará según la página de precios de modernización de AWS mainframe. Esta facturación se basa en la cantidad de datos (GB) transferidos, medida según el tamaño del conjunto de datos.

Visualización de las tareas de transferencia

Para ver las tareas de transferencia en File Transfer, debe seguir estos pasos en la consola de modernización del AWS mainframe.

Para ver las tareas de transferencia

1. Abra la consola de modernización AWS del mainframe en. https://console.aws.amazon.com/m2/
2. En el Región de AWS selector, elija la región a la que desea transferir los archivos de su mainframe a un bucket de Amazon S3.
3. En la página Tareas de transferencia, seleccione el punto de conexión de transferencia de datos para ver sus tareas de transferencia.
4. Para los puntos de conexión que tengan tareas de transferencia preexistentes, aparecerán en la sección Tareas de transferencia. Puede elegir ver los detalles de cualquier tarea de transferencia de esta lista.

Tutorial: Introducción a AWS Mainframe Modernization File Transfer

Transferencia de archivos de AWS Mainframe Modernization le permite transferir y convertir conjuntos de datos de mainframe para casos de uso de modernización, migración y aumento de mainframes.

Siga los pasos de este tutorial para entender cómo funciona AWS Mainframe Modernization File Transfer.

Descripción general

File Transfer consta de lo siguiente:

1. Un agente que se instalará en el mainframe de origen.
2. Acceda a las funciones de descubrimiento, transferencia y conversión de conjuntos de datos directamente desde la consola del servicio de administración de AWS Mainframe Modernization.

Como usuario, puede transferir conjuntos de datos desde el mainframe a su bucket de Amazon S3.

Temas

- Paso 1: Transfiera el paquete tar de los archivos binarios del agente AWS a la partición lógica del mainframe
- Paso 2: Configurar el agente de File Transfer en el mainframe de origen
- Paso 3: Creación de un punto de conexión de transferencia de datos
- Paso 4: Crear una tarea de transferencia
- Paso 5: Ver el progreso de la tarea de transferencia

Paso 1: Transfiera el paquete tar de los archivos binarios del agente AWS a la partición lógica del mainframe

Descargue los archivos tar desde el enlace de M2-agent tar.

Paso 2: Configurar el agente de File Transfer en el mainframe de origen

En este paso, debe configurar e iniciar el agente de AWS Mainframe Modernization File Transfer en el mainframe de origen. El agente debe facilitar las comunicaciones entre la característica del servicio de File Transfer y el mainframe de origen. Se requiere al menos un agente por mainframe. Se puede iniciar más de un agente para obtener una alta disponibilidad y una escalabilidad mejorada.

Siga las instrucciones de la guía de the section called “Configuración del agente de File Transfer” para completar la instalación del agente de File Transfer en el mainframe.

Paso 3: Creación de un punto de conexión de transferencia de datos

Siga los pasos de la página the section called “Creación de puntos de conexión de transferencia de datos” para crear un nuevo punto de conexión de transferencia de datos.

Paso 4: Crear una tarea de transferencia

Siga los pasos de la página the section called “Creación de tareas de transferencia” para crear y gestionar las tareas de transferencia.

Paso 5: Ver el progreso de la tarea de transferencia

Puede ver el progreso de la tarea de transferencia en la consola de modernización del AWS mainframe. Para obtener más detalles, consulte la sección the section called “Visualización de las tareas de transferencia”.

Codificaciones de origen y destino admitidas en la Transferencia de archivos de AWS Mainframe Modernization

La Transferencia de archivos de AWS Mainframe Modernization admite varios tipos de conjuntos de datos y opciones de conversión de páginas de códigos.

Tipos de conjuntos de datos de mainframe

La Transferencia de archivos de AWS Mainframe Modernization admite los siguientes tipos de conjuntos de datos de mainframe:

- No VSAM: secuencial (PS), PDS, GDS, GDG
- Tipos de VSAM: KDS

Páginas de código admitidas

La Transferencia de archivos de AWS Mainframe Modernization admite las siguientes páginas de código para la conversión de conjuntos de datos (origen/destino):

«BIG5", "BIG5_HKSCS», «CESU_8", «EUC_JP», «EUC_KR», "GB18030", «GBK», «IBM00858", «GB2312IBM01140", «IBM01141", «IBM01142", «IBM01143", «IBM01144", «IBM01145", «IBM01146", «IBM01147", «IBM01148", «IBM01149", «IBM037"," 026", "0"," 0", "," 0", "," 0", "0"," 0", "," 00", "0" «,»,» IBM1 IBM1 IBM273 IBM277 IBM278 IBM28 IBM284 IBM285 IBM29 IBM297 IBM42 IBM424 IBM437 IBM5 IBM775 IBM85 IBM852 IBM855«," IBM857 "," IBM86 0", "IBM861", ",IBM862"», IBM863 "», IBM864 "", IBM865 ",", IBM866 "IBM8680," IBM869 IBM87 0", "", IBM871 IBM918 ", «IBM_THAI», «ISO_2022_CN», «ISO_2022_JP», «ISO_2022_JP_2",

«ISO_2022_KR», «ISO_8859_1", «ISO_8859_1", «ISO_8859_1" 59_13", «ISO_8859_15", «ISO_8859_16", «ISO_8859_2", «ISO_8859_3", «ISO_8859_4", «ISO_8859_5", «ISO_8859_6", «ISO_8859_7", «ISO_8859_8", «ISO_8859_9", "JIS_X0201201"», «JIS_X0212_1990"," KOI8 _R», "KOI8_U», «SHIFT_JIS», «TIS_620", «US_ASCII», «UTF_16", «UTF_16BE», «UTF_16LE», «UTF_32", «UTF_32LE», «UTF_8", «WINDOWS_1250", «WINDOWS_1251", «WINDOWS_1252", «WINDOWS «WINDOWS_1253", «WINDOWS_1254", «WINDOWS_1256", «WINDOWS_1257", «WINDOWS_1258", «WINDOWS_31J», «X_ BIG5 _HKSCS_2001", «X_ BIG5 _SOLARIS», «X_EUCJP_OPEN», «X_EUC_JP_LINUX», «X_EUC_TW», «X_EUC_TW», «X_EUC_TW» _ IBM1 006", «X_ IBM1 025", «X_ IBM1 046", «X_ IBM1 097", «X_ IBM1 098", «X_», «X_IBM1112», «X_IBM1122» IBM1123 IBM1124«, «X_IBM1129», «X_IBM1166», «X_IBM1364», «X_IBM1381», «X_ IBM29626 C», «X_ IBM3 00", «X_IBM33722», «X_IBM737», «X_IBM833», «X_IBM834», «X_IBM856», «X_IBM874», «X_IBM921», «X_IBM922», «X_IBM93», «X_IBM933», «X_IBM935», «X_IBM937», «X_IBM939», «X_ IBM942 IBM942 C», «X_», IBM943 «X_ IBM943 C», «X_», IBM948 «X_», IBM949 «X_ IBM949 C», «X_ IBM1383 IBM875 IBM95 0", «X_IBM964», «X_ IBM97 0", «X_», «ISCII91X_ISO_2022_CN_CNS», «X_ISO_2022_CN_GB», «X_ISO_8859_11", «X_JIS0208", «X_JISAUTODETECT», «X_JOHAB», «X_MACARABIC», «X_MACCENTRALEUROPE», «X_MACCROATIAN», «X_MACCIRILLIC», «X_MACDINGBAT», «X_MACGREEK», «X_MACHEBREW», «X_MACICELAND», «X_MACROMAN», «X_MACROMANIA», «X_MACSYMBOL», «X_MACTHAI», «X_MACTURKISH», «X_MACUKRAINE», «X_ MS932 _0213", «X_ MS95 0_HKSCS», «X_ MS95 0_HKSCS_XP», «X_MSWIN_936", «X_PCK», «X_SJIS_0213", «X_UTF_16LE_BOM», «X_UTF_32BE_BOM», «X_UTF_ 32LE_BOM», «X_WINDOWS_50220", «X_WINDOWS_50221", «X_WINDOWS_874", «X_WINDOWS_949", «X_WINDOWS_950", «X_WINDOWS_ ISO2 022j»

AWS Transform para modernizar las aplicaciones de mainframe

AWS Transform para mainframe le permite modernizar las aplicaciones de mainframe COBOL antiguas y convertirlas en aplicaciones Java con mayor rapidez al automatizar las complejas y lentas tareas de analizar las bases de código, planificar la transformación, generar documentación técnica, extraer la lógica empresarial, descomponer el código, planificar la oleada y ejecutar la refactorización y el reforjamiento. Reduce el coste y la complejidad de la modernización del mainframe al aprovechar la IA generativa y la automatización, a la vez que preserva la lógica empresarial esencial. AWS Transform Su interfaz en lenguaje natural y su enfoque orientado a los objetivos le permiten controlar la transformación, lo que le permite centrarse en las prioridades estratégicas, mientras que la automatización se encarga del pesado proceso de modernización.

Para obtener más información sobre las capacidades y características principales, una guía de alto nivel y la participación humana en las entradas y el procesamiento, consulte Migración y modernización de aplicaciones de mainframe en la Guía del usuario.AWS Transform

Ventajas principales

AWS Transform Las capacidades de modernizar las aplicaciones de mainframe tienen múltiples ventajas. Algunas de las cuales incluyen:

- Acelere el proceso de modernización del mainframe con la IA generativa: le AWS Transform ayuda a transformar su código COBOL en código Java moderno en unos pocos meses, en lugar de el cronograma tradicional en años.
- Supere las brechas de conocimiento: AWS Transform genera una documentación completa para sus aplicaciones de mainframe, lo que colma la brecha de conocimiento y permite tomar decisiones mejor informadas.
- Conserve la lógica empresarial fundamental: AWS Transform preserva la lógica empresarial fundamental de sus sistemas heredados y, al mismo tiempo, los refactoriza para convertirlos en aplicaciones Java modernas y optimizadas para la nube.
- Descomposición del dominio lógico empresarial y técnico: descompone AWS Transform automáticamente la base de código del mainframe en distintos dominios empresariales para reducir el esfuerzo manual y el tiempo necesarios para el análisis y la descomposición de la base de código.

- Capacidades de Human in the Loop (HITL): AWS Transform proporcionan un enfoque autónomo y orientado a objetivos que le permite controlar el proceso de modernización del mainframe.

Tutorial sobre la transformación de la consola de aplicaciones de mainframe

En la experiencia AWS Transform web, puede transformar sus aplicaciones de mainframe de COBOL a Java. Para saber cómo utilizar esta función, siga todos los pasos de la página sobre la transformación de las aplicaciones de mainframe de la Guía del AWS Transform usuario.

AWS Modernización de mainframe y replicación de datos con Precily

AWS Mainframe Modernization ofrece una variedad de Amazon Machine Images (AMIs). Esto AMIs facilita el aprovisionamiento rápido de las EC2 instancias de Amazon, lo que crea un entorno personalizado para la replicación de datos desde sistemas mainframe hasta el AWS uso de Precily. Esta guía proporciona los pasos necesarios para acceder a ellas y utilizarlas. AMIs

Requisitos previos

- Asegúrese de tener acceso de administrador a una AWS cuenta en la que pueda crear EC2 instancias de Amazon.
- Compruebe que el servicio de modernización de AWS mainframe esté disponible en la región en la que planea crear las EC2 instancias de Amazon. Consulte la lista de servicios de AWS disponibles por región.
- Identifique la Amazon Virtual Private Cloud (Amazon VPC) en la que se crearán EC2 las instancias de Amazon.
- Al crear EC2 instancias de Amazon en una VPC de Amazon, asegúrese de que la tabla de enrutamiento asociada tenga una puerta de enlace a Internet o una puerta de enlace NAT.

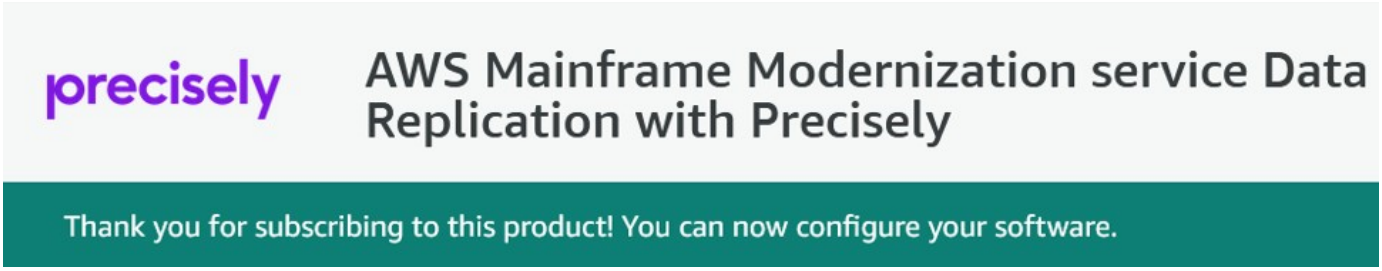
> **Note**
>
> La replicación de datos correcta requiere que la EC2 instancia de AWS tenga acceso de comunicación a AWS Marketplace. Si hay algún problema de conectividad con AWS Marketplace, el proceso de replicación fallará.

Suscripción a imágenes de máquina de Amazon

Al suscribirse a un producto del AWS Marketplace, puede lanzar una instancia desde la AMI del producto.

1. Inicie sesión en https://console.aws.amazon.com/marketplace AWS Management Console y abra la AWS Marketplace consola.

2. Elija Administrar suscripciones.
3. Vaya a cualquiera de los siguientes enlaces en función de su caso de uso:
   - Replicación de datos para IBM z/OS: https://aws.amazon.com/marketplace/pp/prodview-doe2lroefogia
   - Replicación de datos para IBM i: https://aws.amazon.com/marketplace/pp/prodview-iqrkflccxf7ko
4. Elija Continuar con la suscripción.
5. Si los términos y condiciones son aceptables, seleccione Aceptar términos. El proceso de suscripción puede tardar unos minutos.
6. Espere a que aparezca el mensaje de agradecimiento, como se muestra a continuación. Este mensaje confirma que se ha suscrito correctamente al producto.



7. En el panel de navegación izquierdo, elija Administrar suscripciones. Esta vista muestra todas las suscripciones a las que se ha suscrito.

Lance la replicación de datos AWS de Mainframe Modernization con Precisely

1. Abra la AWS Marketplace consola en https://console.aws.amazon.com/market.
2. En el panel de navegación izquierdo, elija Administrar suscripciones.
3. Busque la AMI que desee lanzar y elija Lanzar nueva instancia.
4. En Región, seleccione la región incluida en la lista de permitidos.
5. Selecciona Continuar para iniciar. EC2 Esta acción te lleva a la EC2 consola de Amazon.
6. Escriba un nombre del servidor.
7. Seleccione un tipo de instancia que se ajuste a los requisitos de rendimiento y costo de su proyecto. El punto de partida sugerido para el tamaño de la instancia es `c5.2xLarge`.

8. Elija un par de claves existente o cree y guarde uno nuevo. Para obtener información sobre los pares de claves, consulte los pares de EC2 claves de Amazon y las instancias de Linux en la Guía del EC2 usuario de Amazon.
9. Edite la configuración de red y elija la VPC de la lista de permitidos y la subred adecuada.
10. Seleccione un grupo de seguridad existente o cree uno nuevo. Además de permitir el acceso SSH (de forma predeterminada en el puerto 22), para la replicación de datos con una EC2 instancia de servidor Precily, lo habitual es permitir que el tráfico TCP llegue a su puerto predeterminado 2626.
11. Configura el almacenamiento de la EC2 instancia de Amazon.
12. Revise el resumen y seleccione Lanzar instancia. Para que el lanzamiento se realice correctamente, el tipo de instancia debe ser válido. Si se produce un error en el lanzamiento, seleccione Editar la configuración de la instancia y elija un tipo de instancia distinto.
13. Cuando vea el mensaje de éxito, seleccione Conectarse a la instancia.
14. Abre la EC2 consola de Amazon en https://console.aws.amazon.com/ec2/.
15. En el panel de navegación, bajo Instancias elija Instancias.
16. En el panel principal, compruebe el estado de la instancia.

Creación de una política de IAM

Para operar correctamente las EC2 instancias de modernización de AWS mainframe implementadas a través de nuestra AWS Marketplace lista, debe configurar una función y una política de IAM. Esta configuración de IAM específicamente diseñada no es opcional; autoriza a tus instancias de EC2 Amazon a interactuar con el servicio. AWS Marketplace La función y la política de IAM permiten a AWS Mainframe Modernization registrar con precisión los datos de uso, lo cual es esencial para una facturación precisa. Si no se implementa esta configuración, es posible que se produzcan intentos fallidos de replicación de datos e interrupciones operativas.

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación de la izquierda, elija Políticas.
3. Si es la primera vez que elige Políticas, aparecerá la página Bienvenido a políticas administradas. Elija Comenzar.
4. En la parte superior de la página, seleccione Crear política.
5. En la sección Editor de políticas, seleccione la opción JSON.

6. Introduzca la siguiente política JSON:

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": ["aws-marketplace:MeterUsage"],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Creación de un rol de IAM

1. Abra la consola de IAM en https://console.aws.amazon.com/iam/.
2. En el panel de navegación, seleccione Roles y luego seleccione Crear rol.
3. En la sección Tipo de entidad de confianza, elija Servicio de AWS .
4. En la sección Caso de uso, en Servicio o caso de uso, selecciona Amazon EC2.
5. Elija Siguiente.
6. En la lista de políticas, seleccione Gestionado por el cliente en el menú desplegable Filtrar por tipo e introduzca el nombre de la política que creó. Seleccione la casilla de verificación que hay junto al nombre de la política.
7. Elija Siguiente.
8. Indique un nombre para el rol y, opcionalmente, una descripción.
9. Revise la política de confianza y los permisos y, a continuación, seleccione Crear rol.

Adjunta el rol de IAM a la instancia de Amazon EC2

1. Abre la EC2 consola de Amazon en https://console.aws.amazon.com/ec2/.
2. En el panel de navegación, seleccione Instances (Instancias).
3. Selecciona tu EC2 instancia de Amazon.
4. En el menú Acciones, elija Seguridad, Modificar rol de IAM.
5. Seleccione el rol que desee asociar a la instancia y luego elija Actualizar rol de IAM.

Para obtener más información sobre cómo empezar a utilizar la replicación de AWS datos para IBM i, consulte Descripción general de la replicación de datos.

AWS Mainframe Modernization Conversión de código con MLogica

AWS Mainframe Modernization La conversión de código con mLogica (conversión de código) es una AWS Mainframe Modernization función que convierte z/OS automáticamente el código ensamblador del mainframe en COBOL. Puede utilizar la conversión de código para extraer una imagen de ensamblador utilizando el AWS CodeBuild servicio para la conversión de código que desee realizar con su. Cuenta de AWS

Temas

- ¿Qué es la conversión de ensambladores con mLogica?
- Descripción de la facturación por conversión de código para la conversión de ensambladores
- Conceptos de conversión de código
- Descripción de los componentes y procesos para la conversión de código
- Tutorial: Convierte código de Assembler a COBOL en AWS Mainframe Modernization

¿Qué es la conversión de ensambladores con mLogica?

AWS Mainframe Modernization La conversión de código con mLogica (conversión de código) convierte z/OS automáticamente el código ensamblador del mainframe en COBOL. El servicio se ejecuta dentro de usted Cuenta de AWS y no transmite ni almacena el código fuente de Assembler o COBOL fuera de él. Cuenta de AWS La conversión de código permite a su cuenta autorizada extraer una imagen del ensamblador utilizando el AWS CodeBuild servicio para la conversión de código que pretende convertir.

AWS Mainframe Modernization le permite configurar procesos de creación y integration/continuous distribución continua (CI/CD) para las aplicaciones migradas. Estas compilaciones y canalizaciones utilizan AWS CodeBuild Amazon S3 para proporcionar esta función. AWS CodeBuild es un servicio de compilación totalmente gestionado que compila el código fuente, ejecuta pruebas unitarias y produce artefactos listos para su implementación. Amazon S3 es un servicio de almacenamiento de objetos que ofrece escalabilidad, disponibilidad de datos, seguridad y rendimiento líderes del sector.

Temas

- Compiladores de conversión de código

- Arquitectura de conversión de código
- Enfoque de automatización
- Seguridad
- Recursos adicionales

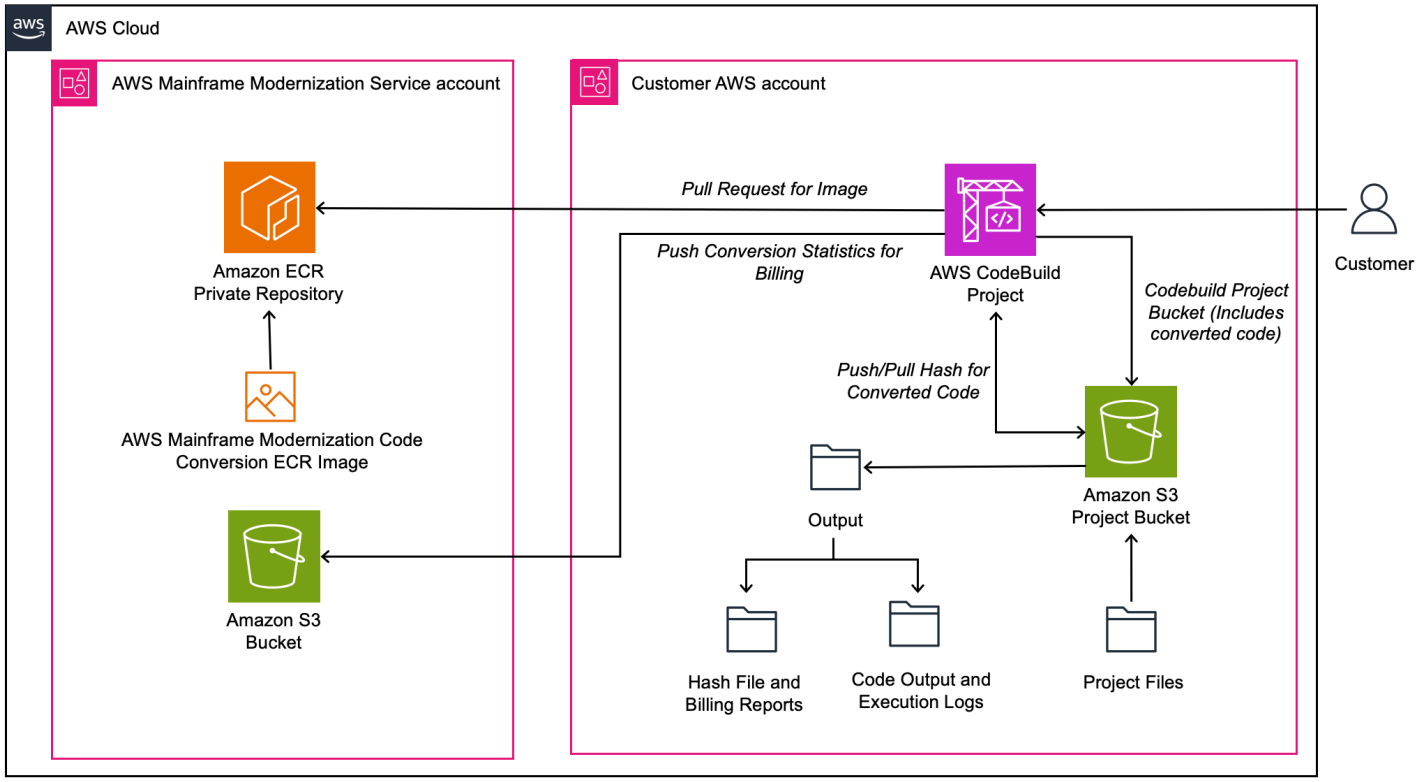
Compiladores de conversión de código

La conversión de código se puede configurar para que emita COBOL adecuado para su compilación y ejecución en varios entornos de destino con diferentes compiladores. Algunos de ellos incluyen:

- M2 cambia de plataforma con Rocket Software (anteriormente Micro Focus) y otros entornos de Rocket Enterprise Server
- Cambio de plataforma de M2 con NTT DATA Enterprise COBOL () UniKix
- mLogica LIBER*COBOL
- z/OSMainframe con IBM Enterprise COBOL
- Veryant isCOBOL

Arquitectura de conversión de código

El siguiente es un diagrama de arquitectura para el proceso de conversión de código:



Enfoque de automatización

Para utilizar la conversión de código con CodeBuild, el código del ensamblador debe cargarse en un bucket de Amazon S3 para, posteriormente, configurar los parámetros de conversión e invocar un CodeBuild proyecto para realizar cada paso del proceso de conversión. El código COBOL de destino se almacena automáticamente en una ruta específica del bucket de Amazon S3.

Seguridad

AWS Mainframe Modernization La conversión de código permite la conversión y, al mismo tiempo, conserva todo el código fuente y de destino en su archivo. Cuenta de AWS El código fuente del ensamblador, el código COBOL de destino y los archivos de configuración se almacenan en el bucket de Amazon S3. La herramienta de conversión automática se ejecuta como un contenedor en su CodeBuild entorno Cuenta de AWS. El código permanece en la cuenta en todo momento.

Para permitir que la herramienta de conversión acceda a su bucket de Amazon S3, debe conceder permisos al bucket a un Servicio de AWS rol. Cuando lo configure CodeBuild, definirá este rol de servicio para que CodeBuild pueda acceder a la imagen del contenedor y acceder a su bucket de Amazon S3.

Recursos adicionales

Además[the section called “Tutorial: conversión de código del ensamblador a COBOL”](), aquí encontrará algunos recursos adicionales en los que podrá obtener información sobre la creación de las AWS CloudFormation plantillas y más información sobre la conversión de Assembler a COBOL.

- Enlace al taller sobre la conversión de código automática del ensamblador a COBOL: https://catalog.workshops.aws/awsm2ccm-assembler-cobol/en-US.
- Publicación de blog: https://aws.amazon.com/blogs/migration-and-modernization/unlocking-new-potential-transform-your-assembler-programs-to-cobol-with-aws-mainframe-modernization/.

Descripción de la facturación por conversión de código para la conversión de ensambladores

Consulte esta página para comprender el alcance y el proceso de facturación de la conversión de código antes de realizar la conversión propiamente dicha. La sección de cálculo de facturación menciona el proceso mediante el cual la conversión del ensamblador a COBOL se cobra por cada línea de código.

Alcance y facturación de conversión de código

La conversión del código del ensamblador genera cargos (informes de facturación) en la Cuenta de AWS solo después de completar el paso de conversión. El cargo se basa en el número de líneas de código convertidas. Si realiza varios pasos de conversión, por ejemplo, después de agregar un nuevo código ensamblador, cambiar la configuración de conversión o aplicar una nueva versión del contenedor, solo se utilizarán las líneas modificadas o las líneas recién añadidas para calcular el cargo. No le cobraremos dos veces por la conversión de la misma línea de código en el mismo programa.

> **Note**
>
> Se cobrarán los módulos con líneas de código modificadas y todas las líneas de código de programas nuevos o que han cambiado de nombre.

Para evitar varios cargos, la conversión de código almacena un archivo binario codificado para cada módulo de macros o ensamblador del bucket de proyectos en `<Project_bucket>/awsm2ccm-`

do-not-delete/<AWS_account_number>/Hash. Estos archivos codificados no contienen ningún código de cliente.

> **Important**
>
> No edite ni elimine estos archivos manualmente. Los cambios pueden dar lugar a varias facturaciones por la conversión de los mismos componentes.

El informe de análisis de conversiones de AWS Mainframe Modernization Code («informe de análisis») proporciona a los clientes detalles sobre el alcance, el resultado y la facturación de la conversión prevista para garantizar una expectativa precisa de la conversión real. La conversión puede provocar que no se conviertan algunas líneas de código, que algunas líneas de código se conviertan parcialmente y que algunas líneas de código se conviertan por completo. El informe de análisis muestra el número de líneas de código para cada categoría. Debe ejecutar y leer el informe de análisis antes de procesar cualquier conversión de programas, macros y cuadernos. Una vez que el cliente revise el informe de análisis y esté de acuerdo con el alcance informado, el resultado esperado y la facturación prevista, podrá continuar con la ejecución de la conversión.

> **Note**
>
> Al ejecutar el comando **`Convert`** de conversión de código de AWS Mainframe Modernization, reconoce que ha ejecutado y leído el informe de análisis y que está de acuerdo con el resultado esperado y el número de líneas de código facturables.

Ámbito de la conversión

AWS Mainframe Modernization La conversión de código procesa todas las líneas de código de todos los componentes del ensamblador, macro y cuaderno de apuntes disponibles en los directorios scrlib y macrolib en la ubicación de origen de S3 configurada. Se incluyen los programas ensambladores, así como todas las macros y cuadernos a los que se haga referencia en un programa ensamblador. Los componentes de macros y cuadernos a los que no haga referencia un programa ensamblador se consideran fuera del alcance y no se convierten. Durante el procesamiento, el convertidor ejecuta algoritmos avanzados que consideran cada componente incluido de manera integral. Todas las líneas de código de estos componentes participan en el procesamiento, independientemente de si se han convertido totalmente, se han convertido parcialmente o no se han convertido. AWS Mainframe Modernization La conversión de código ignora las líneas en blanco y no las cuenta como líneas de

código. Las líneas de comentarios y las líneas que contienen cualquier otro texto (por ejemplo, las instrucciones de JCL para el ensamblador incrustadas en JCL) se cuentan como líneas de código para la facturación.

Cálculo de facturación

AWS Mainframe Modernization La conversión de código cobra por todos los componentes incluidos en el programa en su totalidad. Esto significa que cobra por cada línea de código de cada componente incluido en el ámbito de aplicación, incluidas las líneas que no se han podido convertir, que se han convertido parcialmente o que se han convertido totalmente. AWS Mainframe Modernization La conversión de código suma todas las líneas de código de los componentes suministrados para su procesamiento (incluidos los programas ensambladores, los cuadernos referenciados y las macros referenciadas) y utiliza el número total de líneas de código para la facturación.

> **Note**
>
> Los cuadernos y macros a los que no haga referencia un programa de ensamblador no se consideran incluidos.

Por ejemplo, supongamos que un programa tiene 1000 líneas de código:

- 700 líneas están completamente convertidas
- 200 líneas están convertidas parcialmente
- 100 líneas no se convierten

Se procesarían 1000 líneas de código y se facturarían.

Mejora de la conversión

Si, como cliente, busca una tasa de conversión más alta para las líneas de código o tiene otros requisitos específicos, puede ponerse en contacto con los AWS representantes para obtener opciones de contratación adicionales, como un esfuerzo de calibración o asistencia de servicios profesionales.

Conceptos de conversión de código

Para saber cómo se produce la conversión de código, CodeBuild es importante comprender algunos conceptos clave, como el manejo de macros y las páginas de códigos, etc.

Temas

- Gestión de macros
- Páginas de código (EBCDIC frente a ASCII)
- CodeBuild

Gestión de macros

El código del ensamblador de mainframe utiliza con frecuencia macros para encapsular la funcionalidad y reutilizarla. El comportamiento de las macros se suele determinar en el tiempo de ejecución de la aplicación en función de los parámetros transmitidos desde un programa ensamblador. La conversión de código proporciona varios mecanismos para expandir las macros del ensamblador antes de convertirlas a COBOL.

Páginas de código (EBCDIC frente a ASCII)

El ensamblador de mainframe suele contener literales de caracteres expresados como valores hexadecimales correspondientes a los caracteres EBCDIC. La conversión de código proporciona una capacidad configurable para gestionar automáticamente los literales de caracteres en ASCII al emitir COBOL para entornos ASCII.

CodeBuild

La conversión de código está disponible a través del AWS CodeBuild servicio. AWS CodeBuild es una herramienta de automatización de compilaciones diseñada originalmente como parte de un CI/CD proceso. En AWS Mainframe Modernization, AWS CodeBuild se utiliza para automatizar la herramienta de conversión MCCAC y otras herramientas, como el compilador COBOL de Rocket Software (anteriormente Micro Focus).

Descripción de los componentes y procesos para la conversión de código

AWS Mainframe Modernization El proceso de conversión de código incluye varios componentes, como el AWS Mainframe Modernization contenedor, el depósito de proyectos de S3 y las ubicaciones de los archivos de registro.

Temas

- AWS Mainframe Modernization contenedor
- Bucket de proyectos de S3
- Ubicaciones de archivo de registro
- Información general del proceso

AWS Mainframe Modernization contenedor

AWS Mainframe Modernization El contenedor de conversión de código se ejecuta en el AWS CodeBuild proyecto y proporciona comandos para configurar los directorios y los archivos de configuración del proyecto, evaluar el código del ensamblador, expandir las macros del ensamblador y convertir el código del ensamblador a COBOL.

Tendrá acceso al siguiente repositorio de AWS ECR: `381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod`.

Parar utilizar las imágenes, puede seguir una de las siguientes opciones:

- Utilice la etiqueta más reciente al consumir la imagen a través de. AWS CodeBuild Al usar la imagen, usará esta ruta: `381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod`. Esto significa que AWS CodeBuild recogerá la última imagen introducida en el repositorio.
- Enumeración de la versión y selección desde ella. Para ello, utilice el siguiente comando mediante la CLI para enumerar las diferentes versiones del repositorio:

```
aws ecr describe-images \
  --registry-id 381492161314 \
  --repository-name aws-mlogica-codebuild-prod \
  --query 'imageDetails[*].{ImagePushedAt: imagePushedAt, ImageTags: imageTags}' \
```

```
 --output json | jq '[.[] | {ImageURI: (.ImageTags[] |
"381492161314.dkr.ecr.us-east-1.amazonaws.com/aws-mlogica-codebuild-prod:" + .),
ImagePushedAt: .ImagePushedAt}] | sort_by(.ImagePushedAt) | reverse'
```

Esto mostrará una lista de todas las imágenes con la etiqueta asociada en cada una de ellas y la hora en que se publicó una imagen concreta en el repositorio. Según el código anterior, obtendrá una lista de imágenes en las que la etiqueta de la imagen representa la versión de la utilidad de conversión de código. Según los requisitos, puede seleccionar la imagen adecuada.

Bucket de proyectos de S3

El código de entrada y salida, el código actualizado con macros ampliadas y los informes generados por la conversión de AWS Mainframe Modernization código se almacenan en el depósito de proyectos que haya creado en el suyo AWS Account Management. Para que AWS Mainframe Modernization Code Conversion tenga acceso al bucket, debes conceder permisos a un rol AWS de servicio.

Ubicaciones de archivo de registro

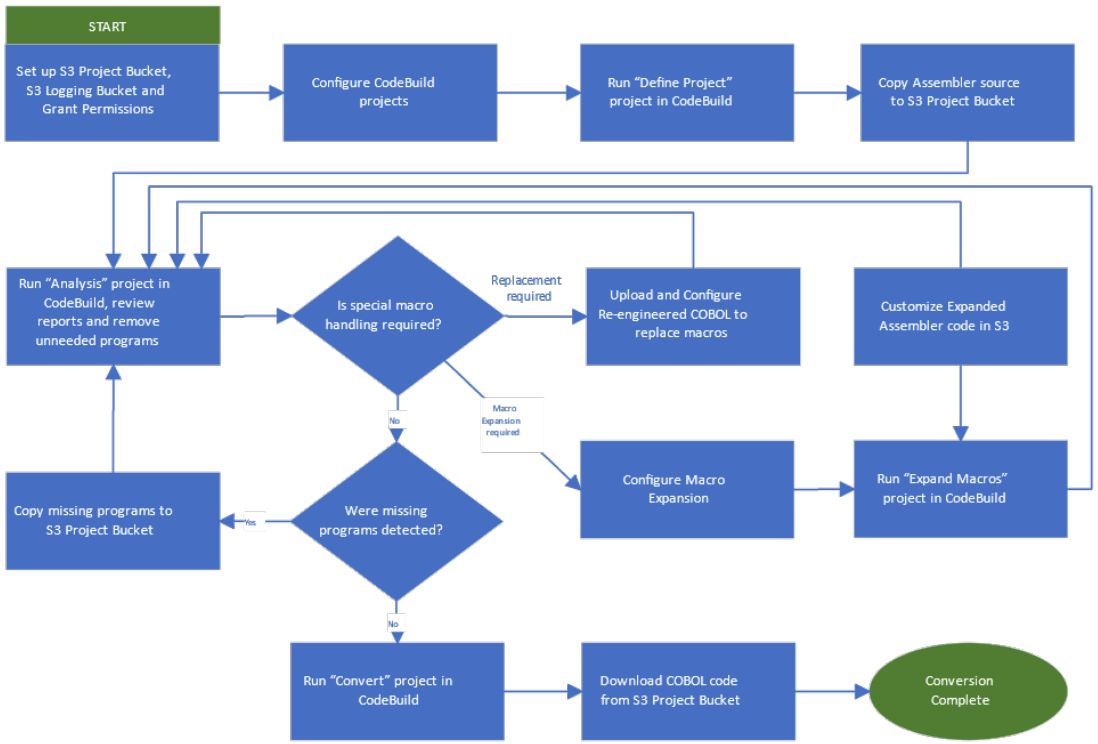
Los archivos de registro se escriben en dos ubicaciones durante la ejecución de cada CodeBuild proyecto:

- Los archivos de registro con los resultados de alto nivel de cada CodeBuild paso se escriben en los archivos de registro del depósito de registro configurado en el CodeBuild. Estos archivos aparecen como archivos gzip con un nombre de archivo de tipo GUID generado por el CodeBuild marco (por ejemplo,`0c03e183-ab40-4fe0-ba77- bc1d87e73b14.gz`). Cada archivo contiene el registro generado por la ejecución de un CodeBuild proyecto. Si se produce un error en la ejecución de un CodeBuild proyecto, este archivo de registro contendrá información importante sobre la solución de problemas.
- Los archivos de registro con resultados de ejecución detallados en el nivel de componente se escriben en los archivos de registro de la ruta principal del bucket del proyecto con el patrón de nombre de archivo `<Project_Bucket_name>_.log` (por ejemplo, `project-bucket_202406131200.log`). Estos registros proporcionan:
  - Un resumen de la configuración que indica las ubicaciones de entrada y salida.
  - Un registro de cada componente de ensamblador o macro procesado con el nombre de archivo de destino.
  - Una lista de informes generados con las ubicaciones de los archivos.

- Para las ejecuciones de conversiones, se proporciona una lista de los cuadernos de texto en tiempo de ejecución.

Información general del proceso

En el siguiente diagrama se muestra el proceso de conversión del ensamblador a COBOL:



Tutorial: Convierte código de Assembler a COBOL en AWS Mainframe Modernization

Puede utilizar este documento como step-by-step guía para comprender cómo convertir el código del ensamblador de modernización del mainframe a COBOL. Además, también puede consultar Automated code conversion from Assembler to COBOL workshop para obtener más información sobre el proceso de conversión.

Temas

- Requisitos previos
- Paso 1: Comparta los activos de construcción con Cuenta de AWS
- Paso 2: creación de buckets de Amazon S3

- Paso 3: creación de la política de IAM
- Paso 4: creación de un rol de IAM
- Paso 5: asociación de las políticas de IAM al rol de IAM.
- Paso 6: Crea el CodeBuild proyecto
  - Paso 6.1: creación del proyecto Definir
  - Paso 6.2: creación del proyecto de Code Analysis
  - Paso 6.3: creación del proyecto de Code Conversion
- Paso 7: definición del proyecto y carga del código fuente
- Paso 8: ejecución del análisis y comprensión de los informes
- Paso 9: ejecución de la conversión del código
- Paso 10: verificación de la conversión de código
- Paso 11: descarga del código convertido
- Eliminar recursos

Requisitos previos

Lea la Descripción de la facturación por conversión de código para la conversión de ensambladores sección para entender cómo la conversión de código de Assembler genera cargos (informes de facturación) y cómo AWS Account Management funciona la facturación.

Paso 1: Comparta los activos de construcción con Cuenta de AWS

En este paso, asegúrate de compartir los activos de construcción con los tuyos Cuenta de AWS, especialmente en la región en la que se utilizan los activos.

1. Abre la AWS Mainframe Modernization consola en https://console.aws.amazon.com/m2/.
2. En el panel de navegación izquierdo, seleccione Herramientas.
3. En AWS Mainframe Modernization Code Code Conversion with MLogica, elija Compartir activos con mi. Cuenta de AWS

> **Important**
>
> Debe realizar este paso una vez en cada AWS región en la que desee realizar las compilaciones.

Paso 2: creación de buckets de Amazon S3

En este paso, creará buckets de Amazon S3. El primer bucket es el del proyecto AWS CodeBuild para contener el código fuente y, a continuación, colocar el bucket de salida para contener la salida de AWS CodeBuild (código convertido). Para obtener más información, consulte Creación, configuración y trabajo con buckets de Amazon S3 en la Guía del usuario de Amazon S3.

1. Para crear el bucket de proyecto, inicie sesión en la consola de Amazon S3 y elija Crear bucket.
2. En la configuración general, proporciona un nombre para el depósito y especifica el Región de AWS lugar en el que quieres crearlo. Un nombre de ejemplo es *`codebuild-regionId-accountId-bucket`*, donde:

   - `regionId`es el Región de AWS del depósito.
   - `accountId`es tu Cuenta de AWS carné de identidad.

   > **Note**
   >
   > Si va a crear el depósito en un lugar diferente al Región de AWS de EE. UU. Este (Virginia del Norte), especifique el `LocationConstraint` parámetro. Para obtener más información, consulte CreateBucket en Amazon Simple Storage Service API Referenc.

3. Retenga todos los demás valores y elija Crear bucket.

Utilice los mismos nombres que elija para los buckets en todo este tutorial.

Paso 3: creación de la política de IAM

En este paso, creará una política de IAM. La política de IAM proporcionada concede permisos específicos AWS CodeBuild para interactuar con Amazon S3, Amazon Elastic Container Registry, CloudWatch los registros de Amazon que CodeBuild genera y Amazon Elastic Compute Cloud los recursos para la conversión de código. Esta política no está personalizada para los clientes. La

política concede permisos para AWS Mainframe Modernization interactuar y obtener las estadísticas de conversión del código para facturar al cliente de forma adecuada.

Para obtener más información sobre la creación de una política de IAM, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Creación de una política

1. Inicie sesión en la consola de IAM; y elija Políticas en el panel de navegación de la izquierda.
2. Elija Crear política.
3. Copie y pegue la siguiente política de JSON en el editor de políticas.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetBucketLocation",
                "s3:ListBucket",
                "s3:PutObjectAcl",
                "s3:GetBucketAcl"
            ],
            "Resource": [
                "arn:aws:s3:::codebuild-regionId-accountId-bucket",
                "arn:aws:s3:::codebuild-regionId-accountId-bucket/*",
                "arn:aws:s3:::aws-m2-repo-*" ],
                    "Effect": "Allow"
          },
          {
              "Action": [
                "ecr:GetAuthorizationToken",
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "logs:*",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DeleteNetworkInterface",
                "ec2:CreateNetworkInterface",
```

```
                "ec2:DescribeDhcpOptions",
                "ec2:DescribeVpcs",
                "ec2:CreateNetworkInterfacePermission"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

4. Si lo desea, puede agregar etiquetas a la política. Las etiquetas son pares clave-valor que pueden ayudarle a organizar, controlar el acceso a la política o realizar su seguimiento.
5. Elija Next: Review.
6. Escriba un nombre para la política, por ejemplo, *`CodeBuildAWSM2CCMPolicy`*.
7. Si lo desea, puede introducir una descripción de la política y revisar el resumen de la política para asegurarse de que es correcto.
8. Elija Crear política.

Paso 4: creación de un rol de IAM

En este paso, debe crear un nuevo rol de IAM que le CodeBuild permita interactuar con AWS los recursos, después de asociar las políticas de IAM que creó anteriormente a este nuevo rol de IAM.

Para obtener información sobre la creación de un rol de servicio, consulte Creación de un rol para delegar permisos a un AWS servicio en la guía del usuario de IAM.

1. Inicie sesión en la consola de IAM y elija Roles en el panel de navegación de la izquierda.
2. Elija Crear rol.
3. En Tipo de entidad de confianza, seleccione Servicio de AWS.
4. En Casos de uso de otros servicios de AWS, elija y CodeBuild, a continuación, CodeBuildvuelva a elegir.
5. Elija Siguiente.
6. Elija Siguiente en la página Agregar permisos. Posteriormente, asignará una política al rol.
7. En Detalles del rol, proporcione un nombre para el rol, por ejemplo, `IAMRoleTaskExecutionRoleForCodeBuild`.
8. En Seleccionar entidades de confianza, compruebe que el documento de política tenga el siguiente aspecto:

```
{
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "codebuild.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
}
```

9. Seleccione Crear rol.

Paso 5: asociación de las políticas de IAM al rol de IAM.

En este paso, asociará la política de IAM que ha creado antes al rol de IAM `IAMRoleTaskExecutionRoleForCodeBuild`.

1. Inicie sesión en la consola de IAM y elija Roles en el panel de navegación de la izquierda.
2. En Roles, elija el rol que ha creado anteriormente, por ejemplo, `IAMRoleTaskExecutionRoleForCodeBuild`.
3. En la sección Políticas de permisos, elija Agregar permisos y, después, Asociar políticas.
4. En Otras políticas de permisos, elija las políticas que ha creado anteriormente, por ejemplo, *`CodeBuildAWSM2CCMPolicy`*.
5. Seleccione Asociar políticas.

Paso 6: Crea el CodeBuild proyecto

En este paso, crearás tres CodeBuild proyectos diferentes basados en el `buildspec.yml` archivo mencionado anteriormente.

Paso 6.1: creación del proyecto Definir

Creación del proyecto Definir

1. Inicie sesión en la CodeBuild consola y elija Crear proyecto de compilación.

2. En la sección Configuración del proyecto, proporciona un nombre para el proyecto, por ejemplo, `1-awsm2ccm-define-project`.
3. En la sección Origen, en Proveedor de origen, deje la selección predeterminada.
4. En la sección Entorno, seleccione Imagen personalizada.
5. En el campo Tipo de entorno, seleccione Linux.
6. En Registro de imágenes, seleccione Otro registro.
7. En el campo URL de registro externo, siga la sección the section called “AWS Mainframe Modernization contenedor”.
8. En Rol de servicio, elija Rol de servicio existente y, en el campo ARN de rol, el rol de servicio que ha creado anteriormente (por ejemplo, `IAMRoleTaskExecutionRoleForCodeBuild`).
9. Amplíe la sección Configuración adicional y haga lo siguiente:

    a. VPC: configúrela si es necesario en función de la configuración.

    b. Tiempo de espera: establézcalo en 60 minutos.

    c. Tiempo de espera en cola: establézcalo en 480 minutos.

    d. Cifrado: elija la configuración de cifrado adecuada (la configuración predeterminada es correcta).

    e. En la sección Variables de entorno, agregue las siguientes variables, una por una:

    - Nombre: PROJECT_BUCKET. Valor: **`codebuild-regionId-accountId- bucket`**. Tipo: Plaintext
    - Nombre: PROJECT_DIR. Valor: **`prj_codebuild_01`**. Tipo: Plaintext
    - Nombre: AWSM2CCM_ACTION. Valor: **`define_project`**. Tipo: Plaintext
    - Nombre: CCM_LOGGING_BUCKET. AWSM2 Valor: **`s3:// codebuild-regionId-accountId-bucket`**. Tipo: Plaintext

10. En Especificación de compilación, elija Insertar comandos de compilación y, a continuación, Cambiar al editor.
11. Reemplace los valores actuales por lo siguiente:

```
version: 0.2
phases:
    build:
        commands:
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
```

```
artifacts:
    files:
        - '**/*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

donde PROJECT_DIR son variables de entorno disponibles en él. CodeBuild Para obtener más información, consulte Variables de entorno en los entornos de compilación.

12. En la sección Artefactos, haga lo siguiente:

    - en Tipo, elija Amazon S3 y, a continuación, el bucket de salida, por ejemplo, `codebuild-regionId-accountId-bucket`.
    - en Ruta, deja este campo vacío.
    - en Nombre, escriba **`prj_codebuild_01`**.
    - en Empaquetado de artefactos, seleccione Ninguno.
    - en Sustituir el nombre del artefacto, desmarque esta opción.
    - en Cifrado, déjelo en la configuración predeterminada.

13. En la sección Registro, haga lo siguiente:

    - CloudWatch registros: deshabilitado
    - Registros de S3: Habilitado
    - Bucket: **`codebuild-regionId-account-bucket`**
    - Ruta de registro: **`CODEBUILD-LOGS`**

14. Elija Crear el proyecto de compilación.

Paso 6.2: creación del proyecto de Code Analysis

Creación del proyecto de Code Analysis

1. Inicie sesión en la CodeBuild consola y elija Crear proyecto de compilación.
2. En la sección Configuración del proyecto, proporciona un nombre para el proyecto, por ejemplo, `2-awsm2ccm-analysis`.

3. En la sección Origen, en Proveedor del origen, elija Amazon S3 y, a continuación, el bucket de entrada que ha creado anteriormente, (por ejemplo, `codebuild-regionId-accountId-bucket`).
4. En el campo Clave de objeto S3 o Carpeta S3, escriba **`prj_codebuild_01`**.
5. En la sección Entorno, seleccione Imagen personalizada.
6. En el campo Tipo de entorno, seleccione Linux.
7. En Registro de imágenes, seleccione Otro registro.
8. En el campo URL de registro externo, siga la sección the section called “AWS Mainframe Modernization contenedor”.
9. En Rol de servicio, elija Rol de servicio existente y, en el campo ARN de rol, el rol de servicio que ha creado anteriormente (por ejemplo, `IAMRoleTaskExecutionRoleForCodeBuild`).
10. Amplíe la sección Configuración adicional y haga lo siguiente:

    a. VPC: configúrela si es necesario en función de la configuración.

    b. Tiempo de espera: establézcalo en 60 minutos.

    c. Tiempo de espera en cola: establézcalo en 480 minutos.

    d. Cifrado: elija la configuración de cifrado adecuada (la configuración predeterminada es correcta).

    e. En la sección Variables de entorno, agregue las siguientes variables, una por una:

    - Nombre: PROJECT_BUCKET. Valor: **`codebuild-regionId-accountId-bucket`**. Tipo: Plaintext
    - Nombre: PROJECT_DIR. Valor: **`prj_codebuild_01`**. Tipo: Plaintext
    - Nombre: AWSM2CCM_ACTION. Valor: **`analysis`**. Tipo: Plaintext
    - Nombre: CCM_LOGGING_BUCKET. AWSM2 Valor: **`s3:// codebuild-regionId-accountId-bucket`**. Tipo: Plaintext

11. En Especificación de compilación, elija Insertar comandos de compilación y, a continuación, Cambiar al editor.
12. Reemplace los valores actuales por lo siguiente:

```
version: 0.2
phases:
    build:
        commands:
            - ln -s $CODEBUILD_SRC_DIR $PROJECT_DIR
```

```
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
artifacts:
    files:
        - '*.log'
        - '_Converted/*/*'
        - '_Reports/*'
    secondary-artifacts:
        reports:
            files:
                - '_Reports/AWSM2CCM*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

donde PROJECT_DIR son variables de entorno disponibles en él. CodeBuild Para obtener más información, consulte Variables de entorno en los entornos de compilación.

13. En la sección Artefactos, haga lo siguiente:

    - en Tipo, elija Amazon S3 y, a continuación, el bucket de salida (p. ej., `codebuild-regionId-accountId-bucket`).
    - en Ruta, escriba ARTIFACTS.
    - en Nombre, escriba **`prj_codebuild_01`**.
    - en Empaquetado de artefactos, seleccione Ninguno.
    - en Sustituir el nombre del artefacto, desmarque esta opción.
    - en Cifrado, déjelo en la configuración predeterminada.

14. En la sección Registro, haga lo siguiente:

    - CloudWatch registros: deshabilitado
    - Registros de S3: Habilitado
    - Bucket: **`codebuild-regionId-account-bucket`**
    - Ruta de registro: **`CODEBUILD-LOGS`**

15. Elija Crear el proyecto de compilación.

Paso 6.3: creación del proyecto de Code Conversion

Creación del proyecto de Code Conversion

1. Inicie sesión en la CodeBuild consola y elija Crear proyecto de compilación.
2. En la sección Configuración del proyecto, proporcione un nombre para el proyecto, (por ejemplo, `3-awsm2ccm-convert`).
3. En la sección Origen, en Proveedor del origen, elija Amazon S3 y, a continuación, el bucket de entrada que ha creado anteriormente, (por ejemplo, `codebuild-regionId-accountId-bucket`).
4. En el campo Clave de objeto S3 o Carpeta S3, escriba **`prj_codebuild_01`**.
5. En la sección Entorno, seleccione Imagen personalizada.
6. En el campo Tipo de entorno, seleccione Linux.
7. En Registro de imágenes, seleccione Otro registro.
8. En el campo URL de registro externo, siga la sección the section called “AWS Mainframe Modernization contenedor”.
9. En Rol de servicio, elija Rol de servicio existente y, en el campo ARN de rol, elija el rol de servicio que ha creado anteriormente; por ejemplo, `IAMRoleTaskExecutionRoleForCodeBuild`.
10. Amplíe la sección Configuración adicional y haga lo siguiente:
    a. VPC: configúrela si es necesario en función de la configuración.
    b. Tiempo de espera: establézcalo en 60 minutos.
    c. Tiempo de espera en cola: establézcalo en 480 minutos.
    d. Cifrado: elija la configuración de cifrado adecuada (la configuración predeterminada es correcta).
    e. En la sección Variables de entorno, agregue las siguientes variables, una por una:
       - Nombre: PROJECT_BUCKET. Valor: **`codebuild-regionId-accountId-bucket`**. Tipo: Plaintext
       - Nombre: PROJECT_DIR. Valor: **`prj_codebuild_01`**. Tipo: Plaintext
       - Nombre: AWSM2CCM_ACTION. Valor: **`conversion`**. Tipo: Plaintext
       - Nombre: CCM_LOGGING_BUCKET. AWSM2 Valor: **`s3:// codebuild-regionId-accountId-bucket`**. Tipo: Plaintext

11. En Especificación de compilación, elija Insertar comandos de compilación y, a continuación, Cambiar al editor.
12. Reemplace los valores actuales por lo siguiente:

```
version: 0.2
phases:
    build:
        commands:
            - export AWSM2CCM_PUSH_RUNTIME_COPYBOOKS=y
            - ln -s $CODEBUILD_SRC_DIR $PROJECT_DIR
            - . /app/awsm2ccm_prod/bin/setup_env.sh
            - run_awsm2ccm.sh $PROJECT_DIR
artifacts:
    files:
        - '*.log'
        - '_Converted/*/*'
        - '_Reports/*'
    discard-paths: no
    base-directory: $PROJECT_DIR
```

    donde PROJECT_DIR son variables de entorno disponibles en él. CodeBuild Para obtener más información, consulte Variables de entorno en los entornos de compilación.
13. En la sección Artefactos, haga lo siguiente:
    - en Tipo, elija Amazon S3 y, a continuación, el bucket de salida (p. ej., `codebuild-regionId-accountId-bucket`).
    - en Ruta, escriba ARTIFACTS.
    - en Nombre, escriba **`prj_codebuild_01`**.
    - en Empaquetado de artefactos, seleccione Ninguno.
    - en Sustituir el nombre del artefacto, desmarque esta opción.
    - en Cifrado, déjelo en la configuración predeterminada.
14. En la sección Registro, haga lo siguiente:
    - CloudWatch registros: deshabilitado
    - Registros de S3: Habilitado
    - Bucket: **`codebuild-regionId-account-bucket`**

- Ruta de registro: **`CODEBUILD-LOGS`**

15. Elija Crear el proyecto de compilación.

Paso 7: definición del proyecto y carga del código fuente

El proyecto Definir configura la carpeta del proyecto y los archivos de configuración, inicializados con las configuraciones predeterminadas. En este paso, iniciará la compilación. Para ello:

1. Inicie sesión en la AWS CodeBuild consola.
2. En el panel de navegación, elija Proyectos de compilación.
3. Seleccione el proyecto creado anteriormente (`1-awsm2ccm-define-project`) para crearlo.
4. Seleccione Iniciar la compilación y, a continuación, Comenzar ahora para definir el proyecto. Una vez que comience la compilación, el estado cambiará a En curso.
5. Seleccione los detalles de la fase para ver el progreso de cada paso orquestado por el AWS CodeBuild proyecto.
6. Espere a que el estado cambie a Realizado correctamente para todos los pasos.
7. Uso de la consola de Amazon S3.
8. Localice y haga clic en el bucket de Amazon S3 llamado `codebuild-regionId-accountId-bucket`.

    - **`CODEBUILD-LOGS/`**la carpeta contiene los AWS CodeBuild registros de los AWS CodeBuild proyectos en ejecución.
    - La carpeta **`prj_codebuild_01/`** que contiene la estructura del proyecto. Se usa durante los pasos de análisis, expand_macros y conversión. Puede seleccionar `prj_codebuild_01/` para explorar los detalles.
    - Archivo de configuración **`cobol_reserved.rsw`** (lista de palabras COBOL) reservado para el conversor. Se usa durante el paso de conversión.
    - La carpeta **`Macro_Expansion/`** contiene macros para expandir en programas de ensamblador. Se usa durante el paso expand_macros.
    - El archivo de configuración **`macro_settings.json`** contiene un reemplazo de macros personalizado. Se usa durante el paso expand_macros.
    - La carpeta **`macrolib/`** contiene las macros de ensamblador que se convertirán. Se usa durante el paso de análisis y conversión.

        1. Seleccione `macrolib/`.

2. De forma predeterminada, se proporciona una macro de ensamblador llamada `MACR01.mac` como archivo de ejemplo. Elimine este archivo, puesto que no es necesario para el análisis.
3. Cargue sus macros en este directorio.

- El archivo de configuración **`project_settings_aux.json`** contiene los ajustes relacionados con la página de códigos. Se usa durante el paso de conversión.
- El archivo de configuración **`project_settings.json`** contiene los ajustes del convertidor. Se usa durante el paso de conversión.
- La carpeta **`srclib/`** contiene los programas de ensamblador que se convertirán. Se usa durante el paso de análisis y conversión.
  1. Elija `srclib/`.
  2. De forma predeterminada, dos programas de ensamblador llamados `SQtest01.asm` y `SQtest02.asm` se proporcionan como ejemplos. Elimine estos archivos, ya que no son necesarios para el análisis y la conversión.
  3. Cargue sus programas de ensamblador en este directorio.

9. Compruebe el estado del paso `1-awsm2ccm-define-project`. Debería haberse realizado correctamente en la pestaña Estado de compilación más reciente.

Ya tiene todo preparado para el siguiente paso: Análisis del código.

Paso 8: ejecución del análisis y comprensión de los informes

> **Note**
>
> AWS Mainframe Modernization El paso de análisis de conversión de código es gratuito.

En este paso, iniciará otra compilación:

1. En el panel de navegación, elija Proyectos de compilación.
2. Elija el proyecto que ha creado en el paso 6.2 para compilar: `2-awsm2ccm-analysis`.
3. Seleccione Iniciar la compilación y, a continuación, Comenzar ahora para generar informes de análisis. Esto iniciará la compilación y cambiará el estado a En curso.

4. Elija los detalles de la fase, donde verá el progreso de cada paso orquestado por el AWS CodeBuild proyecto. Espere a que el estado cambie a Realizado correctamente para todos los pasos.
5. Desde AWS Management Console, vaya a la consola de servicio de Amazon S3.
6. Localice el bucket de Amazon S3 y haga clic en él: `codebuild-regionId-accountId-bucket`.

    a. La carpeta **`ARTIFACTS/`** contiene las salidas de los pasos análisis y conversión.

    b. Elija `ARTIFACTS/prj_codebuild_01/_Reports/`.

    c. Estarán disponibles los siguientes informes:

    - `AWSM2CCM-Analysis-Report-<timestamp>.pdf`es un informe ejecutivo que proporciona la facturación y el alcance de la conversión del AWS Mainframe Modernization código, lo que mejora la conversión, el resumen de la conversión y las estadísticas de conversión detalladas. También resume los recuentos de códigos y los recuentos de códigos facturables en el proyecto y proporciona métricas y listas de los miembros a los que se hace referencia para cada componente. Es fundamental ejecutar y examinar este informe antes de ejecutar la conversión real.
    - `Conversion_Detailed_Statistics.txt` proporciona la frecuencia y el resultado de conversión esperado (que se muestra como Estado de la conversión) para cada instrucción que se encuentra en cada componente. Esto proporciona una manera rápida de identificar si las instrucciones son claras o si el conversor no las admite. Los posibles resultados del Estado de la conversión son:
        - Totalmente convertida: la instrucción se convertirá con precisión a COBOL.
        - Parcialmente convertida: la instrucción se admite, pero utiliza un parámetro o expresión no admitidos. Es probable que sea necesario realizar ajustes manuales después de la conversión.
        - No convertida: el convertidor no admite la instrucción.
        - Instrucciones de precompilación para verificación: normalmente se incluyen dentro de las macros y hacen referencia a lo que probablemente se conozca también como instrucciones de Lenguaje ensamblador condicional (por ejemplo, AIF, AGO) en el mainframe. Estas se gestionan por el precompilador, que depende de dichas instrucciones o directivas, selecciona y produce un código ASM limpio y estático. Estas instrucciones dependen de los valores reales de los parámetros de la macro que se compilan. Por lo tanto, la misma macro puede generar diferentes fragmentos de

código ASM, según los valores de los parámetros pasados. Esto se debe a la presencia de dichas instrucciones de precompilación. En ese caso, considere la posibilidad de expandir o rediseñar la macro.

- `Conversion_Global_Statistics.txt` proporciona un resumen del Estado de la conversión en el componente.
- `CrossReference_PgmToCpyMacro.txt` informa sobre las dependencias del programa de ensamblador con respecto a las macros. Proporciona una manera rápida de determinar si falta alguna macro en el código cargado.
- `CrossReference_PgmToPgm.txt` informa sobre las dependencias del programa de ensamblador con respecto a otros programas de este tipo. Proporciona una manera rápida de determinar si falta algún programa de ensamblador en el código cargado.

7. Regrese a la consola AWS CodeBuild de servicio.
8. Verifique el estado del paso 2-awsm2ccm-analysis. Debería haberse realizado correctamente en la pestaña Estado de compilación más reciente.

Ya tiene todo preparado para el siguiente paso: Conversión del código.

Paso 9: ejecución de la conversión del código

> **Important**
>
> AWS Mainframe Modernization El paso de conversión del código se facturará según el uso. Para obtener más información sobre la facturación, consulte the section called “Descripción de la facturación por conversión de código”.

En este paso, configurará el proceso de conversión y, a continuación, iniciará la compilación.

1. Desde AWS Management Console, vaya al servicio Amazon S3.
2. Localice el bucket de Amazon S3 y haga clic en él: `codebuild-regionId-accountId-bucket`.
   a. Vaya a `prj_codebuild_01/`.
   b. Seleccione `project_settings.json` y elija Descargar.
   c. Abra el archivo `project_settings.json` para ver la siguiente estructura JSON:

```
{
"Source programs directory":"srclib",
"Source copybooks/macros directory":"macrolib",
"Copybook/Macros Conversion":"Called_only",
"Do not regenerate the Copy/Macro if already exists":"false",
"Target Compiler":"IBM",
"Endianess":"Big",
"Converted programs extension":"",
"Converted CICS programs extension":"",
"Converted copies/macros extension":"",
"Trace Level":"STANDARD",
"Trace file open mode":"append",
"Data definition level":5,
"Start picture column":40,
"Generate Sync FILLER with name":"FILL-SYNC",
"Use SYNC clause":"yes",
"Decimal Point Comma":"true",
"Original Source Placement":"RIGHT"
}
```

donde,

- Directorio de programas de origen: contiene los programas de ensamblador necesarios para la conversión.
- Directorio de macros y cuadernos de origen: contiene las macros y los cuadernos de ensamblador necesarios para la conversión.
- La conversión de cuadernos o macros puede realizarse de una de las siguientes maneras:
  - Todos: este botón de radio indica que la conversión completa convertirá todo lo que copybook/Macros esté disponible en el directorio, independientemente de si lo están utilizando los programas o no.
  - Called_only: Este botón de radio indica que la conversión completa solo convertirá lo copybook/Macros que realmente utilizan los programas.
-

> **Important**
>
> No es necesario regenerar el si ya existe. Copy/Macro

Si esto es cierto, la herramienta no volverá a convertir el cuaderno o la macro si ya se ha convertido (existe en la carpeta de salida).

- Destino: la conversión de los programas (código generado) depende del compilador COBOL de destino. Las siguientes opciones son compatibles:
  - IBM para el mainframe de IBM
  - MF para COBOL de Micro Focus
  - VERYANT para isCOBOL de Veryant
  - NTT para NTT DATA Enterprise COBOL (Unikix)
- Endianess y Bitness: la conversión de los programas (código generado) depende de la plataforma de destino (bit/endianess). Esta combinación permite seleccionar las siguientes opciones admitidas:
  - Endianess: Grande (para Big-Endian)/Little (Little-Endian). Por ejemplo, el z/OS mainframe de IBM es Big-Endian, Windows es Little-Endian y Linux varía según la distribución (por ejemplo, Amazon Linux 2 on es Little-Endian). EC2
  - Bitness: 32/64 (si no se indica, el valor predeterminado será 32). La configuración recomendada es de 32 bits.
- Extensión de programa convertida: sirve para establecer la extensión de archivo de los programas COBOL generados. Vacío (“”): sin extensión. En el caso de los objetivos COBOL de Rocket Software (anteriormente Micro Focus), se recomienda utilizar el CBL para permitir que Rocket Enterprise Developer reconozca los archivos correctamente.
- Extensión de programa CICS convertida: sirve para establecer la extensión de archivo de los programas CICS COBOL generados. Vacío (“”): sin extensión. En el caso de los objetivos COBOL de Rocket Software, se recomienda utilizar el CBL para que Rocket Enterprise Developer reconozca correctamente los archivos.
- Extensión de cuadernos o macros convertidos: sirve para establecer la extensión de archivo de los cuadernos COBOL generados. Vacío (“”): sin extensión. En el caso de los objetivos COBOL de Rocket Software, se recomienda utilizar el CPY para que Rocket Enterprise Developer reconozca correctamente los archivos.
- Nivel de rastreo: el rastreo es la información que se registra CodeBuild durante la conversión. El usuario puede seleccionar el nivel de detalle mediante cualquiera de las opciones proporcionadas.
  - ERROR = TRACE ERROR: solo se muestran los errores de conversión.

- STANDARD = TRACE STANDARD: se muestran los errores de conversión y la información estándar. Esta es la configuración predeterminada.
- ALL = TRACE ALL: nivel máximo de seguimiento

- Modo de apertura del archivo de rastreo: no se utiliza. Se recomienda la configuración predeterminada de añadir.
- Nivel de definición de datos: indica el nivel inicial de los subcampos (después del nivel 01) definidos en la sección de trabajo-almacenamiento y vinculación. Debe ser un número.
- Columna de imagen inicial: se trata del formato del código COBOL generado e indica la columna en la que se coloca la cláusula PIC (después de los nombres de los campos). Debe ser un número.
- Ubicación original del origen: indica la posición en la que se colocan los comentarios en el programa. Tiene dos secciones:
  - DERECHA: esta opción colocará el comentario o la información adicional en la posición correcta después de la septuagésima tercera (73) columna. En COBOL, el código se escribe en las primeras setenta y dos (1-72) columnas y todo lo que aparezca desde la septuagésima tercera (>= 73) se considerará un comentario.
  - ARRIBA: esta opción colocará el comentario encima del contenido traducido.
- Generar FILLER de sincronización con nombre: esta opción está relacionada con la alineación en la memoria de campos binarios (tipos de datos del ensamblador H, F y D, que se convierten al tipo de datos COMP de COBOL). Para garantizar el límite de alineación correcto, se añadirán campos de relleno explícitos durante la conversión. Se trata de una opción basada en texto; el valor debe ser una cadena (como FILL-SYNC).
- Usar la cláusula SYNC: esta opción se refiere a la alineación en la memoria de los campos binarios. Sí = todos los campos convertidos a COBOL. COMP se definirá con la cláusula SYNC (por ejemplo, 05 WRKFLD PIC S9 (09) COMP SYNC).
- Coma decimal: si esto es verdadero, se añadirá la cláusula DECIMAL-POINT IS COMMA al párrafo de COBOL “SPECIAL-NAMES”.

d. En función de sus requisitos, cambie los parámetros adecuados y, a continuación, guarde `project_settings.json`.

e. Elimine el archivo `project_settings.json` existente de `prj_codebuild_01/` del bucket de Amazon S3 y, a continuación, cargue la nueva versión.

3. Vuelva al AWS CodeBuild servicio.

4. Seleccione el proyecto para compilar que ha creado anteriormente: `3-awsm2ccm-convert`

a. Seleccione Iniciar la compilación y, a continuación, Comenzar ahora para convertir macros y programas de ensamblador en cuadernos y programas de COBOL.

b. Espere a que el estado de compilación cambie a Realizado correctamente para este proyecto. Aparecerá en la pestaña Estado de compilación más reciente.

Paso 10: verificación de la conversión de código

1. Desde AWS Management Console, vaya al servicio Amazon S3.
2. Localice el bucket de Amazon S3 y haga clic en él: `codebuild-regionId-accountId-bucket`.
3. Navegue hasta **`awsm2ccm-do-not-delete`** . AWS Mainframe Modernization La conversión de código crea archivos binarios codificados para cada módulo ensamblador o macro durante el proceso de conversión. Estos archivos son esenciales para evitar la facturación duplicada a los clientes y también para hacer un seguimiento de cuánto del código de ensamblador proporcionado se analizó y convirtió. Los archivos se almacenan en la siguiente ubicación `codebuild-regionId-accountId- bucket/awsm2ccm-do-not-delete/<your_AWS_account_id>/Hash`: Los archivos codificados no contienen código de ensamblador y tampoco es posible extraer código del cliente de estos archivos.

   > **Important**
   >
   > No edite tampoco estos archivos manualmente ni los elimine. La edición o eliminación de estos archivos puede provocar facturaciones múltiples para los mismos componentes.

   Trate la carpeta **`awsm2ccm-do-not-delete/`** como un directorio administrado por el sistema. Consulte Soporte antes de realizar cualquier cambio en este directorio o en su contenido.
4. Haga clic en `codebuild-regionId-accountId-bucket` para volver al bucket.
5. Elija **`ARTIFACTS/prj_codebuild_01/`**. La carpeta _Converted/ contiene las salidas COBOL generadas como resultado del paso de conversión de código. Tendrá los siguientes subdirectorios:

   - La carpeta copybooks/ contiene los cuadernos COBOL generados.
   - La carpeta programs/ contiene los programas COBOL generados.

- La carpeta runtime_lib/ contiene programas COBOL adicionales y cuadernos proporcionados por la solución.

6. Si los informes de análisis y otros informes indican que la conversión se ha realizado correctamente y el AWS CodeBuild proyecto `3-awsm2ccm-convert` está marcado como realizado correctamente, descargue el código COBOL y los cuadernos del directorio _Converted/.

Paso 11: descarga del código convertido

En este paso, descargue el código de COBOL y los cuadernos del directorio _Converted/ y compílelos en el entorno de COBOL de destino.

1. Desde AWS Management Console, vaya al servicio Amazon S3.
2. Localice el bucket de Amazon S3 y haga clic en él: `codebuild-regionId-accountId-bucket`.
3. Navegue hasta la ubicación: `ARTIFACTS/prj_codebuild_01/_Converted/`.
4. Descargue el código COBOL convertido de todos los subdirectorios de _Converted/. También puede utilizar el siguiente comando CLI para descargarlos a la vez:

```
aws s3 cp s3://codebuild-regionId-accountId-
bucket/ARTIFACTS/prj_codebuild_01/_Converted/ . --recursive
```

5. Analice y compile el COBOL convertido en el entorno de COBOL de destino.

Eliminar recursos

Si ya no necesita los recursos que ha creado para este tutorial, elimínelos para evitar cargos adicionales. Para ello, complete los siguientes pasos.

- Elimine los depósitos de S3 que ha creado para este tutorial. Para obtener más información, consulte Eliminar un bucket en la Guía del usuario de Amazon Simple Storage Service.
- Elija las políticas que ha creado para este tutorial. Para obtener más información, consulte Eliminación de políticas de IAM en la Guía del usuario de IAM.
- Elija el rol de IAM que ha creado para este tutorial. Para obtener más información, consulte Eliminación de roles o perfiles de instancia en la Guía del usuario de IAM.

- Elimine el CodeBuild proyecto que creó para este tutorial. Para obtener más información, consulte Eliminar un proyecto de compilación CodeBuild en la Guía del AWS CodeBuild usuario.

Integración de Charon

Introducción a Charon-SSP

En 1987, Sun Microsystems lanzó el procesador SPARC V7, un procesador RISC de 32 bits. Le siguió el SPARC V8 en 1990, una revisión del SPARC V7 original, cuya inclusión más notable fueron las instrucciones de dividir y multiplicar por hardware. Los procesadores SPARC V8 constituyeron la base de varios servidores y estaciones de trabajo, como los modelos SPARCstation 5, 10 y 20. En 1993, al SPARC V8 le siguió el procesador SPARC V9 de 64 bits. Esta fue también la base de varios servidores y estaciones de trabajo, como los Enterprise 250 y 450.

Debido a la obsolescencia del hardware y a la falta de piezas de repuesto o reacondicionadas, el software y los sistemas desarrollados para estas antiguas estaciones de trabajo y servidores basados en SPARC son cada vez más difíciles de mantener. Para cubrir la necesidad continua de determinados sistemas end-of-life basados en SPARC, Stromasys S.A. desarrolló la línea Charon-SSP de emuladores SPARC. Los siguientes productos son sustitutos de máquinas virtuales basados en software para los sistemas SPARC de hardware nativo especificados. A continuación presentamos una descripción general de las gamas de hardware emulado.

Charon-SSP/4M emula el siguiente hardware SPARC:

- Familia Sun-4m (representada por el Sun SPARCstation 20): originalmente, era una variante Sun-4 multiprocesador, basada en el bus de módulos de procesador introducido en la serie de 600 MP. MBus SPARCServer Posteriormente, la arquitectura Sun-4m también incluyó sistemas que no eran MBus uniprocesadores, como el SPAR Cstation 5, que utilizaba procesadores de arquitectura SPARC V8. Compatible a partir de SunOS 4.1.2 y de Solaris 2.1 a Solaris 9. SPARCServer La compatibilidad con 600 MP se dejó de utilizar después de Solaris 2.5.1.

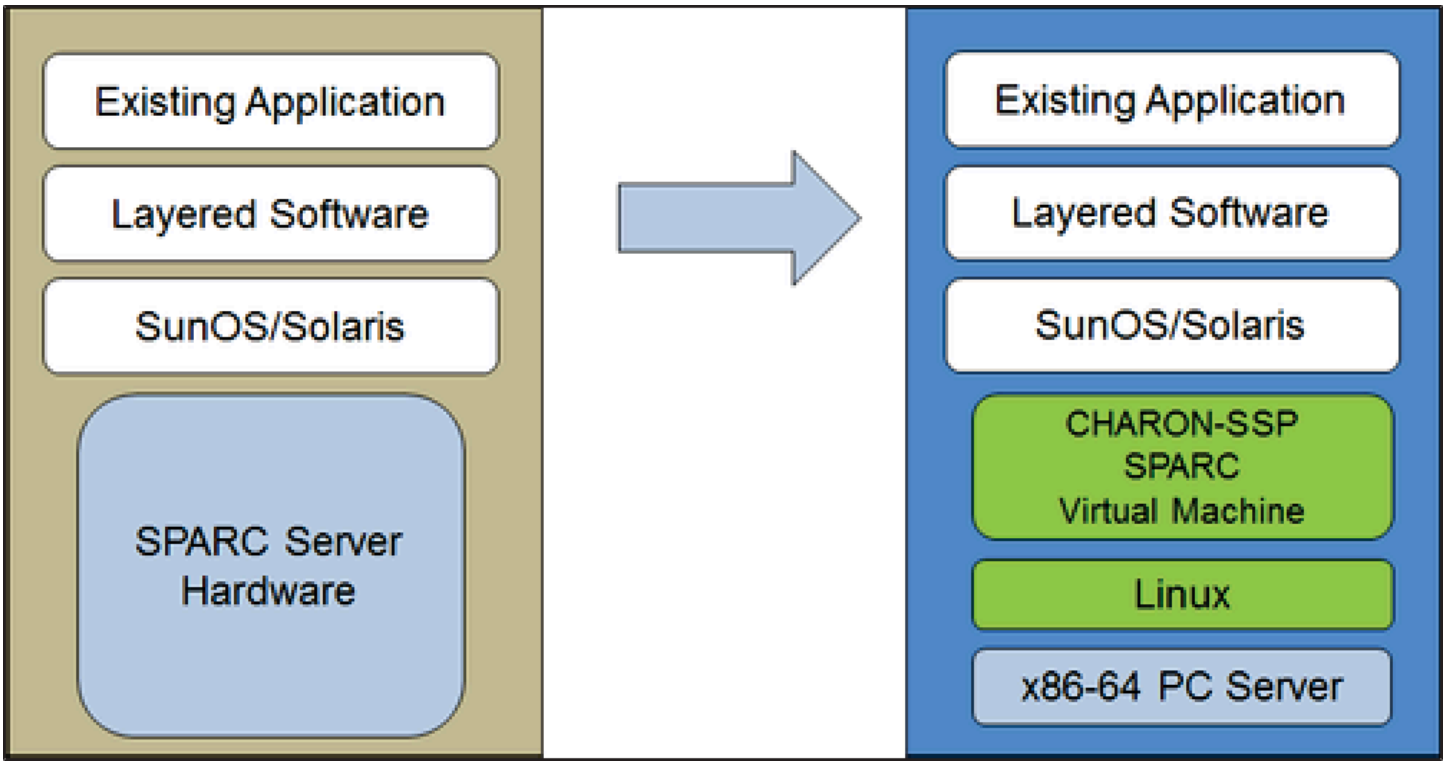
Charon-SSP/4U(+) emula el siguiente hardware SPARC:

- Gama Sun-4u (representada por el Sun Enterprise 450): (U para UltraSPARC): esta variante introdujo la arquitectura del procesador SPARC V9 de 64 bits y la interconexión de procesadores UPA utilizada por primera vez en la serie Sun Ultra. Compatible con las versiones de 32 bits de Solaris a partir de la versión 2.5.1 La primera versión de Solaris de 64 bits para Sun-4u fue la Solaris 7. La compatibilidad con UltraSPARC I se eliminó tras Solaris 9. Solaris 10 admite las implementaciones de Sun-4u, desde UltraP ARC II hasta UltraSPARC IV.

Charon-SSP/4V(+) emula el siguiente hardware SPARC:

- Gama Sun-4v (representada por los SPARC T2 y T4): esta variante incorporó la virtualización del procesador con hipervisor al Sun-4u; se introdujo en el procesador multinúcleo Ultra SPARC T1. El hardware seleccionado era compatible con la versión 10 de Solaris a partir de la versión 3/05 HW2 (la mayoría de los modelos, incluido el hardware emulado por Charon-SSP, requieren versiones más recientes de Solaris 10). También se admiten varias versiones de Solaris 11.

La siguiente imagen muestra el concepto básico de la migración de hardware físico a un emulador.



Las máquinas virtuales Charon-SSP permiten a los usuarios de equipos basados en Sun y Oracle SPARC reemplazar su hardware nativo de una manera que requiere pocos o ningún cambio en la configuración original del sistema. Esto significa que puede seguir ejecutando sus aplicaciones y datos sin necesidad de cambiar o migrar a otra plataforma. El software Charon-SSP se ejecuta en sistemas Intel de 64 bits básicos, lo que garantiza la protección continua de su inversión.

El Charon-SSP/4U+ es compatible con las mismas plataformas SPARC virtuales que el Charon-SSP/4U, y el Charon-SSP/4V+ lo mismo que el Charon-SSP/4V. Sin embargo, las versiones 4U + y 4V+ aprovechan las tecnologías modernas de virtualización asistida por hardware VTx /EPT de Intel y AMD-V/NPT de AMD para ofrecer un mejor rendimiento de la CPU virtual. CPUs Los Charon-SSP/4U+ y Charon-SSP/4v+ requieren compatibilidad con VT-X/EPT o AMD-V/NPT y deben

instalarse en un sistema host dedicado. CPUs No se admite la ejecución de estas variantes de producto en una máquina virtual (por ejemplo, VMware encendida).

> **Note**
>
> Si planea ejecutar Charon-SSP/4U+ o 4V+ en un entorno de nube, póngase en contacto con Stromasys o con un VAR de Stromasys para analizar sus requisitos.

Sistemas operativos huéspedes compatibles

Las máquinas virtuales Charon-SSP/4M son compatibles con las siguientes versiones de sistemas operativos huéspedes:

- SunOS 4.1.3 - 4.1.4
- De Solaris 2.3 a Solaris 9

Las máquinas virtuales Charon-SSP/4U(+) son compatibles con las siguientes versiones de sistemas operativos huéspedes:

- De Solaris 2.5.1 a Solaris 10

Las máquinas virtuales Charon-SSP/4V(+) son compatibles con las siguientes versiones de sistemas operativos huéspedes:

- Solaris 10 (a partir de la actualización 4, 08/07) y de Solaris 11.1 a Solaris 11.4

Para Charon-SSP/4V(+), tenga en cuenta lo siguiente:

- Para el SPARC T4 emulado, las versiones de Solaris 10 compatibles son: Oracle Solaris 10 1/13, Oracle Solaris 10 8/11 y Solaris 10 9/10, o Solaris 10 10/09 con el set de parches Oracle Solaris 10 8/11.
- El modelo SPARC T4 emulado es un requisito previo para ejecutar Solaris 11.4 en el emulador.
- No se admiten las zonas de núcleos de Solaris.

Requisitos previos de la instancia en la nube de Charon-SSP

Al seleccionar un tipo o forma de instancia, seleccione el hardware virtual que se utilizará para la instancia de host de Charon-SSP en la nube. Por lo tanto, la selección de un tipo o forma de instancia determina las características del hardware del host virtual de Charon-SSP (por ejemplo, cuántos núcleos de CPU y cuánta memoria tendrá su sistema de host virtual Charon).

> **Note**
>
> Si utiliza una imagen de mercado de Charon-SSP para lanzar la instancia, se cumplen todos los requisitos del sistema operativo de host Linux.

Los requisitos mínimos de hardware se describen a continuación.

Puntos importantes relacionados con las pautas de tamaño:

- Las siguientes pautas de tamaño, en particular en lo que respecta al número de núcleos de CPU del host y la memoria del host, ilustran los requisitos mínimos. Se debe revisar cada situación de implementación y el tamaño real del host debe adaptarse según sea necesario. Por ejemplo, se debe aumentar la cantidad de núcleos de CPU disponibles para la E/S si las aplicaciones del huésped producen una carga de E/S elevada. Además, un sistema con muchos emulados CPUs suele ser capaz de generar una mayor carga de E/S y, por lo tanto, es posible que sea necesario aumentar la cantidad de núcleos de CPU disponibles para la E/S. En un entorno de hipersubprocesos, para obtener el mejor rendimiento, la cantidad de núcleos de CPU (es decir, reales o físicos CPUs) debe ser suficiente para cumplir con los requisitos de CPU de los emuladores activos, evitando así que los subprocesos de alta carga de trabajo compartan un núcleo de CPU físico.
- La configuración determina la asignación de núcleos de CPU para los núcleos de CPU emulados CPUs y para el procesamiento de E/S. Consulte la configuración de la CPU en la guía general del usuario de Charon-SSP para obtener más información sobre esto y la asignación predeterminada de los núcleos de CPU para el procesamiento de E/S.

> **Información general importante**
>
> - Para facilitar la transferencia rápida de los datos del emulador de una instancia en la nube a otra, se recomienda encarecidamente almacenar todos los datos relevantes del

emulador en un volumen de disco independiente que pueda separarse fácilmente de la instancia anterior y adjuntarse a una nueva instancia.

- Asegúrese de dimensionar la instancia correctamente desde el principio (consulte los requisitos mínimos que se indican a continuación). La licencia de Charon-SSP para Charon-SSP AL se crea cuando la instancia se lanza por primera vez. Si se cambia posteriormente a otro tamaño/tipo de instancia y, por lo tanto, se modifica el número de núcleos de la CPU, se invalidará la licencia y, por lo tanto, se impedirá que las instancias de Charon se inicien (se requiere una nueva instancia). Si planea utilizar la instancia de Charon-SSP AL en modo AutoVE, asegúrese de incluir la información del servidor de AutoVE antes del primer lanzamiento; de lo contrario, se utilizarán los servidores de licencias públicas. La licencia de Charon-SSP VE se crea en función de la huella digital capturada en el servidor de licencias. Si el servidor de licencias se ejecuta directamente en el host del emulador y, posteriormente, el host del emulador requiere, por ejemplo, un cambio en el número de núcleos de la CPU, la licencia se invalidará (se necesitará una nueva licencia y, posiblemente, una nueva instancia).

Requisitos previos para las instancias

Requisitos generales de CPU: Charon-SSP es compatible con instancias Amazon modernas basadas en procesadores de arquitectura x86-64. EC2

Requisitos mínimos para Charon-SSP:

- Número mínimo de núcleos de CPU del sistema host:
  - Al menos un núcleo de CPU para el sistema operativo host, además de:
  - Para cada sistema SPARC emulado:
    - Un núcleo de CPU por cada CPU emulada de la instancia, además de:
    - Al menos un núcleo de CPU adicional para el procesamiento de E/S (al menos dos, si se utiliza la optimización JIT del servidor). Consulte la sección de configuración de la CPU mencionada anteriormente para ver las opciones de configuración. De forma predeterminada, Charon asignará al procesamiento de E/S 1/3 (mínimo 1, redondeado hacia abajo) del número de elementos CPUs visibles para el host de Charon.
- Requisitos mínimos de memoria:
  - 4 GB o más de RAM para el sistema operativo del host Linux. Los requisitos reales pueden ser mayores y dependerán de los requisitos de los servicios no emuladores que se ejecuten en el

host Linux. La recomendación anterior de al menos 2 GB de RAM para el host Linux seguirá siendo válida para muchos sistemas, pero los requisitos cada vez mayores del sistema operativo y las aplicaciones Linux han llevado a actualizar la recomendación para nuevas instalaciones. Además:

  - Para cada sistema SPARC emulado:
    - La memoria configurada de la instancia emulada, además de:
    - 2 GB de RAM (6 GB de RAM si se utiliza el JIT del servidor) para permitir la optimización del DIT, los requisitos del emulador, los búferes en tiempo de ejecución, el SMP y la emulación de gráficos.
- Si la tecnología Hyper-Threading está habilitada en la versión x86-64 moderna CPUs, se pueden ejecutar dos subprocesos en un núcleo físico de la CPU, lo que proporciona dos procesos lógicos al sistema operativo anfitrión. CPUs Si es posible, deshabilite el Hyper-Threading en el host Charon-SSP. Sin embargo, esto no suele ser posible en VMware entornos de nube, o no está claro si se utiliza el hiperproceso o no. La opción de Hyper-Threading de Charon-SSP permite a Charon-SSP adaptarse a dichos entornos. Consulte la sección de configuración de la CPU en la guía general del usuario de Charon-SSP mencionada anteriormente para obtener información detallada sobre la configuración. Tenga en cuenta que, para obtener el mejor rendimiento, los subprocesos Charon-SSP no deben compartir un núcleo de CPU físico; debe haber suficientes núcleos físicos en el sistema host para cumplir con los requisitos de los emuladores configurados.
- Una o más instancias de red, en función de los requisitos del cliente.
- Charon-SSP/4U+ y Charon-SSP/4V+ deben ejecutarse en hardware físico compatible con los procesadores Intel VT-x/EPT o AMD-v/NPT (instancias básicas) y, por lo tanto, no pueden ejecutarse en todos los entornos de nube. Consulte la documentación de su proveedor de servicios en la nube para conocer la disponibilidad de dicho hardware. Además, tenga en cuenta los siguientes puntos:
  - Charon-SSP/4U+ y Charon-SSP/4V+ solo están disponibles cuando se utiliza un núcleo de Linux compatible con Stromasys.
  - Si necesita este tipo de hardware SPARC emulado, póngase en contacto con Stromasys o con su VAR de Stromasys para analizar sus requisitos en detalle.

Creación y configuración de una instancia AWS en la nube para Charon (nueva GUI)

Esta sección refleja lo ocurrido AWS Management Console en la primavera de 2022. Si aún utilizas la consola anterior, consulta el apéndice de la guía de introducción de Charon-SSP AWS .

Requisitos previos generales

Esta descripción muestra la configuración básica de una instancia de Linux en AWS. No incluye requisitos previos específicos. Sin embargo, en función de su caso de uso, tenga en cuenta los siguientes requisitos previos:

- Cuenta y AWS Marketplace suscripciones de Amazon
  - Para configurar una instancia de Linux AWS, necesitas una AWS cuenta con acceso de administrador.
  - Identifique la AWS región en la que planea lanzar la instancia. Asegúrese de que los servicios de AWS que planea usar estén disponibles en esa región. Consulte Servicios de AWS por región.
  - Identifique la VPC en la que planee lanzar la instancia.
  - Si la instancia requiere acceso a Internet, asegúrese de que la tabla de enrutamiento asociada a la VPC tenga una puerta de enlace de Internet. Si su instancia requiere acceso mediante VPN a su red en las instalaciones, asegúrese de que haya una puerta de enlace VPN disponible. La configuración exacta de la VPC y sus subredes dependerá del diseño de la red y de los requisitos de la aplicación.
  - Para suscribirse a un AWS Marketplace servicio específico, elija Suscripciones a AWS Marketplace en AWS Management Console y, a continuación, elija Administrar suscripciones.
  - Busque el servicio que planee usar y suscríbase a él. Cuando la suscripción se haya realizado correctamente, la encontrará en la sección Administrar suscripciones. Desde allí, puede lanzar directamente una nueva instancia.
- Los requisitos previos de hardware y software de la instancia variarán en función del uso previsto para la instancia:
  - Opción 1: la instancia se utilizará como sistema host del emulador de Charon:
    - Consulte las secciones de requisitos previos de hardware y software de la guía del usuario o la guía de introducción de su producto Charon para determinar los requisitos previos exactos de hardware y software que debe cumplir la instancia de Linux. La imagen que utilice para

lanzar la instancia y el tipo de instancia que elija determinan el software y el hardware de la instancia en la nube.

- Se requiere una licencia de producto de Charon para ejecutar sistemas emulados heredados. Consulte la información de licencia en la documentación de su producto Charon o póngase en contacto con su representante de Stromasys o con el VAR de Stromasys para obtener más información.

- Opción 2: la instancia se utilizará como servidor de licencias de VE dedicado:
  - Consulte la guía del servidor de licencias de VE para conocer los requisitos previos detallados.

- Algunos sistemas operativos heredados que pueden ejecutarse en los sistemas emulados proporcionados por los productos emuladores de Charon requieren una licencia del proveedor original del sistema operativo. El usuario es responsable de cualquier obligación de licencia relacionada con el sistema operativo heredado y debe proporcionar las licencias correspondientes.

AWS Management Console Utilizándolo para lanzar una nueva instancia

Para crear una nueva instancia

1. Inicia sesión en la EC2 consola de Amazon AWS Management Console y ábrela en https://console.aws.amazon.com/ec2/.
2. Seleccione Iniciar instancia.
3. Escriba el nombre de la instancia.
4. Seleccione una AMI. Una AMI es una imagen preempaquetada que se utiliza para lanzar instancias en la nube. Incluye el sistema operativo y el software de aplicación correspondiente. La elección de la AMI depende de cómo vaya a utilizar la instancia:

   - Si la instancia se va a utilizar como sistema host del emulador de Charon, son posibles varias opciones de AMI:
     - Instalación del sistema del host de Charon a partir de una imagen de mercado de Charon preempaquetada: contienen el sistema operativo subyacente y el software de Charon preinstalado.
       - Consulte con su representante de Stromasys qué opciones están disponibles actualmente en su mercado de proveedores de servicios en la nube.
       - Según el proveedor de servicios en la nube y los planes de lanzamiento de los productos de Stromasys, puede haber dos variantes:

        - Licencias automáticas (AL) para su uso con un servidor de licencias público operado por Stromasys o con un servidor de licencias AutoVE privado operado por el cliente
        - Entorno virtual (VE) para su uso con un servidor de licencias de VE privado operado por el cliente
    - Instalación del sistema host de Charon mediante una instalación de emulador de Charon convencional con los paquetes RPM de instalación del emulador de Charon para Linux:
        - Elija una AMI de Linux de una distribución compatible con el producto y la versión de Charon que haya seleccionado. Consulte la guía del usuario de su producto en el sitio de documentación de Stromasys.
- Si la instancia se va a utilizar como un servidor de licencias de VE dedicado, consulte la Guía del servidor de licencias de VE en la documentación de licencias para conocer los requisitos de la instancia de Linux.

Después de decidir qué AMI es necesaria, seleccione una AMI de producto de Linux o Charon coincidente. Si no ve la AMI que necesita, elija Explorar más AMIs. Elija la AMI de Linux que coincida con la forma en que tenga previsto usar la instancia. Puede ser una de las siguientes opciones:

- Una imagen de mercado de Charon VE preempaquetada. El nombre de la AMI incluirá la cadena “ve”.
- Una imagen de mercado de Charon AL preempaquetada para Automatic Licensing o AutoVE.
- Una versión de Linux compatible con la instalación de un producto de RPM.
- Una versión de Linux compatible con el servidor de licencias de VE.

5. Seleccione un tipo de instancia. Amazon EC2 ofrece tipos de instancias con distintas combinaciones de CPU, memoria, almacenamiento y capacidad de red. Seleccione un tipo de instancia que coincida con los requisitos del producto Charon que desee utilizar. Algunas imágenes de mercado tienen una selección restringida de tipos de instancias.
6. Seleccione un par de claves existente o cree y guarde uno nuevo. Si selecciona un par de claves existente, asegúrese de tener la clave privada correspondiente. De lo contrario, no podrá conectarse a su instancia.

> **Note**
>
> Si su sistema de administración lo admite, para RHEL 9.x, Rocky Linux 9.x y Oracle Linux 9.x, utilice la clave SSH de tipo ECDSA o. ED25519 Estos tipos le permiten

conectarse a estos sistemas de host Linux de Charon mediante un túnel SSH sin necesidad de cambiar la configuración de política criptográfica predeterminada del host Charon por una configuración menos segura. Por ejemplo, esto es importante para el administrador de Charon-SSP. Consulte Using system-wide cryptographic policies en la documentación de Red Hat.

7. En la sección Configuración de red, elija Editar. Elija la configuración que corresponda a su entorno.

   - Especifique una VPC.
   - Especifique una subred existente o cree una nueva.
   - Habilite o deshabilite la asignación automática de una dirección IP pública a la interfaz principal. La asignación automática solo es posible si la instancia tiene una única interfaz de red.
   - Asigne un grupo de seguridad personalizado nuevo o existente. El grupo de seguridad debe permitir al menos el acceso de SSH a la instancia. También se deben permitir todos los puertos que requieran las aplicaciones que planee ejecutar en la instancia. Puede modificar el grupo de seguridad en cualquier momento después de crear la instancia.

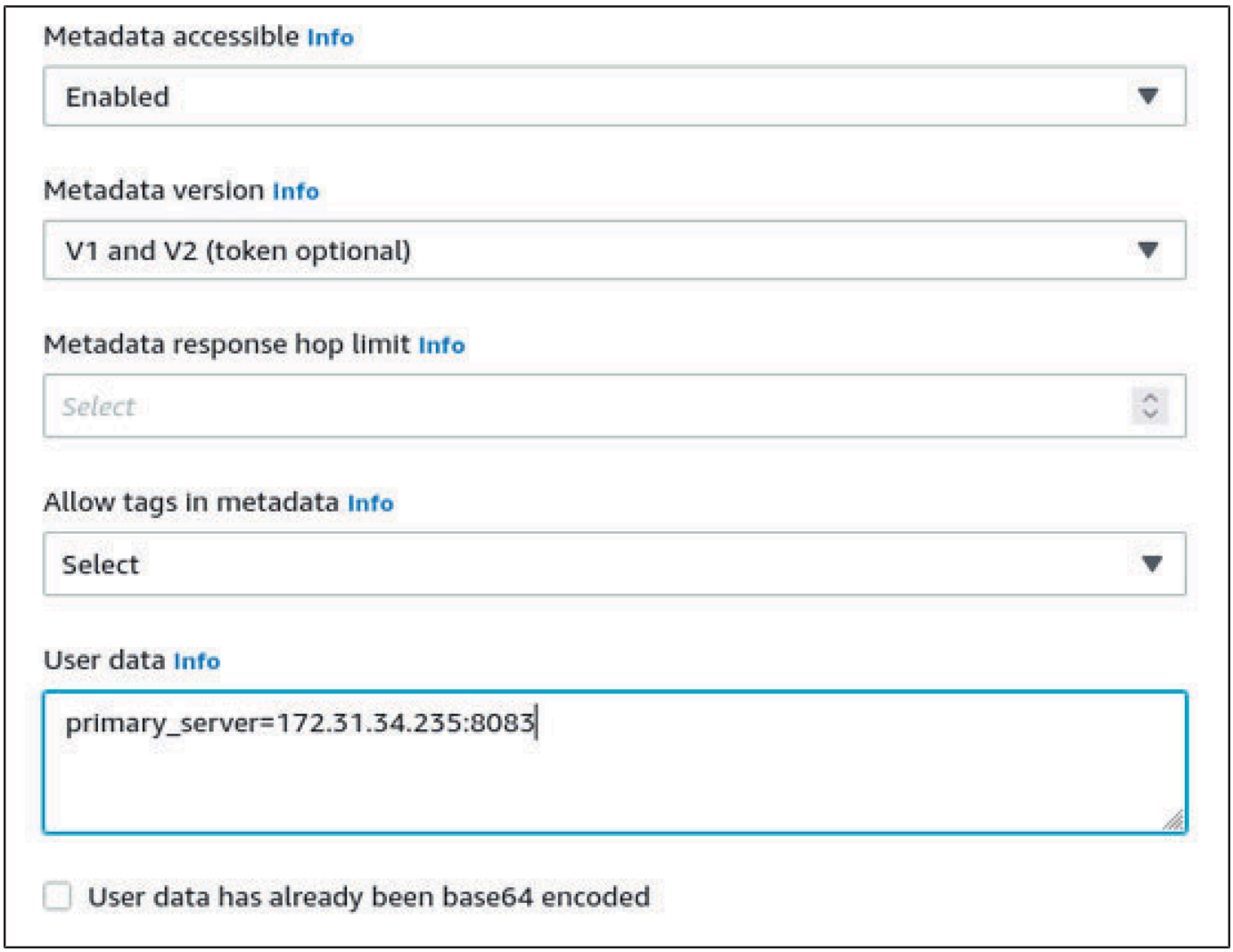
8. En la sección Almacenamiento, para el volumen raíz (el disco del sistema), elija un tamaño adecuado para su entorno. El tamaño mínimo de disco de sistema recomendado para el sistema Linux es de 30 GiB. Para proporcionar espacio para los contenedores de discos virtuales y otros requisitos de almacenamiento, puede agregar más almacenamiento ahora o después de lanzar la instancia. Sin embargo, el tamaño del disco del sistema debe cumplir los requisitos del sistema Linux, incluidas las aplicaciones y utilidades que planee instalar.

   > **Note**
   >
   > Le recomendamos que cree volúmenes de almacenamiento independientes para los datos de las aplicaciones de Charon (por ejemplo, imágenes de disco). Si es necesario, más adelante podrá migrar dichos volúmenes a otra instancia.

9. Expanda la sección Detalles avanzados y active la casilla Especificar opciones de CPU. En la siguiente imagen se muestran a modo de ejemplo tres que tienen más probabilidades de ser útiles en un entorno de emulador de Charon.

10. En el caso de un sistema de servidor de licencias de VE con una versión anterior a la 1.1.23, debe asignar el rol de IAM necesario a la instancia. Debe ser un rol que permita la acción `ListUsers`. Para asignar un rol, en la sección ampliada Detalles avanzados, seleccione un rol en Perfil de instancia de IAM o elija Crear un nuevo perfil de IAM. Para obtener más información, consulta Funciones de IAM para Amazon EC2.
11. Si su instancia está basada en una AWS Marketplace imagen AL de Charon y piensa utilizar los servidores de licencias públicas operados por Stromasys, debe añadir la información correspondiente a la configuración de la instancia antes de lanzarla.

    Introduzca la información del servidor de licencias de AutoVE, tal y como se muestra en la siguiente imagen.



Las siguientes opciones de configuración son válidas:

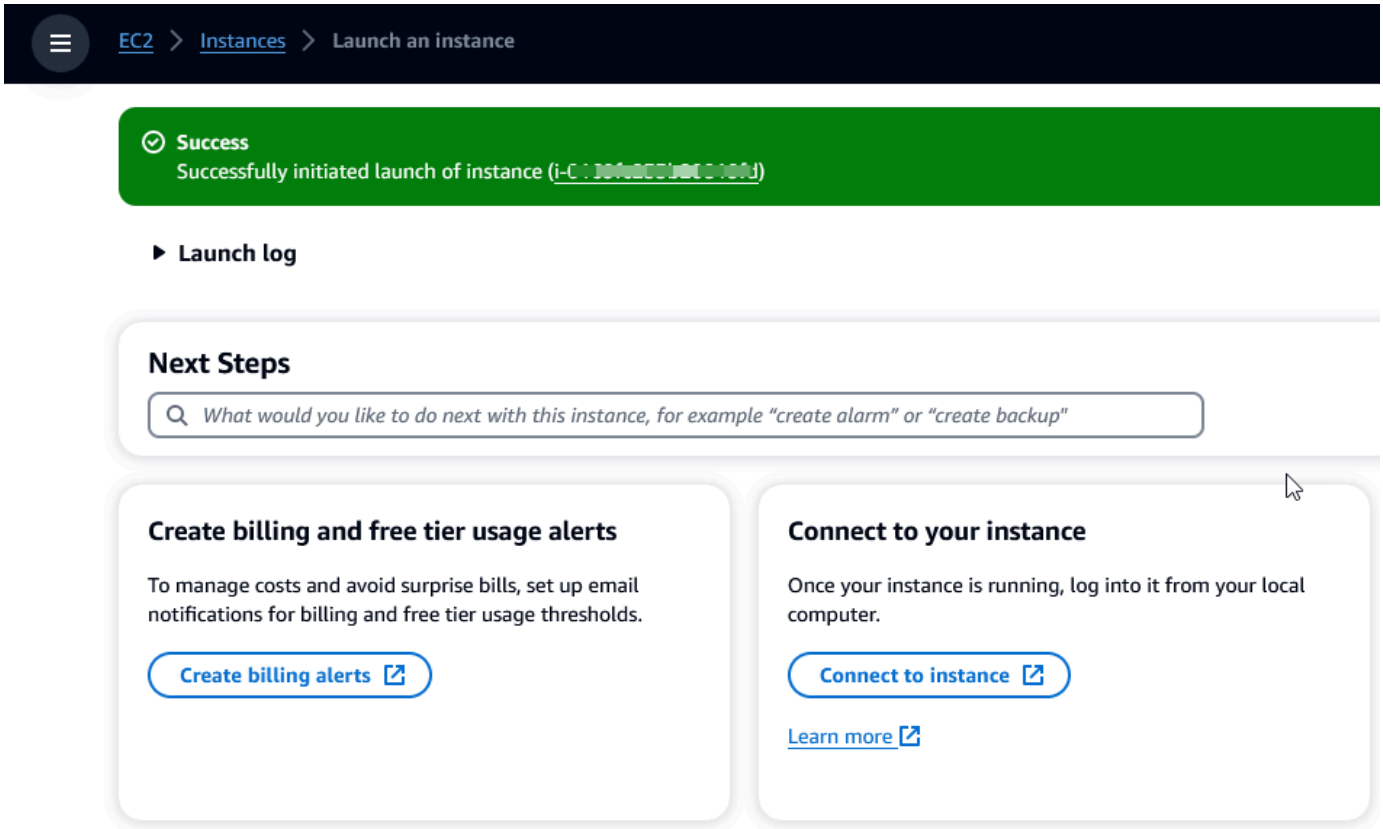
- **`primary_server=`***`<ip-address>`*`[:`*`<port>`*`]`
- **`backup_server=`***`<ip-address>`*`[:`*`<port>`*`]`

Donde

- <ip-address> representa la dirección IP del servidor principal y del servidor de copia de seguridad, según corresponda.
- <port> representa un puerto TCP no predeterminado que se utiliza para comunicarse con el servidor de licencias (predeterminado: TCP/8083).

> **Note**
>
> Se debe configurar al menos un servidor de licencias en el lanzamiento inicial para habilitar el modo AutoVE. De lo contrario, la instancia se vinculará a uno de los servidores de licencias públicos operados por Stromasys.

12. En el panel Resumen, elija Lanzar instancia. Después de un tiempo, verá el siguiente mensaje de éxito:



13. En la esquina inferior derecha de la pantalla, elija Ver todas las instancias.
14. Para ver los detalles de la instancia, seleccione la casilla de verificación situada a la izquierda de la fila que representa la instancia en la tabla Instancias. Los detalles de la instancia se mostrarán en la parte inferior de la pantalla. Para obtener información sobre cómo conectarse a tu instancia, consulta Connect en la Guía del EC2 usuario de Amazon.

AWS Modernización del mainframe y cambio de plataforma con NTT DATA

AWS Mainframe Modernization ofrece una variedad de Amazon Machine Images (AMIs). Esto AMIs facilita el aprovisionamiento rápido de las EC2 instancias de Amazon, lo que crea un entorno personalizado para realojar y cambiar la plataforma de las aplicaciones de mainframe mediante NTT Data. AWS Esta guía proporciona los pasos necesarios para acceder a ellas y utilizarlas. AMIs

Requisitos previos

- Asegúrese de tener acceso de administrador a una AWS cuenta en la que pueda crear EC2 instancias de Amazon.
- Compruebe que el servicio de modernización de AWS mainframe esté disponible en la región en la que planea crear las EC2 instancias de Amazon. Consulte la lista de servicios de AWS disponibles por región.
- Identifique la Amazon VPC en la que quiere crear las instancias de Amazon EC2 .

Suscripción a imágenes de máquina de Amazon

Al suscribirse a un producto del AWS Marketplace, puede lanzar una instancia desde la AMI del producto.

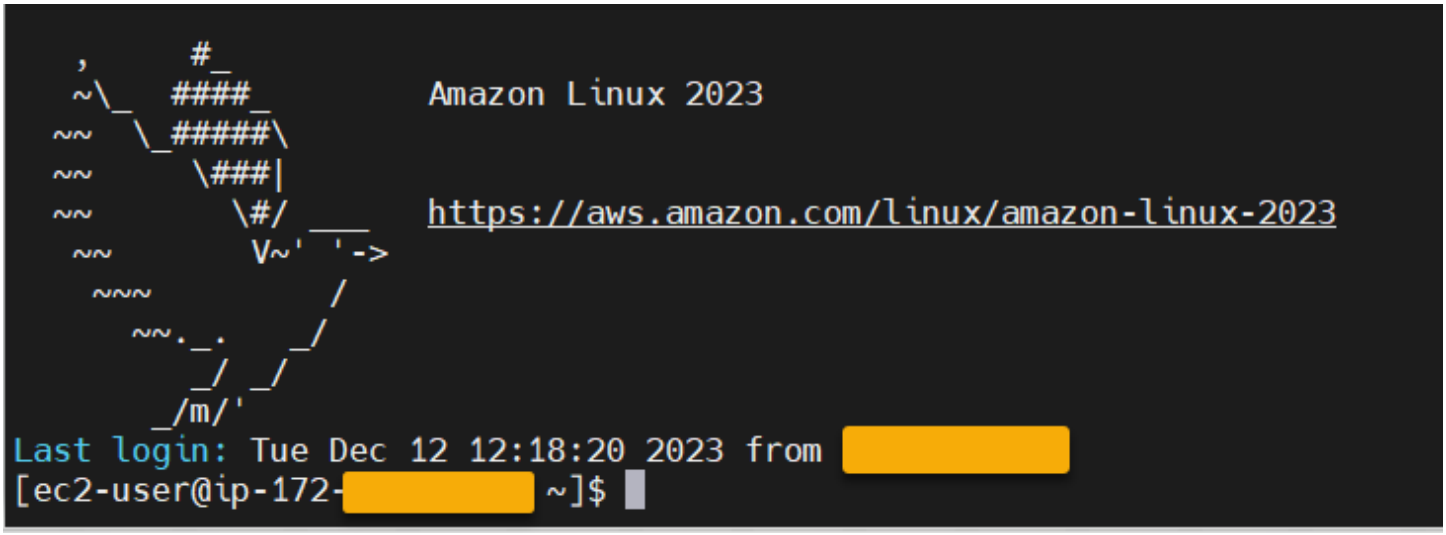
1. Inicie sesión AWS Management Console y abra la AWS Marketplace consola en https://console.aws.amazon.com/market.
2. Elija Administrar suscripciones.
3. Copia y pega el siguiente enlace en la barra de direcciones del navegador: https://aws.amazon.com/marketplace/ pp/prodview-eg227ymldsnx2
4. Elija Continuar con la suscripción.
5. Si los términos y condiciones son aceptables, seleccione Aceptar términos. El proceso de suscripción puede tardar unos minutos.
6. Espera a que aparezca un mensaje de agradecimiento. Este mensaje confirma que se ha suscrito correctamente al producto.
7. En el panel de navegación izquierdo, elija Administrar suscripciones. Esta vista muestra todas las suscripciones.

AWS Lance la replataforma de modernización de mainframe con una instancia de NTT DATA

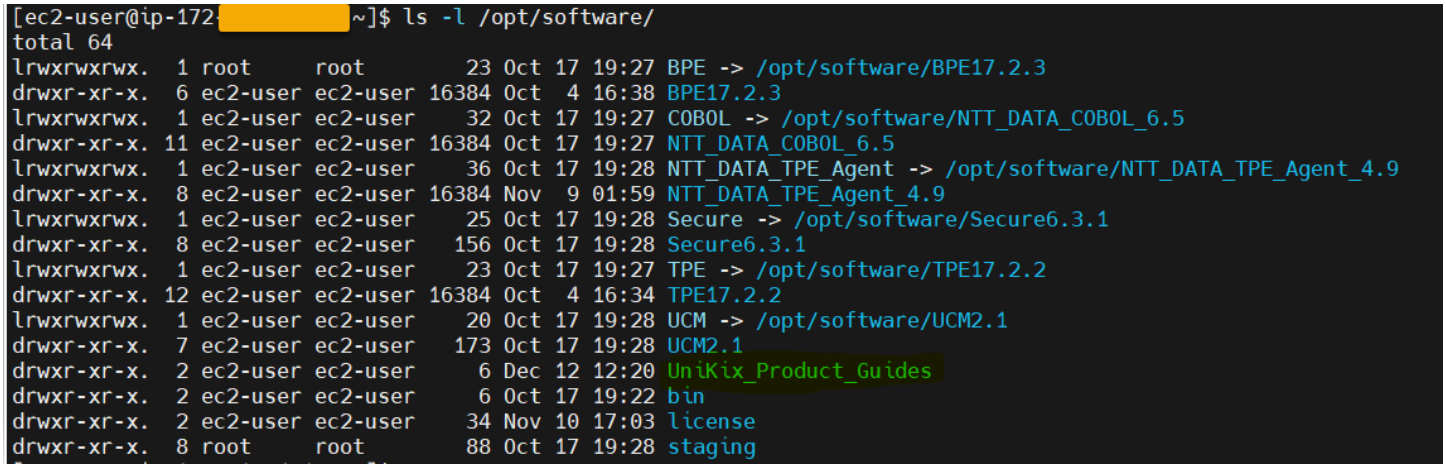
1. Abra la AWS Marketplace consola en /market. https://console.aws.amazon.com
2. En el panel de navegación izquierdo, elija Administrar suscripciones.
3. Busque la AMI que desee lanzar y elija Lanzar nueva instancia.
4. En Región, seleccione la región incluida en la lista de permitidos.
5. Selecciona Continuar para iniciar. EC2 Esta acción te lleva a la EC2 consola de Amazon.
6. Escriba un nombre del servidor.
7. Seleccione un tipo de instancia que se ajuste a los requisitos de rendimiento y costo de su proyecto. El punto de partida sugerido para el tamaño de la instancia es `c5.2xLarge`.
8. Elija un par de claves existente o cree y guarde uno nuevo. Para obtener información sobre los pares de claves, consulte los pares de EC2 claves de Amazon y las instancias de Linux en la Guía del EC2 usuario de Amazon.
9. Edite la configuración de red y elija la VPC de la lista de permitidos y la subred adecuada.
10. Seleccione un grupo de seguridad existente o cree uno nuevo. Si se trata de una EC2 instancia de Enterprise Server Amazon, lo habitual es permitir el tráfico TCP a los puertos 86 y 10086 para administrar la configuración de Rocket Software (anteriormente Micro Focus).
11. Configura el almacenamiento de la EC2 instancia de Amazon.
12. Revise el resumen y seleccione Lanzar instancia. Para que el lanzamiento se realice correctamente, el tipo de instancia debe ser válido. Si se produce un error en el lanzamiento, seleccione Editar la configuración de la instancia y elija un tipo de instancia distinto.
13. Cuando vea el mensaje de éxito, seleccione Conectarse a la instancia.
14. Abre la EC2 consola de Amazon en https://console.aws.amazon.com/ec2/.
15. En el panel de navegación, bajo Instancias elija Instancias.
16. En el panel principal, compruebe el estado de la instancia.

Introducción al uso de NTT Data

Después de aprovisionar la EC2 instancia de Amazon, usa SSH en ella con el nombre `ec2-user` de usuario. La pantalla tendrá un aspecto parecido al que se muestra en la siguiente imagen.



Debajo de la carpeta `/opt/software/`, hay una carpeta con el nombre `UniKix_Product_Guides`, como se muestra en la siguiente imagen.



La `UniKix_Product_Guides` carpeta incluye la documentación de los siguientes componentes que están instalados en esta EC2 instancia de Amazon:

- NTT DATA TPE
- NTT DATA BPE
- NTT DATA Enterprise COBOL
- NTT DATA Secure UniKix
- Administrador central de NTT DATA UniKix

La carpeta `software` que aparece en la imagen anterior contiene los binarios de los componentes que se muestran arriba.

Tras validar correctamente la EC2 instancia de Amazon, comience a utilizar AWS Mainframe Modernization Replatform con NTT DATA siguiendo la documentación de NTT Data.

Tutorial: Implemente CardDemo la aplicación en NTT DATA

Esta página le guía a través del step-by-step proceso de implementación de la aplicación de CardDemo ejemplo en la replataforma de modernización del AWS mainframe con el tiempo de ejecución de NTT DATA Unikix.

La aplicación CardDemo de muestra es una aplicación de mainframe simplificada diseñada y desarrollada para probar y mostrar la tecnología asociada a los casos de uso de AWS migración y modernización de mainframes.

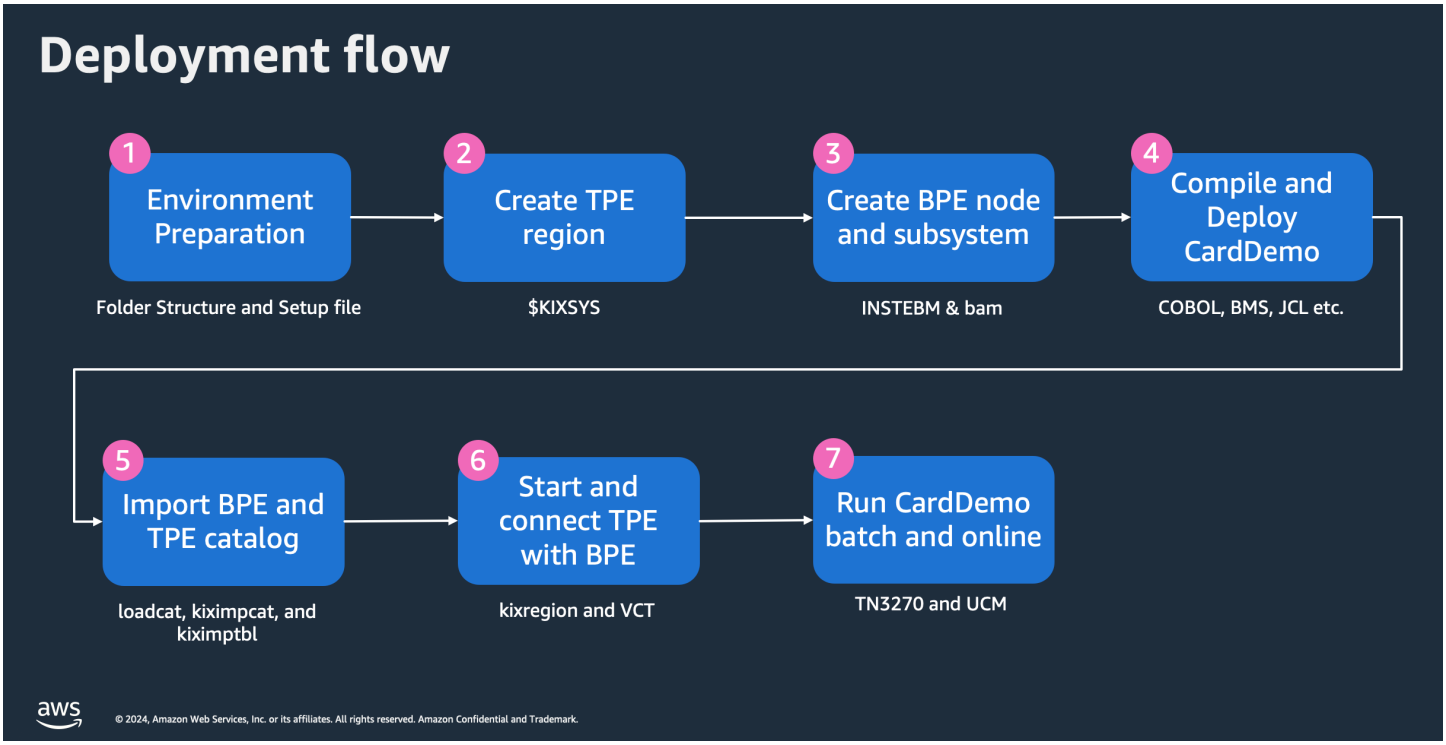
Para obtener más información sobre esta aplicación, consulte el repositorio de. GitHub CardDemo

Temas

- Diagrama de flujo de implementación
- Requisitos previos
- Paso 1: preparación del entorno
- Paso 2: creación de una región de TPE
- Paso 3: creación del subsistema y del nodo de BPE
- Paso 4: Compilar e implementar la aplicación CardDemo
- Paso 5: importación del catálogo de BPE y TPE
- Paso 6: inicio y conexión de TPE con BPE
- Paso 7: ejecuta la aplicación CardDemo
- Solución de problemas

Diagrama de flujo de implementación

En el siguiente diagrama se muestra cada paso del flujo de trabajo para implementar una aplicación en el tiempo de ejecución de UniKix de NTT DATA.



Requisitos previos

- Siga las instrucciones que se proporcionan en la sección Redefinición de plataforma con NTT DATA Uso de la AMI de NTT DATA UniKix Marketplace.
- Modifique la opción de metadatos de la instancia IMDSv2a Opcional como se menciona en Restaurar el uso de IMDSv1 en la guía del EC2 usuario de Amazon.
- Descargue los componentes CardDemo de tiempo de ejecución para NTT DATA UniKix desde GitHub repositorio.
- Inicie sesión en la EC2 instancia UniKix de ejecución como`ec2-user`.
- Extraiga los componentes CardDemo de tiempo de ejecución descargados mediante este enlace: UniKix_ CardDemo _runtime_v1.zip.
  - El directorio extraído debería contener los directorios `migrated_app` y `bin`.
  - Mueva los directorios `migrated_app` y `bin` al directorio `$HOME`. La ruta sería como `/home/ec2-user`.
  - Debería tener los siguientes directorios en `$HOME`:
    - `/home/ec2-user/bin`
    - `/home/ec2-user/migrated_app`

- Mueva todos los archivos dentro del directorio $HOME/bin con el siguiente comando:
- • `chmod +x $HOME/bin/*`

Paso 1: preparación del entorno

Tras completar los requisitos previos, el primer paso es preparar el entorno en el que desea implementar la CardDemo aplicación.

1. Inicie sesión en la EC2 instancia UniKix de ejecución como`ec2-user`.
2. Observe la lista de UniKix software preempaquetado en la AMI, como TPE, BPE y COBOL, junto con otros de la ubicación del UniKix producto de NTT DATA mediante el siguiente comando en su instancia: EC2

   ```
   ls -l /opt/software/
   ```

3. Examine la aplicación migrada. CardDemo Verá todo el código fuente, incluidos los mapas BMS, los programas COBOL, los cuadernos COBOL y. JCLs También encontrará la exportación de los catálogos de BPE y TPE, las definiciones de recursos del CICS y los datos migrados, como los archivos secuenciales y los archivos VSAM, de la siguiente manera:

   ```
   ls  $HOME/migrated_app/*/*
   ```

4. Para crear la estructura de carpetas, ejecute el script `create_project` con el siguiente comando:

   ```
   sh $HOME/bin/create_project
   ```

5. Active el CardDemo entorno buscando el archivo de configuración mediante: `carddemo.env`

   ```
   source $HOME/bin/carddemo.env
   ```

Paso 2: creación de una región de TPE

Una vez que haya activado el entorno en el que desea implementar la aplicación, debe crear una región de TPE.

1. Cree una región de TPE mediante el comando `kixregion createRegion` que requiere entradas como `$KIXSYS`, `$JAVA_HOME` y `$KIXLICDIR`. Estas variables de entorno ya están configuradas en el archivo de configuración `carddemo.env`.

```
kixregion createRegion $KIXSYS $JAVA_HOME $KIXLICDIR
```

2. Configure la región de TPE mediante el comando `kixregion setAttr`.

```
kixregion setAttr $KIXSYS server.tx.languages.cobol.enabled true
kixregion setAttr $KIXSYS server.tx.languages.cobol.flavor vcobol
kixregion setAttr $KIXSYS server.tx.languages.cobol.home $VCOBOL
kixregion setAttr $KIXSYS maps.location $PROJECT_ROOT/maps
kixregion setAttr $KIXSYS programs.location $PROJECT_ROOT/loadlib
kixregion setAttr $KIXSYS environment.KIXDATA $KIXDATA
kixregion setAttr $KIXSYS td.jobq.submission.node $EBMHOME
kixregion setAttr $KIXSYS td.jobq.submission.subsys $EBMSYS
```

3. Para generar el archivo de entorno de usuario específico para esta región de TPE, ejecute el comando `kixregion createScript`. Este comando crea o actualiza `$KIXSYS/bin/userenv` en función de la configuración de la región de TPE.

```
kixregion createScript $KIXSYS
```

4. Active la región de TPE mediante el aprovisionamiento del archivo de entorno de usuario (`$KIXSYS/bin/userenv`).

```
source $KIXSYS/bin/userenv
```

5. Cree la región de TPE ejecutando el comando `kixinstall2`.

```
kixinstall2
```

Paso 3: creación del subsistema y del nodo de BPE

Después de crear una región de TPE, debe crear el subsistema y el nodo de BPE siguiendo estos pasos.

1. Cambie la propiedad y los permisos de `INSTEBM`.

```
sudo chown root $INSTEBM
```

```
sudo chmod 4755 $INSTEBM
```

2. Cree un nodo de BPE mediante el comando `INSTEBM`. El directorio de nodos de BPE se proporciona como el parámetro de entrada.

```
$INSTEBM $EBMHOME
```

3. Active el entorno de lotes mediante el aprovisionamiento del archivo `batchenv` a partir del nodo de BPE recién creado.

```
source $EBMHOME/batchenv
```

4. Cree el subsistema de BPE dentro de este nodo mediante Batch Administration Manager (bam). El comando `bam` abrirá la interfaz de Batch Administration Manager.

```
bam
```

   a. Inicie el nodo de BPE mediante la interfaz de BAM. Elija la opción 2, Entornos del sistema en el menú principal.

   b. Elija la opción 2, Iniciar/detener el nodo de lote para iniciar el nodo de BPE.

```
                System Environments                          2024/09/17 21:27:03




             1   Report System Status
             2   Start/(Stop) Batch Node
             3   Assign the Console
             4   Change the Date
             5   Redirect Job Output
             6   Inter-Node Communications
             7   Job Accounting
             8   Assign the Initial Job Number
             9   Enable/Disable Duplicate Name Execution Delay : Enabled

             H - Help
             R - Return to Main Menu

--------------------------------------------------------------------------------
Type an option and press Return: 2
```

c. Una vez iniciado, pulse la tecla Intro dos veces para volver al menú principal de BAM.

```
                System Environments: Start Batch Node        2024/09/17 21:28:28









              Batch Node Startup Completed.









--------------------------------------------------------------------------------
Press Return to Continue
```

d. Para crear el subsistema de BPE, elija la opción 3, Aplicaciones y subsistemas.

```
                Batch Administration Manager                2024/09/17 21:29:03




             1   Software License Management
             2   System Environments
             3   Applications & Subsystems
             4   Security & Users
             5   Classes & Activities
             6   Problem Determination



             H - Help
             Q - Quit

--------------------------------------------------------------------------------
Type an option and press Return: 3
```

e. A continuación, elija la opción 3, Crear un subsistema.

```
                Applications & Subsystems                   2024/09/17 21:29:57




             1   List All Subsystems
             2   Query a Subsystem
             3   Create a Subsystem
             4   Update a Subsystem
             5   Delete a Subsystem
             6   Change/Show Default Subsystem
             7   Import a Subsystem (from another batch node)
             8   Create a BPESUB Project
             9   Export Subsystem bldsub Config File

             H - Help
             R - Return to Main Menu

--------------------------------------------------------------------------------
Type an option and press Return: 3
```

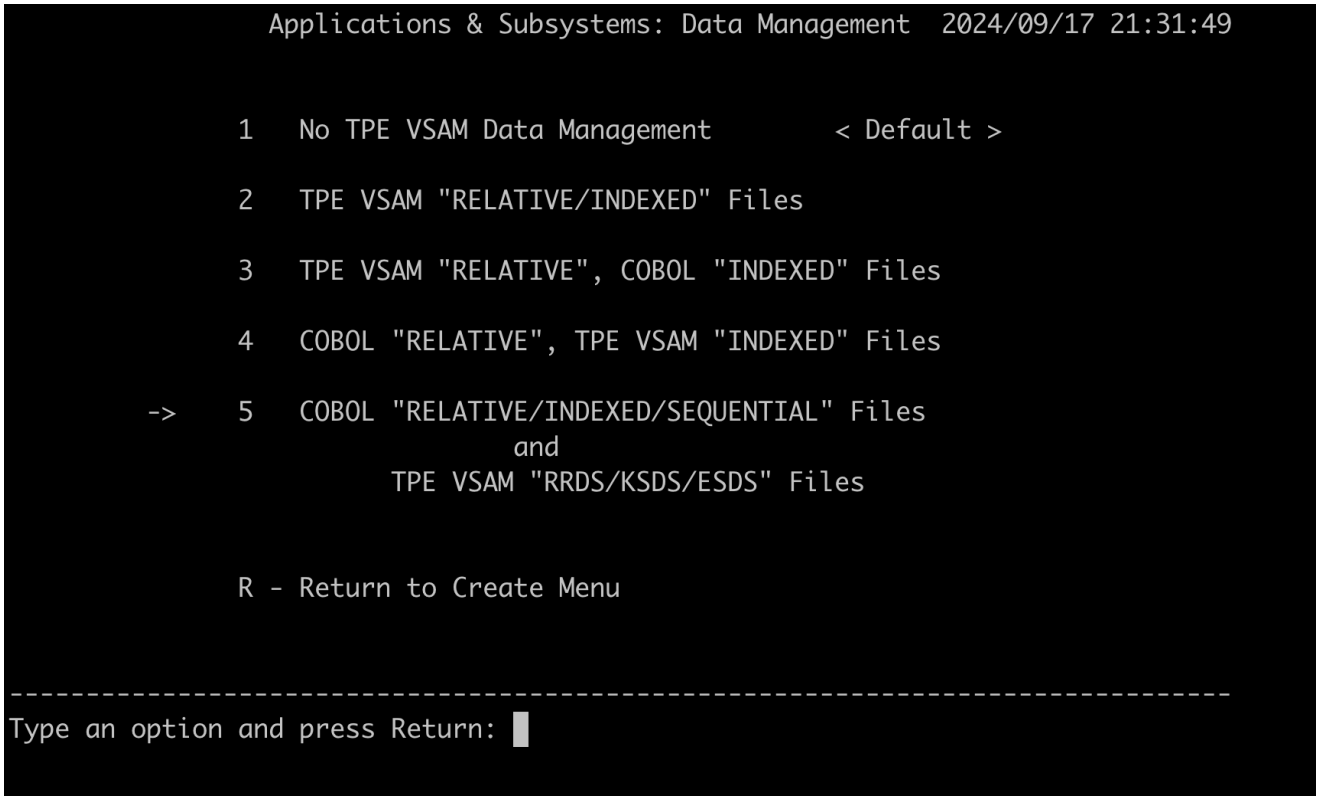
f. Introduzca el nombre del subsistema como `sys1`.

```
                Applications & Subsystems: Create          2024/09/17 21:30:22









         No Subsystems are currently defined.









             R - Return to Previous Screen

--------------------------------------------------------------------------------
Enter the Subsystem's name you want to create: sys1
```

g. Elija la opción 3, Administración de datos.

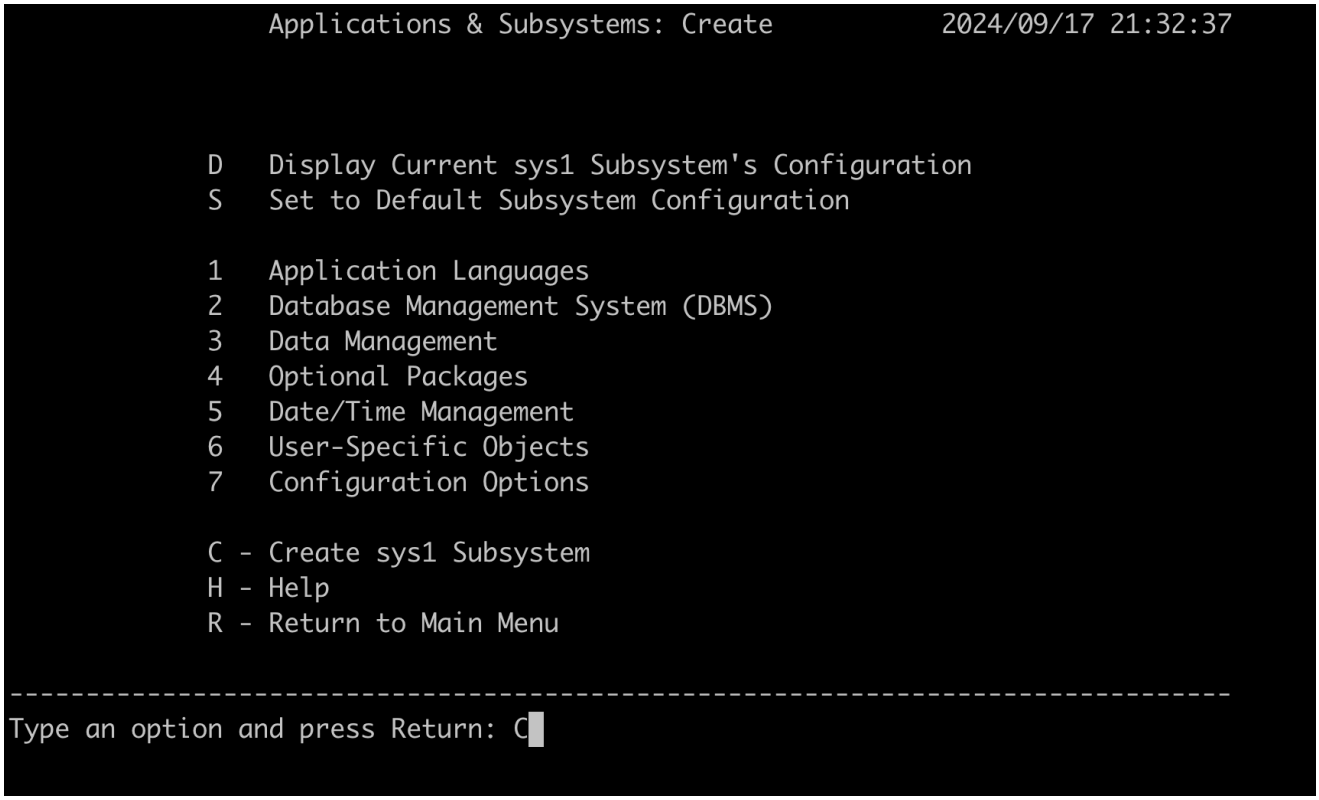
```
                Applications & Subsystems: Create          2024/09/17 21:30:53



             D   Display Current sys1 Subsystem's Configuration
             S   Set to Default Subsystem Configuration

             1   Application Languages
             2   Database Management System (DBMS)
             3   Data Management
             4   Optional Packages
             5   Date/Time Management
             6   User-Specific Objects
             7   Configuration Options

             C - Create sys1 Subsystem
             H - Help
             R - Return to Main Menu

--------------------------------------------------------------------------------
Type an option and press Return: 3
```

h. Elija la opción 5, ya que la CardDemo aplicación incluye archivos secuenciales y VSAM.



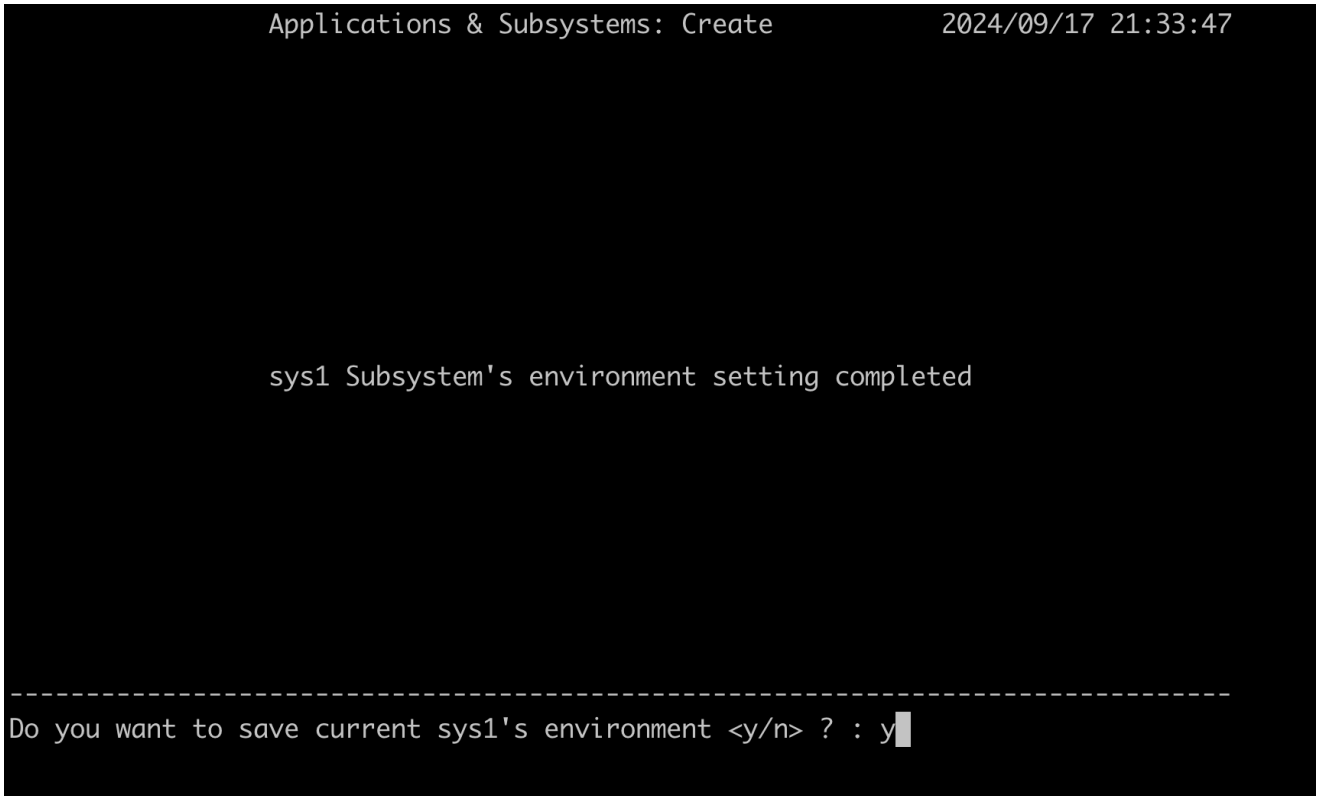
i. (Opcional). Pulse R para volver a la página Crear menú y revise las diferentes opciones de configuración disponibles.

j. En la página Crear, introduzca C para crear el subsistema `sys1`.

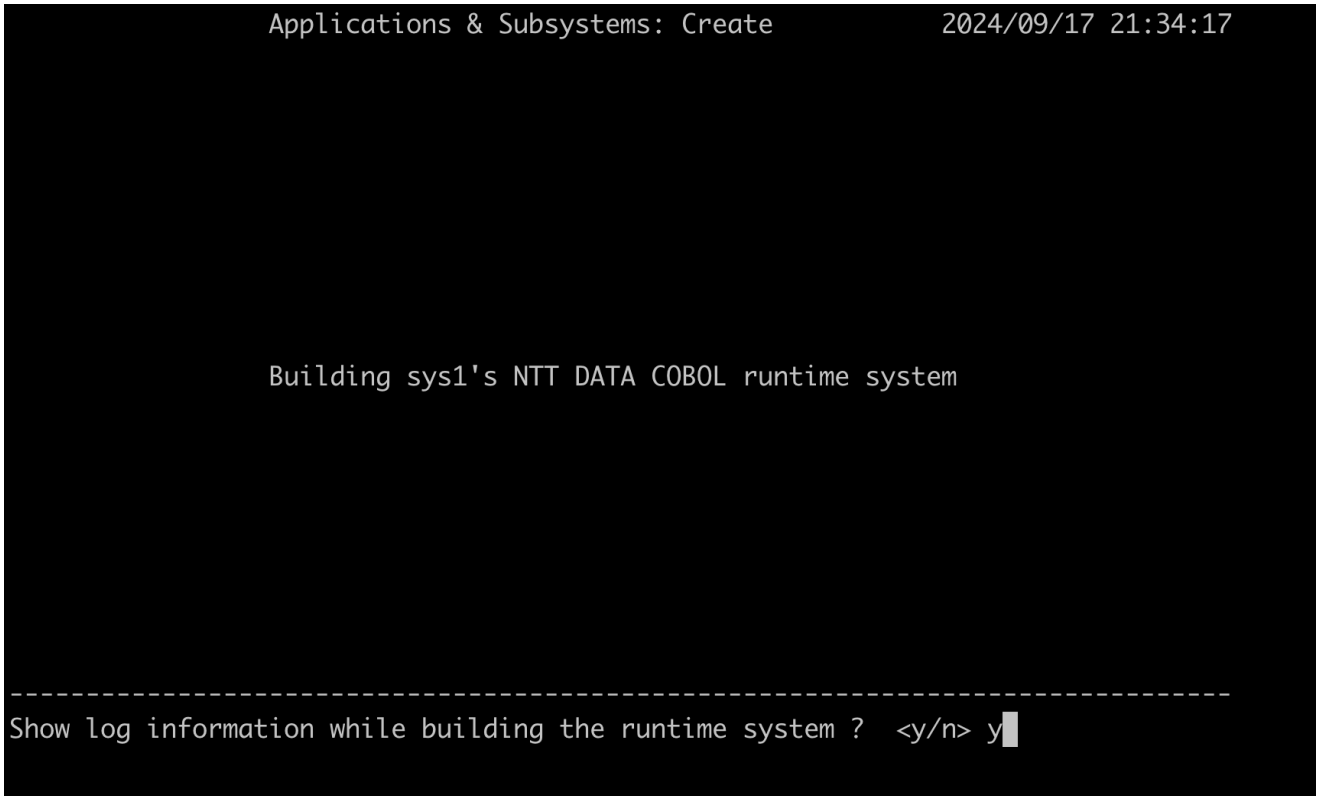


k. Revise la configuración e introduzca C para continuar con el resto de la configuración del entorno. Esta configuración del entorno se rellena previamente porque se han establecido las variables de entorno necesarias definidas en el archivo de configuración `carddemo.env` y la estructura de carpetas recomendada.

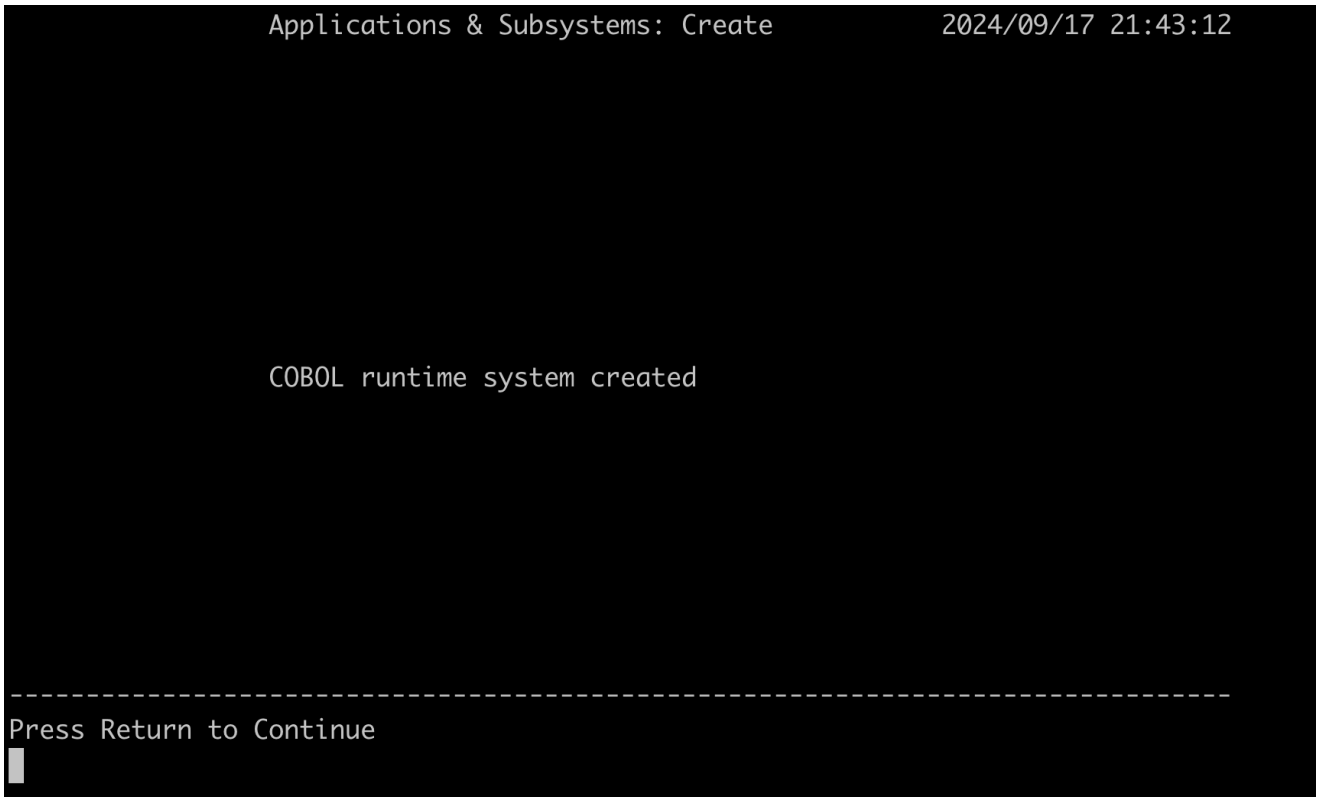
l. Introduzca s para confirmar y guardar la configuración del entorno actual.

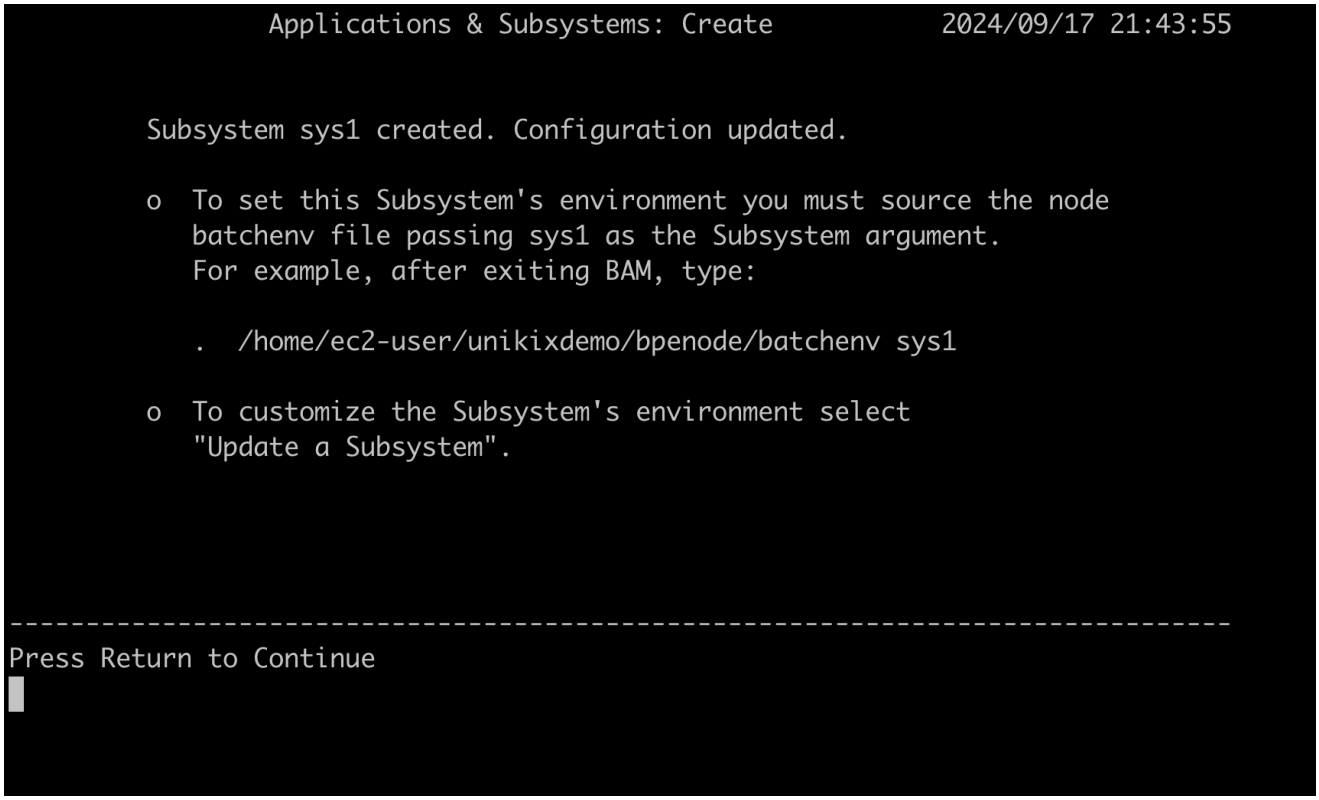


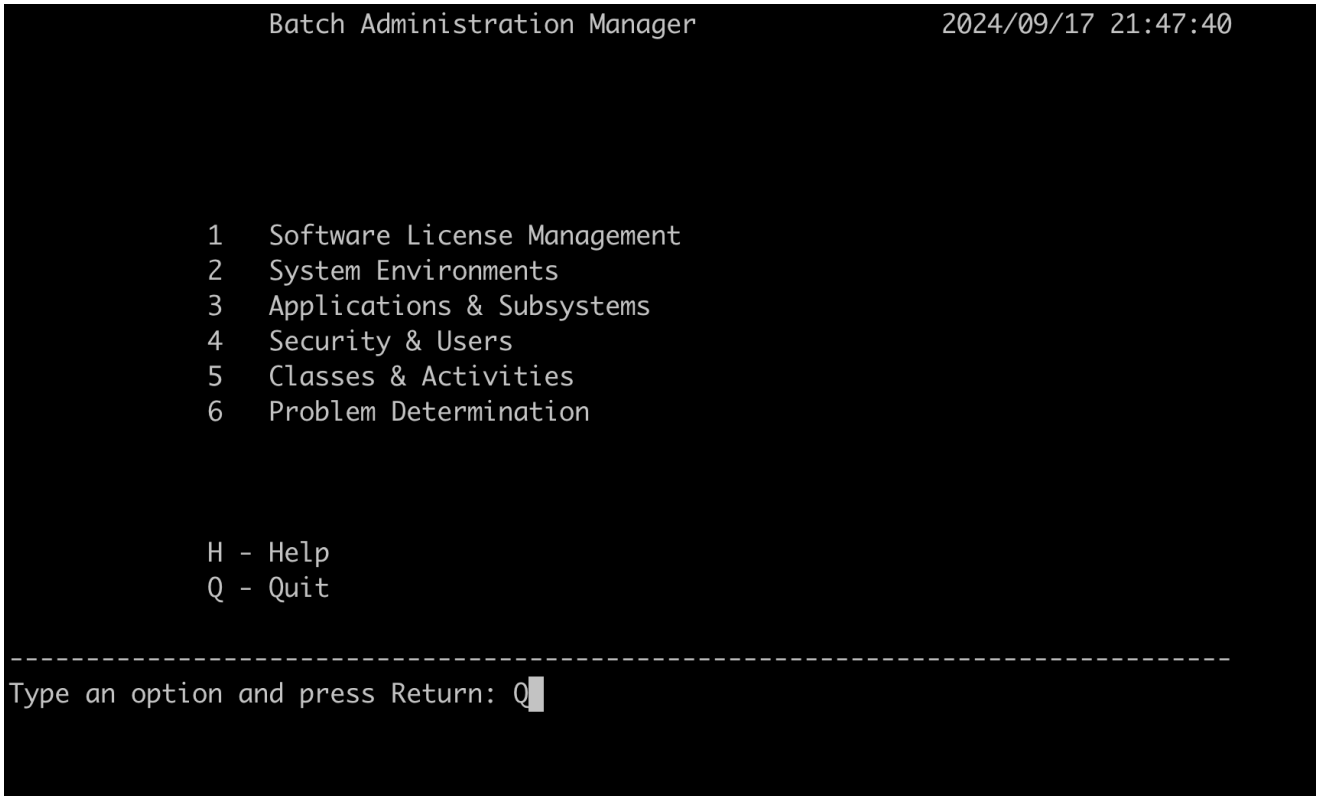
m. Introduzca s para mostrar el registro al crear el subsistema.



n. Pulse la tecla Intro hasta volver al menú principal y salir de la interfaz de BAM seleccionando la opción Salir.







5. Active el subsistema de BPE mediante el aprovisionamiento de `batchenv` con el nombre del subsistema `sys1`.

```
source $EBMHOME/batchenv sys1
```

Paso 4: Compilar e implementar la aplicación CardDemo

En este paso, se compilan los programas de COBOL y se implementan los artefactos de aplicación, como el JCL, los procedimientos, los archivos de datos y las definiciones de recursos del CICS.

1. Vuelva a activar el CardDemo entorno buscando el archivo `carddemo.env` de configuración.

```
source $HOME/bin/carddemo.env
```

2. Vaya al directorio de origen de COBOL.

```
cd $MIGAPP_DIR/cbl
```

3. Compile el programa de Cobol `CBACT01C.cbl` mediante el script `compile`.

```
compile CBACT01C.cbl
```

4. Compile todos los programas de Cobol mediante el script `compile.all`.

```
compile.all
```

5. Navegue hasta el directorio de origen de mapas de BMS.

```
cd $MIGAPP_DIR/bms
```

6. Compile el mapa de BMS `COACTUP.bms` mediante el script `compbms`.

```
compbms COACTUP.bms
```

7. Compile todos los mapas de BMS mediante el script `compbms.all`.

```
compbms.all
```

8. Verifique los binarios compilados para los mapas COBOL y BMS.

```
ls $PROJECT_ROOT/loadlib
ls $PROJECT_ROOT/maps
```

9. Implemente otros artefactos de aplicación, como el JCL, los procedimientos, los archivos de datos y las definiciones de recursos del CICS mediante el script `deploy_app`.

```
deploy_app
```

10. Desplácese hasta el directorio JCL del proyecto.

```
cd $PROJECT_ROOT/jcl
```

11. Traduzca ACCTFILE de JCL a macro de JCL de BPE. Utilice el comando `mvstrans`, con la opción -v para la verificación de JCL y la opción -v para crear la macro.

```
mvstrans ACCTFILE -v
mvstrans ACCTFILE -f
```

12. Traduzca el procedimiento JCL REPROC a la macro de procedimiento de JCL de BPE. Utilice el comando `mvstrans` con la opción -p junto con la opción -v para la verificación y la opción -f para crear la macro.

```
mvstrans REPROC -v -p
mvstrans REPROC -f -p
```

13. Translate all JCLs y los procedimientos de JCL.

```
for file in "./jmvs/*"; do mvstrans $file -f; done > jmvs.out
for file in "./mvsp/*"; do mvstrans $file -p -f; done > mvsp.out
```

Paso 5: importación del catálogo de BPE y TPE

En este paso, se importa el catálogo de BPE y TPE mediante diferentes comandos.

1. Importe el catálogo de BPE mediante el comando `loadcat`.

```
loadcat $MIGAPP_DIR/catlg/bpe/BPECAT*
```

2. Vaya al directorio $KIXSYS.

```
cd $KIXSYS
```

3. Importe el catálogo de TPE mediante el comando `kiximpcat`.

```
kiximpcat -c CATALOG -l CATALOG.lst
```

4. Importe las definiciones de recursos de CICS mediante el comando kiximptbl.

```
kiximptbl
```

Paso 6: inicio y conexión de TPE con BPE

En este paso, debe iniciar la región de TPE creada anteriormente junto con el administrador de BPE y conectarlos para poder ejecutar la aplicación de muestra. CardDemo

1. Ejecute el comando `kixverify` en todos los archivos VSAM para asegurarse de que se restablezcan y de que se cierren todos los archivos abiertos anteriormente.

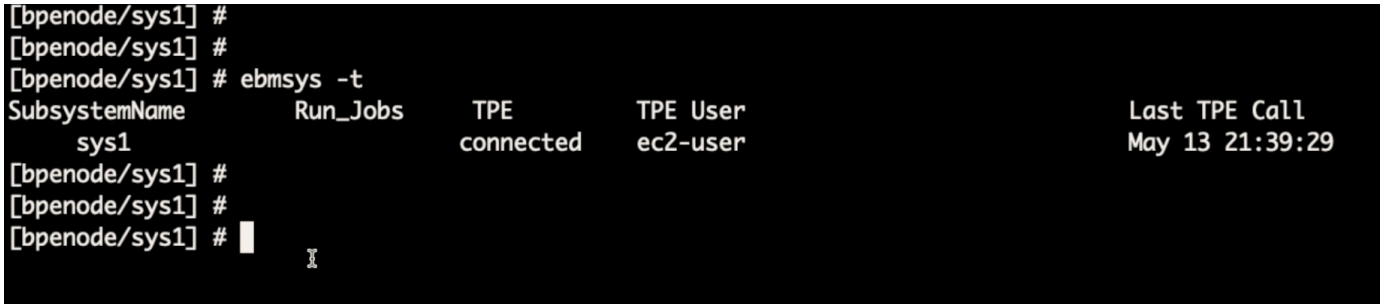
```
kixverify -r ALL
```

2. Inicie la región de TPE.

```
kixregion start $KIXSYS
```

3. Asegúrese de que tanto BPE como TPE estén conectados. Esto es fundamental, ya que los archivos de VSAM son propiedad de TPE y cualquier operación por lotes que acceda al VSAM requerirá una conexión a TPE.

```
ebmsys -t
```



Paso 7: ejecuta la aplicación CardDemo

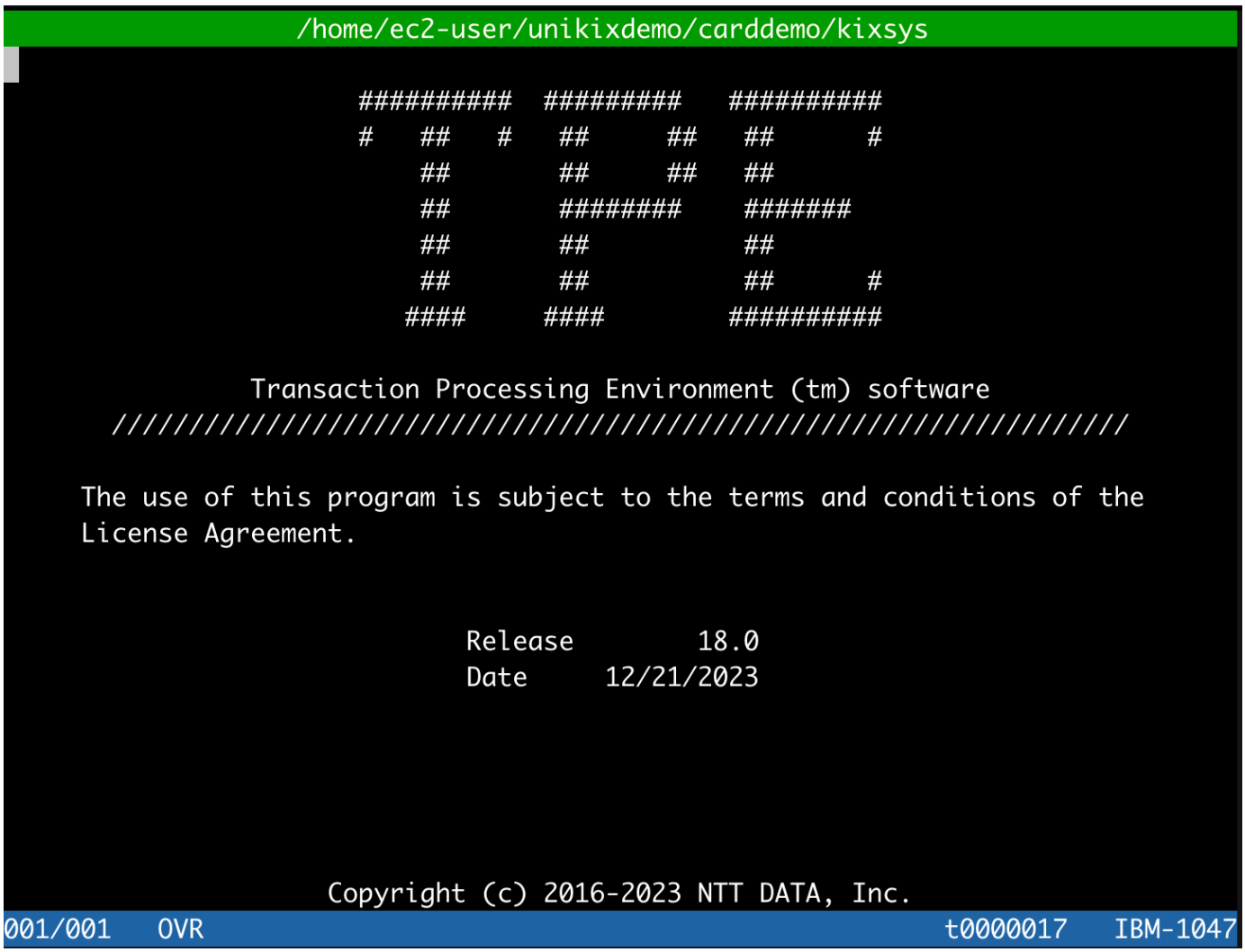
En este paso, ejecuta la CardDemo aplicación en el emulador de terminal TN327 0.

La AMI en tiempo de UniKix ejecución incluye TN327 0 emuladores de terminal que puede iniciar directamente desde la UniKix EC2 instancia.

Conéctese a TPE mediante el emulador de terminal TN327 0

- Inicie el terminal TN327 0 con el `kixterm` comando.

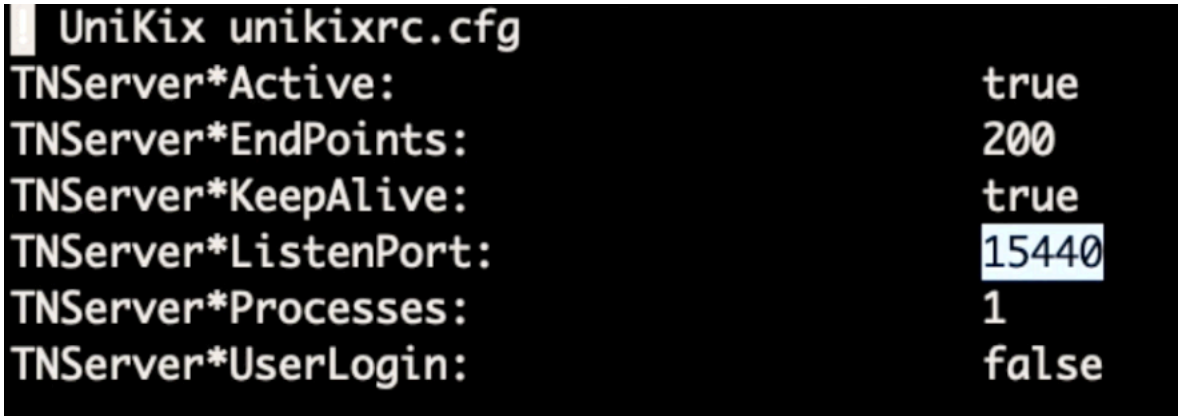
```
kixterm
```



(Opcional). Si desea utilizar su propio emulador de terminal:

1. Obtenga la dirección IP de la instancia de tiempo de UniKix ejecución de la EC2 consola de Amazon.
2. Obtenga el número de puerto para conectarse a la región de TPE mediante el emulador de terminal TN327 0. Puede encontrarlo en el TNServer ListenPort archivo unikixrc.cfg.

```
cat $KIXSYS/unikixrc.cfg
```

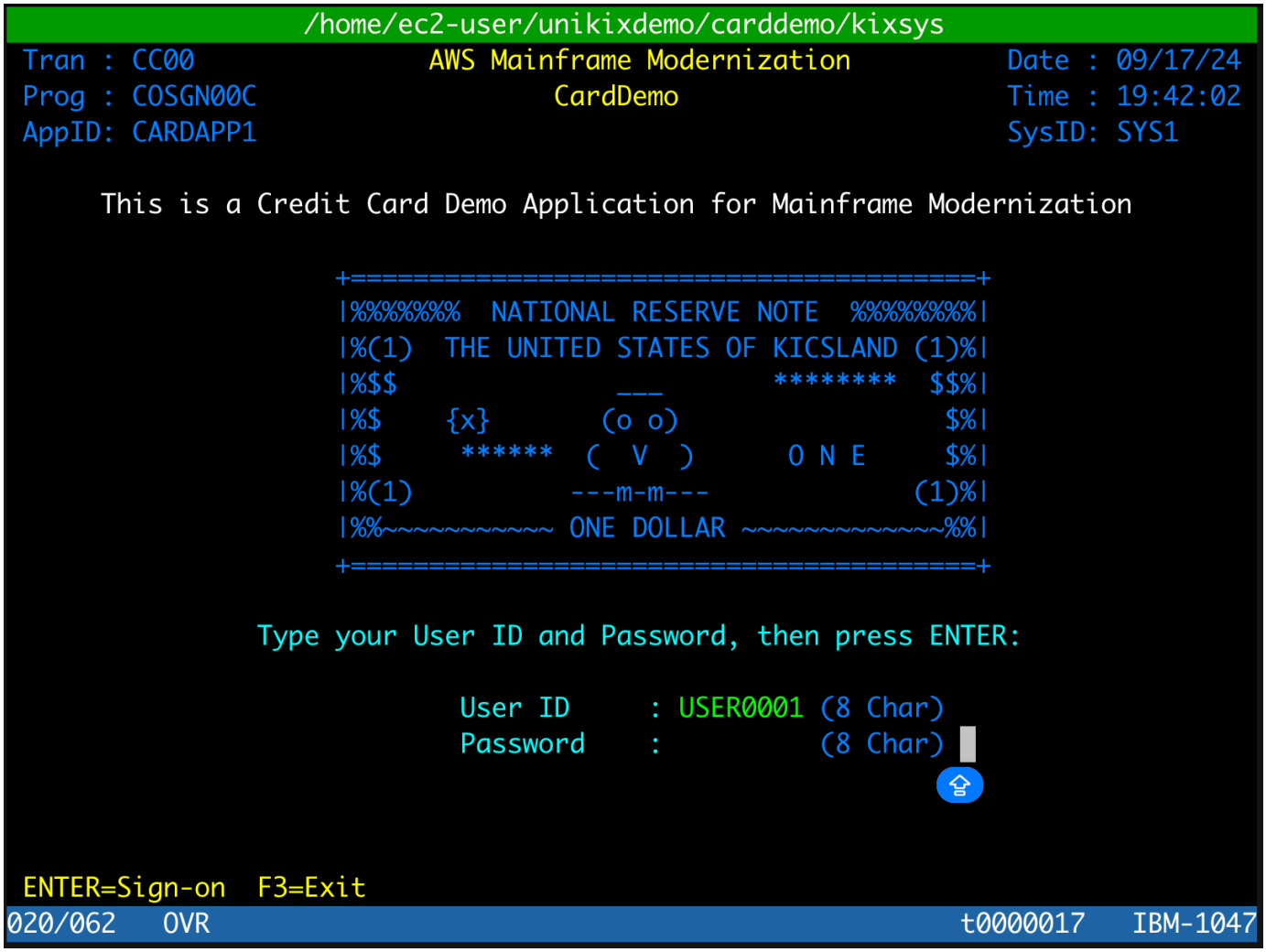


3. Configura tu emulador de terminal TN327 0 para que utilice la dirección IP de la instancia de UniKix ejecución y el número de puerto 15440.

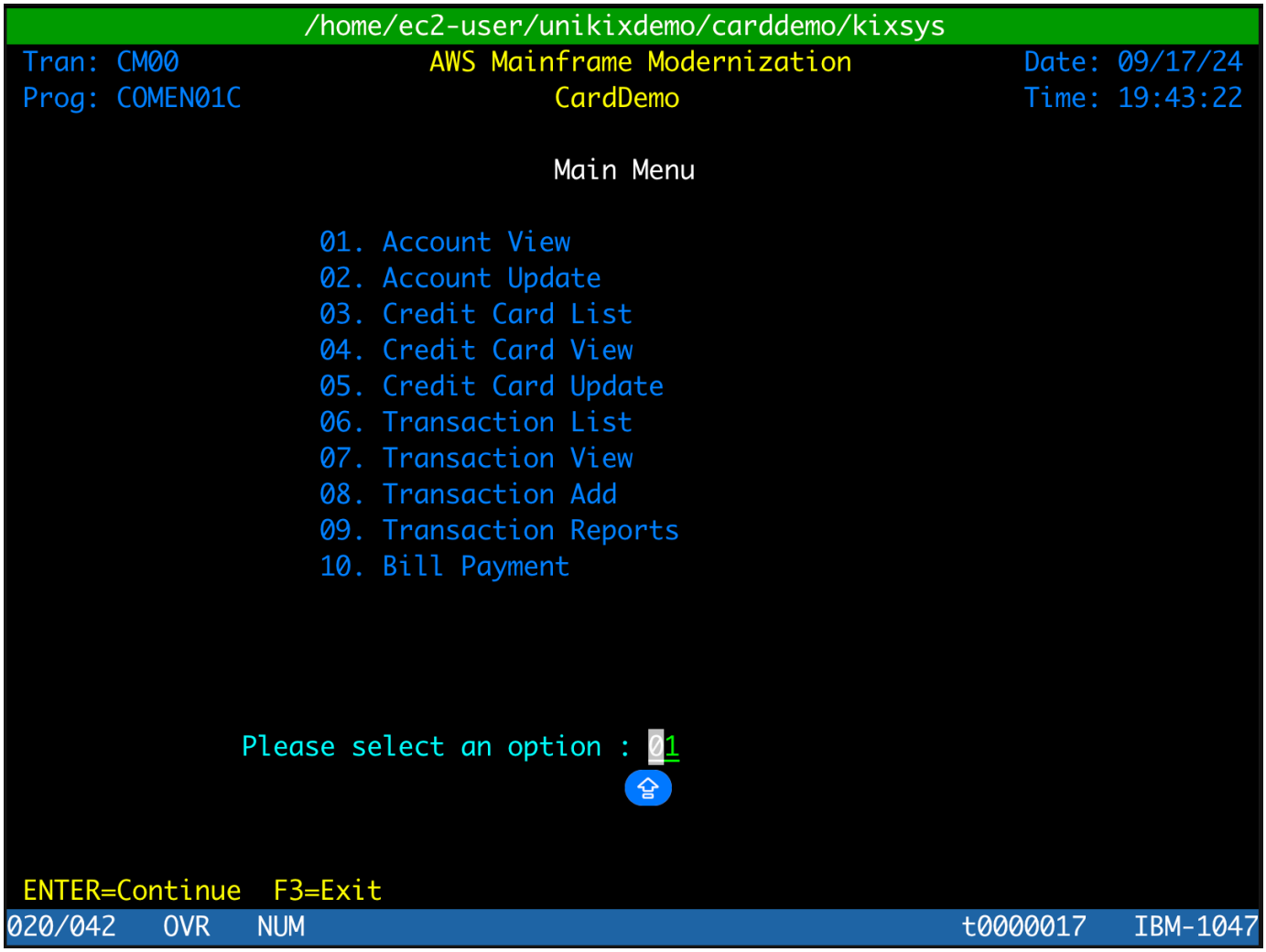
Transacciones en línea

En esta sección se supone que te has conectado al emulador de terminal TN327 0 mediante el `kixterm` comando.

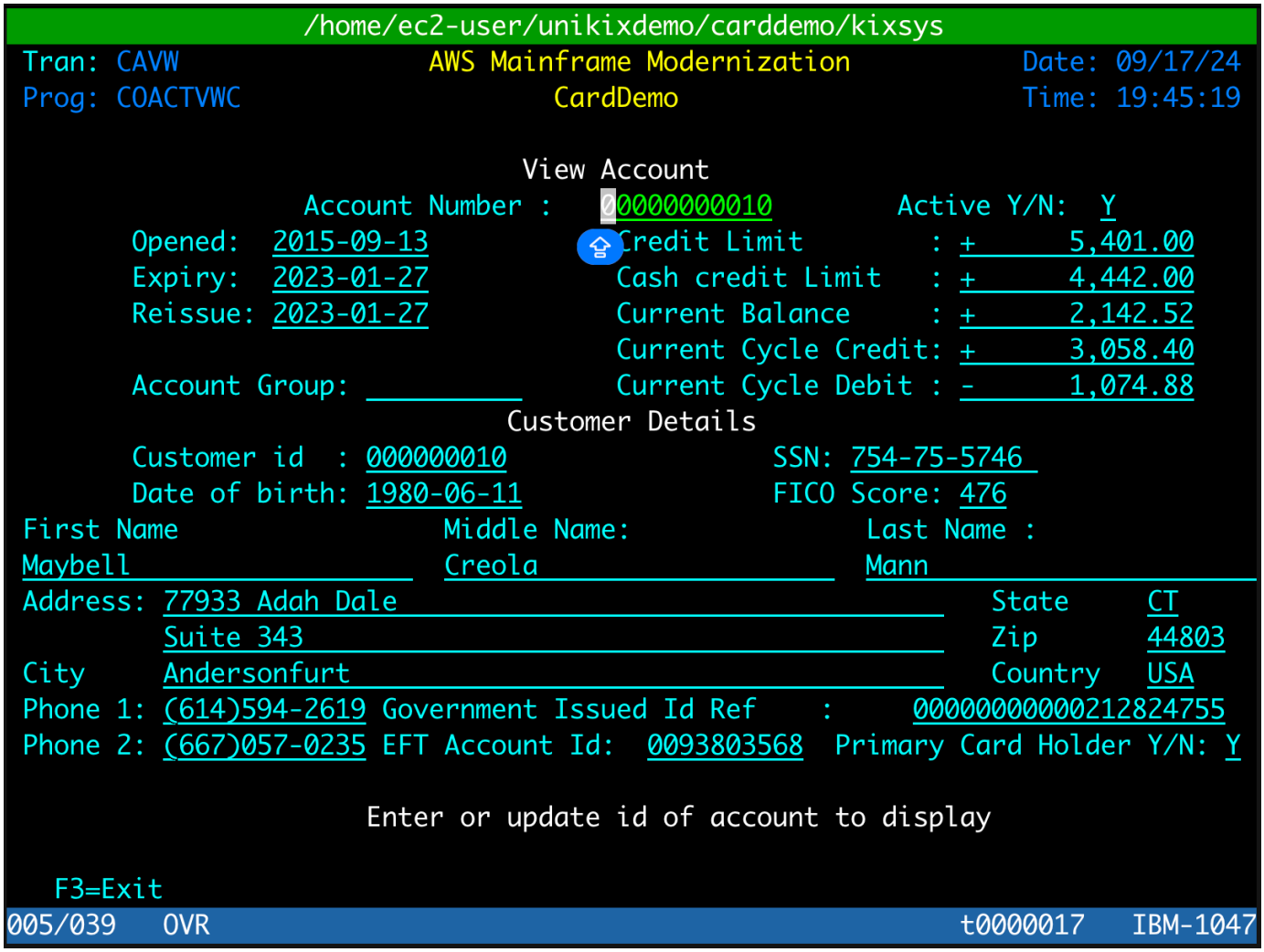
1. Tras conectarte desde el emulador de terminal TN327 0, pulsa la tecla «Enter» para borrar la pantalla del TPE e introducir la transacción inicial.
2. En la transacción inicial CC00 (pantalla de inicio de sesión), introduzca el nombre de usuario `USER001` y la contraseña `PASSWORD`.



3. Elija la opción 01 en el Menú principal para ver las cuentas.



4. En la pantalla Ver cuenta, introduce un número de cuenta (por ejemplo, 00000000010). Debería ver la información de la cuenta rellenada a partir de los datos migrados.



5. Pulse la tecla «PF03» dos veces para volver a la pantalla de inicio de sesión y salga del terminal TN327 0 pulsando «Ctrl+C» (Windows) o «Cmd+C» (MacBook).

Tareas por lotes

1. Vaya al directorio JCL.

```
cd $MBMSUB
```

2. Envíe el trabajo `MFCATGL1` y observe la salida del registro de trabajos.

```
BPESUB READCARD
```

3. Opcionalmente, podría ver los registros de trabajos desde el directorio `$SUBSYS_OUTDIR`.

```
ls -lrt $SUBSYS_OUTDIR/*
```

Ya ha implementado correctamente la CardDemo aplicación en NTT DATA UniKix Runtime y ha verificado la aplicación en ejecución navegando por algunas pantallas en línea de CICS y realizando trabajos por lotes.

Solución de problemas

Los siguientes son algunos errores comunes que puede encontrar al configurar la aplicación. CardDemo

Error: Licensing error

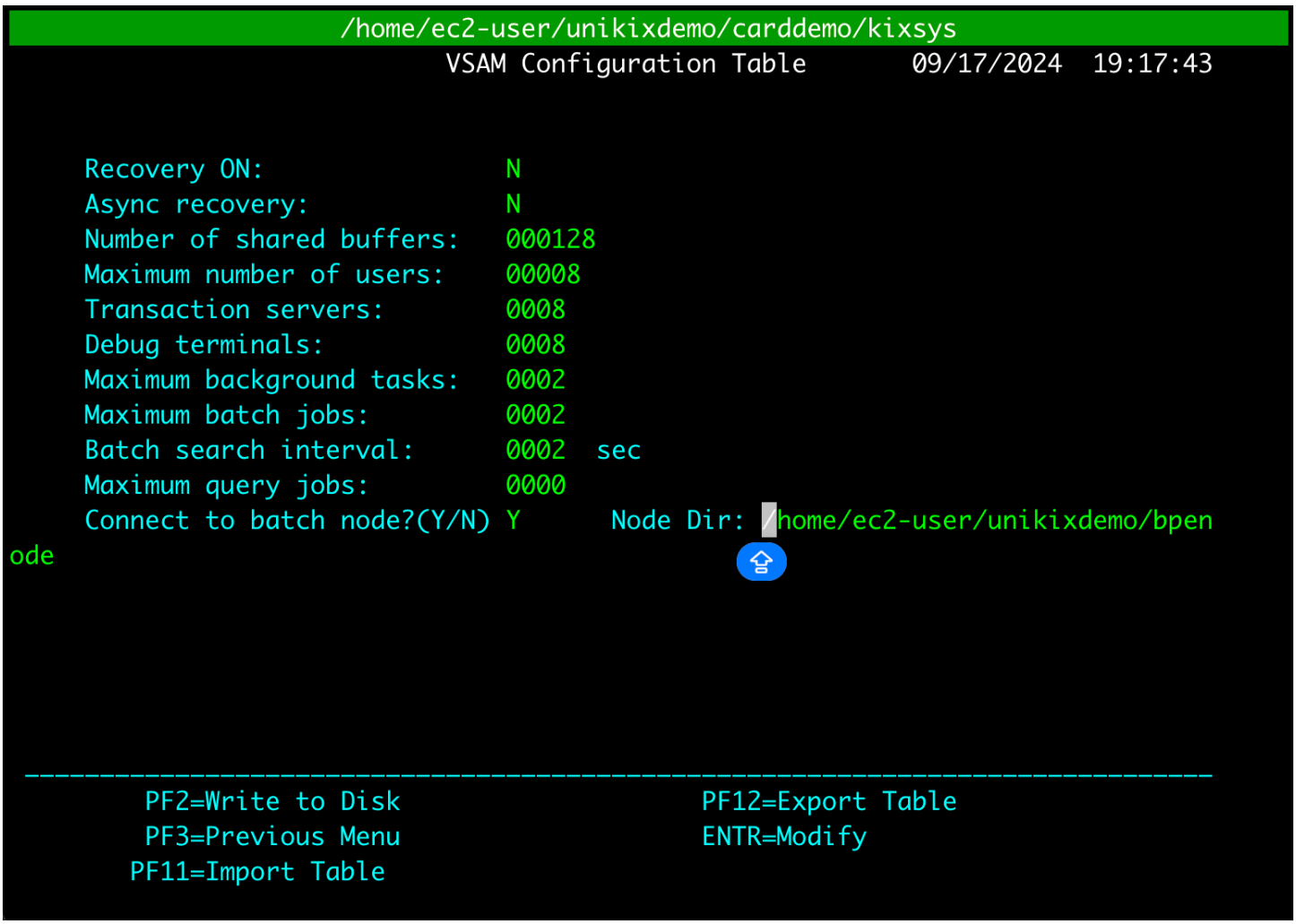
Si recibe un error de licencia mientras sigue este tutorial, es posible que la IMDSv2esté habilitada en su EC2. Puedes resolver este problema modificando la opción de metadatos de la instancia IMDSv2a Opcional, tal y como se menciona en Restaurar el uso de IMDSv1 en la guía del EC2 usuario de Amazon.

Error: TPE is not connected to BPE

Si TPE no está conectado a BPE, asegúrese de que la Tabla de configuración de VSAM esté configurada correctamente con el directorio de nodos de BPE. Para acceder a la tabla de configuración de VSAM, inicie el emulador de terminal TN327 0 con el siguiente comando:

```
kixterm
```

1. Introduzca el nombre de la transacción como `CTBL`.
2. En el menú Administrador de tablas, elija la opción Tablas estándar.
3. En la pantalla Tablas estándar, elija la opción Tabla de configuración de VSAM.
4. Compruebe si la opción ¿Conectarse al nodo por lotes? está establecida en S y el directorio de nodos es correcto.


/home/ec2-user/unikixdemo/carddemo/kixsys
VSAM Configuration Table 09/17/2024 19:17:43
Recovery ON: N
Async recovery: N
Number of shared buffers: 000128
Maximum number of users: 00008
Transaction servers: 0008
Debug terminals: 0008
Maximum background tasks: 0002
Maximum batch jobs: 0002
Batch search interval: 0002 sec
Maximum query jobs: 0000
Connect to batch node?(Y/N) Y Node Dir: /home/ec2-user/unikixdemo/bpen
ode
PF2=Write to Disk
PF3=Previous Menu
PF11=Import Table
PF12=Export Table
ENTR=Modify

Seguridad en la modernización del AWS mainframe

La seguridad en la nube AWS es la máxima prioridad. Como AWS cliente, usted se beneficia de una arquitectura de centro de datos y red diseñada para cumplir con los requisitos de las organizaciones más sensibles a la seguridad.

La seguridad es una responsabilidad compartida entre usted AWS y usted. El modelo de responsabilidad compartida la describe como seguridad de la nube y seguridad en la nube:

- Seguridad de la nube: AWS es responsable de proteger la infraestructura que ejecuta AWS los servicios en la AWS nube. AWS también le proporciona servicios que puede utilizar de forma segura. Los auditores externos prueban y verifican periódicamente la eficacia de nuestra seguridad como parte de los AWS programas de de . Para obtener más información sobre los programas de conformidad que se aplican a la modernización del AWS mainframe, consulte Servicios de AWS en el ámbito de aplicación por programa de conformidad .
- Seguridad en la nube: su responsabilidad viene determinada por el AWS servicio que utilice. También es responsable de otros factores, incluida la confidencialidad de los datos, los requisitos de la empresa y la legislación y los reglamentos aplicables

Esta documentación le ayuda a comprender cómo aplicar el modelo de responsabilidad compartida al utilizar la modernización del AWS mainframe. Le muestra cómo configurar la modernización del AWS mainframe para cumplir sus objetivos de seguridad y conformidad. También aprenderá a utilizar otros AWS servicios que le ayudan a supervisar y proteger sus recursos de modernización del AWS mainframe.

AWS La modernización del mainframe proporciona sus propios recursos protegidos por la IAM (aplicación, entorno, despliegue, etc.), que son los recursos administrativos de modernización del AWS mainframe, sobre los que las políticas de IAM deben permitir cualquier acción.

AWS La modernización del mainframe para la reconfiguración de plataformas también está asegurada por la IAM. IAM concede o deniega el permiso a una entidad principal para realizar una acción específica en un recurso definido, derivado del entorno de la unidad central original, también mediante políticas de IAM estándar. El tiempo de ejecución de la replataforma de AWS Mainframe Modernization llama al servicio de autorización de IAM cuando una aplicación intenta realizar dicha acción en un recurso protegido. IAM devolverá la opcion de permitir o denegar en función de los mecanismos estándar de evaluación de las políticas de IAM.

Contenido

- La protección de datos en la modernización del mainframe AWS
- Identity and Access Management para la modernización del AWS mainframe
- Validación del cumplimiento para la modernización del mainframe AWS
- Resiliencia en la AWS modernización del mainframe
- Seguridad de la infraestructura en AWS Mainframe Modernization
- Acceda AWS Mainframe Modernization mediante un punto final AWS PrivateLink de interfaz

La protección de datos en la modernización del mainframe AWS

El modelo de se aplica a protección de datos en la modernización del AWS mainframe. Como se describe en este modelo, AWS es responsable de proteger la infraestructura global en la que se ejecutan todos los Nube de AWS. Eres responsable de mantener el control sobre el contenido alojado en esta infraestructura. También eres responsable de las tareas de administración y configuración de seguridad para los Servicios de AWS que utiliza. Para obtener más información sobre la privacidad de los datos, consulta las Preguntas frecuentes sobre la privacidad de datos. Para obtener información sobre la protección de datos en Europa, consulta la publicación de blog sobre el Modelo de responsabilidad compartida de AWS y GDPR en el Blog de seguridad de AWS .

Con fines de protección de datos, le recomendamos que proteja Cuenta de AWS las credenciales y configure los usuarios individuales con AWS IAM Identity Center o AWS Identity and Access Management (IAM). De esta manera, solo se otorgan a cada usuario los permisos necesarios para cumplir sus obligaciones laborales. También recomendamos proteger sus datos de la siguiente manera:

- Utiliza la autenticación multifactor (MFA) en cada cuenta.
- Se utiliza SSL/TLS para comunicarse con AWS los recursos. Se recomienda el uso de TLS 1.2 y recomendamos TLS 1.3.
- Configure la API y el registro de actividad de los usuarios con AWS CloudTrail. Para obtener información sobre el uso de CloudTrail senderos para capturar AWS actividades, consulte Cómo trabajar con CloudTrail senderos en la Guía del AWS CloudTrail usuario.
- Utilice soluciones de AWS cifrado, junto con todos los controles de seguridad predeterminados Servicios de AWS.
- Utiliza servicios de seguridad administrados avanzados, como Amazon Macie, que lo ayuden a detectar y proteger los datos confidenciales almacenados en Amazon S3.

- Si necesita módulos criptográficos validados por FIPS 140-3 para acceder a AWS través de una interfaz de línea de comandos o una API, utilice un punto final FIPS. Para obtener más información sobre los puntos de conexión de FIPS disponibles, consulta Estándar de procesamiento de la información federal (FIPS) 140-3.

Se recomienda encarecidamente no introducir nunca información confidencial o sensible, como por ejemplo, direcciones de correo electrónico de clientes, en etiquetas o campos de formato libre, tales como el campo Nombre. Esto incluye cuando trabaja con AWS Mainframe Modernization u otro tipo de Servicios de AWS uso de la consola, la API o. AWS CLI AWS SDKs Cualquier dato que ingrese en etiquetas o campos de texto de formato libre utilizados para nombres se puede emplear para los registros de facturación o diagnóstico. Si proporciona una URL a un servidor externo, recomendamos encarecidamente que no incluya información de credenciales en la URL a fin de validar la solicitud para ese servidor.

Datos que recopila AWS Mainframe Modernization

AWS La modernización del mainframe recopila varios tipos de datos sobre usted:

- `Application configuration`: se trata de un archivo JSON que se crea para configurar la aplicación. Contiene sus opciones para las diferentes opciones que ofrece la modernización de AWS mainframe. El archivo también contiene información sobre los AWS recursos dependientes, como las rutas de Amazon Simple Storage Service donde se almacenan los artefactos de la aplicación o el nombre de recurso de Amazon (ARN) AWS Secrets Manager donde se almacenan las credenciales de la base de datos.
- `Application executable (binary)`: Se trata de un binario que usted compila y que pretende implementar en la modernización del AWS mainframe.
- `Application JCL or scripts`: este código fuente gestiona los trabajos por lotes u otro tipo de procesamiento en nombre de la aplicación.
- `User application data`: Al importar conjuntos de datos, AWS Mainframe Modernization los almacena en la base de datos relacional para que la aplicación pueda acceder a ellos.
- `Application source code`: A través de Amazon AppStream 2.0, AWS Mainframe Modernization proporciona un entorno de desarrollo para escribir y compilar código.

AWS La modernización del mainframe almacena estos datos de forma nativa en. AWS Los datos que recopilamos sobre usted se almacenan en un bucket de Amazon S3 gestionado por AWS Mainframe

Modernization. Al implementar una aplicación, AWS Mainframe Modernization descarga los datos en una instancia de Amazon Elastic Compute Cloud respaldada por Amazon Elastic Block Store. Cuando se activa la limpieza, los datos se eliminan del volumen de Amazon EBS y de Amazon S3. Los volúmenes de Amazon EBS son de un solo inquilino, lo que significa que se utiliza una instancia para un cliente. Las instancias nunca se comparten. Al eliminar un entorno de tiempo de ejecución, también se elimina el volumen de Amazon EBS. Al eliminar una aplicación, los artefactos y la configuración se eliminan de Amazon S3.

Los registros de las aplicaciones se almacenan en Amazon CloudWatch. Los mensajes de registro de las aplicaciones del cliente también CloudWatch se exportan a. Los CloudWatch registros pueden contener datos confidenciales del cliente, como datos empresariales o información de seguridad (en los mensajes de depuración). Para obtener más información, consulte Supervisión de la modernización AWS del mainframe con Amazon CloudWatch.

Además, si decide adjuntar uno o más sistemas de archivos Amazon Elastic File System o Amazon FSx a su entorno de ejecución, los datos de esos sistemas se almacenarán en AWS. Tendrá que limpiar esos datos si decide dejar de usar los sistemas de archivos.

Puede utilizar todas las opciones de cifrado de Amazon S3 disponibles para proteger sus datos al colocarlos en el depósito de Amazon S3 que AWS Mainframe Modernization utiliza para la implementación de aplicaciones y la importación de conjuntos de datos. Además, puede utilizar las opciones de FSx cifrado Amazon EFS y Amazon si adjunta uno o más de estos sistemas de archivos a su entorno de ejecución.

Cifrado de datos en reposo para el servicio de modernización AWS de mainframe

AWS La modernización del mainframe se integra AWS Key Management Service para proporcionar un cifrado transparente del lado del servidor (SSE) en todos los recursos dependientes que almacenan datos de forma permanente, a saber, Amazon Simple Storage Service, Amazon DynamoDB y Amazon Elastic Block Store. AWS Mainframe Modernization crea y administra claves de cifrado AWS KMS simétricas para usted. AWS KMS

El cifrado de los datos en reposo de forma predeterminada ayuda a reducir la sobrecarga operativa y la complejidad que implica la protección de los datos confidenciales. Al mismo tiempo, le permite migrar aplicaciones seguras que requieren estrictos requisitos normativos y de conformidad del cifrado.

No puede deshabilitar esta capa de cifrado ni seleccionar un tipo de cifrado alternativo al crear aplicaciones y entornos de tiempo de ejecución.

Puede utilizar su propia clave gestionada por el cliente para las aplicaciones de modernización de AWS mainframe y los entornos de ejecución para cifrar los recursos de Amazon S3 y Amazon EBS.

Para sus aplicaciones de modernización de AWS mainframe, puede utilizar esta clave para cifrar la definición de la aplicación, así como otros recursos de la aplicación, como los archivos JCL, que se guardan en el depósito de Amazon S3 que se crea en la cuenta del servicio. Para obtener más información, consulte Creación de una aplicación de .

Para sus entornos de ejecución de AWS Mainframe Modernization, AWS Mainframe Modernization utiliza la clave gestionada por el cliente para cifrar el volumen de Amazon EBS que crea y adjunta a su EC2 instancia de Amazon de Mainframe AWS Modernization, que también se encuentra en la cuenta del servicio. Para obtener más información, consulte Creación de un entorno de tiempo de ejecución.

> **Note**
>
> Los recursos de DynamoDB siempre se cifran mediante Clave administrada de AWS una cuenta de servicio de modernización AWS de mainframe. No se pueden cifrar los recursos de DynamoDB con una clave administrada por el cliente.

AWS Mainframe Modernization utiliza la clave gestionada por el cliente para las siguientes tareas:

- Reimplementación de una aplicación.
- Sustitución de una EC2 instancia de Amazon de modernización de AWS mainframe.

AWS Mainframe Modernization no utiliza la clave gestionada por el cliente para cifrar las bases de datos de Amazon Relational Database Service o Amazon Aurora, las colas de Amazon Simple Queue Service y las ElastiCache cachés de Amazon que se crean para dar soporte a una aplicación de modernización de AWS mainframe, ya que ninguna de ellas contiene datos de clientes.

Para más información, consulte las claves administradas por el cliente en la Guía para desarrolladores de AWS Key Management Service .

En la siguiente tabla se resume cómo Mainframe Modernization cifra los datos confidenciales. AWS

Tipo de datos:	Clave administrada de AWS cifrado	Clave de cifrado gestionada por el cliente
`Definition` Contiene la definición de una aplicación en particular.	Habilitado	Habilitado
`EnvironmentSummary` Contiene información sobre el entorno de tiempo de ejecución.	Habilitado	Habilitado
`ApplicationSummary` Contiene información sobre la aplicación de modernización AWS del mainframe.	Habilitado	Habilitado
`DeploymentSummary` Contiene información sobre la implementación de una aplicación de modernización de AWS mainframe.	Habilitado	Habilitado

> **Note**
>
> AWS La modernización del mainframe permite automáticamente el cifrado en reposo Claves administradas por AWS para proteger sus datos confidenciales sin coste alguno. Sin embargo, se aplican AWS KMS cargos por el uso de una clave administrada por el cliente. Para obtener más información sobre los precios, consulte Precios de AWS Key Management Service.

Para obtener más información sobre AWS KMS, consulte AWS Key Management Service.

Cómo utiliza AWS Mainframe Modernization las subvenciones en AWS KMS

AWS La modernización del mainframe requiere una subvención para utilizar la clave gestionada por el cliente.

Cuando crea una aplicación o un entorno de ejecución, o despliega una aplicación en AWS Mainframe Modernization cifrada con una clave gestionada por el cliente, AWS Mainframe Modernization crea una subvención en su nombre enviando una CreateGrantsolicitud a. AWS KMS Las subvenciones AWS KMS se utilizan para permitir que AWS Mainframe Modernization acceda a una clave KMS de la cuenta de un cliente.

AWS La modernización del mainframe requiere la autorización para utilizar la clave gestionada por el cliente en las siguientes operaciones internas:

- Envíe DescribeKeysolicitudes AWS KMS para comprobar que el identificador de clave simétrico gestionado por el cliente introducido al crear una aplicación, un entorno de ejecución o la implementación de una aplicación es válido.
- Envíe GenerateDataKeysolicitudes AWS KMS para cifrar el volumen de Amazon EBS adjunto a las EC2 instancias de Amazon que alojan los entornos de ejecución de AWS Mainframe Modernization.
- Envíe solicitudes de descifrado a AWS KMS para descifrar contenido cifrado en Amazon EBS.

AWS Mainframe AWS KMS Modernization utiliza las subvenciones para descifrar los secretos almacenados en Secrets Manager y al crear un entorno de ejecución, crear o volver a implementar una aplicación y crear una implementación. Las subvenciones que crea AWS Mainframe Modernization respaldan las siguientes operaciones:

- Crear o actualizar una concesión de entorno de tiempo de ejecución:
  - Decrypt
  - Encrypt
  - ReEncryptFrom
  - ReEncryptTo
  - GenerateDataKey
  - DescribeKey
  - CreateGrant

- Crear o reimplementar una concesión para una aplicación:
  - GenerateDataKey
- Crear una concesión para una implementación:
  - Decrypt

Puede revocar el acceso a la concesión o eliminar el acceso del servicio a la clave administrada por el cliente en cualquier momento. Si lo hace, AWS Mainframe Modernization no podrá acceder a ninguno de los datos cifrados por la clave gestionada por el cliente, lo que afectará a las operaciones que dependen de los datos. Por ejemplo, si AWS Mainframe Modernization intentara acceder a una definición de aplicación cifrada por una clave gestionada por el cliente sin conceder esa clave, la operación de creación de la aplicación fallaría.

AWS La modernización del mainframe recopila las configuraciones de las aplicaciones de los usuarios (archivos JSON) y los artefactos (archivos binarios y ejecutables). También crea metadatos que rastrean varias entidades utilizadas para el funcionamiento de AWS Mainframe Modernization y crea registros y métricas. Los registros y las métricas que los clientes pueden ver incluyen:

- CloudWatch registros que reflejan la aplicación y el motor de ejecución (ya sea AWS Blu Age o Rocket Software (anteriormente Micro Focus)).
- CloudWatch métricas para los cuadros de mando de operaciones.

Además, AWS Mainframe Modernization recopila datos y métricas de uso para la medición, la elaboración de informes de actividad, etc., sobre los servicios. Estos datos no son visibles para los clientes.

AWS La modernización del mainframe almacena estos datos en diferentes lugares según el tipo de datos. Los datos del cliente que cargue se almacenan en un bucket de Amazon S3. Los datos del servicio se almacenan en Amazon S3 y DynamoDB. Al implementar una aplicación, tanto los datos como los datos de servicio se descargan en los volúmenes de Amazon EBS. Si decide adjuntar Amazon EFS o Amazon FSx Storage a su entorno de ejecución, los datos almacenados en esos sistemas de archivos también se descargan en el volumen de Amazon EBS.

El cifrado en reposo está configurado de forma predeterminada. No puede deshabilitarlo ni cambiarlo. Actualmente tampoco se puede cambiar su configuración.

Creación de una clave administrada por el cliente

Puede crear una clave simétrica gestionada por el cliente mediante el AWS Management Console o el. AWS KMS APIs

Para crear una clave simétrica administrada por el cliente

Siga los pasos para crear una clave simétrica gestionada por el cliente que se indican en la Guía para desarrolladores de AWS Key Management Service .

Política de claves

Las políticas de clave controlan el acceso a la clave administrada por el cliente. Cada clave administrada por el cliente debe tener exactamente una política de clave, que contiene instrucciones que determinan quién puede usar la clave y cómo puede utilizarla. Cuando crea la clave administrada por el cliente, puede especificar una política de clave. Para obtener más información, consulte Administración del acceso a las claves en la Guía para desarrolladores de AWS Key Management Service .

Para utilizar la clave gestionada por el cliente con los recursos de modernización del AWS mainframe, la política de claves debe permitir las siguientes operaciones de API:

- `kms:CreateGrant`: añade una concesión a una clave administrada por el cliente. Otorga el acceso de control a una clave de KMS específica, que permite acceder a las operaciones que requiere la modernización AWS del mainframe. Para obtener más información sobre el uso de concesiones, consulte la Guía para desarrolladores de AWS Key Management Service .

  Esto permite que la modernización AWS del mainframe haga lo siguiente:

  - Llamar a `GenerateDataKey` para generar una clave de datos cifrada y almacenarla, ya que la clave de datos no se utiliza inmediatamente para cifrar.
  - Llamar a `Decrypt` para usar la clave de datos cifrados almacenada para acceder a los datos cifrados.
  - Configurar una entidad principal que se retire para permitir que el servicio `RetireGrant`.
- `kms:DescribeKey`— Proporciona los detalles clave gestionados por el cliente para permitir que AWS Mainframe Modernization valide la clave.

AWS La modernización del mainframe requiere `kms:CreateGrant` y `kms:DescribeKey` autoriza en la política clave del cliente. AWS Mainframe Modernization utiliza esta política para crear una subvención para sí misma.

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "Enable IAM User Permissions",
        "Effect": "Allow",
        "Principal": {
            "AWS": "arn:aws:iam::AccountId:role/ExampleRole"
        },
        "Action": [
            "kms:CreateGrant",
            "kms:DescribeKey"
        ],
        "Resource": "*"
    }]
}
```

> **Note**
>
> La función que se muestra `Principal` en el ejemplo anterior es la que se utiliza para las operaciones de modernización AWS del mainframe, como `CreateApplication` y. `CreateEnvironment`

Para obtener más información sobre cómo especificar permisos en una política, consulte la Guía para desarrolladores de AWS Key Management Service .

Para obtener información sobre la solución de problemas de acceso a las claves, consulte la Guía para desarrolladores de AWS Key Management Service .

Especificar una clave administrada por el cliente para AWS Mainframe Modernization

Puede especificar una clave administrada por el cliente para los siguientes recursos:

- Aplicación
- Entorno

Al crear un recurso, puede especificar la clave introduciendo un ID de KMS, que AWS Mainframe Modernization utiliza para cifrar los datos confidenciales almacenados en el recurso.

- KMS ID: un identificador de clave para una clave administrada por el cliente. Introduzca el ID de la clave, el ARN de la clave, el nombre de alias o el ARN del alias.

Puede especificar una clave administrada por el cliente mediante el AWS Management Console o el. AWS CLI

Para especificar la clave gestionada por el cliente al crear un entorno de ejecución en el AWS Management Console, consulteCree un entorno de ejecución de modernización AWS de mainframe. Para especificar la clave gestionada por el cliente al crear una aplicación en el AWS Management Console, consulteCree una AWS Mainframe Modernization aplicación.

Para añadir la clave gestionada por el cliente al crear un entorno de ejecución con la AWS CLI, especifique el `kms-key-id` parámetro de la siguiente manera:

```
aws m2 create-environment –engine-type microfocus –instance-type M2.m5.large
--publicly-accessible –engine-version 7.0.3 –name test
--high-availability-config desiredCapacity=2
--kms-key-id myEnvironmentKey
```

Para añadir la clave gestionada por el cliente al crear una aplicación con AWS CLI, especifique el `kms-key-id` parámetro de la siguiente manera:

```
aws m2 create-application –name test-application –description my description
--engine-type microfocus
--definition content="$(jq -c . raw-template.json | jq -R)"
--kms-key-id myApplicationKey
```

AWS Contexto de cifrado de modernización del mainframe

Un contexto de cifrado es un conjunto opcional de pares clave-valor que pueden contener información contextual adicional sobre los datos.

AWS KMS utiliza el contexto de cifrado como datos autenticados adicionales para respaldar el cifrado autenticado. Al incluir un contexto de cifrado en una solicitud de cifrado de datos, AWS KMS vincula el contexto de cifrado a los datos cifrados. Para descifrar los datos, debe incluir el mismo contexto de cifrado en la solicitud.

AWS Modernización del mainframe: contexto de cifrado

AWS La modernización del mainframe utiliza el mismo contexto de cifrado en todas las operaciones AWS KMS criptográficas relacionadas con una aplicación (crear la aplicación y crear el despliegue), donde está la clave `aws:m2:app` y el valor es el identificador único de la aplicación.

Example

```
"encryptionContextSubset": {
        "aws:m2:app": "a1bc2defabc3defabc4defabcd"
}
```

Uso del contexto de cifrado para la supervisión

Si utiliza una clave simétrica administrada por el cliente para cifrar sus aplicaciones o entornos de tiempo de ejecución, también puede utilizar el contexto de cifrado en los registros y registros de auditoría para identificar cómo se está utilizando la clave administrada por el cliente.

Utilizar el contexto de cifrado para controlar el acceso a la clave administrada por el cliente

Puede utilizar el contexto de cifrado en las políticas de claves y las políticas de IAM como `conditions` para controlar el acceso a la clave simétrica administrada por el cliente. Puede usar también una restricción de contexto de cifrado en una concesión.

AWS La modernización del mainframe utiliza una restricción del contexto de cifrado en las subvenciones para controlar el acceso a la clave gestionada por el cliente en su cuenta o región. La restricción de concesión requiere que las operaciones que permite la concesión utilicen el contexto de cifrado especificado. El siguiente ejemplo es una subvención que AWS Mainframe Modernization aprovecha para cifrar los artefactos de una aplicación al crear una aplicación.

```
//This grant is retired immediately after create application finish
{
   "grantee-principal": m2.us-west-2.amazonaws.com,
   "retiring-principal": m2.us-west-2.amazonaws.com,
   "operations": [
       "GenerateDataKey"
   ]
   "condition": {
        "encryptionContextSubset": {
            “aws:m2:app”: “a1bc2defabc3defabc4defabcd”
   }
```

```
}
```

Supervisión de las claves de cifrado de AWS Mainframe Modernization

Cuando utiliza una clave gestionada por el AWS KMS cliente con sus recursos de modernización de AWS mainframe, puede utilizar AWS CloudTrailAmazon CloudWatch Logs para realizar un seguimiento de las solicitudes que envía AWS Mainframe Modernization. AWS KMS

Ejemplos de entornos de tiempo de ejecución

Los siguientes ejemplos son AWS CloudTrail eventos para `DescribeKey` `CreateGrantGenerateDataKey`, y para monitorear las operaciones de KMS `Decrypt` solicitadas por AWS Mainframe Modernization para acceder a los datos cifrados por su clave administrada por el cliente:

DescribeKey

AWS Mainframe Modernization utiliza la `DescribeKey` operación para comprobar si la clave gestionada por el AWS KMS cliente asociada a su entorno de ejecución existe en la cuenta y la región.

El siguiente evento de ejemplo registra la operación `DescribeKey`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
```

```
                "creationDate": "2022-12-06T19:40:26Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:43Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "DescribeKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "205.251.233.182",
    "userAgent": "ExampleDesktop/1.0 (V1; OS)",
    "requestParameters": {
        "keyId": "00dd0db0-0000-0000-ac00-b0c000SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```


CreateGrant

Cuando utiliza una clave administrada por el AWS KMS cliente para cifrar su entorno de ejecución, AWS Mainframe Modernization envía varias `CreateGrant` solicitudes en su nombre para realizar las operaciones de KMS necesarias. Algunas de las subvenciones que crea AWS

Mainframe Modernization se retiran inmediatamente después de su uso. Otras se retiran al eliminar el entorno de tiempo de ejecución.

El siguiente evento de ejemplo registra la operación `CreateGrant` del rol de ejecución de Lambda asociado al flujo de trabajo de Create Environment.


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:11:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
```

```
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

El siguiente evento de ejemplo registra la operación `CreateGrant` del rol vinculado al servicio del grupo Auto Scaling. El rol de ejecución de Lambda asociado al flujo de trabajo de Create Environment llama a esta operación `CreateGrant`. Concede permiso para que el rol de ejecución cree una subconcesión para el rol vinculado a servicios del grupo Auto Scaling.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65MZFUPM4JO:EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
```

```
        "arn": "arn:aws:sts::111122223333:assumed-role/EnvironmentWorkflow-
alpha-CreateEnvironmentLambdaS-1AU4A8VNQEEKN/EnvironmentWorkflow-alpha-
CreateEnvironmentLambda7-HfxDj5zz86tr",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN",
                "accountId": "111122223333",
                "userName": "EnvironmentWorkflow-alpha-
CreateEnvironmentLambdaS-1AU4A8VNQEEKN"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:22:28Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-12-06T20:23:09Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "54.148.236.160",
    "userAgent": "aws-sdk-java/2.18.21 Linux/4.14.255-276-224.499.amzn2.x86_64
 OpenJDK_64-Bit_Server_VM/11.0.14.1+10-LTS Java/11.0.14.1 vendor/Amazon.com_Inc. md/
internal exec-env/AWS_Lambda_java11 io/sync http/Apache cfg/retry-mode/legacy",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "operations": [
            "Encrypt",
            "Decrypt",
            "ReEncryptFrom",
            "ReEncryptTo",
            "GenerateDataKey",
            "GenerateDataKey",
            "DescribeKey",
            "CreateGrant"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
```

```
        "retiringPrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_256_GCM_SHA384",
        "clientProvidedHostHeader": "kms.us-west-2.amazonaws.com"
    }
}
}
```

GenerateDataKey

Cuando habilita una clave administrada por el AWS KMS cliente para su recurso de entorno de ejecución, Auto Scaling crea una clave única para cifrar el volumen de Amazon EBS asociado al entorno de ejecución. Envía una `GenerateDataKey` solicitud a la AWS KMS que se especifica la clave administrada por el AWS KMS cliente para el recurso.

El siguiente evento de ejemplo registra la operación `GenerateDataKey`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
```

```
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65EEXVIEH7D:AutoScaling",
        "arn": "arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForAutoScaling/
AutoScaling",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/aws-service-role/
autoscaling.amazonaws.com/AWSServiceRoleForAutoScaling",
                "accountId": "111122223333",
                "userName": "AWSServiceRoleForAutoScaling"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T20:23:16Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "autoscaling.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:18Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "autoscaling.amazonaws.com",
    "userAgent": "autoscaling.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        },
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "numberOfBytes": 64
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
```

```
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Cuando accede a un entorno de tiempo de ejecución cifrado, Amazon EBS llama a la operación `Decrypt` que utiliza la clave de datos cifrados almacenada para acceder a los datos cifrados.

El siguiente evento de ejemplo registra la operación `Decrypt`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ebs.amazonaws.com"
    },
    "eventTime": "2022-12-06T20:23:22Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ebs.amazonaws.com",
    "userAgent": "ebs.amazonaws.com",
    "requestParameters": {
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT",
        "encryptionContext": {
            "aws:ebs:id": "vol-080f7a32d290807f3"
        }
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "sharedEventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventCategory": "Management"
}
```

Ejemplos de aplicaciones

Los siguientes ejemplos son AWS CloudTrail eventos que AWS Mainframe `GenerateDataKey` Modernization solicita `CreateGrant` y supervisan las operaciones de KMS para acceder a los datos cifrados por la clave gestionada por el cliente:

CreateGrant

Cuando utiliza una clave gestionada por el AWS KMS cliente para cifrar los recursos de la aplicación, la función de ejecución de Lambda envía `CreateGrant` una solicitud en su nombre para acceder a la clave KMS de su cuenta. AWS La concesión permite l rol de ejecución de Lambda cargar los recursos de las aplicaciones del cliente en Amazon S3 mediante la clave administrada por el cliente. Esta concesión se retira inmediatamente después de crear la aplicación.

El siguiente evento de ejemplo registra la operación `CreateGrant`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
```

```
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T22:47:04Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE",
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "operations": [
            "GenerateDataKey"
        ],
        "granteePrincipal": "m2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
```

```
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

GenerateDataKey

Cuando habilita una clave gestionada por el AWS KMS cliente para el recurso de su aplicación, la función de ejecución de Lambda crea una clave que utiliza para cifrar y cargar los datos de los clientes en Amazon Simple Storage Service. La función de ejecución de Lambda envía una `GenerateDataKey` solicitud a la AWS KMS que se especifica la clave administrada por el AWS KMS cliente para el recurso.

El siguiente evento de ejemplo registra la operación `GenerateDataKey`:

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65CLCEKKC7Z:ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "arn": "arn:aws:sts::111122223333:assumed-role/ApplicationWorkflow-
alpha-CreateApplicationVersion-1IZRBZYDG20B/ApplicationWorkflow-alpha-
CreateApplicationVersion-CstWZUn5R4u6",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B",
                "accountId": "111122223333",
                "userName": "ApplicationWorkflow-alpha-
CreateApplicationVersion-1IZRBZYDG20B"
            },
            "webIdFederationData": {},
            "attributes": {
```

```
                "creationDate": "2022-12-06T23:28:32Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:29:08Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKey",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcd",
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcd/1/cics-transaction/ZBNKE35.so"
        },
        "keySpec": "AES_256",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Ejemplos de implementaciones

Los siguientes ejemplos son AWS CloudTrail eventos que AWS Mainframe `Decrypt` Modernization solicita `CreateGrant` y supervisan las operaciones de KMS para acceder a los datos cifrados por la clave gestionada por el cliente:

CreateGrant

Cuando utiliza una clave gestionada por el AWS KMS cliente para cifrar los recursos de implementación, AWS Mainframe Modernization envía dos `CreateGrant` solicitudes en su nombre. La primera concesión se destina a la función de ejecución de Lambda a la que se deba llamar ListBatchJobScriptFiles y se retira inmediatamente después de que finalice la implementación. La segunda subvención se destina a la función de EC2 instancia restringida de Amazon para que Amazon EC2 pueda descargar los recursos de las aplicaciones de los clientes de Amazon S3. Esta concesión se retira cuando la aplicación se elimina del entorno de tiempo de ejecución.

El siguiente evento de ejemplo registra la operación `CreateGrant`:


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
        "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:sts::111122223333:assumed-role/Admin/Sampleuser01",
                "accountId": "111122223333",
                "userName": "Admin"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T21:51:45Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
```

```
    },
    "eventTime": "2022-12-06T23:40:07Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:m2:app": "a1bc2defabc3defabc4defabcd"
            }
        },
        "granteePrincipal": "m2.us-west-2.amazonaws.com",
        "retiringPrincipal": "m2.us-west-2.amazonaws.com",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "responseElements": {
        "grantId":
 "0ab0ac0d0b000f00ea00cc0a0e00fc00bce000c000f0000000c0bc0a0000aaafSAMPLE",
        "keyId": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
    },
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": false,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```

Decrypt

Al acceder a una implementación, Amazon EC2 solicita a la `Decrypt` operación que utilice la clave de datos cifrados almacenada para descifrar y descargar los datos cifrados de los clientes de Amazon S3.

El siguiente evento de ejemplo registra la operación `Decrypt`:


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROA3YPCLM65BSPZ37E6G:m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "arn": "arn:aws:sts::111122223333:assumed-role/
SupernovaEnvironmentInstanceScopeDownRole/m2-hm-bqe367dxtfcpdbzmnhfzranisu",
        "accountId": "111122223333",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE3",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAIGDTESTANDEXAMPLE:Sampleuser01",
                "arn": "arn:aws:iam::111122223333:role/
SupernovaEnvironmentInstanceScopeDownRole",
                "accountId": "111122223333",
                "userName": "SupernovaEnvironmentInstanceScopeDownRole"
            },
            "webIdFederationData": {},
            "attributes": {
                "creationDate": "2022-12-06T23:19:29Z",
                "mfaAuthenticated": "false"
            }
        },
        "invokedBy": "m2.us-west-2.amazonaws.com"
    },
    "eventTime": "2022-12-06T23:40:15Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "Decrypt",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "m2.us-west-2.amazonaws.com",
    "userAgent": "m2.us-west-2.amazonaws.com",
    "requestParameters": {
        "encryptionContext": {
            "aws:m2:app": "a1bc2defabc3defabc4defabcdm",
```

```
            "aws:s3:arn": "arn:aws:s3:::supernova-processedtemplate-111122223333-us-
west-2/111122223333/a1bc2defabc3defabc4defabcdm/1/cics-transaction/BBANK40P.so"
        },
        "encryptionAlgorithm": "SYMMETRIC_DEFAULT"
    },
    "responseElements": null,
    "requestID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "eventID": "ff000af-00eb-00ce-0e00-ea000fb0fba0SAMPLE",
    "readOnly": true,
    "resources": [
        {
            "accountId": "111122223333",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-123456SAMPLE"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "111122223333",
    "eventCategory": "Management"
}
```


Más información

Los siguientes recursos proporcionan más información sobre cifrado de datos en reposo.

- Para obtener más información acerca de los conceptos básicos de AWS Key Management Service, consulte la Guía para desarrolladores de AWS Key Management Service .
- Para obtener más información sobre las prácticas recomendadas de seguridad de AWS Key Management Service, consulte la Guía para desarrolladores de AWS Key Management Service .

Cifrado en tránsito

En el caso de las aplicaciones interactivas que forman parte de cargas de trabajo transaccionales, los intercambios de datos entre el emulador de terminal y el punto final del servicio de modernización de AWS mainframes para el protocolo TN327 0 no se cifran durante el tránsito. Si la aplicación requiere el cifrado en tránsito, es posible que desee implementar algunos mecanismos de tunelización adicionales.

AWS La modernización del mainframe utiliza HTTPS para cifrar el servicio. APIs Todas las demás comunicaciones dentro de la modernización del AWS mainframe están protegidas por la VPC del servicio o el grupo de seguridad, así como por HTTPS. AWS La modernización del mainframe transfiere los artefactos, las configuraciones y los datos de las aplicaciones. Los artefactos de la aplicación se copian de un bucket de Amazon S3 del que es propietario, al igual que los datos de la aplicación. Puede proporcionar las configuraciones de las aplicaciones mediante un enlace a Amazon S3 o cargando un archivo localmente.

El cifrado básico en tránsito está configurado de forma predeterminada, pero no se aplica al protocolo TN327 0. AWS La modernización del mainframe utiliza HTTPS para los puntos finales de la API, que también están configurados de forma predeterminada.

Identity and Access Management para la modernización del AWS mainframe

AWS Identity and Access Management (IAM) es una herramienta Servicio de AWS que ayuda al administrador a controlar de forma segura el acceso a los recursos. AWS Los administradores de IAM controlan quién puede autenticarse (iniciar sesión) y quién puede autorizarse (tener permisos) para usar los recursos de modernización de AWS mainframe. La IAM es una herramienta Servicio de AWS que puede utilizar sin coste adicional.

Temas

- Público
- Autenticación con identidades
- Administración de acceso mediante políticas
- Cómo funciona la modernización AWS del mainframe con la IAM
- Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS
- Solución de problemas de AWS identidad y acceso a la modernización del mainframe
- Cómo utilizar roles vinculados a servicios de AWS Mainframe Modernization

Público

La forma de usar AWS Identity and Access Management (IAM) varía según el trabajo que se realice en la modernización del AWS mainframe.

Usuario del servicio: si utiliza el servicio de modernización del AWS mainframe para realizar su trabajo, el administrador le proporcionará las credenciales y los permisos que necesita. A medida que vaya utilizando más funciones de modernización de AWS mainframe para realizar su trabajo, es posible que necesite permisos adicionales. Entender cómo se administra el acceso puede ayudarle a solicitar los permisos correctos al administrador. Si no puede acceder a una característica de AWS Mainframe Modernization, consulte Solución de problemas de AWS identidad y acceso a la modernización del mainframe.

Administrador de servicios: si está a cargo de los recursos de modernización del AWS mainframe en su empresa, probablemente tenga pleno acceso a la modernización del AWS mainframe. Su tarea es determinar a qué funciones y recursos de modernización AWS del mainframe deben acceder los usuarios del servicio. Luego, debe enviar solicitudes a su gestionador de IAM para cambiar los permisos de los usuarios de su servicio. Revise la información de esta página para conocer los conceptos básicos de IAM. Para obtener más información sobre cómo su empresa puede utilizar la IAM con la modernización del AWS mainframe, consulte. Cómo funciona la modernización AWS del mainframe con la IAM

Administrador de IAM: si es administrador de IAM, puede que desee obtener información detallada sobre cómo redactar políticas para administrar el acceso a la modernización de mainframe. AWS Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad que puede utilizar en la IAM, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Autenticación con identidades

La autenticación es la forma de iniciar sesión con sus AWS credenciales de identidad. Debe estar autenticado (con quien haya iniciado sesión AWS) como usuario de IAM o asumiendo una función de IAM. Usuario raíz de la cuenta de AWS

Puede iniciar sesión AWS como una identidad federada mediante las credenciales proporcionadas a través de una fuente de identidad. AWS IAM Identity Center Los usuarios (Centro de identidades de IAM), la autenticación de inicio de sesión único de su empresa y sus credenciales de Google o Facebook son ejemplos de identidades federadas. Al iniciar sesión como una identidad federada, su gestionador habrá configurado previamente la federación de identidades mediante roles de IAM. Cuando accedes AWS mediante la federación, estás asumiendo un rol de forma indirecta.

Según el tipo de usuario que sea, puede iniciar sesión en el portal AWS Management Console o en el de AWS acceso. Para obtener más información sobre cómo iniciar sesión AWS, consulte Cómo iniciar sesión Cuenta de AWS en su Guía del AWS Sign-In usuario.

Si accede AWS mediante programación, AWS proporciona un kit de desarrollo de software (SDK) y una interfaz de línea de comandos (CLI) para firmar criptográficamente sus solicitudes con sus credenciales. Si no utilizas AWS herramientas, debes firmar las solicitudes tú mismo. Para obtener más información sobre la firma de solicitudes, consulte AWS Signature Versión 4 para solicitudes API en la Guía del usuario de IAM.

Independientemente del método de autenticación que use, es posible que deba proporcionar información de seguridad adicional. Por ejemplo, le AWS recomienda que utilice la autenticación multifactor (MFA) para aumentar la seguridad de su cuenta. Para obtener más información, consulte Autenticación multifactor en la Guía del usuario de AWS IAM Identity Center y Autenticación multifactor AWS en IAM en la Guía del usuario de IAM.

Cuenta de AWS usuario root

Al crear una Cuenta de AWS, comienza con una identidad de inicio de sesión que tiene acceso completo a todos Servicios de AWS los recursos de la cuenta. Esta identidad se denomina usuario Cuenta de AWS raíz y se accede a ella iniciando sesión con la dirección de correo electrónico y la contraseña que utilizaste para crear la cuenta. Recomendamos encarecidamente que no utiliza el usuario raíz para sus tareas diarias. Proteja las credenciales del usuario raíz y utilícelas solo para las tareas que solo el usuario raíz pueda realizar. Para ver la lista completa de las tareas que requieren que inicie sesión como usuario raíz, consulta Tareas que requieren credenciales de usuario raíz en la Guía del usuario de IAM.

Identidad federada

Como práctica recomendada, exija a los usuarios humanos, incluidos los que requieren acceso de administrador, que utilicen la federación con un proveedor de identidades para acceder Servicios de AWS mediante credenciales temporales.

Una identidad federada es un usuario del directorio de usuarios de su empresa, un proveedor de identidades web AWS Directory Service, el directorio del Centro de Identidad o cualquier usuario al que acceda Servicios de AWS mediante las credenciales proporcionadas a través de una fuente de identidad. Cuando las identidades federadas acceden Cuentas de AWS, asumen funciones y las funciones proporcionan credenciales temporales.

Para una administración de acceso centralizada, le recomendamos que utiliza AWS IAM Identity Center. Puede crear usuarios y grupos en el Centro de identidades de IAM, o puede conectarse y sincronizarse con un conjunto de usuarios y grupos de su propia fuente de identidad para usarlos en

todas sus Cuentas de AWS aplicaciones. Para obtener más información, consulta ¿Qué es el Centro de identidades de IAM? en la Guía del usuario de AWS IAM Identity Center .

Usuarios y grupos de IAM

Un usuario de IAM es una identidad propia Cuenta de AWS que tiene permisos específicos para una sola persona o aplicación. Siempre que sea posible, recomendamos emplear credenciales temporales, en lugar de crear usuarios de IAM que tengan credenciales de larga duración como contraseñas y claves de acceso. No obstante, si tiene casos de uso específicos que requieran credenciales de larga duración con usuarios de IAM, recomendamos rotar las claves de acceso. Para más información, consulte Rotar las claves de acceso periódicamente para casos de uso que requieran credenciales de larga duración en la Guía del usuario de IAM.

Un grupo de IAM es una identidad que especifica un conjunto de usuarios de IAM. No puedes iniciar sesión como grupo. Puedes usar los grupos para especificar permisos para varios usuarios a la vez. Los grupos facilitan la administración de los permisos para grandes conjuntos de usuarios. Por ejemplo, puede asignar un nombre a un grupo IAMAdminsy concederle permisos para administrar los recursos de IAM.

Los usuarios son diferentes de los roles. Un usuario se asocia exclusivamente a una persona o aplicación, pero la intención es que cualquier usuario pueda asumir un rol que necesite. Los usuarios tienen credenciales de larga duración permanentes; no obstante, los roles proporcionan credenciales temporales. Para obtener más información, consulte Casos de uso para usuarios de IAM en la Guía del usuario de IAM.

Roles de IAM

Un rol de IAM es una identidad dentro de usted Cuenta de AWS que tiene permisos específicos. Es similar a un usuario de IAM, pero no está asociado a una persona determinada. Para asumir temporalmente un rol de IAM en el AWS Management Console, puede cambiar de un rol de usuario a uno de IAM (consola). Puedes asumir un rol llamando a una operación de AWS API AWS CLI o usando una URL personalizada. Para más información sobre los métodos para el uso de roles, consulta Métodos para asumir un rol en la Guía del usuario de IAM.

Los roles de IAM con credenciales temporales son útiles en las siguientes situaciones:

- Acceso de usuario federado: para asignar permisos a una identidad federada, puede crear un rol y definir sus permisos. Cuando se autentica una identidad federada, se asocia la identidad al rol y se le conceden los permisos define el rol. Para obtener información acerca de roles de

federación, consulte Crear un rol para un proveedor de identidad de terceros (federación) en la Guía de usuario de IAM. Si utiliza el IAM Identity Center, debe configurar un conjunto de permisos. IAM Identity Center correlaciona el conjunto de permisos con un rol en IAM para controlar a qué pueden acceder las identidades después de autenticarse. Para obtener información acerca de los conjuntos de permisos, consulta Conjuntos de permisos en la Guía del usuario de AWS IAM Identity Center .

- Permisos de usuario de IAM temporales: un usuario de IAM puede asumir un rol de IAM para recibir temporalmente permisos distintos que le permitan realizar una tarea concreta.
- Acceso entre cuentas: puede utilizar un rol de IAM para permitir que alguien (una entidad principal de confianza) de otra cuenta acceda a los recursos de la cuenta. Los roles son la forma principal de conceder acceso entre cuentas. Sin embargo, con algunas Servicios de AWS, puedes adjuntar una política directamente a un recurso (en lugar de usar un rol como proxy). Para obtener información acerca de la diferencia entre los roles y las políticas basadas en recursos para el acceso entre cuentas, consulta Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.
- Acceso entre servicios: algunos Servicios de AWS utilizan funciones en otros Servicios de AWS. Por ejemplo, cuando realizas una llamada en un servicio, es habitual que ese servicio ejecute aplicaciones en Amazon EC2 o almacene objetos en Amazon S3. Es posible que un servicio haga esto usando los permisos de la entidad principal, usando un rol de servicio o usando un rol vinculado al servicio.
  - Sesiones de acceso directo (FAS): cuando utilizas un usuario o un rol de IAM para realizar acciones en AWS ellas, se te considera principal. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. El FAS utiliza los permisos del principal que llama Servicio de AWS y los solicita Servicio de AWS para realizar solicitudes a los servicios descendentes. Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.
  - Rol de servicio: un rol de servicio es un rol de IAM que adopta un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.
  - Función vinculada al servicio: una función vinculada a un servicio es un tipo de función de servicio que está vinculada a un. Servicio de AWS El servicio puede asumir el rol para realizar

una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

- Aplicaciones que se ejecutan en Amazon EC2: puedes usar un rol de IAM para administrar las credenciales temporales de las aplicaciones que se ejecutan en una EC2 instancia y realizan AWS CLI solicitudes a la AWS API. Esto es preferible a almacenar las claves de acceso en la EC2 instancia. Para asignar un AWS rol a una EC2 instancia y ponerlo a disposición de todas sus aplicaciones, debe crear un perfil de instancia adjunto a la instancia. Un perfil de instancia contiene el rol y permite que los programas que se ejecutan en la EC2 instancia obtengan credenciales temporales. Para obtener más información, consulte Usar un rol de IAM para conceder permisos a las aplicaciones que se ejecutan en EC2 instancias de Amazon en la Guía del usuario de IAM.

Administración de acceso mediante políticas

El acceso se controla AWS creando políticas y adjuntándolas a AWS identidades o recursos. Una política es un objeto AWS que, cuando se asocia a una identidad o un recurso, define sus permisos. AWS evalúa estas políticas cuando un director (usuario, usuario raíz o sesión de rol) realiza una solicitud. Los permisos en las políticas determinan si la solicitud se permite o se deniega. La mayoría de las políticas se almacenan AWS como documentos JSON. Para obtener más información sobre la estructura y el contenido de los documentos de política JSON, consulte Información general de políticas JSON en la Guía del usuario de IAM.

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

De forma predeterminada, los usuarios y los roles no tienen permisos. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los recursos que necesitan. A continuación, el administrador puede añadir las políticas de IAM a roles y los usuarios pueden asumirlos.

Las políticas de IAM definen permisos para una acción independientemente del método que se utiliza para realizar la operación. Por ejemplo, suponga que dispone de una política que permite la acción `iam:GetRole`. Un usuario con esa política puede obtener información sobre el rol de la API AWS Management Console AWS CLI, la o la AWS API.

Políticas basadas en identidades

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Las políticas basadas en identidades pueden clasificarse además como políticas insertadas o políticas administradas. Las políticas insertadas se integran directamente en un único usuario, grupo o rol. Las políticas administradas son políticas independientes que puede adjuntar a varios usuarios, grupos y roles de su Cuenta de AWS empresa. Las políticas administradas incluyen políticas AWS administradas y políticas administradas por el cliente. Para obtener más información sobre cómo elegir una política administrada o una política insertada, consulte Elegir entre políticas administradas y políticas insertadas en la Guía del usuario de IAM.

Políticas basadas en recursos

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Las políticas basadas en recursos son políticas insertadas que se encuentran en ese servicio. No puedes usar políticas AWS gestionadas de IAM en una política basada en recursos.

Listas de control de acceso () ACLs

Las listas de control de acceso (ACLs) controlan qué responsables (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLs son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Amazon S3 y Amazon VPC son ejemplos de servicios compatibles. AWS WAF ACLs Para obtener más información ACLs, consulte la descripción general de la lista de control de acceso (ACL) en la Guía para desarrolladores de Amazon Simple Storage Service.

Otros tipos de políticas

AWS admite tipos de políticas adicionales y menos comunes. Estos tipos de políticas pueden establecer el máximo de permisos que los tipos de políticas más frecuentes le conceden.

- Límites de permisos: un límite de permisos es una característica avanzada que le permite establecer los permisos máximos que una política basada en identidad puede conceder a una entidad de IAM (usuario o rol de IAM). Puedes establecer un límite de permisos para una entidad. Los permisos resultantes son la intersección de las políticas basadas en la identidad de la entidad y los límites de permisos. Las políticas basadas en recursos que especifiquen el usuario o rol en el campo `Principal` no estarán restringidas por el límite de permisos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para obtener más información sobre los límites de los permisos, consulte Límites de permisos para las entidades de IAM en la Guía del usuario de IAM.
- Políticas de control de servicios (SCPs): SCPs son políticas de JSON que especifican los permisos máximos para una organización o unidad organizativa (OU). AWS Organizations AWS Organizations es un servicio para agrupar y gestionar de forma centralizada varios de los Cuentas de AWS que son propiedad de su empresa. Si habilitas todas las funciones de una organización, puedes aplicar políticas de control de servicios (SCPs) a una o a todas tus cuentas. El SCP limita los permisos de las entidades en las cuentas de los miembros, incluidas las de cada una Usuario raíz de la cuenta de AWS. Para obtener más información sobre Organizations SCPs, consulte las políticas de control de servicios en la Guía del AWS Organizations usuario.
- Políticas de control de recursos (RCPs): RCPs son políticas de JSON que puedes usar para establecer los permisos máximos disponibles para los recursos de tus cuentas sin actualizar las políticas de IAM asociadas a cada recurso que poseas. El RCP limita los permisos de los recursos en las cuentas de los miembros y puede afectar a los permisos efectivos de las identidades, incluidos los permisos Usuario raíz de la cuenta de AWS, independientemente de si pertenecen a su organización. Para obtener más información sobre Organizations e RCPs incluir una lista de Servicios de AWS ese apoyo RCPs, consulte Políticas de control de recursos (RCPs) en la Guía del AWS Organizations usuario.
- Políticas de sesión: las políticas de sesión son políticas avanzadas que se pasan como parámetro cuando se crea una sesión temporal mediante programación para un rol o un usuario federado. Los permisos de la sesión resultantes son la intersección de las políticas basadas en identidades del rol y las políticas de la sesión. Los permisos también pueden proceder de una política en función de recursos. Una denegación explícita en cualquiera de estas políticas anulará el permiso. Para más información, consulte Políticas de sesión en la Guía del usuario de IAM.

Varios tipos de políticas

Cuando se aplican varios tipos de políticas a una solicitud, los permisos resultantes son más complicados de entender. Para saber cómo se AWS determina si se debe permitir una solicitud cuando se trata de varios tipos de políticas, consulte la lógica de evaluación de políticas en la Guía del usuario de IAM.

Cómo funciona la modernización AWS del mainframe con la IAM

Antes de usar la IAM para administrar el acceso a la modernización del AWS mainframe, averigüe qué funciones de IAM están disponibles para usar con la modernización del mainframe. AWS

Funciones de IAM que puede utilizar con la modernización de mainframes AWS

Característica de IAM	AWS Soporte para la modernización de mainframes
Políticas basadas en identidades	Sí
Políticas basadas en recursos	No
Acciones de políticas	Sí
Recursos de políticas	Sí
Claves de condición de política	Sí
ACLs	No
ABAC (etiquetas en políticas)	Sí
Credenciales temporales	Sí
Sesiones de acceso directo (FAS)	Sí
Roles de servicio	Sí
Roles vinculados al servicio	Sí

Para obtener una visión general de cómo funcionan la modernización del AWS mainframe y otros AWS servicios con la mayoría de las funciones de IAM, consulte los AWS servicios que funcionan con IAM en la Guía del usuario de IAM.

Políticas basadas en la identidad para la modernización del mainframe AWS

Compatibilidad con las políticas basadas en identidad: sí

Las políticas basadas en identidad son documentos de políticas de permisos JSON que puede asociar a una identidad, como un usuario de IAM, un grupo de usuarios o un rol. Estas políticas controlan qué acciones pueden realizar los usuarios y los roles, en qué recursos y en qué condiciones. Para obtener más información sobre cómo crear una política basada en identidad, consulte Creación de políticas de IAM en la Guía del usuario de IAM.

Con las políticas basadas en identidades de IAM, puede especificar las acciones y los recursos permitidos o denegados, así como las condiciones en las que se permiten o deniegan las acciones. No es posible especificar la entidad principal en una política basada en identidad porque se aplica al usuario o rol al que está asociada. Para obtener más información sobre los elementos que puede utilizar en una política de JSON, consulte Referencia de los elementos de las políticas de JSON de IAM en la Guía del usuario de IAM.

Ejemplos de políticas basadas en la identidad para la modernización del mainframe AWS

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Políticas basadas en recursos dentro de la modernización del mainframe AWS

Admite políticas basadas en recursos: no

Las políticas basadas en recursos son documentos de política JSON que se asocian a un recurso. Los ejemplos de políticas basadas en recursos son las políticas de confianza de roles de IAM y las políticas de bucket de Amazon S3. En los servicios que admiten políticas basadas en recursos, los administradores de servicios pueden utilizarlos para controlar el acceso a un recurso específico. Para el recurso al que se asocia la política, la política define qué acciones puede realizar una entidad principal especificada en ese recurso y en qué condiciones. Debe especificar una entidad principal en una política en función de recursos. Los principales pueden incluir cuentas, usuarios, roles, usuarios federados o. Servicios de AWS

Para habilitar el acceso entre cuentas, puede especificar toda una cuenta o entidades de IAM de otra cuenta como la entidad principal de una política en función de recursos. Añadir a una política

en función de recursos una entidad principal entre cuentas es solo una parte del establecimiento de una relación de confianza. Cuando el principal y el recurso son diferentes Cuentas de AWS, el administrador de IAM de la cuenta de confianza también debe conceder a la entidad principal (usuario o rol) permiso para acceder al recurso. Para conceder el permiso, adjunte la entidad a una política basada en identidad. Sin embargo, si la política basada en recursos concede acceso a una entidad principal de la misma cuenta, no es necesaria una política basada en identidad adicional. Para obtener más información, consulte Cross account resource access in IAM en la Guía del usuario de IAM.

Acciones políticas para la modernización del AWS mainframe

Compatibilidad con las acciones de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puede realizar acciones en qué recursos y en qué condiciones.

El elemento `Action` de una política JSON describe las acciones que puede utilizar para conceder o denegar el acceso en una política. Las acciones políticas suelen tener el mismo nombre que la operación de AWS API asociada. Hay algunas excepciones, como acciones de solo permiso que no tienen una operación de API coincidente. También hay algunas operaciones que requieren varias acciones en una política. Estas acciones adicionales se denominan acciones dependientes.

Incluya acciones en una política para conceder permisos y así llevar a cabo la operación asociada.

Para ver una lista de las acciones de modernización del AWS mainframe, consulte las acciones definidas por la modernización del AWS mainframe en la Referencia de autorización de servicios.

Las acciones políticas de modernización de AWS mainframes utilizan el siguiente prefijo antes de la acción:

```
m2
```

Para especificar varias acciones en una única instrucción, sepárelas con comas.

```
"Action": [
      "m2:StartApplication",
      "m2:StopApplication"
         ]
```

Puede utilizar caracteres comodín (*) para especificar varias acciones . Por ejemplo, para especificar todas las acciones que comiencen con la palabra `List`, incluya la siguiente acción:

```
"Action": "m2:List*"
```

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Recursos de políticas para la modernización de mainframes AWS

Compatibilidad con los recursos de políticas: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puedes realizar acciones en qué recursos y en qué condiciones.

El elemento `Resource` de la política JSON especifica el objeto u objetos a los que se aplica la acción. Las instrucciones deben contener un elemento `Resource` o `NotResource`. Como práctica recomendada, especifique un recurso utilizando el Nombre de recurso de Amazon (ARN). Puedes hacerlo para acciones que admitan un tipo de recurso específico, conocido como permisos de nivel de recurso.

Para las acciones que no admiten permisos de nivel de recurso, como las operaciones de descripción, utiliza un carácter comodín (*) para indicar que la instrucción se aplica a todos los recursos.

```
"Resource": "*"
```

Puede restringir el acceso a recursos específicos de modernización AWS del mainframe utilizándolos ARNs para identificar el recurso al que se aplica la política de IAM. Para obtener más información sobre el formato de ARNs, consulte Amazon Resource Names (ARNs) en Referencia general de AWS.

Por ejemplo, un entorno de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:m2:regionId:accountId:env/service-generated-unique-identifier"
```

Una aplicación de modernización de AWS mainframe tiene el siguiente ARN.

```
"Resource": "arn:aws:m2:regionId:accountId:app/service-generated-unique-identifier"
```

No todas las acciones de modernización AWS del mainframe admiten permisos a nivel de recursos. Para las acciones que no admiten permisos de recursos, debe utilizar el comodín (*).

Las siguientes acciones de modernización del AWS mainframe no admiten permisos a nivel de recursos.

```
ListApplications
            ListApplicationVersions
            ListBatchJobDefinitions
            ListBatchJobExecutions
            ListDataSetImportHistory
            ListDataSets
            ListDeployments
            ListEngineVersions
            ListEnvironments
            ListTagsForResource
```

Para ver una lista de los tipos de recursos de modernización del AWS mainframe y sus correspondientes ARNs, consulte los recursos definidos por la modernización del AWS mainframe en la referencia de autorización de servicios. Para saber con qué acciones puede especificar el ARN de cada recurso, consulte Acciones definidas por la modernización del AWS mainframe.

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

AWS Permisos de la API de modernización de mainframes: referencia sobre acciones, recursos y condiciones

Puede usar la siguiente tabla como referencia cuando escriba políticas de permisos que vaya a asociar a una identidad de IAM (políticas basadas en identidades). La tabla incluye lo siguiente:

- Cada operación de la API AWS de modernización de mainframe.
- Las acciones correspondientes para las que puede conceder permisos para realizar la acción
- El AWS recurso para el que puede conceder los permisos.

Las acciones se especifican en el campo `Action` de la política y el valor del recurso se especifica en el campo `Resource` de la política.

Puede utilizar claves de condición AWS globales en sus políticas de modernización de AWS mainframe para expresar las condiciones. Para obtener una lista completa de AWS las claves, consulte las claves de condición globales disponibles en la Guía del usuario de IAM.

> **Note**
>
> Para especificar una acción, use el prefijo `m2:` seguido del nombre de operación de la API (por ejemplo, `m2:CreateApplication`).

AWS API de modernización de mainframe y permisos necesarios para realizar acciones

AWS Operaciones de la API de modernización de mainframes	Permisos necesarios (acciones de la API)	Recursos
CancelBatchJobExecution		Aplicación

CreateApplication	`iam:PassRole`
	`kms:DescribeKey`
	`kms:CreateGrant`
	`s3:GetObject`
	`s3:ListBucket`
		Aplicación

CreateDataSetImportTask	`s3:GetObject`	Aplicación
CreateDataSetExportTask	`kms:DescribeKey` `s3:PutObject`	Aplicación

CreateDeployment	`elasticloadbalancing:AddTags`
	`elasticloadbalancing:CreateListener`
	`elasticloadbalancing:CreateTargetGroup`
		Aplicación

AWS Operaciones de la API de moderniza ción de mainframes	Permisos necesarios (acciones de la API)	Recursos

	`elasticloadbalanci ng:RegisterTargets`
	
	`elasticloadbalanci ng:DeleteListener`
	
	`elasticloadbalanci ng:DeleteTargetGroup`
	
	`elasticloadbalanci ng:DeregisterTargets`
	
	`elasticloadbalanci ng:DeleteLoadBalancer`
	
	`logs:CreateLogDelivery`
	
	`logs:GetLogDelivery`
	
	`logs:UpdateLogDelivery`
	
	`logs:DeleteLogDelivery`
	
	`logs:ListLogDeliveries`
	
	`logs:PutResourcePolicy`
	
	`logs:DescribeResou rcePolicies`
	
	`logs:DescribeLogGroups`	

AWS Operaciones de la API de modernización de mainframes	Permisos necesarios (acciones de la API)	Recursos
CreateEnvironment	ec2:CreateNetworkInterface ec2:CreateNetworkInterfacePermission ec2:DescribeNetworkInterfaces ec2:DescribeSecurityGroups ec2:DescribeSubnets ec2:DescribeVpcAttribute ec2:DescribeVpcs ec2:ModifyNetworkInterfaceAttribute elasticfilesystem:DescribeMountTargets elasticloadbalancing:AddTags elasticloadbalancing:CreateLoadBalancer elasticloadbalancing:DeleteLoadBalancer kms:DescribeKey kms:CreateGrant fsx:DescribeFileSystems	Entorno

AWS Operaciones de la API de modernización de mainframes	Permisos necesarios (acciones de la API)	Recursos
	`iam:CreateServiceLinkedRole`	

DeleteApplication	`elasticloadbalancing:DeleteListener`
	`elasticloadbalancing:DeleteTargetGroup`
	`logs:DeleteLogDelivery`
		Aplicación

DeleteApplicationFromEnvironment	`elasticloadbalancing:DeleteListener` `elasticloadbalancing:DeleteTargetGroup`	Aplicación Entorno
DeleteEnvironment	`elasticloadbalancing:DeleteLoadBalancer`	Entorno
GetApplication		Aplicación
GetApplicationVersion		Aplicación
GetBatchJobExecution		Aplicación
GetDataSetDetails		Aplicación
GetDataSetImportTask		Aplicación
GetDataSetExportTask		Aplicación
GetDeployment		Aplicación
GetEnvironment		Entorno

AWS Operaciones de la API de modernización de mainframes	Permisos necesarios (acciones de la API)	Recursos
ListApplications		*
ListApplicationVersions		*
ListBatchJobDefinitions		*
ListBatchJobExecutions		*
ListDataSetImportHistory		*
ListDataSetExportHistory		*
ListDataSets		*
ListDeployments		*
ListEngineVersions		*
ListEnvironments		*
ListTagsForResource		*
StartApplication		Aplicación
StartBatchJob		Aplicación
StopApplication		Aplicación
TagResource		*
UntagResource		*

AWS Operaciones de la API de moderniza ción de mainframes	Permisos necesarios (acciones de la API)	Recursos

UpdateApplication	`s3:GetObject`
	
	`s3:ListBucket`
		Aplicación

UpdateEnvironment	`kms:DescribeKey`	Entorno

Claves de condición de la política para la modernización del AWS mainframe

Compatibilidad con claves de condición de políticas específicas del servicio: sí

Los administradores pueden usar las políticas de AWS JSON para especificar quién tiene acceso a qué. Es decir, qué entidad principal puedes realizar acciones en qué recursos y en qué condiciones.

El elemento `Condition` (o bloque de `Condition`) permite especificar condiciones en las que entra en vigor una instrucción. El elemento `Condition` es opcional. Puedes crear expresiones condicionales que utilizan operadores de condición, tales como igual o menor que, para que la condición de la política coincida con los valores de la solicitud.

Si especifica varios elementos de `Condition` en una instrucción o varias claves en un único elemento de `Condition`, AWS las evalúa mediante una operación `AND` lógica. Si especifica varios valores para una única clave de condición, AWS evalúa la condición mediante una `OR` operación lógica. Se deben cumplir todas las condiciones antes de que se concedan los permisos de la instrucción.

También puedes utilizar variables de marcador de posición al especificar condiciones. Por ejemplo, puedes conceder un permiso de usuario de IAM para acceder a un recurso solo si está etiquetado con su nombre de usuario de IAM. Para más información, consulta Elementos de la política de IAM: variables y etiquetas en la Guía del usuario de IAM.

AWS admite claves de condición globales y claves de condición específicas del servicio. Para ver todas las claves de condición AWS globales, consulte las claves de contexto de condición AWS globales en la Guía del usuario de IAM.

Las siguientes claves de condición son específicas de la modernización del AWS mainframe

```
m2:EngineType
```

```
            m2:InstanceType
```

Para ver una lista de las claves de condición de la modernización del AWS mainframe, consulte las claves de condición para la modernización del AWS mainframe en la Referencia de autorización de servicio. Para saber con qué acciones y recursos puede utilizar una clave de condición, consulte Acciones definidas por la modernización del AWS mainframe.

Para ver ejemplos de políticas de modernización de AWS mainframes basadas en la identidad, consulte. Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

Listas de control de acceso (ACLs) en Mainframe Modernization AWS

Soporta ACLs: No

Las listas de control de acceso (ACLs) controlan qué directores (miembros de la cuenta, usuarios o roles) tienen permisos para acceder a un recurso. ACLs son similares a las políticas basadas en recursos, aunque no utilizan el formato de documento de políticas JSON.

Control de acceso basado en atributos (ABAC) con modernización del mainframe AWS

Admite ABAC (etiquetas en las políticas): sí

El control de acceso basado en atributos (ABAC) es una estrategia de autorización que define permisos en función de atributos. En AWS, estos atributos se denominan etiquetas. Puede adjuntar etiquetas a las entidades de IAM (usuarios o roles) y a muchos AWS recursos. El etiquetado de entidades y recursos es el primer paso de ABAC. A continuación, designa las políticas de ABAC para permitir operaciones cuando la etiqueta de la entidad principal coincida con la etiqueta del recurso al que se intenta acceder.

ABAC es útil en entornos que crecen con rapidez y ayuda en situaciones en las que la administración de las políticas resulta engorrosa.

Para controlar el acceso en función de etiquetas, debe proporcionar información de las etiquetas en el elemento de condición de una política utilizando las claves de condición `aws:ResourceTag/`*`key-name`*, `aws:RequestTag/`*`key-name`* o `aws:TagKeys`.

Si un servicio admite las tres claves de condición para cada tipo de recurso, el valor es Sí para el servicio. Si un servicio admite las tres claves de condición solo para algunos tipos de recursos, el valor es Parcial.

Para obtener más información sobre ABAC, consulte Definición de permisos con la autorización de ABAC en la Guía del usuario de IAM. Para ver un tutorial con los pasos para configurar ABAC, consulte Uso del control de acceso basado en atributos (ABAC) en la Guía del usuario de IAM.

Uso de credenciales temporales con la modernización del AWS mainframe

Compatibilidad con credenciales temporales: sí

Algunas Servicios de AWS no funcionan cuando se inicia sesión con credenciales temporales. Para obtener información adicional, incluidas las que Servicios de AWS funcionan con credenciales temporales, consulta Cómo Servicios de AWS funcionan con IAM en la Guía del usuario de IAM.

Utiliza credenciales temporales si inicia sesión en ellas AWS Management Console mediante cualquier método excepto un nombre de usuario y una contraseña. Por ejemplo, cuando accedes AWS mediante el enlace de inicio de sesión único (SSO) de tu empresa, ese proceso crea automáticamente credenciales temporales. También crea credenciales temporales de forma automática cuando inicia sesión en la consola como usuario y luego cambia de rol. Para obtener más información sobre el cambio de roles, consulte Cambio de un usuario a un rol de IAM (consola) en la Guía del usuario de IAM.

Puedes crear credenciales temporales manualmente mediante la AWS CLI API o. AWS A continuación, puede utilizar esas credenciales temporales para acceder AWS. AWS recomienda generar credenciales temporales de forma dinámica en lugar de utilizar claves de acceso a largo plazo. Para obtener más información, consulte Credenciales de seguridad temporales en IAM.

Sesiones de acceso directo para la modernización del AWS mainframe

Admite sesiones de acceso directo (FAS): sí

Cuando utiliza un usuario o un rol de IAM para realizar acciones en él AWS, se le considera director. Cuando utiliza algunos servicios, es posible que realice una acción que desencadene otra acción en un servicio diferente. FAS utiliza los permisos del principal que llama y los que solicita Servicio de AWS para realizar solicitudes a los servicios descendentes. Servicio de AWS Las solicitudes de FAS solo se realizan cuando un servicio recibe una solicitud que requiere interacciones con otros Servicios de AWS recursos para completarse. En este caso, debe tener permisos para realizar ambas acciones. Para obtener información sobre las políticas a la hora de realizar solicitudes de FAS, consulte Reenviar sesiones de acceso.

> **Important**
>
> Estos tokens permiten a AWS Mainframe Modernization acceder a los datos de los clientes sin su consentimiento explícito; por ejemplo, AWS Mainframe Modernization despliega artefactos de aplicaciones con datos empresariales asociados desde un bucket de Amazon S3 sin obtener el permiso explícito del cliente. Es posible que necesite actualizar la documentación de cumplimiento en consecuencia.

Roles de servicio para AWS Mainframe Modernization

Compatibilidad con roles de servicio: sí

Un rol de servicio es un rol de IAM que asume un servicio para realizar acciones en su nombre. Un administrador de IAM puede crear, modificar y eliminar un rol de servicio desde IAM. Para obtener más información, consulte Creación de un rol para delegar permisos a un Servicio de AWS en la Guía del usuario de IAM.

AWS La modernización del mainframe respalda las funciones de servicio para los grupos de actividad (transacciones, tareas pendientes o finalizadas, etc.).

> **Warning**
>
> Si se cambian los permisos de un rol de servicio, se podría interrumpir la funcionalidad de modernización del AWS mainframe. Edite las funciones de servicio solo cuando AWS Mainframe Modernization proporcione instrucciones para hacerlo.

Selección de un rol de IAM en AWS la modernización del mainframe

Si ya ha creado una función de IAM que EC2 puedan asumir sus aplicaciones que se ejecutan en Amazon, puede elegir esta función al crear una plantilla de lanzamiento o una configuración de lanzamiento. AWS La modernización del mainframe le proporciona una lista de funciones entre las que puede elegir. Al crear estos roles, es importante asociar políticas de menos privilegios de IAM que restrinjan el acceso a las llamadas a la API específicas que requiere la aplicación. Para obtener más información, consulte el rol de IAM para las aplicaciones que se ejecutan en EC2 instancias de Amazon en la Guía del usuario de Amazon EC2 Auto Scaling.

Funciones vinculadas a los servicios para la modernización del mainframe AWS

Admite roles vinculados a servicios: sí

Un rol vinculado al servicio es un tipo de rol de servicio que está vinculado a un. Servicio de AWS El servicio puede asumir el rol para realizar una acción en su nombre. Los roles vinculados al servicio aparecen en usted Cuenta de AWS y son propiedad del servicio. Un administrador de IAM puede ver, pero no editar, los permisos de los roles vinculados a servicios.

Para obtener más información sobre la creación o la administración de funciones vinculadas al servicio de modernización de AWS mainframe, consulte. Cómo utilizar roles vinculados a servicios de AWS Mainframe Modernization

Para más información sobre cómo crear o administrar roles vinculados a servicios, consulta Servicios de AWS que funcionan con IAM. Busque un servicio en la tabla que incluya `Yes` en la columna Rol vinculado a un servicio. Seleccione el vínculo Sí para ver la documentación acerca del rol vinculado a servicios para ese servicio.

Ejemplos de políticas basadas en la identidad para la modernización de mainframes AWS

De forma predeterminada, los usuarios y los roles no tienen permiso para crear o modificar los recursos de modernización del AWS mainframe. Tampoco pueden realizar tareas mediante la AWS Management Console, AWS Command Line Interface (AWS CLI) o la AWS API. Un administrador de IAM puede crear políticas de IAM para conceder permisos a los usuarios para realizar acciones en los recursos que necesitan. A continuación, el administrador puedes añadir las políticas de IAM a roles y los usuarios puedes asumirlos.

Para obtener información acerca de cómo crear una política basada en identidades de IAM mediante el uso de estos documentos de políticas JSON de ejemplo, consulte Creación de políticas de IAM (consola) en la Guía del usuario de IAM.

Para obtener más información sobre las acciones y los tipos de recursos definidos por la modernización del AWS mainframe, incluido el ARNs formato de cada uno de los tipos de recursos, consulte Acciones, recursos y claves de condición para la modernización del AWS mainframe en la Referencia de autorización de servicios.

Temas

- Prácticas recomendadas sobre las políticas

- Uso de la consola de modernización de mainframe AWS
- Cómo permitir a los usuarios consultar sus propios permisos

Prácticas recomendadas sobre las políticas

Las políticas basadas en la identidad determinan si alguien puede crear, acceder o eliminar los recursos de modernización del AWS mainframe de su cuenta. Estas acciones pueden generar costos adicionales para su Cuenta de AWS. Siga estas directrices y recomendaciones al crear o editar políticas basadas en identidades:

- Comience con las políticas AWS administradas y avance hacia los permisos con privilegios mínimos: para empezar a conceder permisos a sus usuarios y cargas de trabajo, utilice las políticas AWS administradas que otorgan permisos para muchos casos de uso comunes. Están disponibles en su. Cuenta de AWS Le recomendamos que reduzca aún más los permisos definiendo políticas administradas por el AWS cliente que sean específicas para sus casos de uso. Con el fin de obtener más información, consulta las políticas administradas por AWS o las políticas administradas por AWS para funciones de tarea en la Guía de usuario de IAM.
- Aplique permisos de privilegio mínimo: cuando establezca permisos con políticas de IAM, conceda solo los permisos necesarios para realizar una tarea. Para ello, debe definir las acciones que se pueden llevar a cabo en determinados recursos en condiciones específicas, también conocidos como permisos de privilegios mínimos. Con el fin de obtener más información sobre el uso de IAM para aplicar permisos, consulta Políticas y permisos en IAM en la Guía del usuario de IAM.
- Utilice condiciones en las políticas de IAM para restringir aún más el acceso: puede agregar una condición a sus políticas para limitar el acceso a las acciones y los recursos. Por ejemplo, puede escribir una condición de políticas para especificar que todas las solicitudes deben enviarse utilizando SSL. También puedes usar condiciones para conceder el acceso a las acciones del servicio si se utilizan a través de una acción específica Servicio de AWS, por ejemplo AWS CloudFormation. Para obtener más información, consulta Elementos de la política de JSON de IAM: Condición en la Guía del usuario de IAM.
- Utiliza el analizador de acceso de IAM para validar las políticas de IAM con el fin de garantizar la seguridad y funcionalidad de los permisos: el analizador de acceso de IAM valida políticas nuevas y existentes para que respeten el lenguaje (JSON) de las políticas de IAM y las prácticas recomendadas de IAM. El analizador de acceso de IAM proporciona más de 100 verificaciones de políticas y recomendaciones procesables para ayudar a crear políticas seguras y funcionales. Para más información, consulte Validación de políticas con el Analizador de acceso de IAM en la Guía del usuario de IAM.

- Requerir autenticación multifactor (MFA): si tiene un escenario que requiere usuarios de IAM o un usuario raíz en Cuenta de AWS su cuenta, active la MFA para mayor seguridad. Para exigir la MFA cuando se invoquen las operaciones de la API, añada condiciones de MFA a sus políticas. Para más información, consulte Acceso seguro a la API con MFA en la Guía del usuario de IAM.

Para obtener más información sobre las prácticas recomendadas de IAM, consulte Prácticas recomendadas de seguridad en IAM en la Guía del usuario de IAM.

Uso de la consola de modernización de mainframe AWS

Para acceder a la consola de modernización del AWS mainframe, debe tener un conjunto mínimo de permisos. Estos permisos deben permitirle enumerar y ver detalles sobre los recursos de modernización del AWS mainframe que tiene en su cuenta. Cuenta de AWS Si crea una política basada en identidades que sea más restrictiva que el mínimo de permisos necesarios, la consola no funcionará del modo esperado para las entidades (usuarios o roles) que tengan esa política.

No es necesario conceder permisos mínimos de consola a los usuarios que solo realizan llamadas a la API AWS CLI o a la AWS API. En su lugar, permite el acceso únicamente a las acciones que coincidan con la operación de API que intentan realizar.

Para garantizar que los usuarios y los roles puedan seguir utilizando la consola de modernización del AWS mainframe, adjunte también la política `ReadOnly` AWS gestionada `ConsoleAccess` o de modernización del AWS mainframe a las entidades. Para obtener más información, consulte Adición de permisos a un usuario en la Guía del usuario de IAM:

Cómo permitir a los usuarios consultar sus propios permisos

En este ejemplo, se muestra cómo podría crear una política que permita a los usuarios de IAM ver las políticas gestionadas e insertadas que se asocian a la identidad de sus usuarios. Esta política incluye permisos para completar esta acción en la consola o mediante programación mediante la API o. AWS CLI AWS

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
```

```
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Solución de problemas de AWS identidad y acceso a la modernización del mainframe

Utilice la siguiente información como ayuda para diagnosticar y solucionar los problemas más comunes que pueden surgir al trabajar con la modernización de AWS mainframes y la IAM.

Temas

- No estoy autorizado a realizar la iam: PassRole
- Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis recursos de modernización de AWS mainframe

No estoy autorizado a realizar la iam: PassRole

Si recibe un mensaje de error que indica que no está autorizado a realizar la `iam:PassRole` acción, sus políticas deben actualizarse para que pueda transferir una función a AWS Mainframe Modernization.

Algunas Servicios de AWS permiten transferir una función existente a ese servicio en lugar de crear una nueva función de servicio o una función vinculada al servicio. Para ello, debe tener permisos para transferir el rol al servicio.

En el siguiente ejemplo, el error se produce cuando un usuario de IAM denominado `marymajor` intenta utilizar la consola para realizar una acción en AWS Mainframe Modernization. Sin embargo, la acción requiere que el servicio cuente con permisos que otorguen un rol de servicio. Mary no tiene permisos para transferir el rol al servicio.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

En este caso, las políticas de Mary se deben actualizar para permitirle realizar la acción `iam:PassRole`.

Si necesita ayuda, póngase en contacto con su administrador. AWS El gestionador es la persona que le proporcionó las credenciales de inicio de sesión.

Quiero permitir que personas ajenas a mí accedan Cuenta de AWS a mis recursos de modernización de AWS mainframe

Puede crear un rol que los usuarios de otras cuentas o las personas externas a la organización puedan utilizar para acceder a sus recursos. Puede especificar una persona de confianza para que asuma el rol. En el caso de los servicios que respaldan las políticas basadas en recursos o las listas de control de acceso (ACLs), puede usar esas políticas para permitir que las personas accedan a sus recursos.

Para obtener más información, consulte lo siguiente:

- Para saber si la modernización AWS del mainframe admite estas funciones, consulte. Cómo funciona la modernización AWS del mainframe con la IAM
- Para obtener información sobre cómo proporcionar acceso a los recursos de su Cuentas de AWS propiedad, consulte Proporcionar acceso a un usuario de IAM en otro usuario de su propiedad Cuenta de AWS en la Guía del usuario de IAM.

- Para obtener información sobre cómo proporcionar acceso a tus recursos a terceros Cuentas de AWS, consulta Cómo proporcionar acceso a recursos que Cuentas de AWS son propiedad de terceros en la Guía del usuario de IAM.
- Para obtener información sobre cómo proporcionar acceso mediante una federación de identidades, consulta Proporcionar acceso a usuarios autenticados externamente (identidad federada) en la Guía del usuario de IAM.
- Para conocer sobre la diferencia entre las políticas basadas en roles y en recursos para el acceso entre cuentas, consulte Acceso a recursos entre cuentas en IAM en la Guía del usuario de IAM.

Cómo utilizar roles vinculados a servicios de AWS Mainframe Modernization

AWS Mainframe Modernization usa roles vinculados al AWS Identity and Access Management servicio (IAM). Un rol vinculado a un servicio es un tipo único de rol de IAM al que se vincula directamente. AWS Mainframe Modernization Los roles vinculados al servicio están predefinidos AWS Mainframe Modernization e incluyen todos los permisos que el servicio requiere para llamar a otros AWS servicios en su nombre.

Un rol vinculado a un servicio facilita la configuración AWS Mainframe Modernization , ya que no es necesario añadir manualmente los permisos necesarios. AWS Mainframe Modernization define los permisos de sus funciones vinculadas al servicio y, a menos que se defina lo contrario, solo AWS Mainframe Modernization puede asumir sus funciones. Los permisos definidos incluyen las políticas de confianza y de permisos, y que la política de permisos no se pueda adjuntar a ninguna otra entidad de IAM.

Solo es posible eliminar un rol vinculado a un servicio después de eliminar sus recursos relacionados. Esto protege sus AWS Mainframe Modernization recursos porque no puede eliminar inadvertidamente el permiso de acceso a los recursos.

Para obtener información sobre otros servicios que admiten funciones vinculadas a servicios, consulte AWS Servicios que funcionan con IAM y busque los servicios con la palabra Sí en la columna Funciones vinculadas a servicios. Seleccione una opción Sí con un enlace para ver la documentación acerca del rol vinculado al servicio en cuestión.

Permisos de roles vinculados a servicios de AWS Mainframe Modernization

AWS Mainframe Modernization utiliza el rol vinculado al servicio denominado AWSServiceRoleForAWSM2: configure la red para que se conecte a su VPC y acceda a recursos como los sistemas de archivos.

El rol AWSService RoleFor AWSM2 vinculado al servicio confía en los siguientes servicios para asumir el rol:

- `m2.amazonaws.com`

La política de permisos de roles denominada AWSM2 ServicePolicy permite AWS Mainframe Modernization realizar las siguientes acciones en los recursos especificados:

- Cree, elimine, describa y adjunte permisos a las interfaces de EC2 red de Amazon para que el AWS Mainframe Modernization entorno establezca la conectividad con la VPC del cliente.
- Registra o anula el registro de las entradas de Elastic Load Balancing, que es la forma en que los clientes se conectan al AWS Mainframe Modernization entorno.
- Describa el Amazon EFS o el sistema de FSx archivos de Amazon, si se utiliza.
- Envía métricas al cliente CloudWatch desde el entorno de ejecución.

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Effect": "Allow",
   "Action": [
    "ec2:DescribeSubnets",
    "ec2:CreateNetworkInterface",
    "ec2:DeleteNetworkInterface",
    "ec2:DescribeNetworkInterfaces",
    "ec2:CreateNetworkInterfacePermission",
    "ec2:ModifyNetworkInterfaceAttribute"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
```

```
    "elasticfilesystem:DescribeMountTargets"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "elasticloadbalancing:RegisterTargets",
    "elasticloadbalancing:DeregisterTargets"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "fsx:DescribeFileSystems"
   ],
   "Resource": "*"
  },
  {
   "Effect": "Allow",
   "Action": [
    "cloudwatch:PutMetricData"
   ],
   "Resource": "*",
   "Condition": {
    "StringEquals": {
     "cloudwatch:namespace": [
      "AWS/M2"
     ]
    }
   }
  }
 ]
}
```

Debe configurar permisos para permitir a una entidad de IAM (como un usuario, grupo o rol) crear, editar o eliminar un rol vinculado a servicios. Para obtener más información, consulte Permisos de roles vinculados a servicios en la Guía del usuario de IAM.

Creación de un rol vinculado a un servicio de AWS Mainframe Modernization

No necesita crear manualmente un rol vinculado a servicios. Al crear un entorno de ejecución en la AWS Management Console AWS CLI, la API o la AWS API, AWS Mainframe Modernization crea automáticamente el rol vinculado al servicio.

Si elimina este rol vinculado a servicios y necesita crearlo de nuevo, puede utilizar el mismo proceso para volver a crear el rol en su cuenta. Al crear un entorno de tiempo de ejecución, AWS Mainframe Modernization crea el rol vinculado a servicios para usted de nuevo.

Modificación de un rol vinculado a servicios de AWS Mainframe Modernization

AWS Mainframe Modernization no permite editar el rol vinculado al AWSService RoleFor AWSM2 servicio. Después de crear un rol vinculado al servicio, no podrá cambiar el nombre del rol, ya que varias entidades podrían hacer referencia al rol. Sin embargo, sí puede editar la descripción del rol con IAM. Para obtener más información, consulte Editar un rol vinculado a servicios en la Guía del usuario de IAM.

Eliminación de un rol vinculado a un servicio de AWS Mainframe Modernization

Si ya no necesita usar una característica o servicio que requieran un rol vinculado a un servicio, le recomendamos que elimine dicho rol. Así no tendrá una entidad no utilizada que no se supervise ni mantenga de forma activa. Sin embargo, debe limpiar los recursos de su rol vinculado al servicio antes de eliminarlo manualmente.

> **Note**
>
> Si el AWS Mainframe Modernization servicio utiliza el rol al intentar eliminar los recursos, es posible que la eliminación no se realice correctamente. En tal caso, espere unos minutos e intente de nuevo la operación.

Para eliminar AWS Mainframe Modernization los recursos utilizados por el AWSService RoleFor AWSM2

- Elimine los entornos de ejecución en AWS Mainframe Modernization. Asegúrese de eliminar las aplicaciones de un entorno antes de eliminar el entorno en sí.

Para eliminar manualmente el rol vinculado a servicios mediante IAM

Utilice la consola de IAM AWS CLI, la o la AWS API para eliminar la función vinculada al AWSService RoleFor AWSM2 servicio. Para obtener más información, consulte Eliminación de un rol vinculado a servicios en la Guía del usuario de IAM.

Regiones admitidas para los roles vinculados a un servicio de AWS Mainframe Modernization

AWS Mainframe Modernization admite el uso de funciones vinculadas al servicio en todas las regiones en las que el servicio está disponible. Para obtener más información, consulte Puntos de enlace y regiones de AWS.

Validación del cumplimiento para la modernización del mainframe AWS

Los auditores externos evalúan la seguridad y el cumplimiento de la modernización del AWS mainframe como parte de varios programas de AWS cumplimiento. Estos incluyen SOC, PCI, FedRAMP, HIPAA y otros.

Para ver una lista de AWS los servicios incluidos en el ámbito de los programas de conformidad específicos, consulte Servicios de AWS incluidos . Para obtener información general, consulta Programas de conformidad de AWS.

Puede descargar informes de auditoría de terceros utilizando AWS Artifact. Para obtener más información, consulte Descarga de informes en AWS Artifact .

Su responsabilidad en materia de cumplimiento al utilizar la modernización de AWS mainframes viene determinada por la confidencialidad de sus datos, los objetivos de cumplimiento de su empresa y las leyes y reglamentos aplicables. AWS proporciona los siguientes recursos para ayudar con el cumplimiento:

- Security and Compliance Quick Start Guides (Guías de inicio rápido de seguridad y conformidad) (Guías de inicio rápido de seguridad y conformidad): Estas guías de implementación analizan las consideraciones en materia de arquitectura y proporcionan los pasos para implementar los entornos de referencia centrados en la seguridad y la conformidad en AWS.
- Documento técnico sobre cómo diseñar una arquitectura basada en la seguridad y el cumplimiento de la HIPAA: en este documento técnico se describe cómo pueden utilizar las empresas para crear aplicaciones que cumplan con la HIPAA. AWS

- AWS Recursos de cumplimiento Recursos de de trabajo y guías puede aplicarse a su sector y ubicación.
- Evaluación de los recursos con las reglas de la guía para AWS Config desarrolladores: AWS Config evalúa en qué medida las configuraciones de sus recursos cumplen con las prácticas internas, las directrices del sector y las normas.
- AWS Security Hub— Este AWS servicio proporciona una visión integral del estado de su seguridad AWS que le ayuda a comprobar el cumplimiento de los estándares y las mejores prácticas del sector de la seguridad.

Resiliencia en la AWS modernización del mainframe

La infraestructura AWS global se basa en AWS regiones y zonas de disponibilidad. Las regiones proporcionan varias zonas de disponibilidad físicamente independientes y aisladas que se encuentran conectadas mediante redes con un alto nivel de rendimiento y redundancia, además de baja demora. Con las zonas de disponibilidad, puede diseñar y utilizar aplicaciones y bases de datos que realizan una conmutación por error automática entre las zonas sin interrupciones. Las zonas de disponibilidad tienen una mayor disponibilidad, tolerancia a errores y escalabilidad que las infraestructuras tradicionales de uno o varios centros de datos.

Para obtener más información sobre AWS las regiones y las zonas de disponibilidad, consulte Infraestructura AWS global.

Seguridad de la infraestructura en AWS Mainframe Modernization

Como servicio gestionado, AWS Mainframe Modernization está protegido por la seguridad de la red AWS global. Para obtener información sobre los servicios AWS de seguridad y cómo se AWS protege la infraestructura, consulte Seguridad AWS en la nube. Para diseñar su AWS entorno utilizando las mejores prácticas de seguridad de la infraestructura, consulte Protección de infraestructuras en un marco de buena AWS arquitectura basado en el pilar de la seguridad.

Utiliza las llamadas a la API AWS publicadas para acceder a AWS Mainframe Modernization través de la red. Los clientes deben admitir lo siguiente:

- Seguridad de la capa de transporte (TLS). Exigimos TLS 1.2 y recomendamos TLS 1.3.
- Conjuntos de cifrado con confidencialidad directa total (PFS) como DHE (Ephemeral Diffie-Hellman) o ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). La mayoría de los sistemas modernos como Java 7 y posteriores son compatibles con estos modos.

Además, las solicitudes deben estar firmadas mediante un ID de clave de acceso y una clave de acceso secreta que esté asociada a una entidad principal de IAM. También puedes utilizar AWS Security Token Service (AWS STS) para generar credenciales de seguridad temporales para firmar solicitudes.

Acceda AWS Mainframe Modernization mediante un punto final AWS PrivateLink de interfaz

Puede usarlo AWS PrivateLink para crear una conexión privada entre su VPC y. AWS Mainframe Modernization Puede acceder AWS Mainframe Modernization como si estuviera en su VPC, sin el uso de una puerta de enlace a Internet, un dispositivo NAT, una conexión VPN o AWS Direct Connect una conexión. Las instancias de la VPC no necesitan direcciones IP públicas para acceder a AWS Mainframe Modernization.

Esta conexión privada se establece mediante la creación de un punto de conexión de interfaz alimentado por AWS PrivateLink. Creamos una interfaz de red de punto de conexión en cada subred habilitada para el punto de conexión de interfaz. Se trata de interfaces de red administradas por el solicitante que sirven como punto de entrada para el tráfico destinado a AWS Mainframe Modernization.

Para obtener más información, consulte Acceso Servicios de AWS directo AWS PrivateLink en la AWS PrivateLink Guía.

Consideraciones sobre AWS Mainframe Modernization

Antes de configurar un punto final de interfaz para AWS Mainframe Modernization, consulte las consideraciones de la AWS PrivateLink guía.

AWS Mainframe Modernization permite realizar llamadas a todas sus acciones de API a través del punto final de la interfaz.

Cree un punto final de interfaz para AWS Mainframe Modernization

Puede crear un punto final de interfaz para AWS Mainframe Modernization usar la consola de Amazon VPC o AWS Command Line Interface ()AWS CLI. Para obtener más información, consulte Creación de un punto de conexión de interfaz en la Guía de AWS PrivateLink .

Cree un punto final de interfaz para AWS Mainframe Modernization usar el siguiente nombre de servicio:

```
com.amazonaws.region.m2
```

Si habilita DNS privado para el punto de conexión de interfaz, puede realizar solicitudes a la API para AWS Mainframe Modernization usando su nombre de DNS predeterminado para la región. Por ejemplo, `m2.us-east-1.amazonaws.com`.

Creación de una política de puntos de conexión para el punto de conexión de interfaz

Una política de punto de conexión es un recurso de IAM que puede adjuntar al punto de conexión de su interfaz. La política de punto final predeterminada permite el acceso total a AWS Mainframe Modernization través del punto final de la interfaz. Para controlar el acceso permitido AWS Mainframe Modernization desde su VPC, adjunte una política de punto final personalizada al punto final de la interfaz.

Una política de punto de conexión especifica la siguiente información:

- Las entidades principales (Cuentas de AWS, usuarios y roles de IAM) que puede realizar acciones
- Las acciones que se pueden realizar.
- El recurso en el que se pueden realizar las acciones.

Para obtener más información, consulte Control del acceso a los servicios con políticas de punto de conexión en la Guía del usuario de AWS PrivateLink .

Ejemplo: política de puntos finales de VPC para acciones AWS Mainframe Modernization

A continuación, se muestra un ejemplo de una política de un punto de conexión personalizada. Cuando se asocia con un punto de conexión, esta política concede acceso a las acciones de AWS Mainframe Modernization mostradas para todas las entidades principales en todos los recursos.

```
//Example of an endpoint policy where access is granted to the
//listed AWS Mainframe Modernization actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Allow",
         "Action": [
            "m2:ListApplications",
            "m2:ListEnvironments",
```

```
            "m2:ListDeployments"
         ],
         "Resource":"*"
      }
   ]
}

//Example of an endpoint policy where access is denied to all the
//AWS Mainframe Modernization CREATE actions for all principals on all resources
{"Statement": [
      {"Principal": "*",
         "Effect": "Deny",
         "Action": [
            "m2:Create*"
         ],
         "Resource":"*"
      }
   ]
}
```

Supervisión de la AWS modernización del mainframe

La supervisión es una parte importante del mantenimiento de la fiabilidad, la disponibilidad y el rendimiento de la modernización del AWS mainframe y del resto de las soluciones de AWS. AWS proporciona las siguientes herramientas de supervisión para supervisar la modernización del AWS mainframe, informar cuando algo va mal y tomar medidas automáticas cuando sea necesario:

- Amazon CloudWatch monitorea tus AWS recursos y las aplicaciones en las que AWS ejecutas en tiempo real. Puede recopilar métricas y realizar un seguimiento de las métricas, crear paneles personalizados y definir alarmas que le advierten o que toman medidas cuando una métrica determinada alcanza el umbral que se especifique. Por ejemplo, puedes CloudWatch hacer un seguimiento del uso de la CPU u otras métricas de tus EC2 instancias de Amazon y lanzar automáticamente nuevas instancias cuando sea necesario. Para obtener más información, consulta la Guía del CloudWatch usuario de Amazon.
- Amazon CloudWatch Logs le permite supervisar, almacenar y acceder a sus archivos de registro desde EC2 instancias de Amazon y otras fuentes. CloudTrail CloudWatch Los registros pueden monitorear la información de los archivos de registro y notificarle cuando se alcanzan ciertos umbrales. También se pueden archivar los datos del registro en un almacenamiento de larga duración. Para obtener más información, consulta la Guía del usuario CloudWatch de Amazon Logs.
- AWS CloudTrailcaptura las llamadas a la API y los eventos relacionados realizados por su AWS cuenta o en su nombre y entrega los archivos de registro a un bucket de Amazon S3 que especifique. Puede identificar qué usuarios y cuentas llamaron AWS, la dirección IP de origen desde la que se realizaron las llamadas y cuándo se produjeron. Para obtener más información, consulte la Guía del usuario de AWS CloudTrail.

Supervisión de la modernización AWS del mainframe con Amazon CloudWatch

Puede monitorizar la modernización AWS del mainframe mediante CloudWatch una herramienta que recopila datos sin procesar y los procesa para convertirlos en métricas legibles prácticamente en tiempo real. Estas estadísticas se mantienen durante 15 meses, de forma que pueda obtener acceso a información histórica y disponer de una mejor perspectiva sobre el desempeño de su aplicación web o servicio. También puede establecer alarmas que vigilen determinados umbrales

y enviar notificaciones o realizar acciones cuando se cumplan dichos umbrales. Para obtener más información, consulta la Guía del CloudWatch usuario de Amazon.

En las siguientes tablas se enumeran las métricas y las dimensiones de la modernización AWS del mainframe. El namespace para estas métricas es `AWS/M2`.

Métricas del entorno de tiempo de ejecución

Métrica	Descripción
CPUUtilization	El uso de la CPU de las instancias del entorno. Dimensión: environmentId Unidad: porcentaje Estadísticas válidas: promedio, mínimo, máximo
InboundNetworkThroughput	Rendimiento de red entrante de las instancias del entorno. Dimensión: environmentId Unidades: bytes por segundo Estadísticas válidas: promedio, mínimo, máximo
MemoryUtilization	El uso de la memoria de las instancias del entorno. Dimensión: environmentId Unidad: porcentaje Estadísticas válidas: promedio, mínimo, máximo
OutboundNetworkThroughput	Rendimiento de red saliente de las instancias del entorno.

Métrica	Descripción
	Dimensión: environmentId Unidades: bytes por segundo Estadísticas válidas: promedio, mínimo, máximo

Métricas de aplicación

Métrica	Descripción
BatchJobCompletedCount	El número de trabajos completados durante el intervalo de tiempo. Esta métrica está disponible para Rocket Software (anteriormente Micro Focus) y para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: recuento Estadísticas válidas: Sum
BatchJobFailedCount	El número de trabajos con error durante el intervalo de tiempo. Esta métrica está disponible para Rocket Software y para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: recuento Estadísticas válidas: Sum

Métrica	Descripción
JvmMemoryFree	Lcantidad de memoria disponible que la máquina virtual Java no está utilizando en ese momento. Esta métrica solo está disponible para el motor de ejecución AWS Blu Age. Está disponible para AWS Blu Age 3.7.0 y versiones posterior es. Dimensión: applicationId Unidades: bytes Estadísticas válidas: promedio, mínimo, máximo
JvmMemoryMax	La cantidad máxima de memoria permitida para la máquina virtual Java. Esta métrica solo está disponible para el motor de tiempo de ejecución AWS Blu Age. Está disponible para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: bytes Estadísticas válidas: promedio, mínimo, máximo

Métrica	Descripción
JvmMemoryUsed	La cantidad de memoria utilizada activamente por la máquina virtual Java. Esta métrica solo está disponible para el motor de tiempo de ejecución AWS Blu Age. Está disponible para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: bytes Estadísticas válidas: promedio, mínimo, máximo
ProcessesActiveCount	El número activo de procesos de ejecución de servicios simultáneos que procesan solicitudes. Esta métrica solo está disponible para el motor de tiempo de ejecución de Rocket Software. Dimensión: applicationId Unidades: recuento Estadísticas válidas: Sum

Métrica	Descripción
SessionCount	El número de sesiones HTTP de la aplicación. Esta métrica solo está disponible para el motor de ejecución AWS Blu Age. Está disponible para AWS Blu Age 3.7.0 y versiones posterior es. Dimensión: applicationId Unidades: recuento Estadísticas válidas: promedio, mínimo, máximo
SharedMemoryFree	La memoria disponible para que el servidor empresarial almacene toda la información que necesita para ejecutar transacciones y trabajos. Esta métrica solo está disponible para el motor de tiempo de ejecución de Rocket Software. Dimensión: applicationId Unidades: kilobytes Estadísticas válidas: promedio, mínimo, máximo

Métrica	Descripción
SharedMemoryTotal	La cantidad total de memoria compartid a asignada al servidor empresarial para almacenar toda la información que necesita para ejecutar transacciones y tareas. Esta métrica solo está disponible para el motor de ejecución de Rocket Software. Dimensión: applicationId Unidades: kilobytes Estadísticas válidas: promedio, mínimo, máximo
ThreadActiveCount	El número de subprocesos del motor que procesan las solicitudes. Esta métrica solo está disponible para el motor de ejecución AWS Blu Age. Está disponible para AWS Blu Age 3.7.0 y versiones posterior es. Dimensión: applicationId Unidades: recuento Estadísticas válidas: promedio, mínimo, máximo

Métrica	Descripción
TransactionCompletedCount	El número de transacciones confirmadas durante el intervalo de tiempo. Esta métrica está disponible para Rocket Software y para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: recuento Estadísticas válidas: Sum
TransactionFailedCount	El número de transacciones con error durante el intervalo de tiempo. Esta métrica está disponible para Rocket Software y para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: recuento Estadísticas válidas: Sum

Métrica	Descripción
TransactionResponseTime	El tiempo transcurrido desde el momento en que un usuario envía una solicitud hasta el momento en que la aplicación indica que la solicitud se ha completado. Esta métrica está disponible para Rocket Software y para AWS Blu Age 3.7.0 y versiones posteriores. Dimensión: applicationId Unidades: milisegundos Estadísticas válidas: promedio, mínimo, máximo

Dimensiones

Dimensión	Descripción
applicationId	Esta dimensión filtra la métrica hasta la aplicación identificada por ID.
environmentId	Esta dimensión filtra la métrica hasta el entorno identificado por ID.

Registro de llamadas a la API de modernización de AWS mainframe mediante AWS CloudTrail

AWS La modernización del mainframe está integrada con AWS CloudTrail un servicio que proporciona un registro de las acciones realizadas por un usuario, un rol o un AWS servicio en el marco de la modernización del AWS mainframe. CloudTrail captura todas las llamadas a la API para la modernización del AWS mainframe como eventos. Las llamadas capturadas incluyen llamadas desde la consola de modernización del AWS mainframe y llamadas en código a las operaciones de

la API de modernización del AWS mainframe. Si crea una ruta, puede habilitar la entrega continua de CloudTrail eventos a un bucket de Amazon S3, incluidos los eventos para la modernización del AWS mainframe. Si no configura una ruta, podrá ver los eventos más recientes en la CloudTrail consola, en el historial de eventos. Con la información recopilada por CloudTrail, puede determinar la solicitud que se realizó a AWS Mainframe Modernization, la dirección IP desde la que se realizó la solicitud, quién la realizó, cuándo se realizó y detalles adicionales.

Para obtener más información CloudTrail, consulte la Guía del AWS CloudTrail usuario.

AWS Información sobre la modernización de mainframes en CloudTrail

CloudTrail está habilitada en su AWS cuenta al crear la cuenta. Cuando se produce una actividad en la modernización AWS del mainframe, esa actividad se registra en un CloudTrail evento junto con otros eventos de AWS servicio en el historial de eventos. Puede ver, buscar y descargar los eventos recientes en su AWS cuenta. Para obtener más información, consulte Visualización de eventos con el historial de CloudTrail eventos.

Para obtener un registro continuo de los eventos de su AWS cuenta, incluidos los relacionados con la modernización AWS del mainframe, cree un registro. Un rastro permite CloudTrail entregar archivos de registro a un bucket de Amazon S3. De forma predeterminada, cuando se crea un registro de seguimiento en la consola, el registro de seguimiento se aplica a todas las regiones de AWS. La ruta registra los eventos de todas las regiones de la AWS partición y envía los archivos de registro al bucket de Amazon S3 que especifique. Además, puede configurar otros AWS servicios para analizar más a fondo los datos de eventos recopilados en los CloudTrail registros y actuar en función de ellos. Para más información, consulte los siguientes temas:

- Introducción a la creación de registros de seguimiento
- CloudTrail servicios e integraciones compatibles
- Configuración de las notificaciones de Amazon SNS para CloudTrail
- Recibir archivos de CloudTrail registro de varias regiones
- Recibir archivos de CloudTrail registro de varias cuentas

Todas las acciones de modernización AWS del mainframe se registran CloudTrail y se documentan en la referencia de la API de modernización del AWS mainframe. Por ejemplo, las llamadas a `CreateEnvironment` y `CreateDeployment` las acciones generan entradas en los archivos de CloudTrail registro. `CreateApplication`

Cada entrada de registro o evento contiene información sobre quién generó la solicitud. La información de identidad del usuario le ayuda a determinar lo siguiente:

- Si la solicitud se realizó con las credenciales del usuario raíz o del usuario.
- Si la solicitud se realizó con credenciales de seguridad temporales de un rol o fue un usuario federado.
- Si la solicitud la realizó otro AWS servicio.

Para obtener más información, consulte el elemento userIdentity de CloudTrail .

Descripción de las entradas de los archivos de registro de modernización de AWS mainframe

Un rastro es una configuración que permite la entrega de eventos como archivos de registro a un bucket de Amazon S3 que usted especifique. CloudTrail Los archivos de registro contienen una o más entradas de registro. Un evento representa una solicitud única de cualquier fuente e incluye información sobre la acción solicitada, la fecha y la hora de la acción, los parámetros de la solicitud, etc. CloudTrail Los archivos de registro no son un registro ordenado de las llamadas a la API pública, por lo que no aparecen en ningún orden específico.

En el siguiente ejemplo, se muestra una entrada de CloudTrail registro que demuestra la `CreateApplication` acción.

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AROAII6WZTHGYAEXAMPLE",
        "arn": "arn:aws:sts::444455556666:assumed-role/Admin/Mary_Major",
        "accountId": "444455556666",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAII6WZTHGYAEXAMPLE",
                "arn": "arn:aws:iam::444455556666:role/Admin",
                "accountId": "444455556666",
                "userName": "Admin"
            },
            "webIdFederationData": {},
```

```
            "attributes": {
                "creationDate": "2022-06-01T20:38:22Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2022-06-01T20:40:39Z",
    "eventSource": "m2.amazonaws.com",
    "eventName": "CreateApplication",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "72.21.196.65",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:91.0) Gecko/20100101
 Firefox/91.0",
    "requestParameters": {
        "clientToken": "1abc23de-f45g-6789-h01i-jkl2m3456789",
        "name": "MyApp",
        "description": "",
        "engineType": "microfocus",
        "definition": {
            "content": "{}"
        },
        "tags": {}
    },
    "responseElements": {
        "applicationVersion": 1,
        "Access-Control-Expose-Headers": "x-amzn-RequestId,x-amzn-ErrorType,x-amzn-
ErrorMessage,Date",
        "applicationArn": "arn:aws:m2:us-east-1:444455556666:app/
lsfhmwhw7fffrosff2lncwqcua",
        "applicationId": "lsfhmwhw7fffrosff2lncwqcua"
    },
    "requestID": "36982d38-fcde-4bfe-a89a-7bd78d43c926",
    "eventID": "d7f0fc36-46ae-4157-9a79-c79f385fda98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "444455556666",
    "eventCategory": "Management"
}
```

Solución de problemas en la AWS modernización de mainframe

Utilice la información de esta sección como ayuda para solucionar errores comunes en las aplicaciones de modernización de AWS mainframes y en los entornos de ejecución mediante los motores de software AWS Blu Age y Rocket.

Temas

- Troubleshooting error: Time out while waiting for data set name to be unlocked
- Troubleshooting error: Cannot access an application URL
- Solución de problemas: AWS Blu Insights no se abre desde la consola
- Troubleshooting error: Environment unhealthy
- Solución de problemas de licencia para Rocket Software (anteriormente Micro Focus)

Troubleshooting error: Time out while waiting for data set name to be unlocked

En esta página, se describe cómo puede resolver un error cuando vea que otra aplicación de un entorno bloquea un conjunto de datos compartido.

- Motor: AWS Blu Age
- Componente: Blusam

Si ve este error en los CloudWatch registros de Amazon de una aplicación de modernización de AWS mainframe que utiliza el motor AWS Blu Age y se ejecuta en un entorno con el patrón de alta disponibilidad, indica que otra aplicación está bloqueando un conjunto de datos compartido. Por lo general, esta situación se produce si la otra aplicación se bloquea o se produce un error y no libera el bloqueo.

Busque una aplicación con errores y compruebe si utiliza el mismo conjunto de datos mencionado en el mensaje de error. Compruebe si la aplicación se ejecuta en un entorno de tiempo de ejecución con el patrón de alta disponibilidad. La aplicación que ha generado la excepción de tiempo de espera no puede continuar y mostrará el estado `Failed`.

Causa habitual

La aplicación `example-app-1` intenta bloquear un registro `example-record-1` para una operación de escritura. Esta operación crea tanto un bloqueo en el conjunto de datos `example-dataset-1`, que es propietario de `example-record-1`, como un bloqueo en el mismo `example-record-1`. Ahora, otra aplicación, `example-app-2`, intenta bloquear el mismo registro `example-record-1`. El conjunto de datos y el registro ya están bloqueados, por lo que `example-app-2` espera a que se libere el bloqueo. Si `example-app-1` falla, el conjunto de datos bloqueado `example-dataset-1` sigue existiendo, lo que provoca que `example-app-2` cancele su intento de escritura y se genere una excepción de tiempo de espera. Esta situación de bloqueo impide el acceso de todas las aplicaciones a `example-dataset-1`.

Resolución

Para resolver la situación de inmediato, puede forzar la liberación del bloqueo. Para evitar que ocurra una situación similar en el futuro, puede configurar dos parámetros que controlan el mecanismo de reparación automática de Blusam.

Forzar la liberación del bloqueo

El administrador de bloqueos Blusam utiliza Amazon ElastiCache (Redis OSS) para proporcionar bloqueos compartidos entre aplicaciones. Para liberar los bloqueos ElastiCache, utilice la utilidad CLI de Redis. No puede eliminar bloqueos de registros individuales. Debe eliminar todos los bloqueos del conjunto de datos propietario. Realice los siguientes pasos:

1. Conéctese a su ElastiCache mediante el siguiente comando:

   ```
   redis-cli -h hostname -p port
   ```

   Puede encontrar sus detalles ElastiCache en la ElastiCache consola en https://console.aws.amazon.com/elasticache/.
2. Introduzca su contraseña.
3. Introduzca el comando que desee ejecutar, de la siguiente manera:

Comando	Finalidad
`KEYS *`	Obtener todas las claves existentes.

Comando	Finalidad
LLAVES * *YOUR_DATASET_NAME*	Obtener una clave de bloqueo de conjunto de datos.
DEL *THE_RETURNED_KEY*	Eliminar un bloqueo de conjunto de datos.
FLUSHDB	Limpiar todo el Redis. **Warning** Se perderán todos los datos de la memoria caché de Redis. Si el Redis se utiliza para otros fines, como gestionar sesiones http, puede que no desee utilizar `FLUSHDB`.

Configuración del mecanismo de reparación automática de Blusam

El administrador de bloqueos de Blusam incluye un mecanismo de reparación automática para evitar bloqueos en conjuntos de datos o registros. Puede ajustar los siguientes parámetros en la definición de la aplicación (`application-main.yml`) para configurar el mecanismo de reparación automática:

- `locksDeadTime`: se refiere al tiempo máximo que una aplicación puede mantener un bloqueo. Cuando pasa este tiempo, el bloqueo se declara caducado y se libera inmediatamente. El valor `locksDeadTime` se expresa en milisegundos y el valor predeterminado es 1000.
- `locksCheck`: define la estrategia de administrador de bloqueos de Blusam para comprobar los bloqueos. Todos los bloqueos de Blusam ElastiCache tienen una marca de tiempo y una fecha de caducidad. El valor del parámetro `locksCheck` determina si se eliminan los bloqueos caducados.
  - `off`: no se ejecuta ninguna comprobación en ningún momento. Se pueden producir puntos muertos. (No recomendado)
  - `reboot`: las comprobaciones se ejecutan cuando se inicia o se reinicia una instancia de aplicación de modernización de AWS mainframe que se ejecuta en un entorno de ejecución de modernización de AWS mainframe. Todos los bloqueos caducados se liberan inmediatamente. (predeterminado)

- `timeout`: las comprobaciones se ejecutan cuando se inicia o reinicia una AWS instancia de aplicación de modernización de AWS mainframe que se ejecuta en un entorno de ejecución de modernización de mainframe, o cuando se agota el tiempo de espera durante un intento de bloquear un conjunto de datos. Los bloqueos caducados se liberan inmediatamente.

Para obtener más información sobre la definición de una aplicación de AWS Blu Age, consulte. AWS Ejemplo de definición de aplicación de Blu Age

Administrador de bloqueos de Blusam

En el contexto de un entorno de ejecución de modernización de AWS mainframe que utiliza el patrón de alta disponibilidad, una aplicación de AWS Blu Age puede implementarse varias veces. En el caso de las aplicaciones que gestionan conjuntos de datos de Blusam, pueden producirse problemas de acceso simultáneo. El administrador de bloqueos Blusam garantiza la integridad de los datos y gestiona el acceso de lectura y escritura a los registros y conjuntos de datos al proporcionar bloqueos compartidos entre las aplicaciones que utilizan. ElastiCache Este mecanismo permite que más de una aplicación lea el registro de manera concurrente y garantiza que solo una aplicación escriba el registro simultáneamente.

Bloqueos de escritura

Para actualizar o eliminar un registro específico, la aplicación debe bloquear primero el conjunto de datos propietario del registro y, a continuación, bloquear el propio registro. Cuando el registro está bloqueado, se libera el bloqueo del conjunto de datos y quedan disponibles para su uso otros registros del mismo conjunto de datos. Cuando se completa la operación de actualización o eliminación, se libera el bloqueo de registro retenido. Solo una aplicación a la vez puede actualizar el registro, lo que impide que otras aplicaciones lean o escriban hasta que se libere el bloqueo, si la política de aplicación definida permite esperar a que se libere.

Bloqueos de lectura

Mientras no se mantenga ningún bloqueo de escritura en el registro o el conjunto de datos, varias aplicaciones pueden leer los mismos registros al mismo tiempo. Para bloquear un registro para una operación de escritura, se deben liberar todos los bloqueos de lectura.

> **Note**
>
> El gestor de bloqueos de Blusam gestiona el acceso desde varios subprocesos de una aplicación determinada mediante el mismo mecanismo de bloqueo.

Troubleshooting error: Cannot access an application URL

En esta página se describe cómo resolver un error cuando no puede acceder a la URL de una aplicación de AWS Mainframe Modernization en ejecución.

- Motor: AWS Blu Age y Rocket Software (anteriormente Micro Focus)
- Componentes: aplicaciones

Si no puede acceder a la URL de una aplicación de modernización de AWS mainframe en ejecución que creó e implementó en un entorno de ejecución de AWS Mainframe Modernization, es posible que deba configurar las reglas de entrada en el grupo de seguridad que asoció al entorno de ejecución.

Causa habitual

Al crear un entorno de tiempo de ejecución, el grupo de seguridad que proporcione, incluido el grupo de seguridad predeterminado, debe tener reglas de entrada configuradas para permitir el tráfico a las aplicaciones implementadas desde fuera de la VPC, si desea permitir este tipo de acceso.

Resolución

Compruebe si el grupo de seguridad de Amazon VPC asociado al entorno de tiempo de ejecución permite el tráfico al entorno en los puertos de aplicación correspondientes. Para comprobar las reglas de grupo de seguridad, siga estos pasos:

1. Abra la consola de modernización de AWS mainframe en. https://console.aws.amazon.com/m2/
2. En el panel de navegación izquierdo, seleccione Entornos.
3. Haga clic en el entorno de tiempo de ejecución que aloja la aplicación a la que desea conectarse.
4. Elija Configuración.
5. En Seguridad y red, elija el grupo de seguridad. El enlace abre los detalles del grupo de seguridad en la consola de Amazon VPC.

6. Si es necesario, elija Edición de reglas entrantes y añada la siguiente regla si aún no está presente:

   Tipo

   TCP personalizada

   Puerto

   8196 o el puerto que coincida con las propiedades del oyente especificadas en la definición de la aplicación. Para obtener más información, consulte Paso 2: Crear la definición de aplicación.

   Origen

   La dirección IP desde la que se llama a la aplicación. Puede elegir myIP en el menú desplegable. Si sigue teniendo problemas con el tiempo de espera, pruebe a elegir Anywhere IPV4 o Anywhere. IPV6 Asegúrese de detener la aplicación y volver a iniciarla después de añadir la regla de entrada al grupo de seguridad.

   Para obtener más información, consulte Trabajar con reglas de grupos de seguridad en la Guía del usuario de Amazon VPC.

Solución de problemas: AWS Blu Insights no se abre desde la consola

En esta página se describe cómo solucionar el problema de que la página de Blu Insights no se abra desde la consola de modernización de AWS mainframe.

- Motor: AWS Blu Age
- Componente: Blu Insights

Cuando intenta acceder a Blu Insights desde la consola de modernización de AWS mainframe, no se abre y la nueva pestaña se cierra inmediatamente.

Causa habitual

El rol que está utilizando para acceder a Blu Insights no tiene permisos suficientes.

Resolución

Adjunte una política de IAM al rol para permitirle acceder a Blu Insights. Asegúrese de que la política incluya los siguientes permisos.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "m2:GetSignedBluinsightsUrl"
            ],
            "Resource": "*"
            }
    ]
}
```

Asegúrese de sustituir `region` y por `account` el Región de AWS y Cuenta de AWS correcto.

Troubleshooting error: Environment unhealthy

En esta página, se describe cómo resolver un error cuando recibe una notificación de que uno de sus entornos de modernización del AWS mainframe no funciona correctamente.

- Motor: AWS Blu Age y Rocket Software (anteriormente Micro Focus)
- Componente: entornos

Si recibe una notificación en la que se indica que uno de sus entornos de modernización de AWS mainframe no funciona correctamente, es su caso. Se le notifica a través de uno de estos orígenes:

- El estado del entorno en mal estado se muestra en la consola de modernización AWS del mainframe.
- Notificación por correo electrónico sobre el estado del entorno en mal estado de AWS Health.
- Puede ver un evento relacionado con AWS Mainframe Modernization en su AWS Health panel de control, en el estado de su cuenta.

Causa habitual

El error se produce cuando no se puede acceder a los recursos de su AWS cuenta asociados al entorno de modernización del AWS mainframe. Un motivo habitual para este problema es que los recursos relacionados con el entorno se están modificando o eliminando.

Resolución

Para obtener orientación específica, utilice el código de error que se proporciona en el correo electrónico de AWS Health o a través de la consola de AWS Mainframe Modernization.

Código de error:

- Storage unreachable

Este error indica que el almacenamiento adjunto (Amazon Elastic File System o Amazon FSx file systems) del entorno no se ha montado correctamente. Para comprobar los detalles de un entorno en mal estado, siga estos pasos:

1. Abra la consola de modernización AWS de mainframe en https://console.aws.amazon.com/m2/.
2. Seleccione el entorno en mal estado y elija Configuración.
3. Elija Almacenamiento adjunto para ver los recursos de almacenamiento asociados a este entorno.
4. Compruebe las configuraciones relacionadas con la red, como el grupo de seguridad, la subred y Amazon VPC asociados al almacenamiento. Si estas configuraciones son incorrectas, intente restaurarlas para solucionar el problema.

> **Note**
>
> Si se ha eliminado el almacenamiento, no se podrá recuperar el entorno. En este caso, debería plantearse la posibilidad de eliminar el entorno en mal estado.

Solución de problemas de licencia para Rocket Software (anteriormente Micro Focus)

En esta página se describe cómo resolver los problemas de licencia con el motor Rocket Software Runtime

- Motor: Rocket Software
- Componente: Amazon EC2

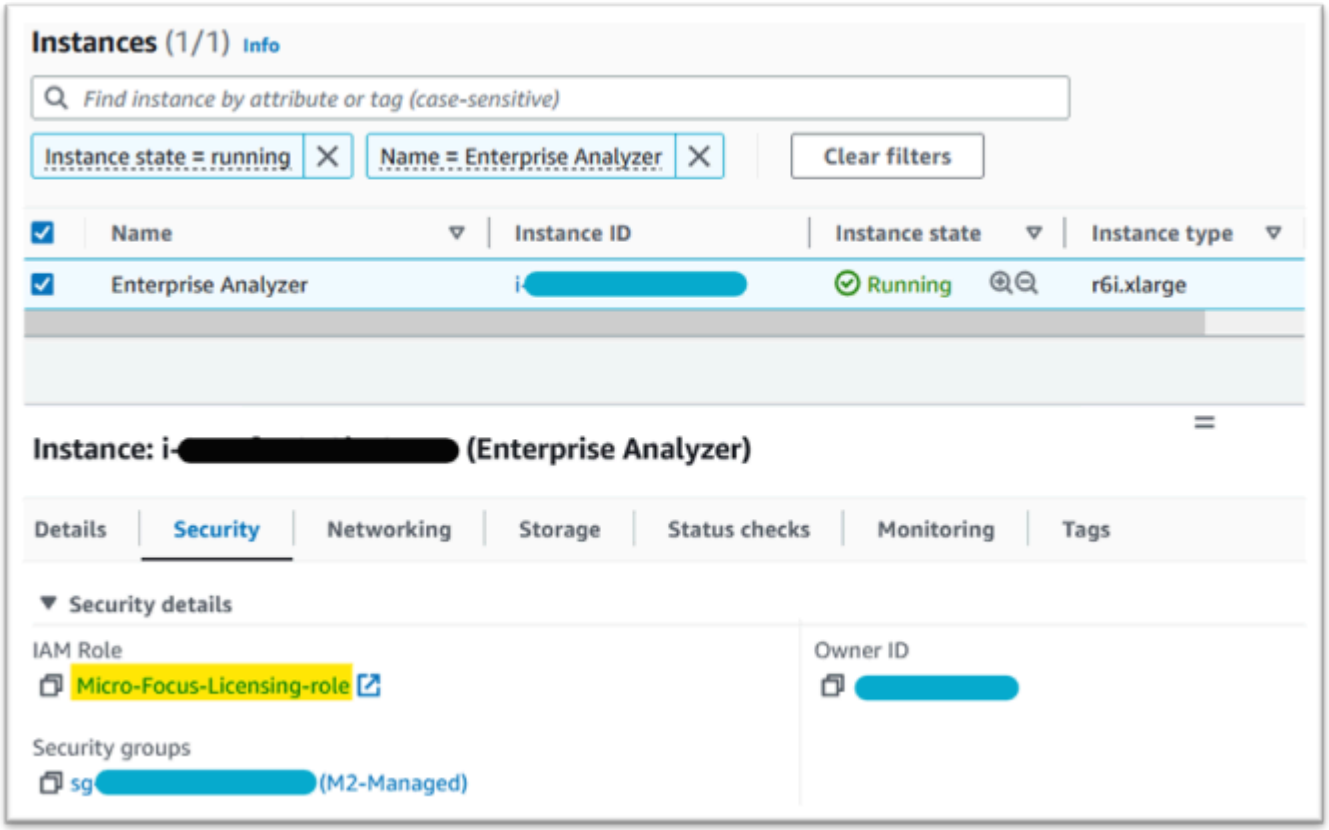
Si tiene problemas para acceder o utilizar el AMIs, la siguiente información puede ayudarle.

Temas

- Verifica que la EC2 instancia de Amazon tenga la función de licencia de IAM
- Uso del Analizador de accesibilidad
- Ejecute el daemon de licencias
- Problemas de licencia con Enterprise Server o las herramientas de compilación empresarial en Linux después de aplicar parches al sistema operativo

Verifica que la EC2 instancia de Amazon tenga la función de licencia de IAM

Esto se puede comprobar en la pestaña Seguridad de los detalles de la EC2 instancia de Amazon. Se puede cambiar mediante la opción de seguridad del menú desplegable Acciones.



Uso del Analizador de accesibilidad

Busque el Reachability Analyzer en la página de la consola. AWS Network Manager

Cree y analice una ruta entre la EC2 instancia de Amazon creada a partir de la AMI y el punto de enlace de VPC de Amazon S3.

Si la EC2 instancia de Amazon no tiene acceso a Internet, repita el análisis de la ruta en los 4 puntos finales.

Para obtener más información sobre el Analizador de accesibilidad, consulte Getting started with Reachability Analyzer en la guía del Analizador de accesibilidad.

Ejecute el daemon de licencias

En Windows Enterprise Developer, utilice el siguiente comando desde una línea de comandos:

```
“C:\Program Files (x86)\Micro Focus\Enterprise Developer\AdoptOpenJDK\bin\java” -jar
 "C:\Program Files (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

y examine la salida. Ignore los mensajes SLF4 J y busque la primera excepción.

En Enterprise Analyzer, utilice el siguiente comando desde una línea de comandos:

```
"C:\Program Files (x86)\Micro Focus\AdoptOpenJDK\bin\java" -jar "C:\Program Files
 (x86)\Micro Focus\Licensing\aws-license-daemon.jar"
```

y examine la salida. Ignore los mensajes SLF4 J y busque la primera excepción.

En Linux, ejecute .

```
java -jar /var/microfocuslicensing/bin/aws-license-daemon.jar
```

Ignore los mensajes SLF4 J y busque la primera excepción.

Por ejemplo, si el recurso de Amazon S3 no está disponible, la excepción es la siguiente:

```
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
```

```
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.

Exception in thread "main" software.amazon.awssdk.services.s3.model.S3Exception: Access
 Denied (Service: S3, Status Code: 403, Request ID: P6
```

El mensaje de excepción indica qué recurso no está disponible. Compare los valores de configuración con los que se muestran en este tema.

Problemas de licencia con Enterprise Server o las herramientas de compilación empresarial en Linux después de aplicar parches al sistema operativo

Si tiene problemas de licencia con Enterprise Server o Enterprise Build Tools en Linux después de aplicar los parches al sistema operativo, descargue y ejecute un script de parche para actualizar el daemon de licencias. Para ello, use los siguientes comandos en el símbolo del sistema:

```
sudo curl https://d148y999krizvm.cloudfront.net/patch/v8/linux/patch.sh -o /var/
microfocuslicensing/bin/patch.sh
sudo chmod +x /var/microfocuslicensing/bin/patch.sh
sudo /var/microfocuslicensing/bin/patch.sh
sudo ./startmfcesd.sh
```

> **Note**
>
> Este script de parche también funcionará con la versión 9, incluso si la ruta de descarga es para la versión 8.

Historial de documentos de la Guía del usuario de modernización de AWS mainframe

En la siguiente tabla se describen las versiones de la documentación sobre la modernización de AWS mainframes.

Cambio	Descripción	Fecha
AWS Notas de la versión 4.7.0 y 4.8.0 de Blu Age	La versión 4.7.0 de los motores AWS Blu Age Runtime y AWS Blu Age Transformation incluye actualizaciones de dependenc ia principales para Spring Boot 3.4.2 y Angular 19, que mejoran la seguridad y el rendimiento y, al mismo tiempo, modernizan el marco de aplicaciones frontend. La versión 4.8.0 incluye funciones clave, como la compatibilidad con la transformación de bases de datos IMS de IBM y una nueva función de caché en memoria que mejora el rendimiento de los programas con uso intensivo de datos para la modernización 00. AS4	18 de mayo de 2025
AWS Notas de la versión 4.6.0 de Blu Age	Esta versión del motor de transformación AWS Blu Age Runtime y AWS Blu Age se centra en las nuevas funciones y mejoras para zOS y AS4 00.	24 de enero de 2025

AWS Notas de la versión 4.5.0 de Blu Age	Esta versión de AWS Blu Age Runtime y del motor de transformación AWS Blu Age se centra en las caracterí sticas clave del soporte para JCL, el soporte de directorios y grupos de activación para las aplicaciones modernizadas del AS/400 y las dependencias actualizadas.	20 de diciembre de 2024
AWS Preguntas frecuentes sobre Blue Age	Conozca la capacidad de refactorización de AWS Blu Age con esta lista completa de. FAQs	20 de diciembre de 2024
Transformación de Amazon Q Developer para mainframe	Puede obtener más informaci ón sobre la función de transformación para mainframe de Amazon Q Developer, que le permite modernizar sus aplicacio nes de mainframe COBOL antiguas para convertirlas en aplicaciones Java.	2 de diciembre de 2024
AWS Notas de la versión 4.4.0 de Blu Age	Esta versión de los motores de ejecución y transformación de AWS Blu Age se centra en actualizar las dependenc ias críticas y las tecnologías compatibles, al tiempo que se mejora el rendimiento en múltiples funcionalidades.	13 de noviembre de 2024

Implemente AWS Blu Age Runtime en contenedores	Puede aprender a configura r AWS Blu Age Runtime en contenedores para implement arlo en Amazon ECS (gestiona do por Amazon EC2 o AWS Fargate) y Amazon EKS gestionado por Amazon EC2.	28 de octubre de 2024
Credenciales de usuario de LDAP actualizadas	Ahora puede crear y administr ar las credenciales de usuario de LDAP para la autenticación y la autorización mediante la AWS consola o AWS CLI (o el SDK).	21 de octubre de 2024
Configure la aplicación gestionada por Rocket Software	Ahora puede configurar sus aplicaciones con el motor de ejecución de Rocket Software para personalizar propiedad es adicionales, incluidas las integraciones.	21 de octubre de 2024
Cambio de plataforma con NTT DATA Unikix Runtime	Obtenga información sobre cómo usar Amazon Machine Images (AMIs) para crear un entorno personalizado para realojar y cambiar la plataforma de las aplicaciones de mainframe AWS mediante NTT DATA.	25 de septiembre de 2024

AWS Modernización de mainframes y pruebas de aplicaciones (IAM)	Puede obtener información sobre cómo administrar el acceso a las pruebas de aplicaciones de modernización de AWS mainframes con las funciones y políticas de IAM disponibles.	20 de septiembre de 2024
AWS Lanzamientos de Blu Age	En esta sección de la guía del usuario sobre la modernización de ordenadores AWS centrales se incluyen todos los detalles sobre las versiones de AWS Blu Age, las notas de lanzamiento de AWS Blu Age, las instrucciones de actualización de AWS Blu Age y el ciclo de vida general de AWS Blu Age.	16 de septiembre de 2024
AWS Ciclo de vida de los componentes de modernización de ordenadores	Esta página muestra el ciclo de vida de cada componente de modernización del AWS mainframe, incluidas sus actualizaciones de versión, el plan general de lanzamiento y los planes de fin del soporte y retirada.	5 de septiembre de 2024

Conversión de ensambladores con mLogica	AWS La conversión del código de modernización del mainframe con MLogica es una función de moderniza ción del AWS mainframe que convierte automáticamente el código del ensamblador de mainframe de z/OS en COBOL.	22 de julio de 2024
Versión de GA de Prueba de aplicaciones	Documentos de disponibi lidad general para las pruebas de aplicaciones. AWS Las pruebas de aplicacio nes de modernización de mainframe proporcionan pruebas automatizadas de equivalencia funcional para sus proyectos de migración . Esta versión incluye una página de protección de datos, flujos de trabajo de consola y actualizaciones de otras páginas de documentos desde la versión preliminar.	12 de junio de 2024
Tutorial actualizado sobre Managed Runtime for Rocket Software	Este tutorial muestra cómo implementar y ejecutar la aplicación de CardDemo ejemplo en un entorno de tiempo de ejecución gestionad o por la modernización de AWS Mainframe con el motor de tiempo de ejecución de Rocket Software.	5 de febrero de 2024

Notas de publicación de la versión 3.9.0 de AWS Blu Age Runtime and Modernization Tools.	Esta versión de AWS Blu Age Runtime and Modernization Tools se centra en múltiples mejoras transversales en todo el producto, con el objetivo de aumentar el rendimien to en arquitecturas de alta disponibilidad, junto con nuevas capacidades para llevar la ejecución de los trabajos al siguiente nivel.	18 de diciembre de 2023
Transfiera archivos entre el mainframe y AWS	Se ha lanzado una nueva función para transferir archivos desde el mainframe de origen a AWS.	27 de noviembre de 2023
Gestión de las transacciones de las aplicaciones	Se ha lanzado una nueva característica para mostrar y editar las transacciones de las aplicaciones de AWS Mainframe Modernization.	16 de octubre de 2023

Notas de publicación de la versión 3.6.0 de AWS Blu Age Runtime and Modernization Tools.	Esta versión de las herramientas de modernización y tiempo de ejecución de AWS Blu Age incluye nuevas funciones tanto para las migraciones de zOS como para las migraciones anteriores, orientadas principalmente a ampliar los mecanismos de soporte del CICS, complementar las capacidades de JCL, optimizar el rendimiento de las funciones simultáneas y de gran volumen y añadir funciones. AS4 multi-data-source	4 de agosto de 2023
Ahora puede implementar una nueva versión de una aplicación cuando la aplicación esté detenida.	Anteriormente, para implementar una nueva versión de una aplicación, tenía que eliminar la versión implementada. Ahora puede simplemente detener la versión implementada e implementar una nueva versión.	26 de julio de 2023
AWS Paquete de ejecución de Blu Age para facilitar la EC2 implementación de Amazon	AWS La modernización del mainframe con el tiempo de ejecución de AWS Blu Age ahora está disponible con más flexibilidad para configurar la pila completa y el despliegue en las EC2 instancias de Amazon de su Cuenta de AWS empresa.	6 de julio de 2023

Inicio de sesión único en AWS Blu Age Blu Insights.	AWS Blu Age Blu Insights está disponible a AWS Management Console través del inicio de sesión único.	31 de marzo de 2023
Versión de disponibilidad general	Versión general de la guía del usuario de modernización de AWS mainframes.	8 de junio de 2022
Versión inicial	Versión inicial (versión preliminar pública) de la Guía del usuario de modernización de AWS mainframes.	30 de noviembre de 2021

Las traducciones son generadas a través de traducción automática. En caso de conflicto entre la traducción y la version original de inglés, prevalecerá la version en inglés.