Agentäre KI-Muster und Workflows auf AWS

AWS Präskriptive Leitlinien

AWS Präskriptive Leitlinien: Agentäre KI-Muster und Workflows auf AWS

Table of Contents

Agentische KI-Muster und Workflows auf AWS

Aaron Sempf und Andrew Hooker, Amazon Web Services

Juli 2025 (Geschichte der Dokumente)

Organizations setzen umfangreiche Sprachmodelle (LLMs) und Softwareagenten ein, um dynamische, domänenübergreifende Probleme mithilfe einer neuen Architekturdisziplin, den sogenannten Agentenmustern, zu lösen. Agentenmuster sind grundlegende Baupläne und modulare Konstrukte, die verwendet werden, um zielgerichtete KI-Agenten in vielen Kontexten zu entwerfen und zu orchestrieren.

Zielgruppe

Dieser Leitfaden richtet sich an Architekten, Entwickler und Produktleiter, die intelligente Anwendungen entwickeln möchten, die über statische Logik, symbolische Logik und deterministische Automatisierung hinausgehen.

Ziele

Dieser Leitfaden bietet einen Entwurfsrahmen und einen Implementierungsansatz für KI-Agentensysteme, die autonom arbeiten und gleichzeitig kontrollierbar bleiben und auf Ihre Ziele ausgerichtet sind. Es verbindet ereignisgesteuerte Architekturmuster mit verschiedenen agentischen Alternativen und zeigt, wie Sie mithilfe von Cloud-nativen Architekturen Agentensysteme für die Produktion aufbauen können. In diesem Leitfaden werden die folgenden Themen behandelt:

- Agentenmuster — Agentenmuster sind wiederverwendbare Entwurfsvorlagen, die die Struktur und das Verhalten einzelner Agenten beschreiben. Dazu gehören Agenten zur Argumentation, Agenten mit erweiterter Suchfunktion, Codierungsagenten, Sprachschnittstellen, Workflow-Orchestratoren und kollaborative Systeme mit mehreren Agenten. Jedes Muster veranschaulicht, wie Agenten wahrnehmen, argumentieren, handeln und lernen, und zwar anhand von Zuordnungen. AWS-Services
- LLM-Workflows — Workflows konzentrieren sich darauf, wie Agenten sie LLMs zum Argumentieren verwenden. Sie untersuchen Aufforderungsstrategien und Planungsmechanismen und skizzieren, wie LLMs sie nicht nur zur Textgenerierung, sondern auch zur Förderung strukturierter, interpretierbarer und zuverlässiger Verhaltensweisen innerhalb einer Agentenschleife eingesetzt werden.

- Agentische Workflow-Muster — Workflow-Muster beschreiben, wie mehrere Agenten, Tools und Umgebungen interagieren, um autonome Systeme zu bilden. Dazu gehören Muster für die Orchestrierung von Aufgaben, die Delegierung von Subagenten, die ereignisbasierte Koordination, Beobachtbarkeit und Kontrolle. Diese Aspekte fördern skalierbare, zusammensetzbare und überprüfbare KI-Architekturen.

Über diese Inhaltsserie

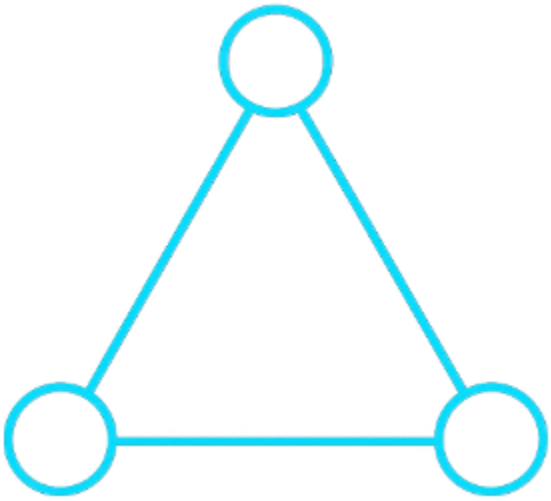
Dieser Leitfaden ist Teil einer Reihe über agentic AI on. AWS Weitere Informationen und die anderen Leitfäden dieser Reihe finden Sie unter Agentic AI auf der Prescriptive Guidance-Website. AWS

Agentenmuster

Agentenmuster sind wiederverwendbare, zusammensetzbare Bausteine, die auf bestimmte Domänen, Anwendungsfälle und Komplexitätsstufen zugeschnitten werden können. Agentensysteme unterscheiden sich jedoch von herkömmlichen Anwendungen. Im Mittelpunkt aller Designs von KI-Agenten steht ein konzeptionelles Modell, das in den folgenden drei Grundprinzipien verankert ist:

- Asynchron — Agenten arbeiten in lose gekoppelten, ereignisreichen Umgebungen
- Autonomie — Agenten agieren unabhängig, ohne menschliche oder externe Kontrolle
- Entscheidungsfreiheit — Agenten handeln zielgerichtet, im Namen eines Benutzers oder Systems, um bestimmte Ziele zu erreichen

Das Dreieck in der folgenden Abbildung stellt die Kernbausteine eines Softwareagenten dar: Wahrnehmung, Vernunft und Handlung. Dies ermöglicht es einem Agentensystem, seine Umgebung zu beobachten, Entscheidungen zu treffen und in ihr zu agieren.



Agentenmuster bieten von Natur aus eine modulare Designsprache für den Aufbau von KI-Systemen, was bedeutet, dass sie zugänglich, betriebsbereit, erweiterbar und produktionsbereit sind. Bei der Entwicklung dieser Systeme müssen die folgenden drei miteinander verbundenen Dimensionen sorgfältig beachtet werden, auf die weiter unten in diesem Leitfaden näher eingegangen wird.

In diesem Abschnitt

- Toolbasierte Agenten für Server
- Agenten, die Computer verwenden
- Codierungsagenten
- Sprach- und Sprachagenten
- Agenten für die Workflow-Orchestrierung
- Agenten mit erweitertem Speicher
- Agenten für Simulation und Testumgebung
- Beobachter- und Überwachungsagenten
- Zusammenarbeit mit mehreren Agenten

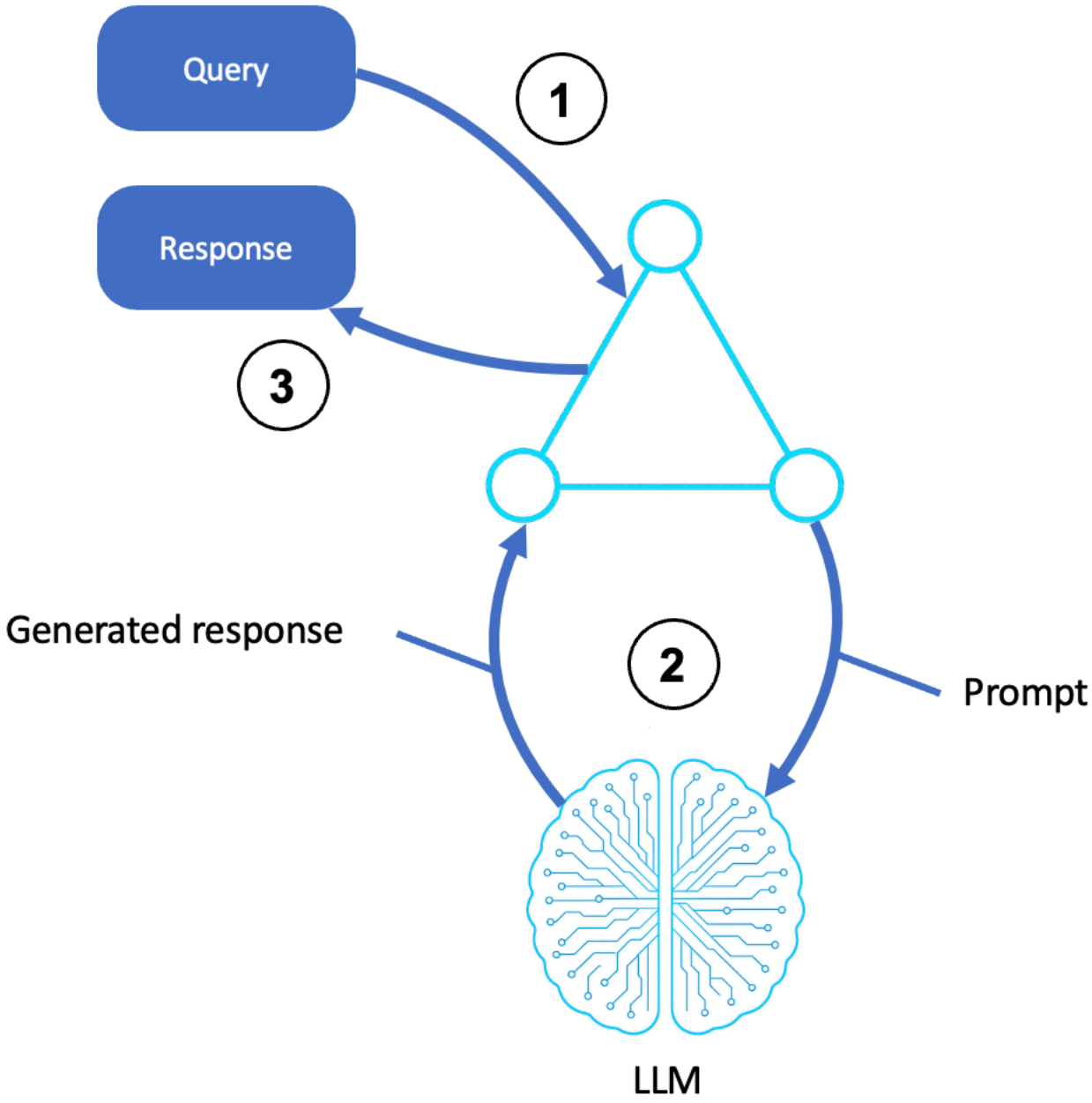
Grundlagen der Argumentation

Ein Agent für grundlegendes Denken ist die einfachste Form der agentischen KI, die als Antwort auf eine Anfrage logische Folgerungen oder Entscheidungen trifft. Es akzeptiert Eingaben von Benutzern oder Systemen und verarbeitet Abfragen und generiert Antworten mithilfe strukturierter Eingabeaufforderungen.

Dieses Muster ist nützlich für Aufgaben, die eine einstufige Argumentation, Klassifizierung oder Zusammenfassung auf der Grundlage eines bestimmten Kontextes erfordern. Es verwendet keinen Arbeitsspeicher, keine Tools oder Statusverwaltung, wodurch es zustandslos, leichtgewichtig und in großen Workflows gut zusammensetzbar ist.

Architektur

In der folgenden Abbildung wird der Ablauf eines Argumentationsmittels dargestellt:



Description

1. Empfängt eine Eingabe
   - Ein Benutzer, ein System oder ein Upstream-Agent sendet eine Anfrage oder Anweisung.
   - Die Eingabe wird an die Agenten-Shell oder die Orchestrierungsebene übergeben.
   - Dieser Schritt umfasst die Vorverarbeitung, die Erstellung von Vorlagen für Eingabeaufforderungen und die Identifizierung der Ziele.

2. Ruft das LLM auf
    - Der Agent wandelt die Anfrage in eine strukturierte Aufforderung um und sendet sie an ein LLM (z. B. über Amazon Bedrock).
    - Das LLM generiert auf der Grundlage der Aufforderung eine Antwort, wobei vorab geschultes Wissen und Kontext verwendet werden.
    - Die generierte Ausgabe kann Argumentationsschritte (chain-of-thought), endgültige Antworten oder Ranglisten enthalten.
3. Gibt eine Antwort zurück
    - Die generierte Ausgabe wird an die Schnittstelle des Agenten weitergeleitet.
    - Dies kann Formatierung, Nachbearbeitung oder eine API-Antwort beinhalten.

Capabilities

- Unterstützt natürliche Sprache oder strukturierte Eingabe
- Nutzt schnelle Technik, um das Verhalten zu steuern
- Zustandslos und skalierbar
- Kann in UI, CLI und Pipelines eingebettet werden APIs

Einschränkungen

- Kein Gedächtnis oder Geschichtsbewusstsein
- Keine Interaktion mit externen Tools oder Datenquellen
- Beschränkt auf das, was der LLM zum Zeitpunkt der Schlussfolgerung weiß

Häufige Anwendungsfälle

- Konversationsfragen und Antworten
- Erläuterungen und Zusammenfassungen der Richtlinien
- Leitlinien für die Entscheidungsfindung
- Leichte und automatisierte Chatbot-Abläufe
- Klassifizierung, Kennzeichnung und Bewertung

Implementierungsleitfaden

Sie können die folgenden Tools und Dienste verwenden, um ein grundlegendes Argumentationsinstrument zu erstellen:

- Amazon Bedrock für LLM-Aufrufe (Anthropic, Meta) AI21
- Amazon API Gateway oder AWS Lambda um es als zustandslosen Microservice verfügbar zu machen
- Rufen Sie Vorlagen auf, die im Parameter Store, AWS Secrets Manager, oder als Code gespeichert sind

Zusammenfassung

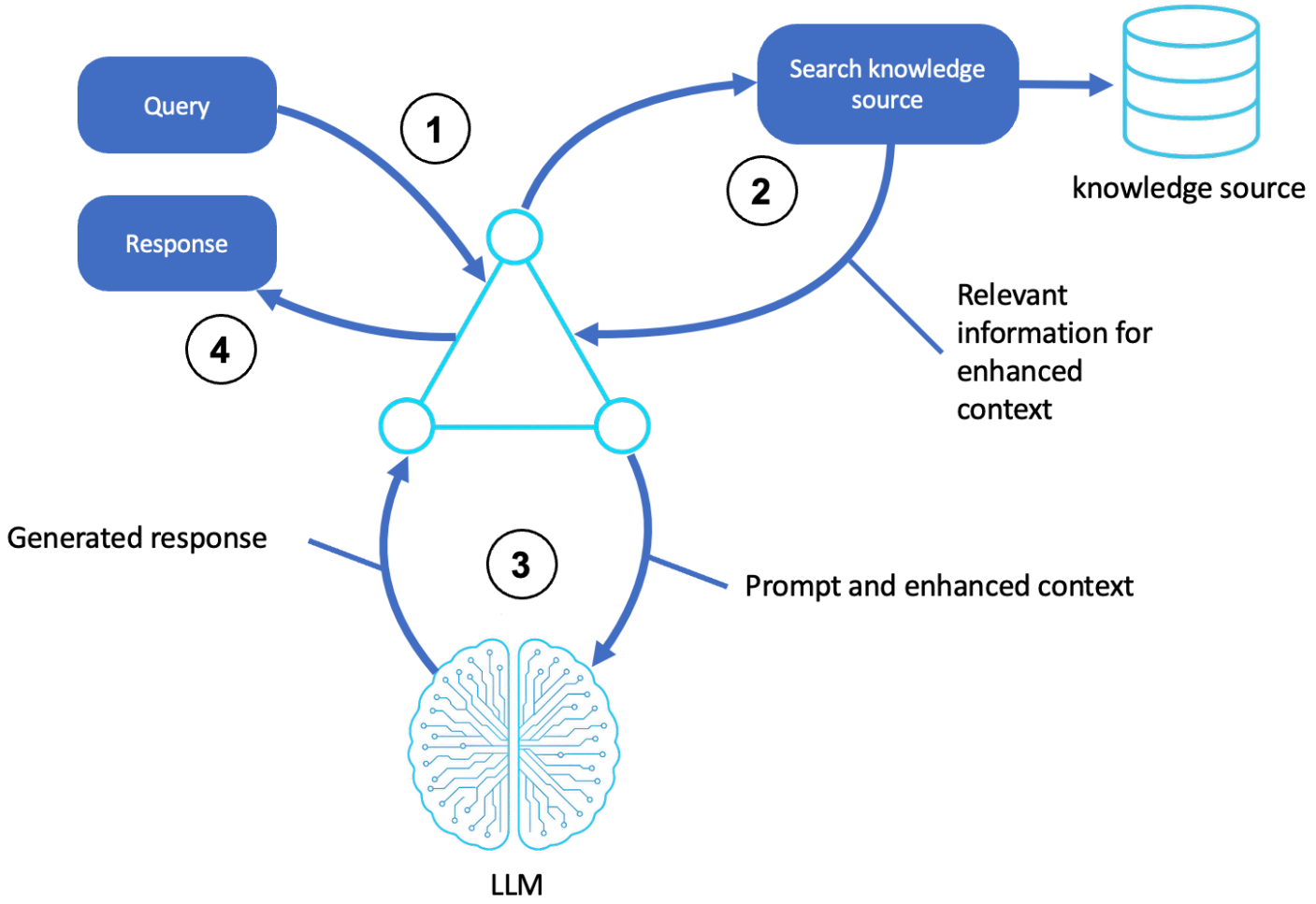
Das grundlegende Argumentationsmittel ist aufgrund seiner einfachen Struktur von grundlegender Bedeutung. Es verfügt über Kernfunktionen, die Ziele in Argumentationswege umwandeln, die zu intelligenten Ergebnissen führen. Dieses Muster ist häufig der Ausgangspunkt für fortgeschrittene Muster, wie z. B. toolbasierte Agenten und Agenten, die Retrieval-Augmented Generation (RAG) verwenden. Es ist auch eine zuverlässige und modulare Komponente großer Workflows.

Agentin RAG

Retrieval-Augmented Generation (RAG) ist eine Technik, die den Informationsabruf mit der Textgenerierung kombiniert, um präzise und kontextbezogene Antworten zu erhalten. RAG ermöglicht es Agenten, relevante externe Informationen abzurufen, bevor sie das LLM in Anspruch nehmen. Es erweitert das effektive Gedächtnis und die Argumentationsgenauigkeit eines Agenten, indem es seine Entscheidungen auf sachliche oder up-to-date domänenspezifische Informationen stützt. Im Gegensatz zu Stateless, LLMs die sich ausschließlich auf vortrainierte Gewichte verlassen, verfügt RAG über eine externe Wissenssuchebene, die Eingabeaufforderungen dynamisch an den Kontext anpasst.

Architektur

Die Logik des RAG-Musters wird in der folgenden Abbildung veranschaulicht:



Description

1. Empfängt eine Anfrage
   - Ein Benutzer oder ein Upstream-System sendet eine Anfrage oder ein Ziel an den Agenten.
   - Die Agenten-Shell akzeptiert die Anfrage und formatiert sie als Aufforderung zur Begründung.
2. Durchsucht eine externe Quelle
   - Der Agent identifiziert Konzepte und Absichten anhand der Abfrage.
   - Er fragt mithilfe der semantischen Suche oder des Stichwortabgleichs eine Wissensquelle ab, z. B. einen Vektorspeicher, eine Datenbank oder einen Dokumentenindex.
   - Die relevantesten Passagen, Dokumente oder Entitäten werden zur Verwendung im nächsten Schritt abgerufen.
3. Generiert eine kontextuelle Antwort

- Der Agent erweitert die Eingabeaufforderung um die abgerufenen Informationen und bildet so eine kontexterweiterte Eingabe für das LLM.
- Das LLM verarbeitet alle Eingaben mithilfe von generativem Denken (zum Beispiel chain-of-thought oder Reflexion), um eine genaue Antwort zu erhalten.

4. Gibt die endgültige Ausgabe zurück
   - Der Agent bereitet die Ausgabe vor, indem er sie in alle Kommunikationskopfzeilen oder erforderlichen Formatierungen verpackt und sie dann an den Benutzer oder das aufrufende System zurückgibt.
   - (Optional) Die abgerufenen Dokumente und die LLM-Ausgabe können protokolliert, bewertet und für future Abfragen im Speicher gespeichert werden.

Capabilities

- Faktengestützte Ausgabe, auch in Long-Tail- oder unternehmensspezifischen Bereichen
- Speichererweiterung ohne Feinabstimmung des Modells
- Dynamischer Kontext, der auf jeder Abfrage und jedem Benutzerstatus basiert
- Vollständig kompatibel mit Vektordatenbanken, semantischen Indizes und Metadatenfilterung

Häufige Anwendungsfälle

- Wissensassistenten für Unternehmen
- Bots zur Einhaltung gesetzlicher Vorschriften
- Copiloten für den Kundensupport
- Chatbots mit verbesserter Suchfunktion
- Agenten für die Dokumentation für Entwickler

Implementierungsleitfaden

Verwenden Sie die folgenden Tools und Dienste, um einen Agenten zu erstellen, der RAG verwendet:

- Amazon Bedrock für LLM-Aufruf

- Amazon Kendra oder Amazon Aurora für Dokumentation oder strukturierte Datensuche OpenSearch
- Amazon Simple Storage Service (Amazon S3) für die Aufbewahrung von Dokumenten
- AWS Lambda um Suche, Eingabeaufforderung und LLM-Inferenz zu orchestrieren
- Wissensbasierte Integrationen mit Agenten (mithilfe von Speicher-Plugins, semantischen Retrievern oder Amazon Bedrock)

Zusammenfassung

Agent RAG verbindet statische Modellargumentation mit dynamischer, realer Intelligenz. Es gibt Agenten die Möglichkeit, nachzuschlagen, was sie nicht wissen, Antworten aus abgerufenem Wissen zu synthetisieren und zuverlässige, überprüfbare Antworten zu erstellen.

RAG-Muster sind eine Grundlage für die Entwicklung intelligenter Agenten, die den Wissenszugang ohne Umschulung skalieren können. Es ist oft ein Vorläufer für komplexere Orchestrierungsmuster, die den Einsatz von Tools, die Planung und das Langzeitgedächtnis beinhalten.

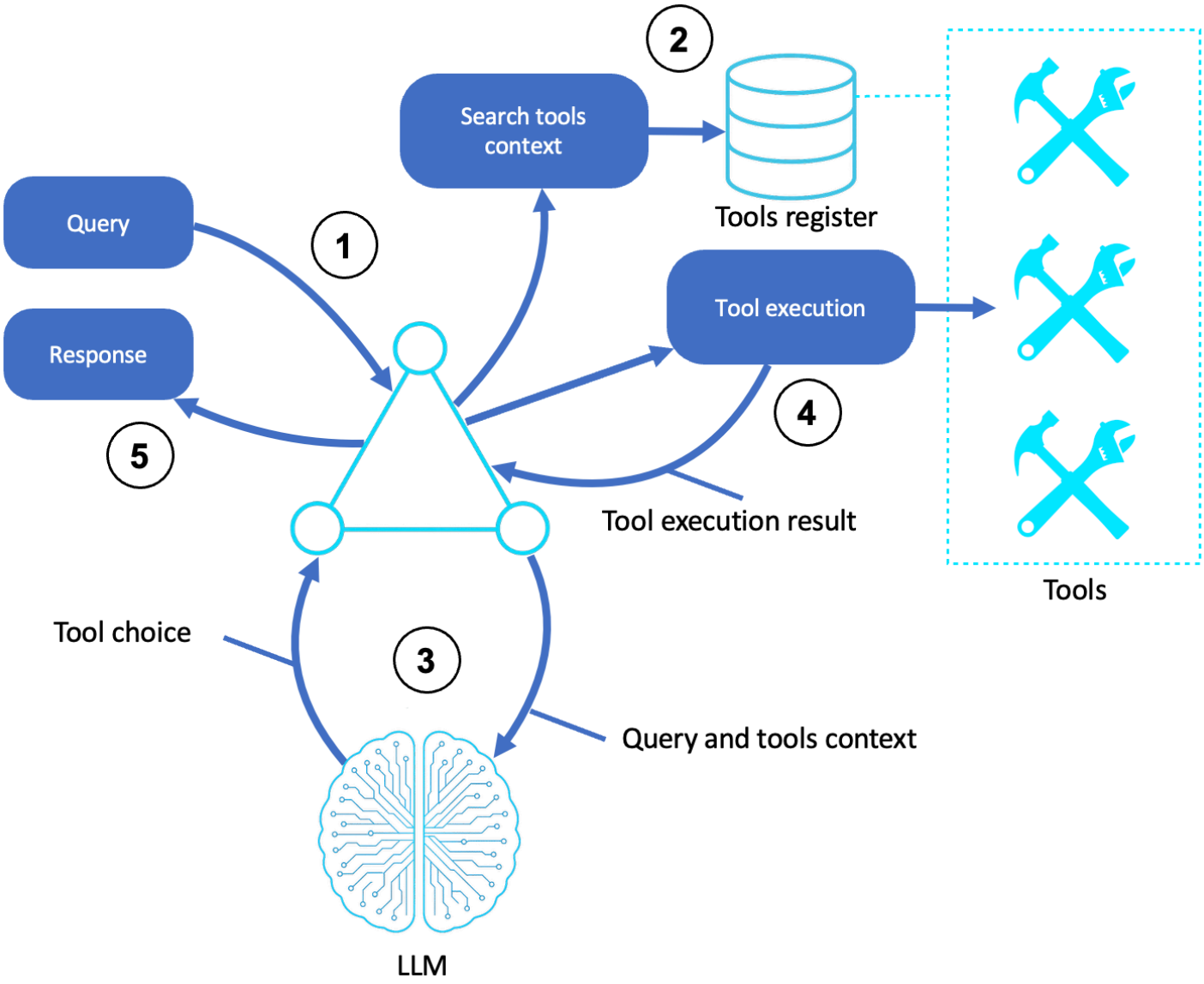
Toolbasierte Agenten zum Aufrufen von Funktionen

Toolbasierte Agenten erweitern die Fähigkeiten von Argumentationsagenten, indem sie externe Funktionen aufrufen oder Aufgaben erledigen, die über das rein APIs sprachliche Denken hinausgehen. Dieses Muster verwendet ein LLM, um zu entscheiden, welches Tool verwendet werden soll, generiert dann Anrufargumente und bezieht die Ergebnisse eines Tools in seine Argumentationsschleife ein.

Dieses Muster ermöglicht es den Agenten, zu handeln, anstatt nur Antworten zu geben. Die Tool-Schnittstelle steht für jede aufrufbare Funktion, von arithmetischen Berechnungen und Datenbankabfragen bis hin zu externen APIs Diensten und Cloud-Diensten.

Architektur

In der folgenden Abbildung ist ein toolbasierter Agent zum Aufrufen von Funktionen dargestellt:



Description

1. Empfängt eine Anfrage
   - Der Agent erhält eine Anfrage oder Aufgabe in natürlicher Sprache vom Benutzer oder vom aufrufenden System.
2. Sucht nach Tools
   - Der Agent verwendet interne Metadaten oder eine Tool-Registry, um nach verfügbaren Tools, Schemas und relevanten Funktionen zu suchen.
3. Wählt Tools aus und ruft sie auf

- Das LLM empfängt die Abfrage- und Tool-Metadaten (z. B. Funktionsnamen, Eingabetypen und Beschreibungen) in seiner Eingabeaufforderung.
- Es wählt das relevanteste Tool aus, konstruiert Eingabeargumente und gibt einen strukturierten Funktionsaufruf zurück.

4. Führt das gewählte Tool aus
   - Die Agent-Shell oder der Tool Runner führt die ausgewählte Funktion aus und gibt das Ergebnis zurück (z. B. eine API-Ausgabe, einen Datenbankwert oder eine Berechnung).
5. Gibt eine Antwort zurück
   - Das LLM leitet die Ergebnisse entweder direkt oder als Teil einer aktualisierten Aufforderung an den Agenten weiter. Anschließend wird ein Ergebnis in natürlicher Sprache zurückgegeben.

Capabilities

- Dynamische Werkzeugauswahl basierend auf dem Aufgabenkontext
- Schemabasierte Eingabeaufforderung (OpenAPI, JSON-Schema, Funktionsschnittstelle) AWS
- Interpretation der Ergebnisse und Verkettung der Ergebnisse zu Argumenten
- Zustandslose oder sitzungsabhängige Operationen

Häufige Anwendungsfälle

- Virtuelle Assistenten mit externem Datenzugriff
- Finanzrechner und Schätzer
- API-basierte Wissensarbeiter
- LLMs die SageMaker Amazon-Endpunkte und SaaS-Dienste aufrufen AWS Lambda

Implementierungsleitfaden

Verwenden Sie Folgendes, um toolbasierte Agenten zum Aufrufen von Funktionen zu erstellen:

- Amazon Bedrock mit Unterstützung für Funktionsaufrufe (Anthropic Claude)
- AWS Lambda als Backend für die Ausführung von Tools
- Amazon API Gateway oder AWS Step Functions für die Orchestrierung von Tools

- Amazon DynamoDB oder Amazon Relational Database Service (Amazon RDS) für kontextsensitive Tool-Metadaten
- EventBridge Amazon-Pipelines oder AWS Step Functions die Staaten zuordnen, um Ausgaben weiterzuleiten

Zusammenfassung

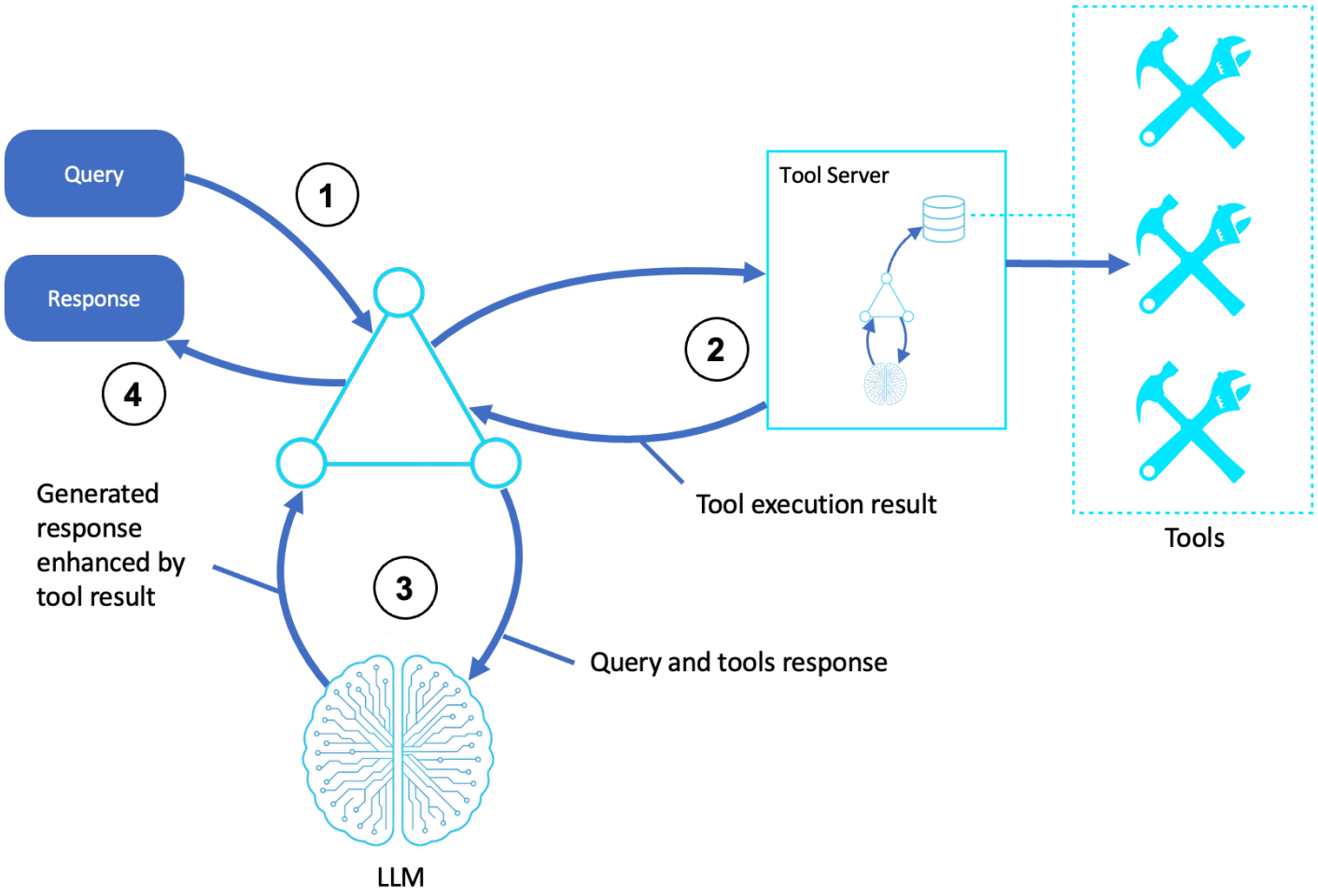
Agenten, die über Tools Funktionen aufrufen, bedeuten eine Verlagerung vom Sprachverständnis hin zur Ausführung von Aktionen. Diese Agenten nutzen dynamische, kontextsensitive Tools und behalten gleichzeitig die LLM-Argumentation bei. Sie verwandeln passive Assistenten in Systeme, die Aufgaben erledigen, auf Dienste zugreifen und Geschäftsabläufe integrieren. Dieses Muster ist ein wichtiger Bestandteil der agentischen KI in Unternehmensumgebungen, insbesondere in Kombination mit deklarativen Schemas, Autorisierungs-Frameworks und Systemen mit mehreren Agenten.

Toolbasierte Agenten für Server

Toolbasierte Agenten für Server verbessern die Funktion von Agenten, die Funktionen aufrufen, indem sie die Ausführung von Tools an einen externen Server delegieren, der über eine spezielle Laufzeitumgebung für Tools, Skripts und kombinierte Agenten verfügt. Im Gegensatz zu Inline-Funktionsaufrufen, die in der Agentenschleife ausgewählt und aufgerufen werden, lagern serverbasierte Agenten die Logik und die Ausführungspipeline an andere Agenten oder Systeme aus. Dies bietet erweiterte Funktionen wie Multitool-Verkettung, isolierte Ausführung und spezielle Argumentation. Toolserver eignen sich ideal für komplexe, zustandsbehaftete oder ressourcenintensive Aktionen, bei denen die Tools selbst separate KI-Modelle, Geschäftsregeln oder Umgebungen beinhalten können.

Architektur

Im Folgenden finden Sie ein Muster für toolbasierte Agenten für Server:



Description

1. Empfängt eine Anfrage
   - Ein Benutzer oder ein System sendet eine Anfrage an die Agenten-Shell.
   - Der Agent interpretiert die Abfrage und bereitet ihren Versand an einen Toolserver vor.
2. Führt Toolserver-Prozesse aus
   - Der Agent sendet die Aufgabe zusammen mit strukturierten Parametern an einen Toolserver.
   - Der Tool-Server kann dann:
     - Führen Sie Skripts oder Logik in speziellen Rechensystemen aus (AWS Lambda z. B. Container oder Amazon SageMaker)
     - Verwenden Sie einen eigenen Subagenten mit LLM-Argumentation, um Tools auszuwählen und auszuführen
     - Verwalten Sie Abhängigkeiten, Wiederholungsversuche oder mehrstufige Ausführungsabläufe

    - Gibt die Ergebnisse an den primären Agenten aus, wenn die Aufgabe abgeschlossen ist

3. Verwendet LLM-Argumentation bei der Toolausgabe
    - Der Agent ruft ein LLM auf und übergibt die ursprüngliche Abfrage und das Ergebnis des Toolservers als Teil der Eingabeaufforderung.
    - Das LLM synthetisiert eine Antwort, die die neu erfassten Informationen enthält.
4. Gibt eine Antwort zurück
    - Der Agent gibt eine Antwort in natürlicher Sprache oder in strukturierter Form an den Benutzer oder das anrufende System zurück.
    - (Optional) Die Ergebnisse können im Speicher oder in Auditprotokollen gespeichert werden.

Capabilities

- Tools werden außerhalb der Ausführungsschleife des primären Agenten aufgerufen
- Die Ausführung von Tools kann LLM-Aufrufe, Logikketten oder Subagenten beinhalten
- Der Agent fungiert als Controller oder Dispatcher, nicht nur als Tool-Wrapper
- Ermöglicht Zusammensetzbarkeit, Skalierbarkeit und Isolierung der Logik

Häufige Anwendungsfälle

- Orchestrierung von Modellketten (z. B. durch Kombination von LLM, Vision und Code)
- KI-gesteuerte Automatisierungspipelines
- DevOps Assistenzagenten mit Script-Runnern
- Agenten für komplexe Finanzberechnungen, Simulationen oder Optimierungen
- Multimodale Tools (z. B. durch die Kombination von Audio, Dokumentation und Aktion)

Implementierungsleitfaden

Sie können dieses Muster wie folgt AWS-Services erstellen:

- Amazon Bedrock (Agentenhost und LLM-Inferenz)
- AWS Lambda, Amazon ECS oder SageMaker Amazon-Endpoints als Tool-Server-Runtime AWS Fargate
- Amazon API Gateway oder AWS App Runner um den Tool-Server verfügbar zu machen APIs

- Amazon EventBridge für entkoppeltes Messaging agent-to-tool
- AWS Step Functions oder AWS AppFabric für die Erstellung von Multi-Agent-Logik auf dem Tool-Server

Zusammenfassung

Toolbasierte Agenten, die Server verwenden, sind hochgradig modular und skalierbar. Sie entkoppeln die Entscheidungslogik von der Ausführung, was es dem primären Agenten ermöglicht, schlank zu bleiben und gleichzeitig komplexe oder sensible Aktionen auf andere Systeme auszulagern. Dies ist wichtig für agentische KI auf Unternehmensebene, insbesondere in Umgebungen, in denen Steuerung, Beobachtbarkeit, Isolierung, dynamische Zusammensetzung oder eine beliebige Kombination davon erforderlich sind.

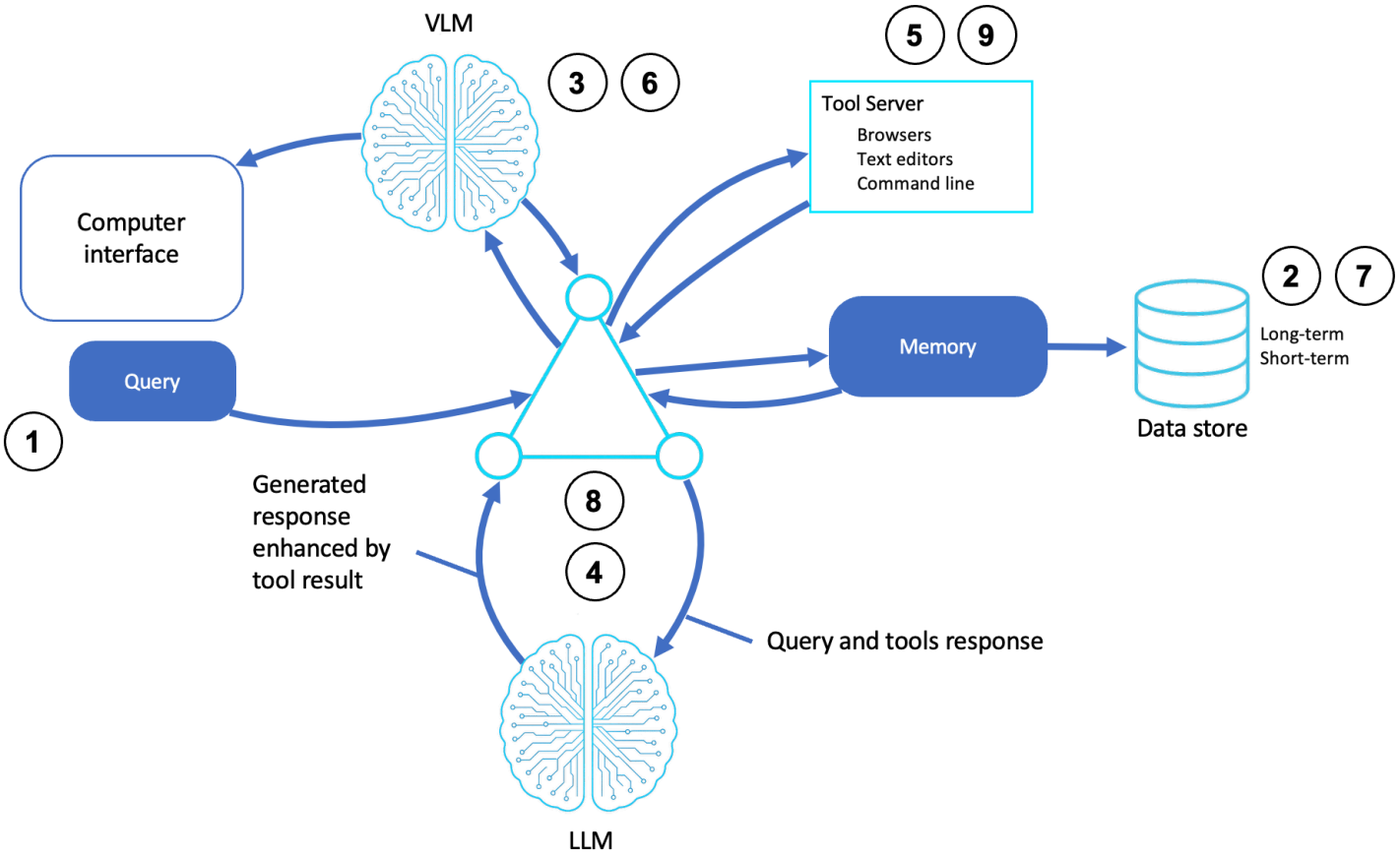
Agenten, die Computer verwenden

Computergestützte Agenten können digitale Umgebungen wie Browser, Terminals, Dateisysteme und Anwendungen simulieren oder steuern. Diese Agenten interpretieren Benutzerabsichten, interagieren mit visuellen und textuellen Benutzeroberflächen und führen zielgerichtete Aktionen durch, indem sie LLM-Argumentation, visuelle Sprachmodelle (VLMs) und Toolserver kombinieren, die Befehle ausführen oder Eingabeereignisse simulieren.

Dieses Muster ist wichtig für praktische KI-Automatisierungen, bei denen der Agent nicht nur als Assistent, sondern auch als Proxy fungiert, der Aktionen wie ein Mensch ausführt, häufig unter Verwendung derselben Tools und Umgebungen.

Architektur

In der folgenden Abbildung ist ein Muster für die Computernutzung durch Agenten dargestellt:



Description

1. Empfängt eine Anfrage
   - Eine Aufgabe oder Anfrage wird über eine Benutzeroberfläche, eine API oder eine Schnittstelle in natürlicher Sprache bereitgestellt.
2. Greift auf den Speicher zu
   - Der Agent ruft das Kurz- und Langzeitgedächtnis ab, um vergangene Befehle, Ziele und Systemzustände abzurufen.
3. Analysiert den visuellen Kontext
   - Ein VLM beobachtet den Computerbildschirm, den Systemstatus oder Elemente der Benutzeroberfläche, um einen bestimmten Kontext zu verstehen und umsetzbare Elemente zu identifizieren.
4. Gründe anhand eines LLM
   - Das LLM kombiniert die Abfrage, den Speicherstatus, das Tool und die Serverantwort, um die nächste Aktion zu bestimmen.

5. Interagiert mit dem Tool-Server
   - Der Agent ruft Tools auf, die auf einem Server gehostet werden. Dazu können folgende Tools gehören:
     - Browser (z. B. Headless Chrome) und Shell-Umgebungen
     - Text- und Code-Editoren
     - Benutzerdefinierte Skriptschnittstellen
6. Aktualisiert visuelle Eingaben
   - Wenn sich die Benutzeroberfläche des Systems ändert oder weitere Beobachtungen erforderlich sind, analysiert das VLM möglicherweise den Bildschirmstatus oder die Textpuffer erneut.
7. Aktualisiert den Speicher
   - Neue Erkenntnisse, Systemzustände oder Benutzerfeedback werden in das Kurz- und Langzeitgedächtnis geschrieben.
8. Formuliert endgültige Entscheidungen und Erklärungen
   - Das LLM fasst die Ergebnisse zusammen oder empfiehlt Maßnahmen auf der Grundlage der Abfrage und der Toolausgabe.
9. Gibt eine Antwort zurück
   - Der Agent gibt Ergebnisse an die Schnittstelle zurück (z. B. eine abgeschlossene Aufgabe, eine Bestätigung oder generierter Inhalt).

Capabilities

- Multimodales Denken mit visuellen und textuellen Eingaben
- Kontrolle über Anwendungen durch simulierte oder API-gesteuerte Eingaben
- Speicherverwaltung für persistenten Zustand
- Autonomie bei der Sequenzausführung (mehrstufige Abläufe)

Häufige Anwendungsfälle

- KI-Entwickler, die Code schreiben und ausführen in IDEs
- Computergestützte Agenten für sich wiederholende digitale Workflows
- Simulierte Benutzer für Softwaretests und Qualitätssicherung

- Agenten für Barrierefreiheit, mit denen Sie UIs mithilfe von Sprachanweisungen oder Anweisungen auf hoher Ebene navigieren können
- Intelligente robotergestützte Prozessautomatisierung (RPA), die durch logisches Denken erweitert wird

Implementierungsleitfaden

- Sie können dieses Muster wie folgt erstellen: AWS-Services
- Amazon Bedrock für LLM-basierte Planung und Argumentation
- Amazon Elastic Compute Cloud (Amazon EC2) oder SageMaker Amazon-Notebooks zum Ausführen von Toolservern mit simulierten UI-Umgebungen AWS Lambda
- Amazon Simple Storage Service (Amazon S3) oder Amazon DynamoDB für Speicherpersistenz
- Amazon Rekognition (oder benutzerdefinierte Modelle) für die UI-Bildanalyse in hybriden Szenarien
- Amazon CloudWatch Logs oder AWS X-Ray für Beobachtbarkeit und Prüfprotokolle

Zusammenfassung

Agenten, die Computer nutzen, agieren als autonome digitale Operatoren und überbrücken die Lücke zwischen Interaktionen zwischen Mensch und Computer und KI-gesteuerten Aktionen. Durch die Integration von Speicher, Werkzeugorchestrierung und können diese Agenten adaptiv mit Systemen interagieren VLMs, die für Menschen konzipiert sind, Aktionen ausführen, Dateien aktualisieren, in Menüs navigieren und Antworten generieren.

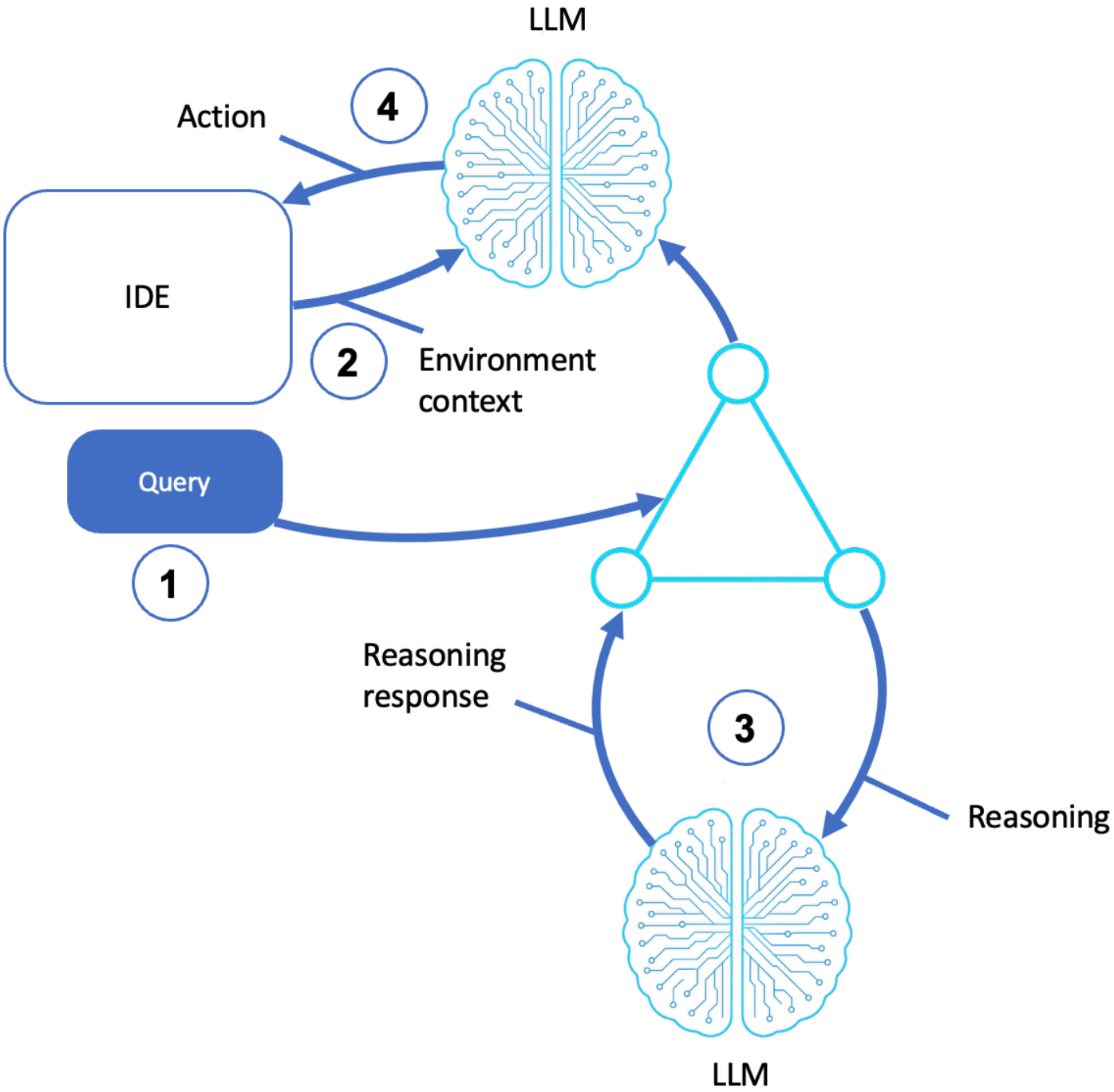
Codierungsagenten

Programmieragenten können über Programmieraufgaben nachdenken, Code generieren oder ändern und mit Entwicklerumgebungen wie IDEs und interagieren. CLIs Diese Agenten kombinieren natürliches Sprachverständnis mit strukturiertem Denken, um die Softwareentwicklung zu unterstützen, zu erweitern und zu automatisieren — von der Funktionsgenerierung über die Fehlerbehebung bis hin zur Erstellung von Tests.

Im Gegensatz zu Tools zur automatischen Vervollständigung interpretieren Programmieragenten aktiv die Ziele der Benutzer, fragen die Entwicklungsumgebung nach Kontext ab (sie öffnet beispielsweise Dateien und verfolgt Fehler), identifizieren Anforderungen und schlagen dann Aktionen vor und führen sie durch.

Architektur

Das folgende Diagramm zeigt ein Muster für Codierungsagenten:



Description

1. Empfängt eine Anfrage

- Der Benutzer stellt Anweisungen in natürlicher Sprache über eine Befehlspalette, ein Chat-Fenster oder eine CLI bereit (z. B. „Protokollierung zu dieser Funktion hinzufügen“ oder „Aus Gründen der Lesbarkeit umgestalten“).

2. Extrahiert den Umgebungskontext
   - Der Agent sammelt den Kontext aus der IDE, einschließlich aktiver Dateien, Cursorposition, Codefragmente und Symboltabellen.
   - Er gibt Fehlermeldungen, Testergebnisse und Ausgaben von anderen Agenten aus.
3. LLM-Argumentation
   - Der Agent sendet eine Aufforderung, einschließlich der Anfrage und des Umgebungskontextes, an einen LLM.
     - Das LLM führt einen Argumentationsprozess durch, um Folgendes festzustellen:
     - Was muss sich ändern
     - Wie generiert man eine Lösung
     - Alle Schritte zum Refactoring, Umschreiben oder Codieren
4. Führt Aktionen aus
   - Das LLM gibt die Ausgabe an den Agenten zurück und importiert sie in die IDE- oder Laufzeitumgebung.
   - Dies kann das Einfügen oder Ändern von Code, das Generieren von Kommentaren oder Dokumentation und das Auslösen nachgelagerter Build-, Test- und Lint-Aufgaben beinhalten.

Capabilities

- Hohe Kontextsensitivität (z. B. IDE-Status, Cursor und Syntaxbaum)
- Iterative Argumentation von Zielen und Feedback
- Optionale Codeplanung und Trennung von Aktionen (z. B. zuerst begründen und dann handeln)
- Funktioniert in synchronen oder asynchronen Entwickler-Workflows

Häufige Anwendungsfälle

- Codegenerierung aus Aufgabenbeschreibungen
- Code-Refactoring und Optimierung
- Generierung und Validierung von Testfällen

- Fehlererklärungen und Debugging
- Assistenten für die Dokumentation
- Gepaarte Programmier-Copiloten

Implementierungsleitfaden

- Sie können dieses Muster mit den folgenden Tools erstellen und: AWS-Services
- Amazon Bedrock für LLM-gestützte Generierung und Argumentation
- Amazon Q Developer für Vorschläge und Ergänzungen zur Codierung
- AWS Lambda oder Amazon Elastic Container Service (Amazon ECS) zum Ausführen und Testen von Sandbox-Umgebungen
- AWS Cloud9, VS Code-Erweiterungen oder benutzerdefinierte IDE-Integrationen zum Hosten und Auswerten von Kontext
- Amazon Simple Storage Service (Amazon S3) zum Speichern von Zwischenaufforderungen, Antworten und dem Versionsverlauf

Zusammenfassung

Coding Agents sind neue KI-gestützte Entwicklungstools, die in der Lage sind, natürliche Sprache zu interpretieren, den Kontext zu analysieren, mehrstufige Codeänderungen zu generieren und sich in den Softwareentwicklungszyklus zu integrieren.

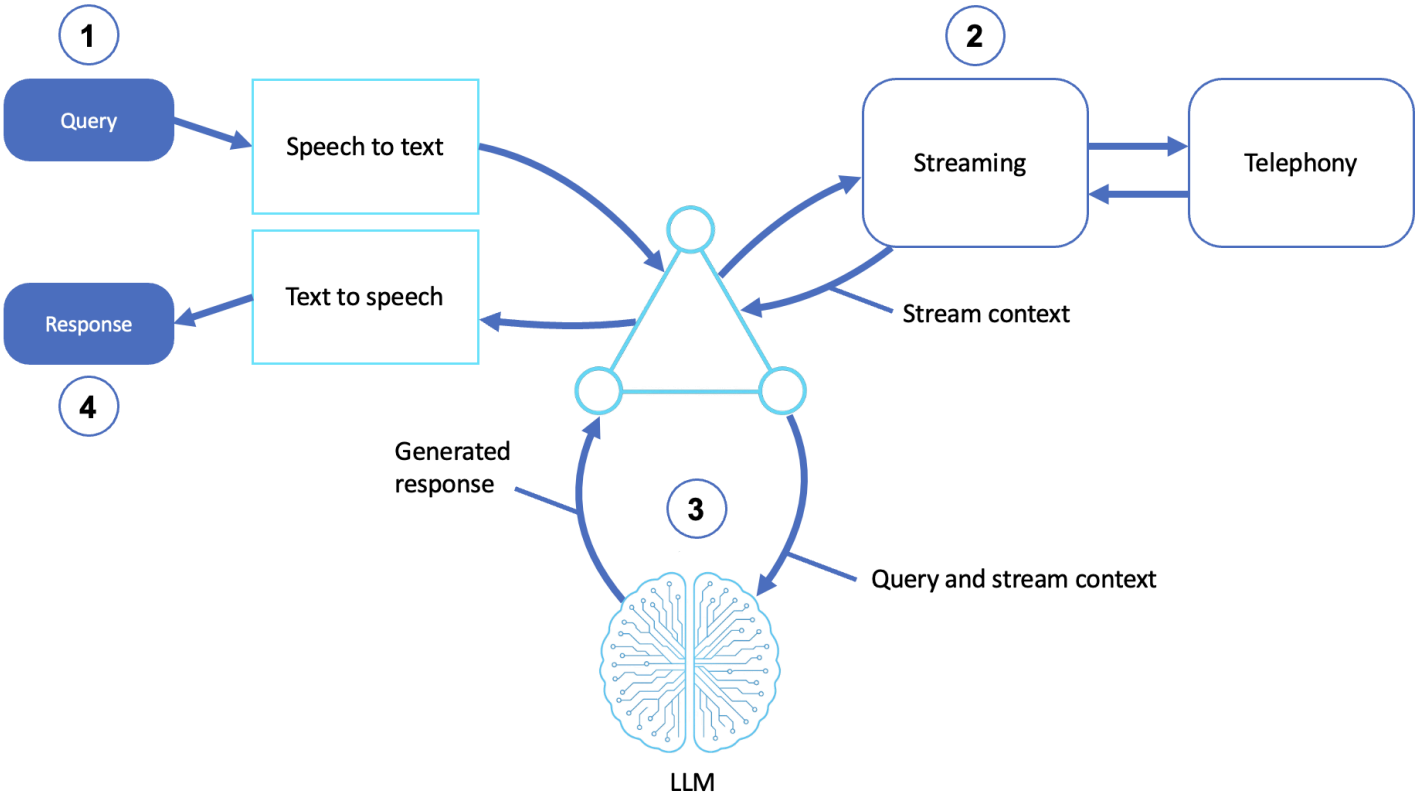
Sprach- und Sprachagenten

Sprach- und Sprachagenten interagieren mit Benutzern im Rahmen eines gesprochenen Dialogs. Diese Agenten integrieren Spracherkennung, natürliches Sprachverständnis und Sprachsynthese, um Konversations-KI auf Telefonie-, Mobil-, Web- und eingebetteten Plattformen zu ermöglichen.

Sprachagenten sind besonders effektiv in Umgebungen mit Freisprechfunktion, Echtzeit- oder Barrierefreiheit. Durch die Kombination von Streaming-Schnittstellen mit LLM-gestützter Argumentation ermöglichen sie umfangreiche, dynamische Interaktionen, die sich für Benutzer natürlich anfühlen.

Architektur

In der folgenden Abbildung ist ein Sprach- und Stimmagent dargestellt:



Description

1. Empfängt eine Sprachanfrage
   - Der Benutzer stellt eine Anfrage an ein Telefon, ein Mikrofon oder ein eingebettetes System.
   - Ein speech-to-text (STT-) Modul konvertiert das Audio in Text.
2. Integriert Streaming- und Telefoniekontext
   - Der Agent verwendet eine Streaming-Schnittstelle, um Audio I/O in Echtzeit zu verwalten.
   - Wenn sie in einem Contact Center oder im Telekommunikationskontext eingesetzt wird, übernimmt die Telefonieintegration das Sitzungsrouting, die zweifarbige Mehrfrequenzeingabe (DTMF) und den Medientransport.

Hinweis: DTMF bezieht sich auf die Töne, die erzeugt werden, wenn Sie Tasten auf einer Telefontastatur drücken. Im Zusammenhang mit der Integration von Streaming- und Telefoniekontexten innerhalb von Sprachagenten wird DTMF als Signaleingabemechanismus während eines Telefonanrufs verwendet, insbesondere in Interactive Voice Response (IVR) -Systemen. DTMF-Eingaben ermöglichen dem Agenten:

- Erkennen Sie Menüauswahlen (z. B. „Drücken Sie 1 für die Abrechnung. Drücken Sie 2, um Unterstützung zu erhalten.“)
- Sammeln Sie numerische Eingaben (z. B. Kontonummern und Bestätigungsnummern) PINs
- Lösen Sie Workflows oder Zustandsübergänge in Anrufabläufen aus
- Wechseln Sie bei Bedarf von der Sprach- zur Tonwiedergabe

1. Gründe dafür sind der LLM-Stream-Kontext
   - Die Abfrage wird an den Agenten gesendet, der sie zusammen mit allen Sitzungsmetadaten (z. B. Anrufer-ID, vorheriger Kontext) an ein LLM weiterleitet.
   - Das LLM generiert eine Antwort, möglicherweise mithilfe einer chain-of-thought Strategie oder eines Multiturn-Speichers, wenn die Interaktion andauert.
2. Gibt eine Sprachantwort zurück
   - Der Agent wandelt seine Antwort mithilfe von text-to-speech (TTS) in Sprache um.
   - Er gibt Audio über einen Sprachkanal an den Benutzer zurück.

Capabilities

- Sprachverständnis und Sprachgenerierung in Echtzeit
- Mehrsprachig I/O mit STT- und TTS-Unterstützung
- Integration mit Telefonie oder Streaming APIs
- Sitzungsbewusstsein und Gedächtnisübergabe zwischen den Runden

Häufige Anwendungsfälle

- IVR-Systeme für Konversationen
- Virtuelle Rezeptionisten und Terminplaner
- Sprachgesteuerte Helpdesk-Agenten
- Tragbare Sprachassistenten
- Sprachschnittstellen für Smart Homes und Tools zur Barrierefreiheit

Implementierungsleitfaden

Sie können dieses Muster mit den folgenden Tools erstellen und AWS-Services:

- Amazon Lex V2 oder Amazon Transcribe für STT
- Amazon Polly für TTS
- Amazon Chime SDK, Amazon Connect oder Amazon Interactive Video Service (Amazon IVS) für Streaming und Telefonie
- Amazon Bedrock zum Argumentieren mit anthropischen oder anderen AI21 Stiftungsmodellen
- AWS Lambda um STT, LLM, TTS und den Sitzungskontext zu verbinden

(Optional) Zusätzliche Verbesserungen können Folgendes umfassen:

- Amazon Kendra oder OpenSearch für kontextsensitives RAG
- Amazon DynamoDB für Sitzungsspeicher
- Amazon CloudWatch Logs und AWS X-Ray zur Rückverfolgbarkeit

Zusammenfassung

Sprach- und Sprachagenten sind intelligente Systeme, die über natürliche Konversationen miteinander interagieren. Durch die Integration von Sprachschnittstellen mit LLM-Argumentation und Echtzeit-Streaming-Infrastruktur ermöglichen Sprachagenten nahtlose, zugängliche und skalierbare Interaktionen.

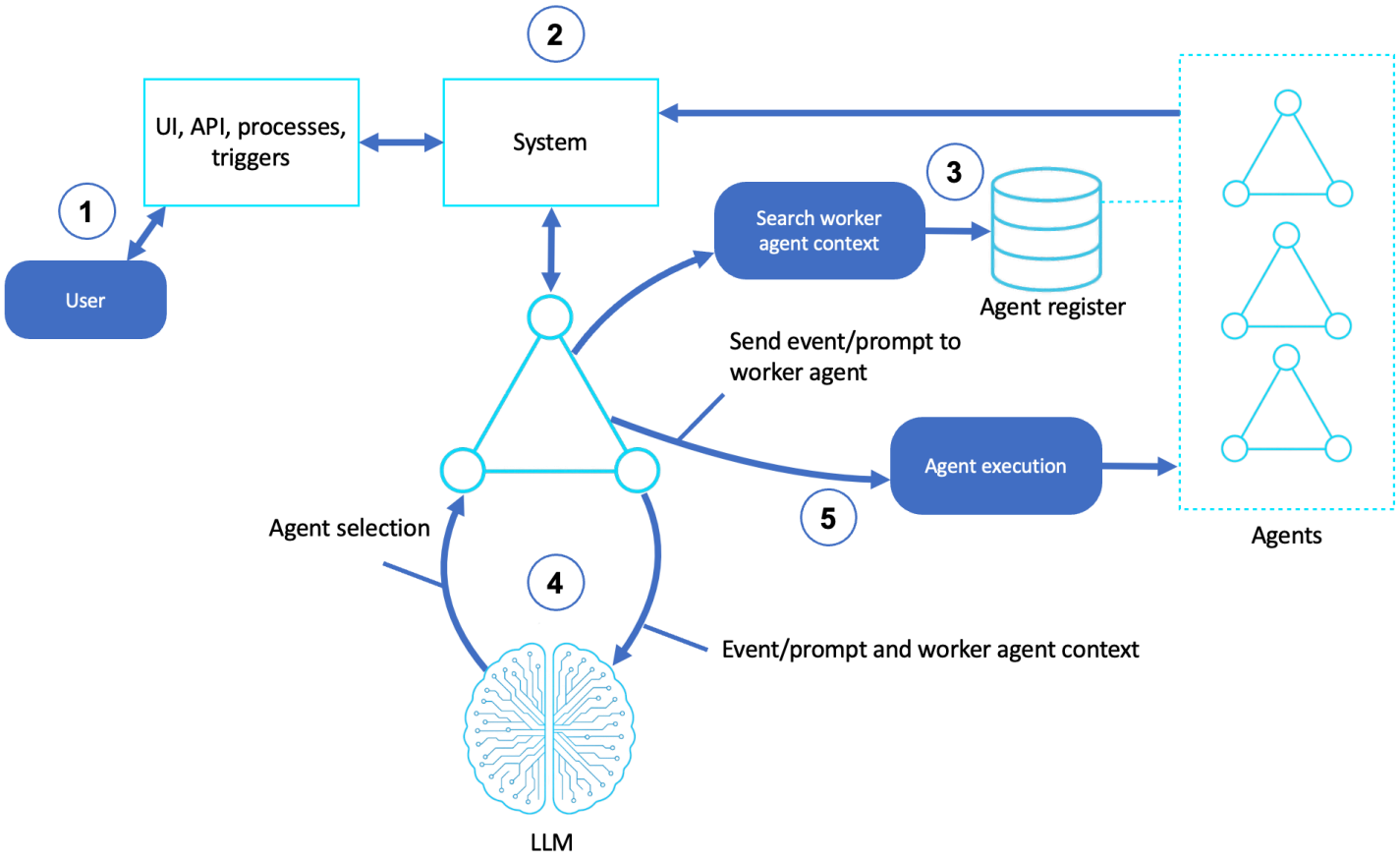
Agenten für die Workflow-Orchestrierung

Agenten zur Workflow-Orchestrierung verwalten und koordinieren mehrstufige Aufgaben, Prozesse und Dienste in verteilten Systemen. Anstatt isoliert zu argumentieren und zu handeln, delegieren diese Agenten Arbeit an Unteragenten oder andere Systeme, behalten den Ausführungskontext bei und passen sich auf der Grundlage von Zwischenergebnissen an.

Diese Agenten sind ein grundlegender Bestandteil von Automatisierungsabläufen. Sie sind besonders nützlich bei der Bearbeitung von Aufgaben mit langer Laufzeit, der Zusammenstellung mehrerer Agenten und domänenübergreifenden Integrationen, bei denen verschiedene Agenten und Tools nacheinander oder bedingt aufgerufen werden müssen.

Architektur

In der folgenden Abbildung ist ein Agent zur Workflow-Orchestrierung dargestellt:



Description

1. Empfängt Benutzereingaben
   - Ein Benutzer (oder ein externer Trigger) initiiert eine Aufgabe über eine Benutzeroberfläche, eine API oder ein Systemereignis.
2. Behandelt Systemereignisse
   - Eine Systemkomponente empfängt die Anforderung und gibt ein Ereignis oder einen Befehl aus, für das eine Orchestrierung erforderlich ist.
3. Ruft den Kontext ab
   - Der Workflow-Agent fragt Wissensdatenbanken und Agentenregister ab, um anhand von Metadaten, Domäne und früherer Erfolgsquote den richtigen Worker Agent für die Aufgabe zu finden.
4. Wählt einen LLM-Agenten aus

- Ein LLM hilft bei der Auswahl des besten Agenten oder Workflow-Plans, indem er die Aufgabenbeschreibung und die verfügbaren Optionen analysiert.
- Es kann auch aufgabenspezifische Aufforderungen formulieren, die an einen ausgewählten Agenten gesendet werden sollen.

5. Delegiert und führt aus
   - Der gewählte Worker-Agent empfängt das Ereignis oder die Aufforderung und beginnt mit der Ausführung von Befehlen.
   - Er kann den Ausführungsstatus verfolgen, bei einem Fehler den Vorgang wiederholen und Zwischenergebnisse an den nächsten Agenten in der Sequenz weiterleiten.

Capabilities

- Zusammensetzung der Agenten (z. B. Vorgesetzte, Mitarbeiter und Tools)
- Ereignisgesteuerte oder geplante Ausführung
- Speicher- und Statusverfolgung im Zeitverlauf
- Hierarchische oder parallel Aufgabenorchestrierung (synchron im Vergleich zu asynchronen Workflows)
- Dynamische Agentenauswahl und Verkettung

Häufige Anwendungsfälle

- Mehrstufige Automatisierung (z. B. Datenerfassung und Berichterstattung)
- Weiterleitung und Eskalation des Kundendienstes (z. B.) agent-as-coordinator
- KI-Agenten koordinieren sich mit Menschen und Bots innerhalb derselben Schleife
- Automatisiert Unternehmensprozesse mithilfe von LLM-gestützter Logik
- Hybridsysteme kombinieren KI-Agenten und traditionelle Orchestrierungstools

Implementierungsleitfaden

Sie können dieses Muster mit den folgenden Tools erstellen und AWS-Services:

- Amazon Bedrock zur Argumentation und Agentenauswahl
- AWS Step Functions oder Amazon EventBridge für die Workflow-Zusammenstellung

- AWS Lambda als Ausführungseinheiten oder Task-Runner
- Amazon DynamoDB, Amazon Simple Storage Service (Amazon S3) oder Amazon RDS zur Nachverfolgung von Status und Ergebnissen
- AWS AppFabric oder Amazon AppFlow für die systemübergreifende Koordination
- (Optional) Verwenden Sie Amazon SageMaker Run Agent, um domänenspezifische Worker-Agents zu hosten

Zusammenfassung

Workflow-Agenten koordinieren, passen sich an und stimmen Ziele in Umgebungen mit mehreren Agenten ab. Das bedeutet, dass KI-Agenten zusammenarbeiten, sich an die Laufzeitbedingungen anpassen und mithilfe modularer, erklärbarer Workflows komplexe Ergebnisse erzielen können.

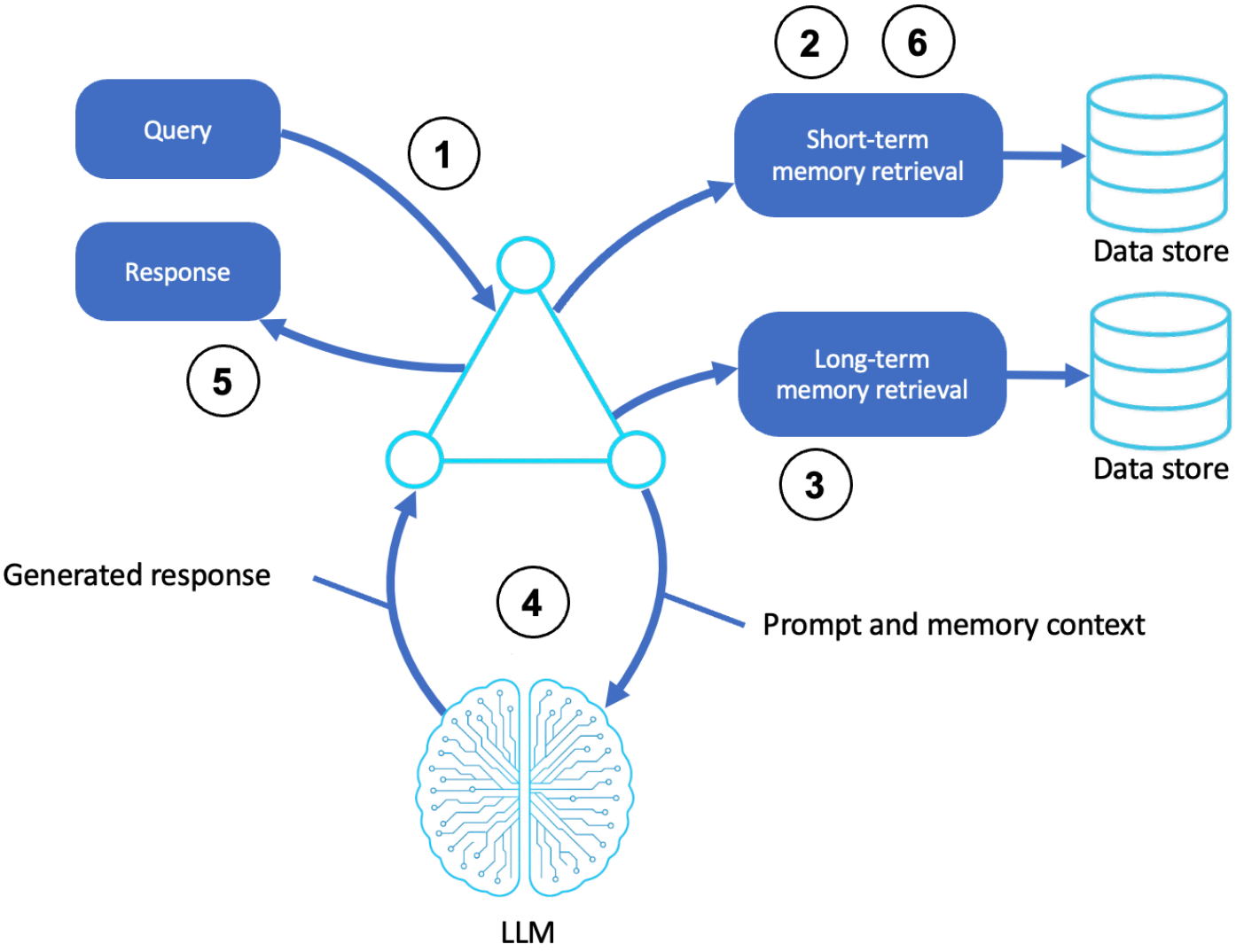
Agenten mit erweitertem Speicher

Agenten mit erweitertem Speicher sind nun in der Lage, das Kurz- und Langzeitgedächtnis zu speichern, abzurufen und zu analysieren. Auf diese Weise können sie den Kontext für mehrere Aufgaben, Sitzungen und Interaktionen aufrechterhalten, was zu kohärenteren, personalisierteren und strategischeren Antworten führt.

Im Gegensatz zu Agenten ohne Status passen sich Agenten mit erweitertem Speicher an, indem sie auf historische Daten zurückgreifen, aus früheren Ergebnissen lernen und Entscheidungen treffen, die den Zielen, Vorlieben und der Umgebung des Benutzers entsprechen.

Architektur

Ein Agent mit erweitertem Speicher ist in der folgenden Abbildung dargestellt:



Description

1. Empfängt Eingaben oder Ereignisse
   - Der Agent empfängt eine Benutzerabfrage oder ein Systemereignis. Dies kann ein Text, ein API-Trigger oder eine Änderung der Umgebung sein.
2. Ruft Kurzzeitgedächtnis ab
   - Der Agent ruft den aktuellen Gesprächsverlauf, den Aufgabenkontext oder den Systemstatus ab, der für die Sitzung oder den Workflow relevant ist.
3. Ruft das Langzeitgedächtnis ab
   - Der Agent fragt das Langzeitgedächtnis (z. B. Vektordatenbanken und Schlüsselwertspeicher) ab, um historische Erkenntnisse wie die folgenden zu erhalten:

    - Benutzereinstellungen
    - Frühere Entscheidungen und Ergebnisse
    - Gelernte Konzepte, Zusammenfassungen oder Erfahrungen
4. Gründe durch das LLM
    - Der Speicherkontext ist in die LLM-Eingabeaufforderung eingebettet, sodass der Agent sowohl auf der Grundlage aktueller Eingaben als auch auf Vorkenntnissen argumentieren kann.
5. Generiert Ausgaben
    - Der Agent erstellt eine kontextsensitive Antwort, einen Plan oder eine Aktion, die entsprechend dem Aufgabenverlauf und den Eingaben des Benutzers personalisiert ist.
6. Aktualisiert den Speicher
    - Neue Informationen wie aktualisierte Ziele, Erfolgs- und Fehlschlagssignale und strukturierte Antworten werden für future Aufgaben gespeichert.

Capabilities

- Kontinuität der Sitzung über Konversationen oder Ereignisse hinweg
- Beharrlichkeit des Ziels im Laufe der Zeit
- Kontextuelles Bewusstsein auf der Grundlage eines sich entwickelnden Zustands
- Anpassungsfähigkeit basiert auf früheren Erfolgen und Misserfolgen
- Personalisierung, abgestimmt auf Benutzerpräferenzen und -historie

Häufige Anwendungsfälle

- Konversations-Copiloten, die sich Benutzerpräferenzen merken
- Codierungsagenten, die Änderungen an der Codebasis verfolgen
- Workflow-Agenten, die sich je nach Aufgabenverlauf anpassen
- Digitale Zwillinge, die sich aus Systemwissen entwickeln
- Forschungsagenten, die redundante Abfragen vermeiden

Implementierung von Agenten mit erweitertem Speicher

Verwenden Sie die folgenden Tools und AWS-Services für Agenten mit erweitertem Speicher:

Speicherschicht	AWS-Service	Zweck
Kurzfristig	Amazon DynamoDB-, Redis- und Amazon Bedrock-Kontext	Schnelles Abrufen der letzten Interaktionsstatus
Langfristig (strukturiert)	Amazon Aurora, Amazon DynamoDB, Amazon Neptune	Fakten, Beziehungen und Protokolle
Langfristig (semantisch)	OpenSearch, PostgreSQL, Tannenzapfen	Auf Einbettung basierender Abruf (d. h. RAG)
Speicher	Amazon S3	Speichern von Transkripten, strukturierten Speichern und Dateien
Orchestrierung	AWS Lambda oder AWS Step Functions	Verwaltung der Speicheri njektion und des Aktualisi erungszyklus
Reasoning	Amazon Bedrock	Anthropic Claude oder Mistral mit Erinnerungsaufford erungen

Implementierung von Eingabeaufforderungen mit integriertem Speicher

Um das Gedächtnis in die Argumentation der Agenten zu integrieren, verwenden Sie eine Kombination aus strukturiertem Zustand und abruf-erweiterter Kontexteinspeisung:

- Beziehen Sie bei der Erstellung der Eingabeaufforderung für das Sprachmodell den aktuellen Status des Agenten und den Verlauf der letzten Dialoge als strukturierte Eingabe mit ein, sodass der Mitarbeiter mit vollständigem Kontext argumentieren kann.
- Verwenden Sie Retrieval-Augmented Generation (RAG), um relevante Dokumente oder Fakten aus dem Langzeitgedächtnis abzurufen.
- Fassen Sie frühere Pläne, Kontexte und Interaktionen zusammen, um sie zu komprimieren und relevant zu machen.
- Fügen Sie während der Inferenz externe Speichermodule wie Vektorspeicher oder strukturierte Protokolle ein, um die Entscheidungsfindung zu unterstützen.

Zusammenfassung

Agenten mit erweitertem Gedächtnis sorgen für Kontinuität im Denken, indem sie aus Erfahrungen lernen und sich den Benutzerkontext merken. Diese Agenten übertreffen reaktive Intelligenz, indem sie auf langfristige Zusammenarbeit, Personalisierung und strategisches Denken setzen. In Bezug auf agentische KI ermöglicht das Gedächtnis den Agenten, sich eher wie adaptive digitale Gegenstücke und weniger wie zustandslose Tools zu verhalten.

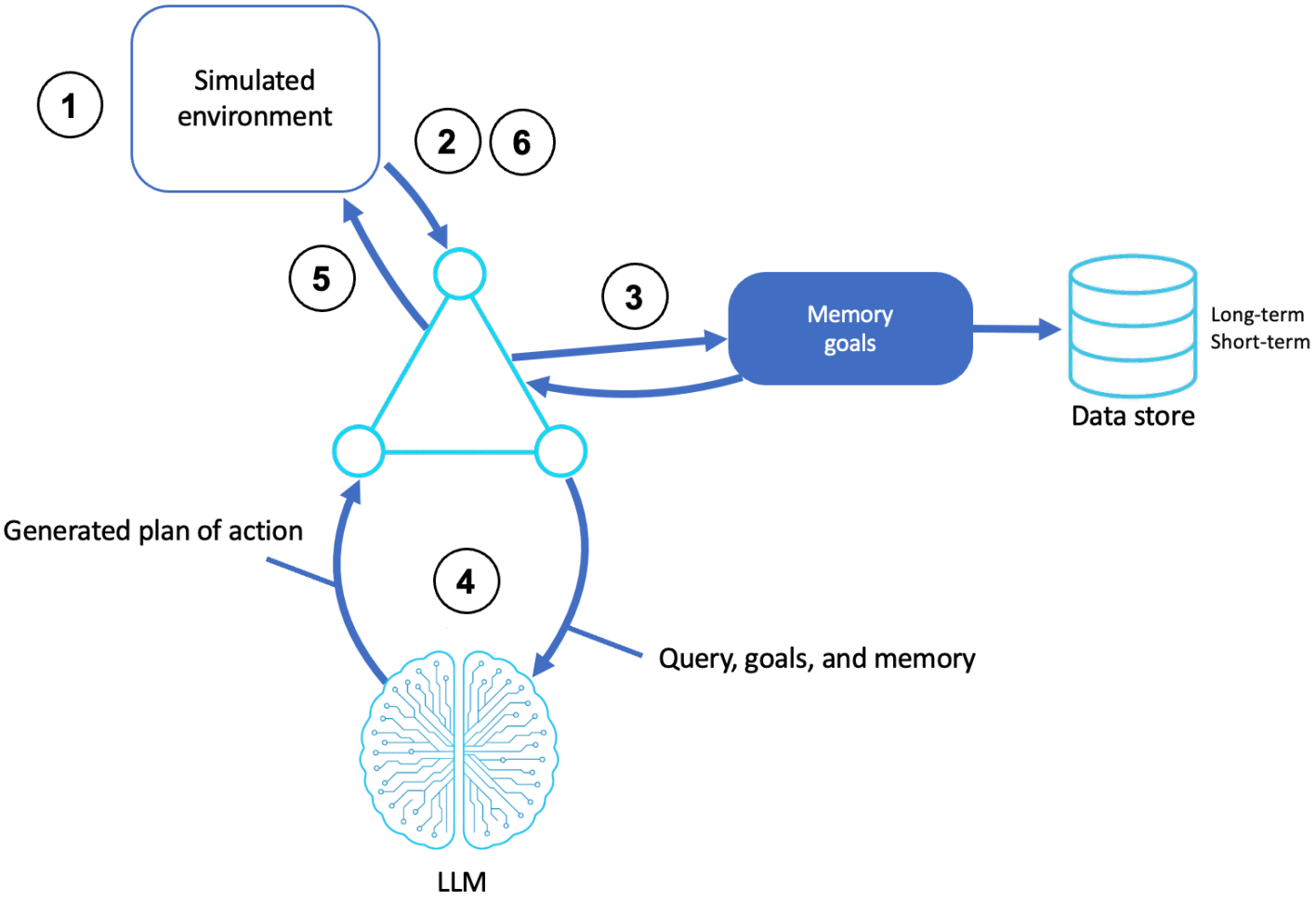
Agenten für Simulation und Testumgebung

Simulations- und Testbed-Agenten arbeiten in virtualisierten oder kontrollierten Umgebungen, in denen sie denken, handeln und lernen. Diese Agenten simulieren Verhalten, modellieren Ergebnisse und trainieren Strategien in wiederholbaren Umgebungen, bevor sie sie in realen Umgebungen anwenden.

Dieses Muster ist nützlich für iterative Entwicklung, verstärkendes Lernen (RL), die Bewertung autonomer Entscheidungen und das Testen von emergentem Verhalten. Simulationsagenten arbeiten oft in geschlossenen Kreisläufen, erhalten Feedback von ihrer Umgebung und passen ihr Verhalten entsprechend an. Deshalb sind sie unverzichtbar für Aufgaben, die räumliches Denken, Echtzeitsteuerung oder komplexe Systemdynamik beinhalten.

Architektur

Das folgende Diagramm zeigt einen Simulations- oder Testbed-Agenten:



Description

1. Initiiert eine Umgebung
   - Der Agent initiiert eine simulierte Umgebung (z. B. eine 3D-Welt, eine Physik-Engine, eine CLI-Sandbox oder einen synthetischen Datenstrom).
   - Der Agent wird mit einer ersten Aufgabe, einem Ziel oder einer Richtlinie in die Umgebung geladen.
2. Nimmt den Agenten wahr
   - Der Agent nimmt den aktuellen Status durch Simulationstelemetrie wahr (z. B. Sensoremulation, virtuelle Kamera und strukturierte Protokolle).
3. Ruft Ziel und Erinnerung ab
   - Der Agent ruft das ihm zugewiesene Ziel, seine Szenarioanweisungen oder sein kontextuelles Ziel ab.

- Er kann auch vorherigen Speicher abrufen, einschließlich der folgenden:
  - Langfristige Strategien oder Richtlinien
  - Umweltkarten oder bekannte Einschränkungen
  - Frühere Erfolge oder Misserfolge ähnlicher Simulationen

4. Gründe und Pläne
   - Ein LLM interpretiert den simulierten Zustand, die Aufgabenziele und das erlernte Wissen.
   - Es generiert einen Aktionsplan oder einen Kontrollbefehl.
5. Führt simulierte Aktionen aus
   - Der Agent führt den Plan aus, ändert den Status, navigiert im Raum oder interagiert mit virtuellen Entitäten.
6. Lernt
   - Der Agent bewertet die Ergebnisse der Maßnahmen
   - Je nach Konfiguration des Agenten kann er Folgendes tun:
     - RL ausführen
     - Protokollieren Sie die Ergebnisse für future Feinabstimmungen
     - Passen Sie Strategien in Echtzeit an

Capabilities

- Arbeitet in synthetischen oder virtuellen Umgebungen
- Unterstützt trial-and-error Lernen, die Verfeinerung von Richtlinien und die Systemmodellierung
- Tests mit geringem Risiko in Bezug auf Verhalten, Fehlerbehandlung und Sonderfälle
- Ermöglicht die Analyse des Verhaltens neu auftretender Agenten in Konfigurationen mit mehreren Agenten
- Unterstützt sowohl die Steuerung als auch die Erkundung von Daten im geschlossenen Regelkreis human-in-the-loop

Häufige Anwendungsfälle

- Reinforcement-Learning für Robotik, Drohnen und Spiele
- Training autonomer Fahrzeuge auf virtuellen Straßen
- Simulierte UIs oder CLIs für- DevOps und Prüfstandsszenarien

- Experimente mit neu auftretendem Verhalten in sozialen Simulationen
- Sicherheitsvalidierung der Entscheidungslogik vor der Produktion

Implementierungsleitfaden

Sie können einen Simulations- und Testbed-Agenten mit den folgenden Tools erstellen und: AWS-Services

Komponente	AWS-Service	Zweck
Umgebung	Amazon ECS EC2, Amazon oder ein benutzerdefinierte r Simulator im Amazon SageMaker Studio Lab	Führen Sie virtuelle Welten (Gazebo, Unity, Unreal) oder eine Sandbox aus CLIs
Agentenlogik	Amazon Bedrock SageMaker, Amazon oder AWS Lambda	LLM-basierte Planer oder RL-Agenten
Feedback-Schleife	Amazon SageMaker Reinforce ment-Learning CloudWatc h, Amazon oder benutzerd efinierte Protokolle	Nachverfolgung von Prämien, Ergebnisbewertung und Verhaltensprotokollierung
Erinnerung und Wiederholung	Amazon S3, Amazon DynamoDB oder Amazon RDS	Persistenter Status, Episodenv erlauf oder Szenariodaten
Visualisierung	CloudWatch Amazon-Da shboards oder Amazon-No tizbücher SageMaker	Beobachten Sie politische Änderungen, Ergebnisse und Schulungskennzahlen

Im Folgenden sind zusätzliche Anwendungen aufgeführt:

- AWS SimSpace Weaverfür groß angelegte räumliche Simulationen
- AWS IoT Corezum Testen von Schattengeräten
- Amazon SageMaker Experiments zur Bewertung und zum Benchmarking von Agenten

Zusammenfassung

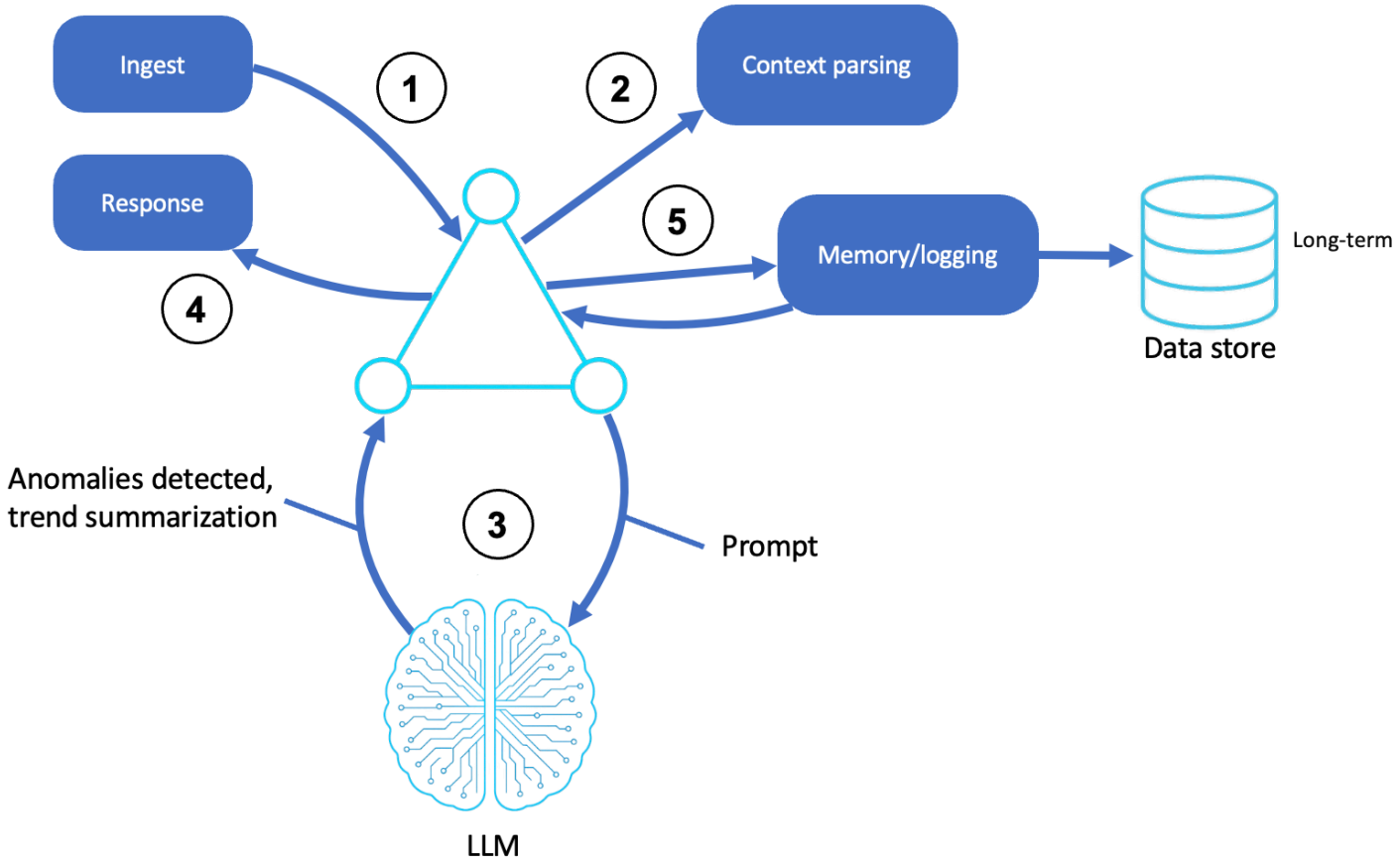
Simulations- und Prüfstandsagenten dienen der strukturierten Erkundung, bevor sie in Produktionssystemen eingesetzt werden. Verwenden Sie diese Agenten, um autonome Navigationsrichtlinien zu trainieren, Geschäftsprozesse in synthetischen Umgebungen zu testen und Schwärme auf Koordinationsmuster zu untersuchen.

Beobachter- und Überwachungsagenten

Beobachter- und Monitoring-Agenten beobachten passiv Systeme, Umgebungen und Interaktionen, um Muster zu erkennen, Erkenntnisse zu gewinnen und Aktionen auszulösen. Als intelligente Beobachter verbessern sie Warnmeldungen, Diagnosen und Audits, ohne direkt ein Verhalten auszulösen.

Diese Agenten zeichnen sich dort aus, wo es der herkömmlichen Überwachung an Anpassungsfähigkeit oder Argumentation mangelt, insbesondere in Bezug auf AI-in-the-loop Überwachung, Erkennung von Anomalien, Compliance-Überwachung und Sicherheitsinformationen. Observer Agents sind Ereignis-Listener, die Systemtelemetrie und Benutzerinteraktionen kontinuierlich überwachen. Der Agent ist auf seine Wahrnehmung, Interpretation und bedingte Eskalation oder Berichterstattung angewiesen.

Architektur



Description

1. Telemetrie aufnehmen
   - Der Agent empfängt Eingaben von einer oder mehreren Systemquellen, z. B. aus den folgenden:
     - Protokolle (Anwendung, Infrastruktur, Sicherheit)
     - Metriken (Leistung, Latenz, Nutzung)
     - Ereignisse (API-Aufrufe, Benutzeraktionen, Sensordaten)
2. Kontext analysieren
   - Rohdaten werden analysiert, strukturiert und mit Metadaten wie Zeitstempel, Akteuridentität, Systemstatus und Trace-ID angereichert.
3. Gründe für die Verwendung von LLM

- Der Agent verwendet ein LLM- oder Logikmodul, um analysierte Eingaben zu interpretieren, indem er Anomalien identifiziert, Trends zusammenfasst und Korrelationen zwischen verteilten Traces oder Zeitfenstern erstellt.

4. Klassifizieren oder warnen
   - Der Agent stellt fest, ob das beobachtete Verhalten Folgendes rechtfertigt:
     - Eine Warnung oder Eskalation
     - Ein Bericht oder ein Dashboard-Update
     - Ein Reaktionsauslöser (z. B. automatische Problembehebung und Richtliniendurchsetzung)
5. Speicher- oder Feedback-Schleifen protokollieren
   - Der Agent speichert Ereignisse und Entscheidungen für langfristiges Lernen, Audits oder future Referenzzwecke für andere Agenten.

Capabilities

- Passiv und nicht invasiv (der Agent wirkt nicht direkt)
- Hochgradig skalierbar und asynchron
- KI-gestützte Korrelation zwischen verrauschten oder verteilten Signalen
- Unterstützt Audits, Compliance und Einblicke in Echtzeit
- Kann nachgeschaltete Mitarbeiter oder menschliche Arbeitsabläufe unterstützen

Häufige Anwendungsfälle

- KI-gestützte Beobachtbarkeit für Microservices und APIs
- Überwachung von Modellabweichungen, Richtlinienverstößen oder Verhalten out-of-band
- Analyse der Kundenaktivitäten oder Zusammenfassungen der Interaktionen
- Code-Review-Agenten, die Commits oder Deployments überwachen
- Überwachung von Sicherheits- oder Compliance-Protokollen mithilfe von LLM-Argumentation

Implementierungsleitfaden

Sie können mit den folgenden Tools einen Beobachter- und Monitoring-Agenten erstellen: AWS-Services

Komponente	AWS-Service	Zweck
Erfassung von Ereignissen	Amazon EventBridge, CloudWatch Amazon-Protokolle, Amazon Kinesis, Amazon S3	Erfassen Sie strukturierte und unstrukturierte Telemetrie
Vorverarbeitung	AWS Lambda, AWS Glue, AWS Step Functions	Wandeln Sie Rohdaten in strukturierte Eingabeaufforderungen um
Maschine zum Argumentieren	Amazon Bedrock, Amazon SageMaker, AWS Lambda	Ereignisse analysieren, Verhalten klassifizieren, Erkenntnisse gewinnen
Speicher und Arbeitsspeicher	Amazon S3, Amazon DynamoDB, OpenSearch	Ständige Beobachtungen, Zusammenfassungen und Ergebnisse
Warnung und Eskalation	Amazon SNS AWS AppFabric, Amazon EventBridge	Lösen Sie nachgeschaltete Systeme oder Agenten aus

Im Folgenden sind zusätzliche Anwendungen aufgeführt:

- AWS Security Hub CSPMfür die Überwachung von Sicherheitsprotokollen
- Amazon Quick Suite zur Visualisierung von Agentenausgaben

Zusammenfassung

Beobachter- und Monitoring-Agenten verfolgen Systeme und Verhalten in Echtzeit. Sie erkennen Anomalien, überprüfen die Sicherheit und sammeln Betriebsinformationen, indem sie Muster identifizieren, die von Menschen oder Regeln möglicherweise übersehen werden. Diese Fähigkeit hilft bei der Entwicklung von Systemen, die sich an sich ändernde Bedingungen anpassen und Entscheidungen auf der Grundlage umfassender Datenanalysen treffen können.

Zusammenarbeit mit mehreren Agenten

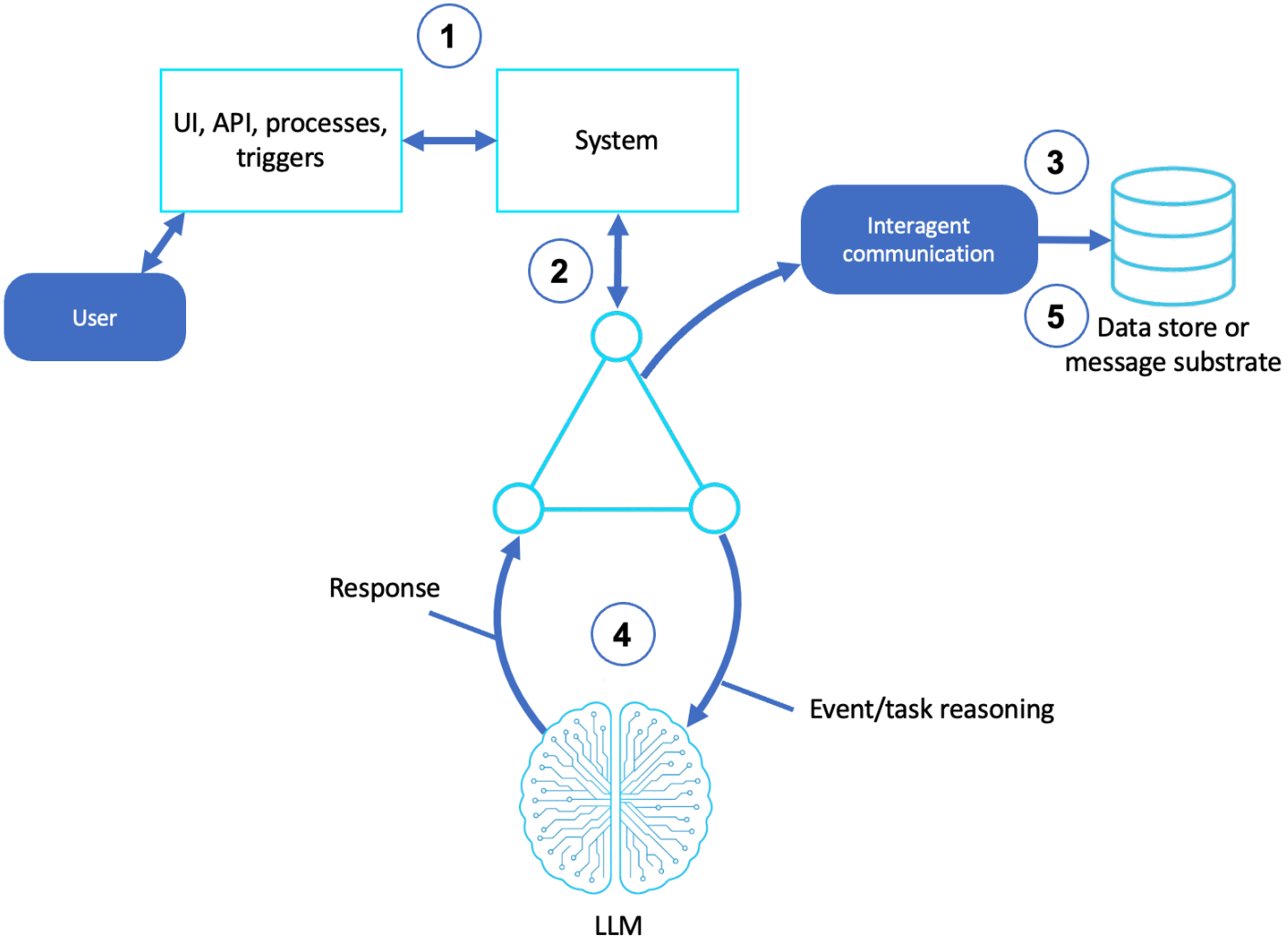
Die Zusammenarbeit mehrerer Agenten bezieht sich auf ein Muster, bei dem mehrere autonome Agenten, von denen jeder eine eigene Rolle, Spezialisierung oder Zielsetzung hat, miteinander verhandeln, um komplexe Aufgaben zu lösen. Diese Agenten können unabhängig voneinander oder mit anderen Agenten zusammenarbeiten, indem sie Informationen austauschen, Verantwortlichkeiten aufteilen und gemeinsam ein Ziel verfolgen.

Dieses Muster unterscheidet sich von Workflow-Agenten, die Aufgaben zentral koordinieren und in einem strukturierten Ablauf an untergeordnete Agenten delegieren. Im Gegensatz dazu betont die Zusammenarbeit mehrerer Agenten die sogenannte peer-to-peer Emergenzkoordination, indem sie Adaptivität, Parallelität und kognitive Teilung ermöglicht. In der folgenden Tabelle wird die Zusammenarbeit mehrerer Agenten mit Workflow-Agenten verglichen:

Funktion	Workflow-Agenten	Zweck
Kontrolle	Zentralisierter Koordinator	Dezentrale, verteilte oder rollenbasierte Kollegen
Interaktionen	Ein Agent delegiert und verfolgt die Ausführung	Mehrere Agenten verhandeln, tauschen sich aus und passen sich an
Design	Vordefinierte Reihenfolge der Aufgaben	Emergente, flexible Aufgabenverteilung
Koordination	Prozedurale Orchestrierung	Kooperative oder kompetitive Interaktionen
Anwendungsfälle	Automatisierung von Unternehmensprozessen	Komplexes Denken, Erkundung und neue Strategien

Architektur

Das folgende Diagramm zeigt die Zusammenarbeit mehrerer Agenten:



Description

1. Leitet eine Aufgabe ein
   - Ein Benutzer oder ein System gibt ein übergeordnetes Ziel oder Problem aus.
   - Ein „Manager“, ein Agent oder ein initiierender Kontext definiert das Ziel.
2. Weist Rollen zu oder entdeckt sie
   - Agenten weisen sich selbst zu (symbolische Logik oder Argumentation) oder werden delegiert (Eventbroker) an andere Rollen, z. B. als Planer, Forscher, Testamentsvollstrecker, Kritiker oder Erklärer.
3. Kommuniziert mit anderen Agenten
   - Agenten kommunizieren über gemeinsamen Speicher, Nachrichtenwarteschlangen oder die Verkettung von Eingabeaufforderungen.

- Sie können sich gegenseitig diskutieren, Fragen stellen oder Unteraufgaben vorschlagen.

4. Verwendet spezielle Argumentation
   - Jeder Agent verwendet sein eigenes Modell oder seine eigene Domänenlogik, um seinen Teil des Problems zu lösen.
   - Agenten können LLMs mit rollenspezifischen Eingabeaufforderungen und Arbeitsspeicher arbeiten.
5. Koordiniert Ergebnisse oder Ziele
   - Die Agenten fassen die Beiträge zu einer endgültigen Antwort, einem Plan oder einer Maßnahme zusammen.
   - (Optional) Ein Supervisor kann das synthetisierte Ergebnis validieren oder zusammenfassen.

Capabilities

- Agenten auf Fachebene mit speziellen Rollen oder Fähigkeiten
- Aufkommendes Verhalten durch Kommunikation oder Verhandlung
- Parallele Bearbeitung komplexer oder vielschichtiger Probleme
- Unterstützt Überlegung, Selbstkorrektur und reflektierende Iteration
- Modellieren Sie soziale Dynamiken, wissenschaftliche Zusammenarbeit oder Rollen in Unternehmensteams

Häufige Anwendungsfälle

- Autonome Forschungsteams (Suchagent, Summarizer und Validator)
- Softwareentwicklung (Planer, Programmierer und Tester)
- Modellierung von Geschäftsszenarien (Finanzen, Richtlinien und Compliance)
- Verhandlung, Ausschreibung oder Argumentation durch mehrere Parteien
- Multimodale Aufgaben (Bild, Text und Logik)

Implementierungsleitfaden

Sie können ein System mit mehreren Agenten mithilfe der folgenden Tools erstellen und: AWS-Services

Komponente	AWS-Service	Zweck
Agenten-Hosting	Amazon Bedrock, Amazon SageMaker, AWS Lambda	Hosten Sie einzelne LLM-gesteuerte Agenten
Kommunikationsebene	Amazon SQS, Amazon EventBridge, AWS AppFabric	Nachrichtenübermittlung und Koordination zwischen den Agenten
Gemeinsamer Speicher	Amazon DynamoDB, Amazon S3 oder OpenSearch	Speicher für mehrere Agenten oder Blackboard-System
Orchestrierungsebene	AWS Step Functions, AWS Lambda Rohrleitungen	Kickoff-, Timeout-, Fallback- und Wiederholungslogik
Identifizierung des Agenten	Amazon Bedrock Agents (rollendefiniert) AWS AppConfig und Amazon Bedrock Converse API (Agenten außerhalb von Amazon Bedrock)	Rollenbasiertes Tool oder Agentenaufruf und Durchsetz ung von Grenzen
Emergente Interaktion	EventBridge Amazon-Pi pelines oder Agentenregister	Aktivieren Sie die dynamische Weiterleitung oder Eskalation von Aufgaben

Zusammenfassung

Durch die Zusammenarbeit mehrerer Agenten werden die Aufgaben zur Problemlösung auf modulare, rollengesteuerte Agenten verteilt. Im Gegensatz zur Workflow-Orchestrierung nutzen Kooperationsmuster neue Intelligenz, Belastbarkeit und Skalierbarkeit, die die Art und Weise widerspiegeln, wie Menschen Probleme lösen. Dies ist besonders wertvoll für offene Bereiche, kreative Aufgaben, multimodales Denken und Umgebungen, die von unterschiedlichen Perspektiven profitieren.

Schlussfolgerung

Die zuvor erörterten Muster veranschaulichen grundlegende Ansätze für reale Implementierungen agentischer KI. Von grundlegenden Argumenten bis hin zu gedächtnisgestützter Intelligenz ist jedes Muster einzigartig für Wahrnehmung, Kognition und Handlung konfiguriert und basiert auf Autonomie, Asynchronität und Entscheidungsfreiheit.

Diese Muster haben gemeinsame Vokabeln und technische Pläne für den Aufbau intelligenter, zielgerichteter Systeme. Unabhängig davon, ob ein Muster in eine Benutzeroberfläche eingebettet, über Cloud-Dienste orchestriert oder von Agententeams koordiniert wird, jedes Muster ist anpassbar und modular.

Imbissbuden

- Agentenmuster sind zusammensetzbar — Die meisten Agenten aus der realen Welt kombinieren zwei oder mehr Muster (z. B. ein Sprachagent mit toolgestütztem Denken und Gedächtnis).
- Das Agentendesign ist kontextuell — Wählen Sie Muster auf der Grundlage der Interaktionsoberfläche, der Aufgabenkomplexität, der Latenztoleranz und domänenspezifischen Einschränkungen aus.
- AWSEine native Implementierung ist möglich — Mit Amazon Bedrock, Amazon SageMaker AWS Lambda AWS Step Functions, und ereignisgesteuerten Architekturen kann jedes Agentenmuster in großem Umfang bereitgestellt werden.

LLM-Workflows

Im Rahmen von Agentenmustern haben wir die gängigen Muster von KI-Agenten untersucht, die jeweils auf einer Reihe modularer Funktionen basieren: Wahrnehmung, Handlung, Lernen und Kognition. Das Herzstück des kognitiven Moduls vieler Agentenmuster ist ein großes Sprachmodell (LLM), das in der Lage ist, zu argumentieren, zu planen und Entscheidungen zu treffen. Der alleinige Aufruf eines LLM reicht jedoch nicht aus, um intelligentes, zielgerichtetes Verhalten zu erzeugen.

Um komplexe Aufgaben zuverlässig ausführen zu können, müssen Agenten das LLM in einen strukturierten Workflow einbetten, in dem die Funktionen des Modells durch Tools, Speicher, Planungsschleifen und Koordinationslogik erweitert werden. Diese LLM-Workflows ermöglichen es einem Agenten, Ziele aufzuschlüsseln, Unteraufgaben weiterzuleiten, externe Dienste in Anspruch zu nehmen, Ergebnisse zu analysieren und sich mit anderen Agenten abzustimmen.

In diesem Kapitel werden die wichtigsten Entwurfsmuster für die Erstellung robuster, erweiterbarer und intelligenter LLM-gestützter kognitiver Module vorgestellt, die auf wiederverwendbaren Workflows basieren.

In diesem Abschnitt

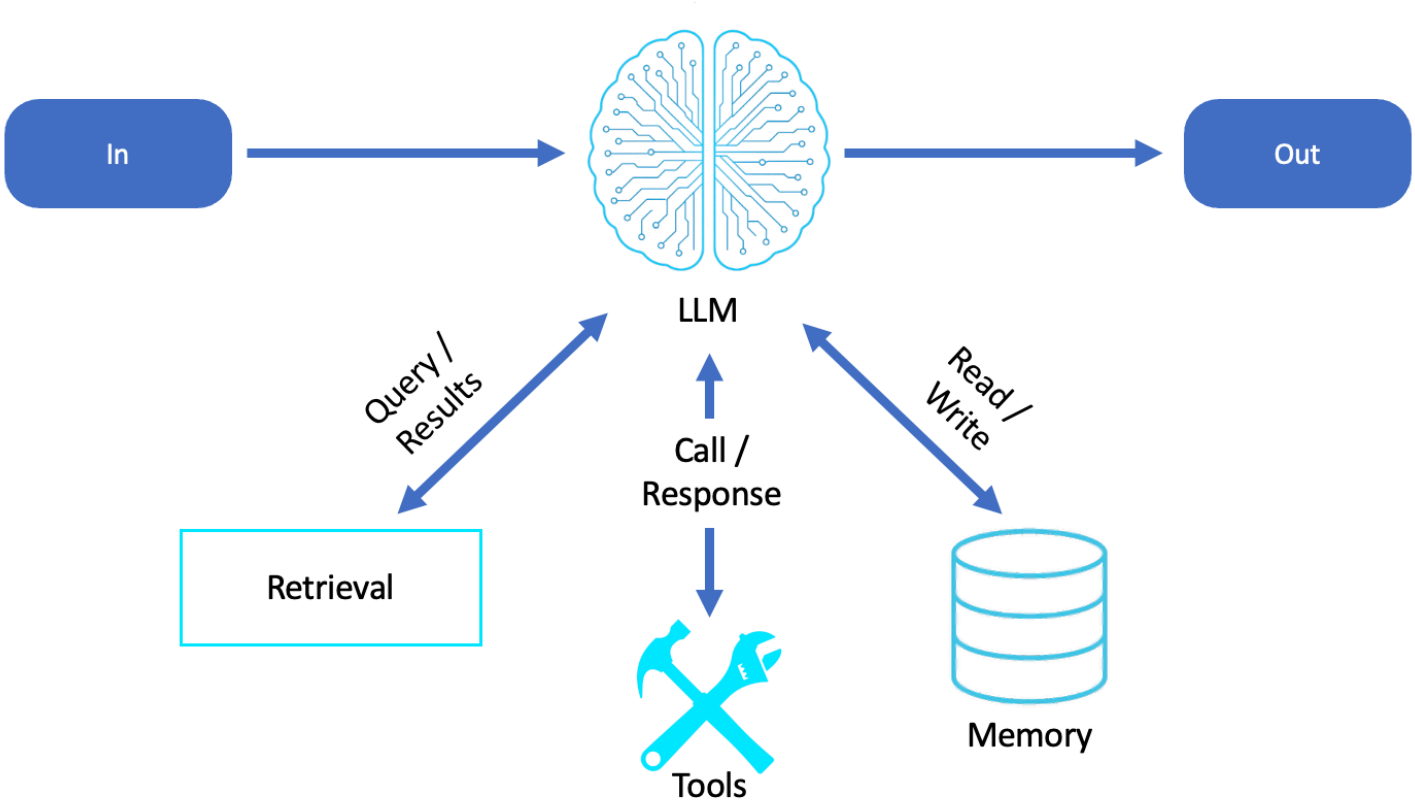
Überblick über LLM-erweiterte Kognition

Im Kern kann das kognitive Modul eines Softwareagenten als ein in Erweiterungen verpacktes LLM betrachtet werden. Der Agent kann die folgenden Bausteine verwenden, um in seiner Umgebung effektiv zu argumentieren:

- Eingabeaufforderung — Eingaben anhand von Kontext, Anweisungen, Beispielen und Speicher strukturieren

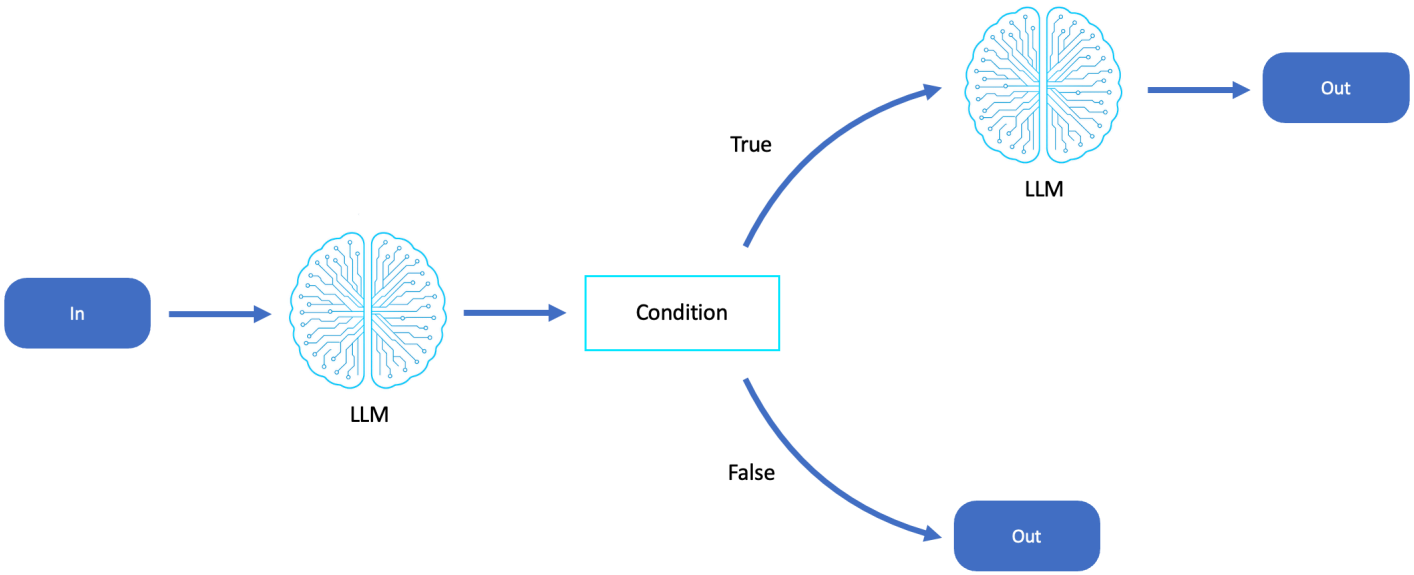
- Abruf — Bereitstellung von up-to-date domänenspezifischem Wissen an die LLM-Eingabeaufforderung durch Vektorsuche oder semantisches Gedächtnis, z. B. durch Retrieval-Augmented Generation (RAG)
- Verwendung von Tools — Ermöglicht es dem LLM, Funktionen aufzurufen oder aufzurufen, um Informationen abzurufen oder darauf zu reagieren APIs
- Speicher — Einbindung eines dauerhaften Zustands oder eines sitzungsbasierten Zustands in die Argumentationsschleife, entweder mithilfe strukturierter Datenbanken oder kontextueller Zusammenfassungen

Diese Erweiterungen bestehen aus Workflows, die definieren, wie das LLM im Laufe der Zeit und aufgabenübergreifend genutzt wird, wodurch es von einer zustandslosen Engine in einen dynamischen Argumentationsmechanismus umgewandelt wird.



Arbeitsablauf für schnelle Verkettung

Prompt Chaining zerlegt komplexe Aufgaben in eine Abfolge von Schritten, wobei jeder Schritt ein diskreter LLM-Aufruf ist, der die Ausgabe des vorherigen verarbeitet oder darauf aufbaut.



Der Workflow zur Verkettung von Eingabeaufforderungen eignet sich für Szenarien, in denen Aufgaben logisch in sequentielle Argumentationsschritte unterteilt werden können und in denen Zwischenergebnisse als Grundlage für die nächste Phase dienen. Er eignet sich hervorragend für Workflows, die strukturiertes Denken, fortschreitende Transformation oder mehrschichtige Analysen erfordern, wie z. B. die Überprüfung von Dokumenten, die Codegenerierung, die Extraktion von Wissen und die Verfeinerung von Inhalten.

Beschreibung

- Die Komplexität der Aufgabe übersteigt das Kontextfenster oder die Argumentationstiefe eines einzelnen LLM-Anrufs.
- Die Ergebnisse eines Schritts (z. B. Analyse, Zusammenfassung oder Planung) werden zu Inputs für eine Folgeentscheidungs- oder Generierungsphase.
- Sie benötigen Transparenz und Kontrolle in allen Phasen der Argumentation (z. B. überprüfbare Zwischenergebnisse).
- Sie möchten zwischen den Schritten eine externe Validierungs-, Filter- oder Anreicherungslogik einbauen.
- Es ist ideal für Agenten, die in Denkschleifen im Pipeline-Stil arbeiten, wie z. B. Rechercheagenten, Redaktionsassistenten, Planungssysteme und mehrstufige Copiloten.

Funktionen

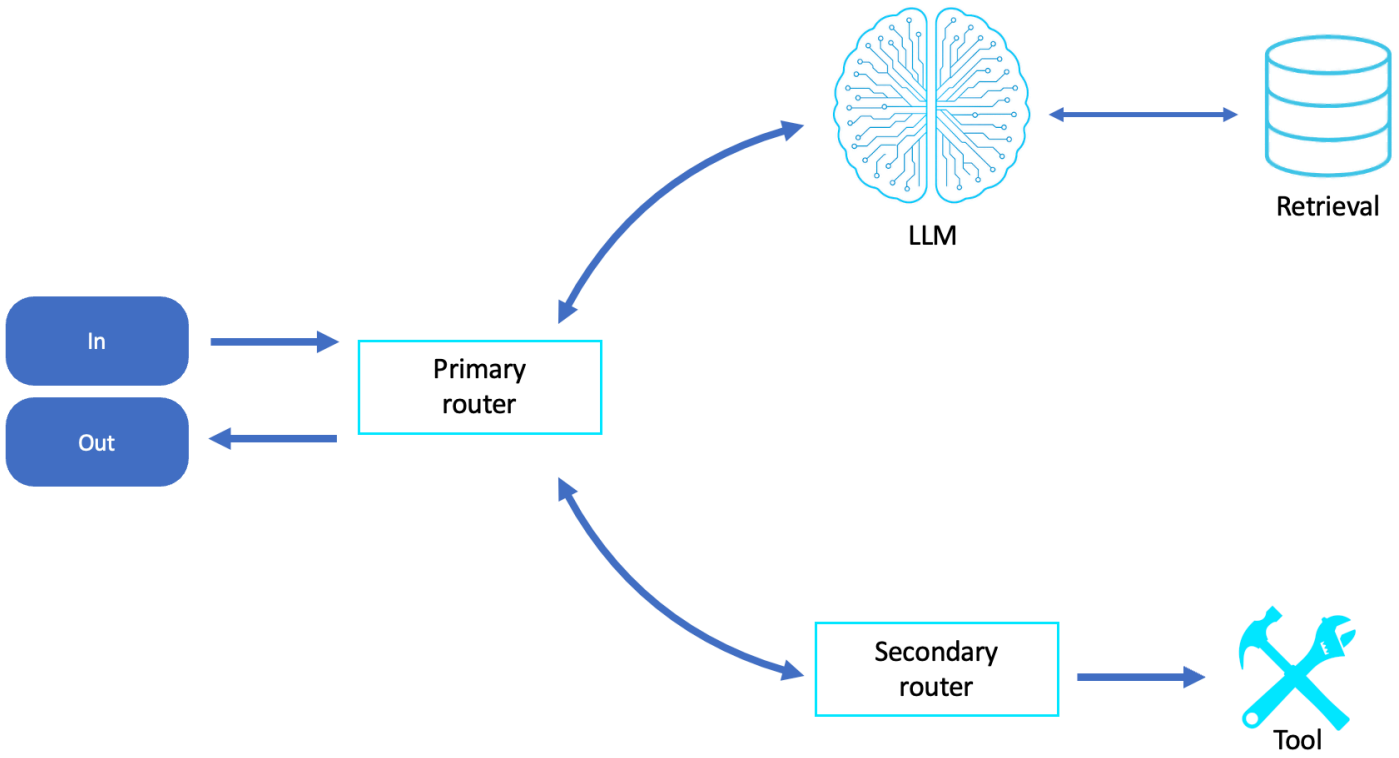
- Lineare oder verzweigte Ketten von LLM-Aufrufen
- Zwischenergebnisse wurden als strukturierte Eingabe weitergegeben oder in Folgeaufforderungen eingebettet
- Kann mit, oder agentenspezifischen AWS Step Functions Runnern AWS Lambda orchestriert werden

Häufige Anwendungsfälle

- Mehrstufige Aufgaben zum Argumentieren (z. B. „Kritik zusammenfassen, neu schreiben“)
- Wissenschaftliche Mitarbeiter, die mehrschichtige Ergebnisse zusammenfassen (z. B. „Suchen, Fakten extrahieren, Frage beantworten“)
- Pipelines zur Codegenerierung („Plan generieren, Code schreiben, Testcode erklären, Ausgabe erklären“)

Arbeitsablauf für das Routing

Im Routing-Muster verwendet ein Classifier oder Router-Agent ein LLM, um die Absicht oder Kategorie einer Abfrage zu interpretieren, und leitet die Eingabe dann an eine spezialisierte Downstream-Aufgabe oder einen Agenten weiter.



Der Routing-Workflow wird in Szenarien verwendet, in denen ein Agent die Eingabeabsicht, den Aufgabentyp oder die Domäne schnell klassifizieren und die Anfrage dann an einen speziellen Subagenten, ein Tool oder einen Workflow delegieren muss. Er ist besonders nützlich für Capability Agents, z. B. solche, die als allgemeine Assistenten, Eingangstüren zu Unternehmensfunktionen oder domänenübergreifende KI-Benutzeroberflächen für Benutzer dienen.

Routing ist besonders effektiv, wenn:

- Sortierung von Anfragen anhand einer Vielzahl von Aufgaben (z. B. Suche, Zusammenfassung, Buchung, Berechnungen).
- Eingaben müssen vorverarbeitet oder normalisiert werden, bevor sie in speziellere Workflows aufgenommen werden können.
- Verschiedene Eingabetypen (z. B. Bilder im Vergleich zu Text, strukturierte und unstrukturierte Abfragen) erfordern eine individuelle Handhabung.
- Ein Agent fungiert als Konversationszentrale und delegiert Aufgaben an spezialisierte Agenten oder Microservices.
- Dieser Workflow ist bei domänenspezifischen Copiloten, Bots für den Kundensupport, Enterprise Service Routern und multimodalen Agenten üblich, bei denen intelligentes Dispatching sowohl die Qualität als auch die Effizienz des Agentenverhaltens bestimmt.

Funktionen

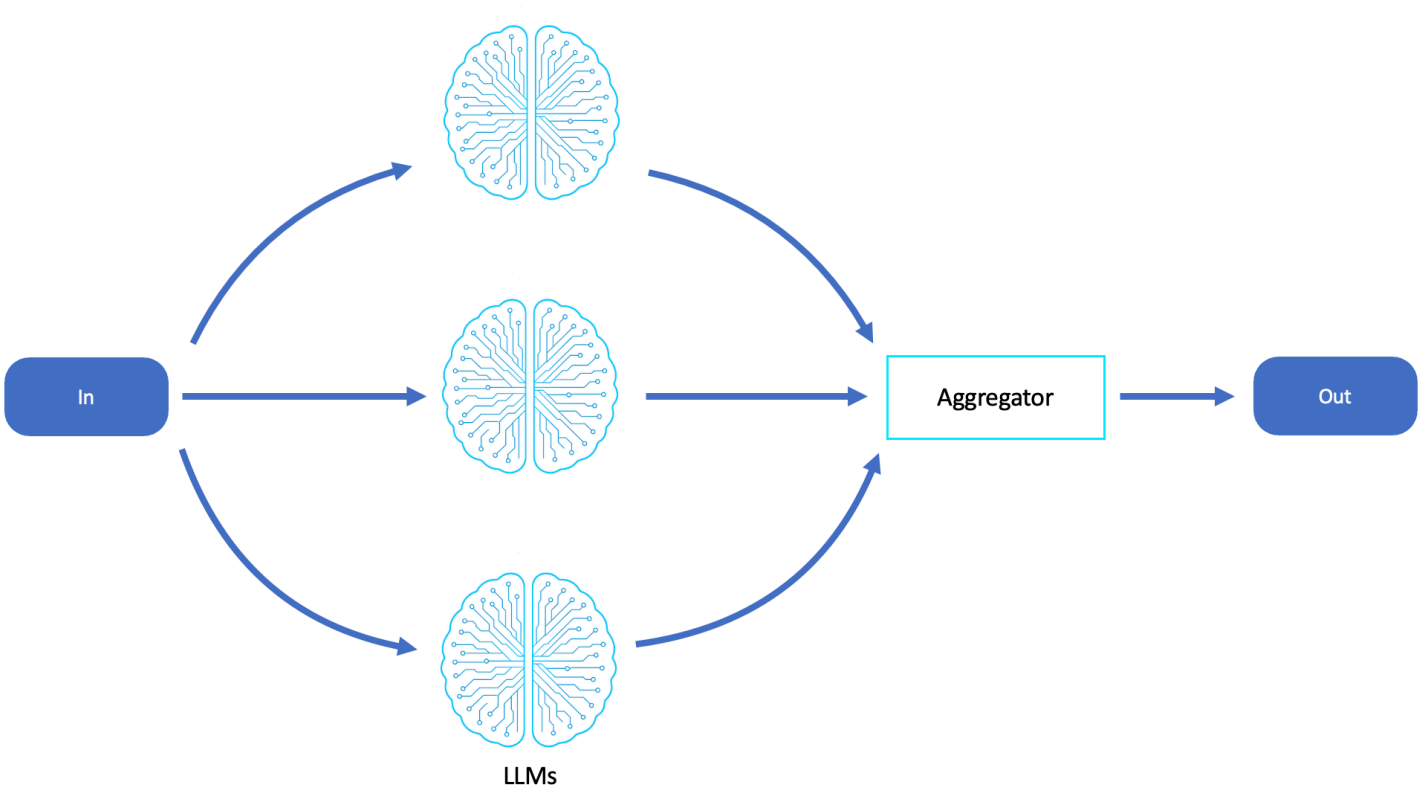
- Ein First-Pass-LM fungiert als Dispatcher
- Routen können unterschiedliche Workflows oder sogar andere Agentenmuster aufrufen
- Unterstützt die modulare Erweiterung von Funktionen

Häufige Anwendungsfälle

- Multidomain-Assistenten („Ist das eine rechtliche, medizinische oder finanzielle Frage?“)
- Die Entscheidungsbäume wurden durch LLM-Argumentation erweitert
- Dynamische Werkzeugauswahl (z. B. Suche im Vergleich zur Codegenerierung)

Workflow für die Parallelisierung

Dieser Workflow beinhaltet die Unterteilung einer Aufgabe in unabhängige Unteraufgaben, die gleichzeitig von mehreren LLM-Aufrufen oder Agenten bearbeitet werden können. Die Ergebnisse werden dann programmgesteuert aggregiert und zu einem Ergebnis zusammengefasst.



Der Parallelisierungs-Workflow wird verwendet, wenn eine Aufgabe in unabhängige, nicht sequentielle Unteraufgaben aufgeteilt werden kann, die gleichzeitig verarbeitet werden können, wodurch Effizienz, Durchsatz und Skalierbarkeit erheblich verbessert werden. Er ist besonders leistungsstark in datenintensiven, stapelorientierten oder multiperspektivischen Problembereichen, in denen der Agent Inhalte für mehrere Eingaben analysieren oder generieren muss.

Parallelisierung ist besonders effektiv, wenn:

- Teilaufgaben hängen nicht von den Zwischenergebnissen der jeweils anderen ab, sodass sie ohne Koordination parallel ausgeführt werden können.
- Eine Aufgabe besteht darin, denselben Argumentationsprozess für viele Aufgaben zu wiederholen (z. B. das Zusammenfassen mehrerer Dokumente oder das Auswerten einer Liste von Optionen).
- Mehrere Hypothesen oder Perspektiven werden parallel untersucht, um Vielfalt, Kreativität oder Robustheit zu fördern.
- Sie müssen die Latenz für Anfragen mit hohem Volumen oder hoher Frequenz durch gleichzeitige LLM-Ausführung reduzieren.
- Dieser Workflow wird häufig in Dokumentenverarbeitungsagenten, Umfrage- oder Vergleichs-Engines, Batchzusammenfassungen, Brainstormern mit mehreren Agenten und skalierbaren Klassifizierungs- oder Kennzeichnungsaufgaben verwendet, insbesondere dort, wo schnelles, paralleles Denken einen Leistungsvorteil darstellt.

Funktionen

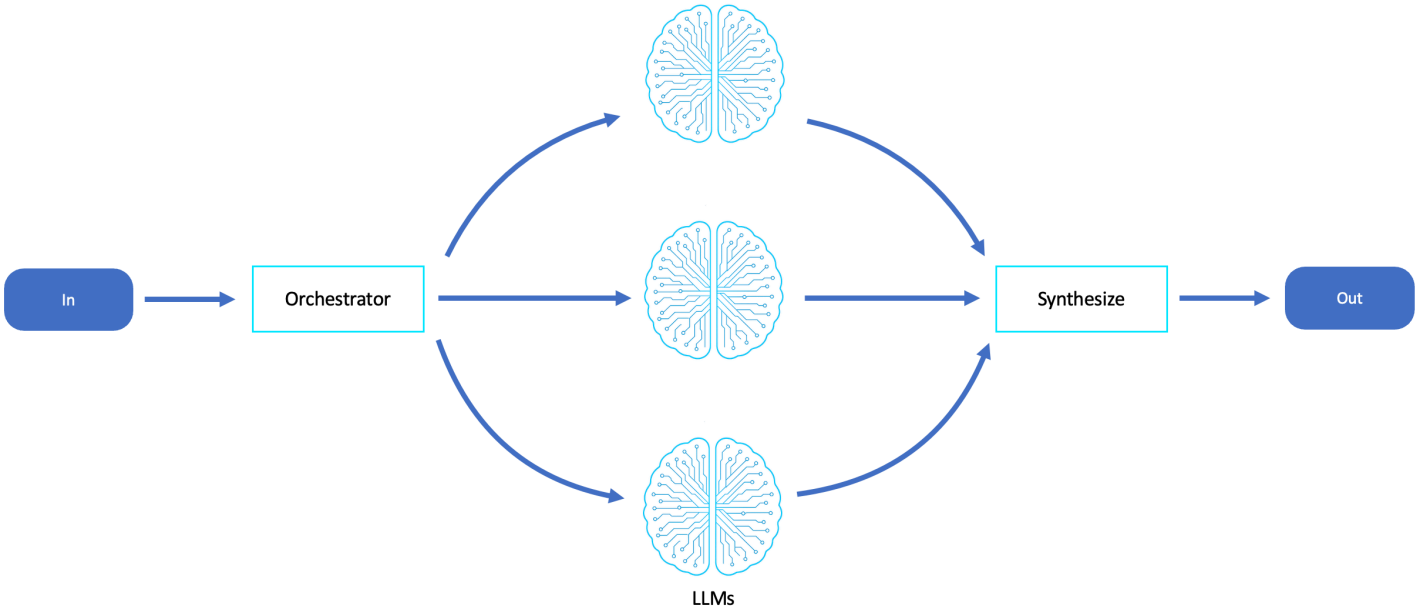
- Parallele Ausführung von LLM-Aufgaben (mithilfe von, oder einem Map-Status) AWS Lambda AWS Fargate AWS Step Functions
- Erfordert die Abstimmung, Validierung oder Deduplizierung der Ergebnisse in der Synthesephase
- Gut geeignet für Stateless Agent-Loops

Häufige Anwendungsfälle

- parallel Analyse mehrerer Dokumente oder Perspektiven
- Generierung verschiedener Entwürfe, Zusammenfassungen oder Pläne
- Beschleunigung des Durchsatzes bei Batch-Aufträgen

Workflow für die Orchestrierung

Ein zentraler Orchestrator-Agent verwendet ein LLM, um Unteraufgaben zu planen, zu zerlegen und an spezialisierte Worker Agents oder Modelle zu delegieren, die jeweils über eine bestimmte Rolle oder Fachkompetenz verfügen. Dies spiegelt die menschlichen Teamstrukturen wider und unterstützt aufkommendes Verhalten mehrerer Agenten.



Der Orchestrierungs-Workflow ist ideal für komplexe, hierarchische oder multidisziplinäre Szenarien, die eine strukturierte Zerlegung und spezielle Ausführung erfordern. Er eignet sich besonders gut für Aufgaben, die Arbeitsteilung erfordern und bei denen verschiedene Teilkomponenten einer Aufgabe am besten von Mitarbeitern mit unterschiedlichen Fähigkeiten, Kenntnissen oder Toolsets erledigt werden können.

Dieser Workflow ist besonders effektiv, wenn:

- Aufgaben können in Unteraufgaben unterteilt werden, die sich in Umfang, Art oder Argumentation unterscheiden (z. B. Planung, Recherche, Implementierung und Test).
- Ein LLM oder Metaagent muss andere Agenten koordinieren, den Fortschritt überwachen und die Ergebnisse zusammenfassen.
- Sie möchten die Zuständigkeiten der Agenten modularisieren, um Skalierbarkeit, Wiederverwendung und spezielle Anpassungen zu ermöglichen.
- Das System erfordert ein rollenbasiertes Verhalten, das nachahmt, wie menschliche Teams (z. B. Projektmanager, Entwickler und Prüfer) zusammenarbeiten.

Die Orchestrierung ist ideal für Multiturn-Planungsagenten, Co-Piloten für die Softwareentwicklung, Prozessagenten im Unternehmen und autonome Projektausführer. Sie ist besonders nützlich bei der Implementierung von Systemen mit mehreren Agenten, die eine zentrale Aufschlüsselung der Aufgaben, aber eine verteilte Ausführungslogik erfordern, wodurch Erweiterbarkeit und ein besser erklärbares Verhalten auf allen Agentenebenen ermöglicht werden.

Funktionen

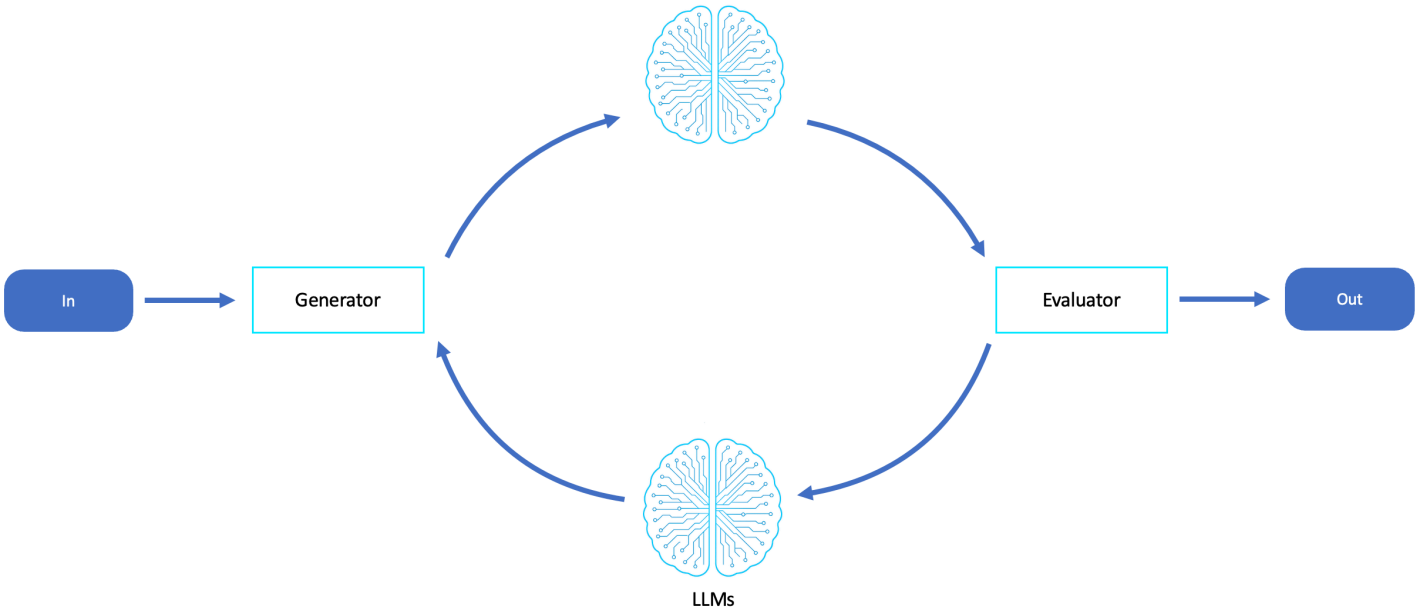
- Orchestrator führt Meta-Argumentation zum Ziel durch
- Worker-Agents können Toolzugriff, Speicher- oder domänenspezifische Eingabeaufforderungen beinhalten
- Kann hierarchisch sein (d. h. mehrstufige Aufgabendelegierung)

Häufige Anwendungsfälle

- Projektmanager, koordinierende Forscher, Autoren und Qualitätssicherungsbeauftragte
- Codierungs-Copiloten, die Planung, Ausführung und Testen kombinieren
- Agenten, die Toolchains oder API-Zugriffsmuster überwachen

Arbeitsablauf für Evaluatoren und Reflect-Refine-Schleifen

Dieser Workflow bietet eine Feedback-Schleife, in der ein LLM ein Ergebnis generiert und ein anderer das Ergebnis bewertet oder kritisiert. Dies fördert Selbstreflexion, Optimierung und iterative Verbesserungen.



Der Arbeitsablauf für Evaluatoren ist ideal für Szenarien, in denen Qualität, Genauigkeit und Ausrichtung der Ergebnisse wichtig sind und in denen die Generierung in einem Durchgang unzuverlässig oder unzureichend ist. Dieser Workflow eignet sich hervorragend, wenn Agenten ihre Ergebnisse selbst kritisieren, iterieren und verfeinern müssen — entweder, um einen höheren Korrektheitsstandard zu erreichen oder um auf der Grundlage von Feedback verbesserte Alternativen zu untersuchen.

Dieser Workflow ist besonders effektiv, wenn:

- Das Ergebnis umfasst subjektive Qualitätskennzahlen (z. B. Stil, Ton und Lesbarkeit) oder objektive Kriterien (z. B. Richtigkeit, Sicherheit und Leistung).
- Der Mitarbeiter muss Kompromisse abwägen, Einschränkungen abwägen oder auf ein Ziel hin optimieren.
- Sie benötigen integrierte Redundanz und Qualitätssicherung, insbesondere in regulierten, kundenorientierten oder kreativen Bereichen.
- Human-in-the-loop Eine Überprüfung ist teuer oder nicht verfügbar, und eine unabhängige Validierung ist erwünscht.

Dieser Workflow wird für die Generierung von Inhalten, die Codesynthese und -überprüfung, die Durchsetzung von Richtlinien, die Überprüfung der Ausrichtung, die Anpassung von Anweisungen und die RAG-Nachbearbeitung verwendet. Er eignet sich auch für Mitarbeiter, die sich selbst

verbessern, da kontinuierliches Feedback dazu beiträgt, im Laufe der Zeit bessere Antworten zu entwickeln, um vertrauenswürdige, autonome Entscheidungsschleifen aufzubauen.

Häufige Anwendungsfälle

- Agenten im roten Team im Vergleich zu Agenten mit blauem Team
- Agenten, die Code oder Pläne generieren, auswerten und überarbeiten
- Qualitätssicherung, Erkennung von Halluzinationen und Durchsetzung von Stilen

Funktionen

- Unterstützt die entkoppelte Generierung und Auswertung mithilfe verschiedener Modelle (z. B. Claude für die Generierung und Mistral für die Bewertung)
- Feedback wird strukturiert und verwendet, um zu überarbeiteten Ergebnissen zu gelangen
- Unterstützt mehrere Iterationen oder Konvergenzschwellenwerte

Schlussfolgerung

LLMs bilden den kognitiven Kern moderner Softwareagenten, aber der Aufruf von Rohmodellen reicht nicht aus, um zielgerichtete, robuste und kontrollierbare Intelligenz zu erreichen. Um von der Generierung von Output zu strukturiertem Denken und zielgerichtetem Verhalten überzugehen, LLMs muss der Übergang in bewusste Workflow-Muster erfolgen, die definieren, wie Modelle Eingaben verarbeiten, Kontexte verwalten und Aktionen koordinieren.

LLM-Workflows bieten Grundlagen für den Aufbau des kognitiven Moduls eines Agenten:

- Prompt Chaining unterteilt komplexes Denken in modulare, überprüfbare Schritte.
- Routing ermöglicht eine intelligente Aufgabenklassifizierung und gezielte Delegierung.
- Parallelisierung beschleunigt den Durchsatz und fördert vielfältige Argumentationsmöglichkeiten.
- Die Agentenorchestrierung strukturiert die Zusammenarbeit mehrerer Agenten durch Aufgabenzerlegung und rollenbasierte Ausführung.
- Der Evaluator (Reflect-Refine-Loop) ermöglicht Selbstverbesserung, Qualitätskontrolle und Überprüfung der Ausrichtung.

Jeder Workflow stellt ein zusammensetzbares Muster dar, das an die Bedürfnisse des Agenten, die Komplexität der Aufgabe und die Erwartungen des Benutzers angepasst werden kann. Diese Workflows schließen sich nicht gegenseitig aus. Sie sind Bausteine, die häufig zu Hybridarchitekturen kombiniert werden, die dynamisches Denken, Koordination mehrerer Agenten und Zuverlässigkeit auf Unternehmensniveau unterstützen.

Wenn Sie zum nächsten Kapitel über behördliche Workflow-Muster übergehen, werden diese LLM-Workflows wieder als eingebettete Strukturen in größeren Systemen auftauchen und die Delegierung von Zielen, die Orchestrierung von Tools, Entscheidungsschleifen und die Autonomie des Lebenszyklus unterstützen. Die Beherrschung dieser LLM-Workflows ist entscheidend für die Entwicklung von Softwareagenten, die nicht nur Text vorhersagen, sondern auch logisch denken, sich anpassen und zielgerichtet handeln.

Agentäre Workflow-Muster

Agentic Workflow Patterns integrieren modulare Softwareagenten mit strukturierten LLM-Workflows (Large Language Model) und ermöglichen so autonomes Denken und Handeln. Diese Muster sind zwar von traditionellen serverlosen und ereignisgesteuerten Architekturen inspiriert, verlagern aber die Kernlogik von statischem Code auf LLM-erweiterte Agenten und sorgen so für verbesserte Anpassungsfähigkeit und kontextuelle Entscheidungsfindung. Diese Entwicklung transformiert konventionelle Cloud-Architekturen von deterministischen Systemen hin zu Systemen, die dynamische Interpretation und intelligente Erweiterung ermöglichen, wobei die grundlegenden Prinzipien der Skalierbarkeit und Reaktionsfähigkeit beibehalten werden.

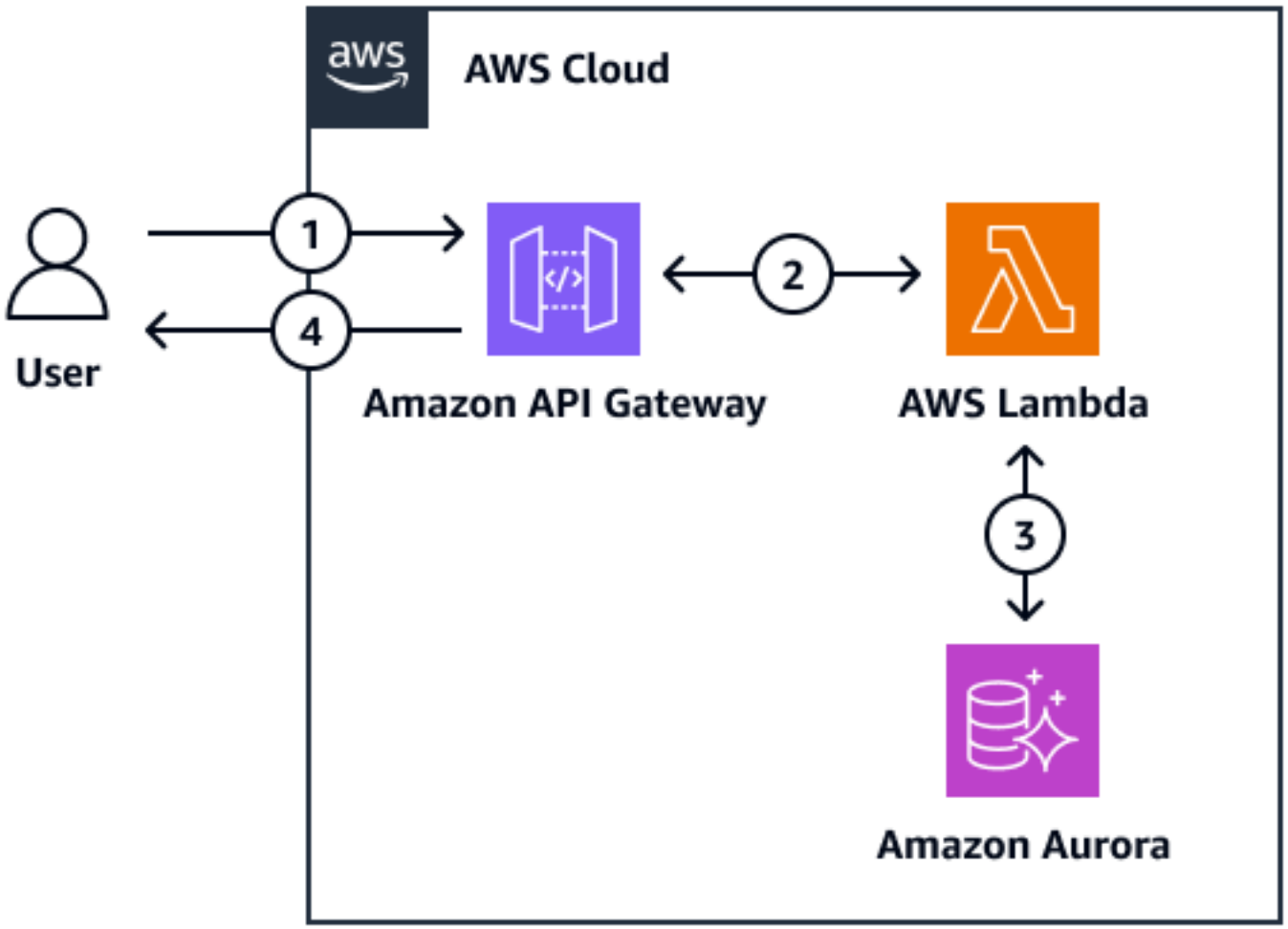
In diesem Abschnitt

Von ereignisgesteuerten zu kognitionserweiterten Systemen

Moderne Cloud-Architekturen, insbesondere solche, die auf serverlosen und ereignisgesteuerten Prinzipien basieren, verlassen sich traditionell auf Muster wie Routing, Fan-Out und Anreicherung, um reaktionsschnelle, skalierbare Systeme zu schaffen. Agentische KI-Systeme bauen auf diesen Grundlagen auf und orientieren sie gleichzeitig an LLM-gestütztem Denken und kognitiver Flexibilität. Dieser Ansatz ermöglicht ausgefeiltere Problemlösungs- und Automatisierungsfunktionen und könnte die Art und Weise, wie komplexe Aufgaben in Cloud-Umgebungen gehandhabt werden, revolutionieren.

Ereignisgesteuerte Architektur

Das folgende Diagramm zeigt ein typisches verteiltes System:

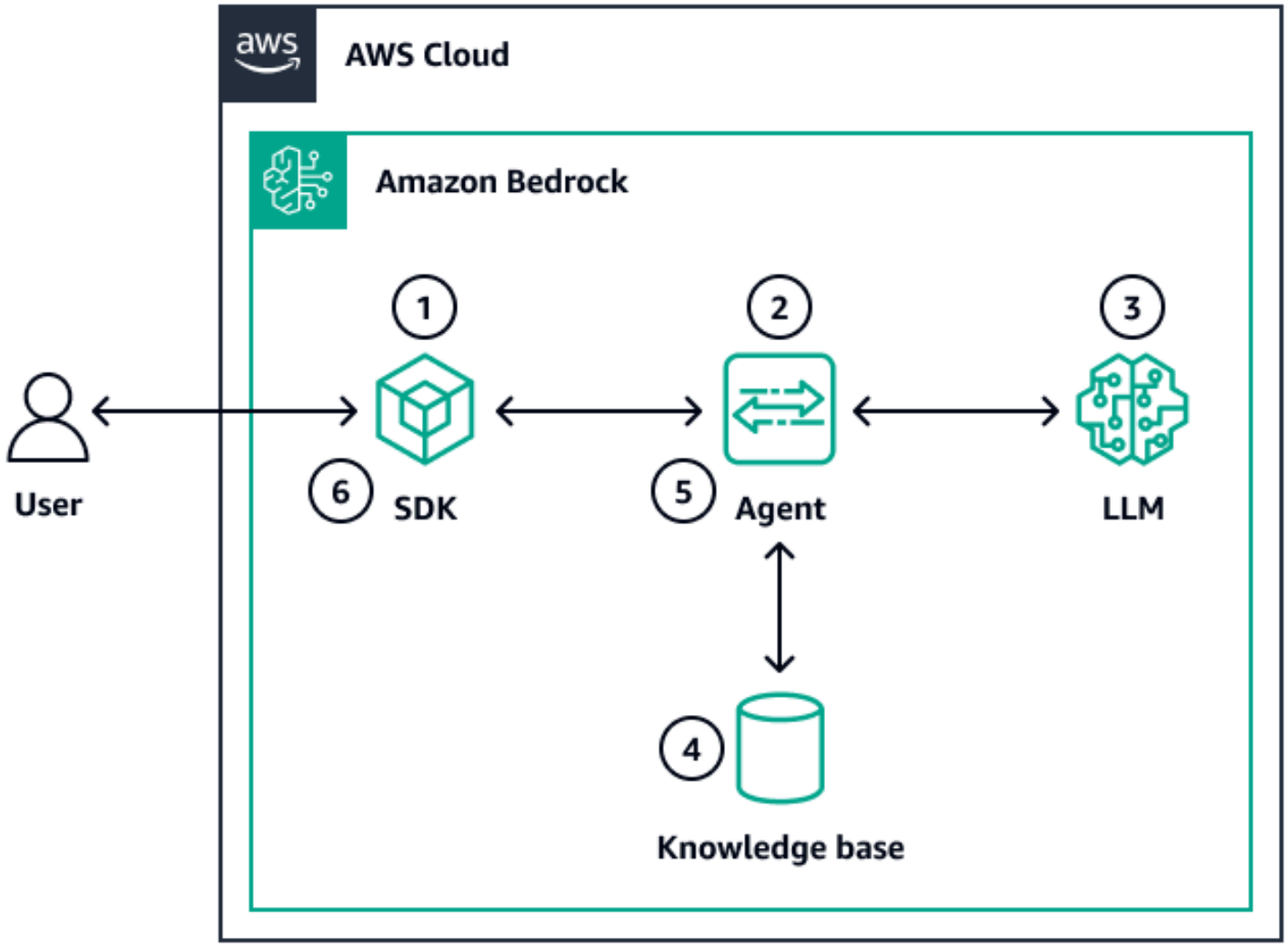


1. Ein Benutzer sendet eine Anfrage an Amazon API Gateway.
2. Amazon API Gateway leitet die Anfrage an eine AWS Lambda Funktion weiter.
3. AWS Lambda führt eine Datenanreicherung durch Abfrage einer Amazon Aurora Aurora-Datenbank durch
4. Amazon API Gateway gibt die angereicherte Nutzlast an den Aufrufer zurück.

Diese Struktur ist sowohl zuverlässig als auch skalierbar, aber sie ist grundsätzlich statisch. Geschäftsregeln und logische Pfade müssen explizit codiert werden, und die Anpassung an sich ändernde Kontexte oder unvollständige Informationen ist begrenzt.

Kognitionsgestützte Arbeitsabläufe

Agentenarchitekturen erweitern ein ereignisgesteuertes System um kognitive Augmentation. Das folgende Diagramm zeigt ein agentisches Äquivalent:



1. Ein Benutzer sendet eine Anfrage über einen SDK- oder API-Aufruf.
2. Ein Amazon Bedrock-Agent erhält die Anfrage.
3. Der Agent interpretiert die Anfrage, indem er ein LLM aufruft
4. Der Agent führt eine semantische Anreicherung durch, indem er die Amazon Bedrock-Wissensdatenbank oder andere externe Datenquellen durchsucht.
5. Das LLM synthetisiert eine kontextreiche, zielgerichtete Antwort.
6. Das System gibt dem Benutzer eine synthetisierte Antwort zurück.

In diesem Ablauf verwendet das LLM Logik, versteht die Absicht, ruft den relevanten Kontext ab, kombiniert ihn und entscheidet dann, wie am besten reagiert werden soll. Dieses Muster spiegelt das traditionelle Anreicherungsmuster wider, bei dem Nachrichten mit externen Daten angereichert werden, bevor sie weitergeleitet werden. In Agentensystemen handelt es sich bei dieser

Anreicherung jedoch nicht um eine statische Suche. Stattdessen ist die Anreicherung dynamisch, semantisch gesteuert und zielgerichtet.

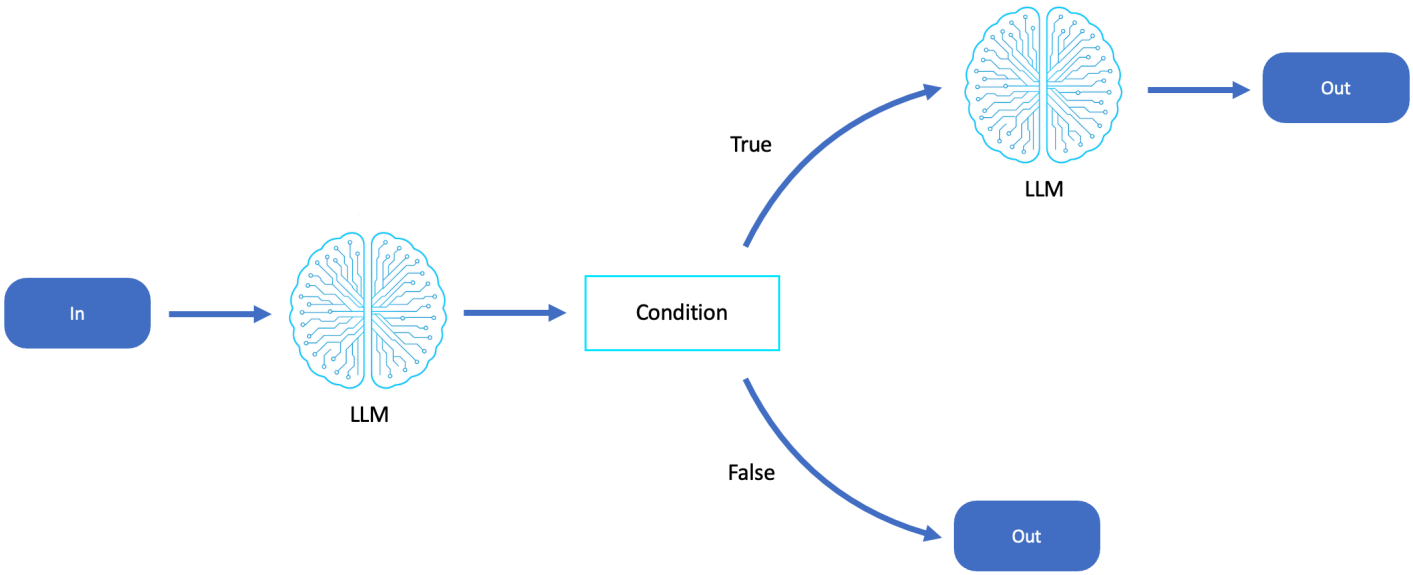
Die wichtigsten Erkenntnisse

Jeder LLM-Workflow kann einem zentralen Workflow-Muster zugeordnet werden, das traditionelle ereignisgesteuerte Architekturstile widerspiegelt und weiterentwickelt. Ein grundlegender Baustein agentischer Workflows ist die Fähigkeit, den Kontext eines LLM um Daten, Tools und Speicher zu erweitern. Dadurch entsteht eine Argumentationsschleife, die fundiert, anpassungsfähig und auf die Benutzerabsichten abgestimmt ist. Während herkömmliche Systeme Nachrichten mit Suchdaten anreichern, ermöglichen es Agentensysteme der Software, sich weniger wie Skripte, sondern eher wie intelligente Kollaborateure zu verhalten.

Sofortige Verkettung von Saga-Mustern

Indem wir LLM-Prompt-Chaining als ereignisgesteuerte Saga neu erfinden, erschließen wir ein neues Betriebsmodell: Workflows werden auf autonome Agenten verteilt, wiederherstellbar und semantisch koordiniert. Jeder Prompt-Response-Schritt wird als atomare Aufgabe neu formuliert, als Ereignis ausgegeben, von einem speziellen Agenten verarbeitet und mit kontextuellen Metadaten angereichert.

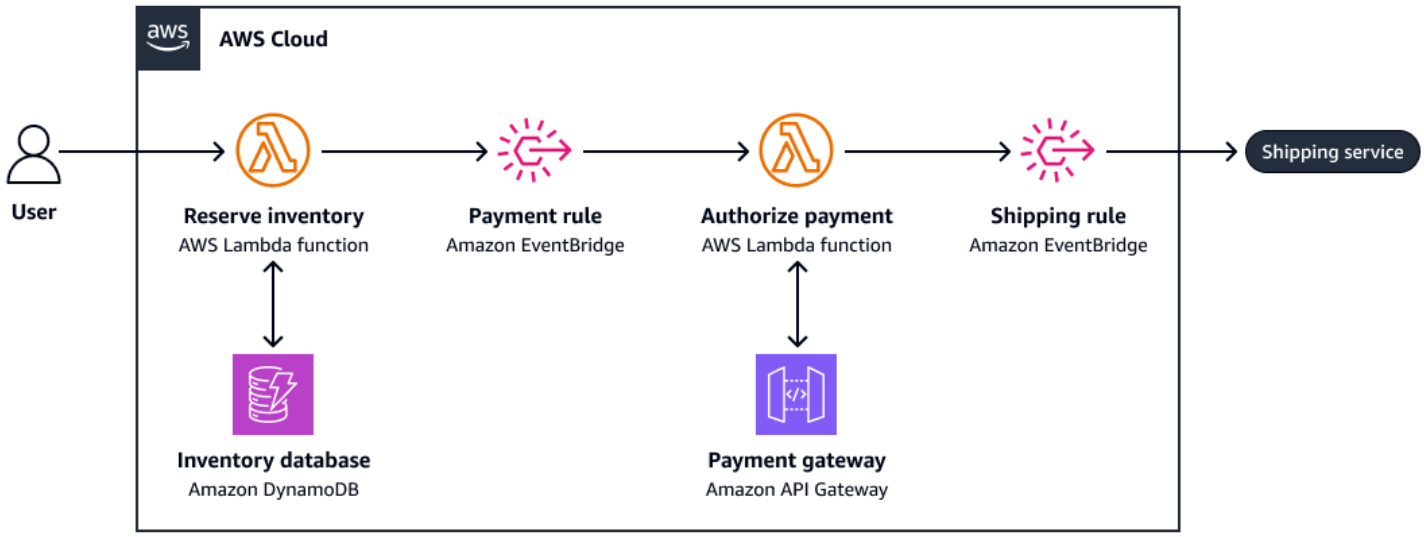
Das folgende Diagramm ist ein Beispiel für die Verkettung von LLM-Eingabeaufforderungen:



Saga-Choreographie

Das Saga-Choreographie-Muster ist ein Implementierungsansatz in verteilten Systemen, für den es keinen zentralen Koordinator gibt. Stattdessen veröffentlicht jeder Dienst oder jede Komponente Ereignisse, die die nächste Workflow-Aktion auslösen. Dieses Muster wird häufig in verteilten Systemen zur Verwaltung von Transaktionen über mehrere Dienste hinweg verwendet. In einer Saga führt das System eine Reihe koordinierter lokaler Transaktionen durch. Wenn eine davon ausfällt, löst das System Ausgleichsmaßnahmen aus, um die Konsistenz aufrechtzuerhalten.

Das folgende Diagramm ist ein Beispiel für eine Saga-Choreographie:



1. Inventar reservieren
2. Zahlung autorisieren
3. Versandauftrag erstellen

Schlägt Schritt 3 fehl, leitet das System Ausgleichsmaßnahmen ein (z. B. eine Zahlung stornieren oder Inventar freigeben).

Dieses Muster ist besonders nützlich in ereignisgesteuerten Architekturen, in denen Dienste lose miteinander verknüpft sind und Zustände im Laufe der Zeit konsistent behoben werden müssen, selbst bei teilweisem Ausfall.

Muster für die schnelle Verkettung

Die schnelle Verkettung ähnelt in ihrer Struktur und ihrem Zweck dem Saga-Muster. Es führt eine Reihe von Argumentationsschritten aus, die nacheinander aufgebaut werden, wobei der Kontext erhalten bleibt und Rollbacks und Überarbeitungen möglich sind.

Choreographie der Agenten

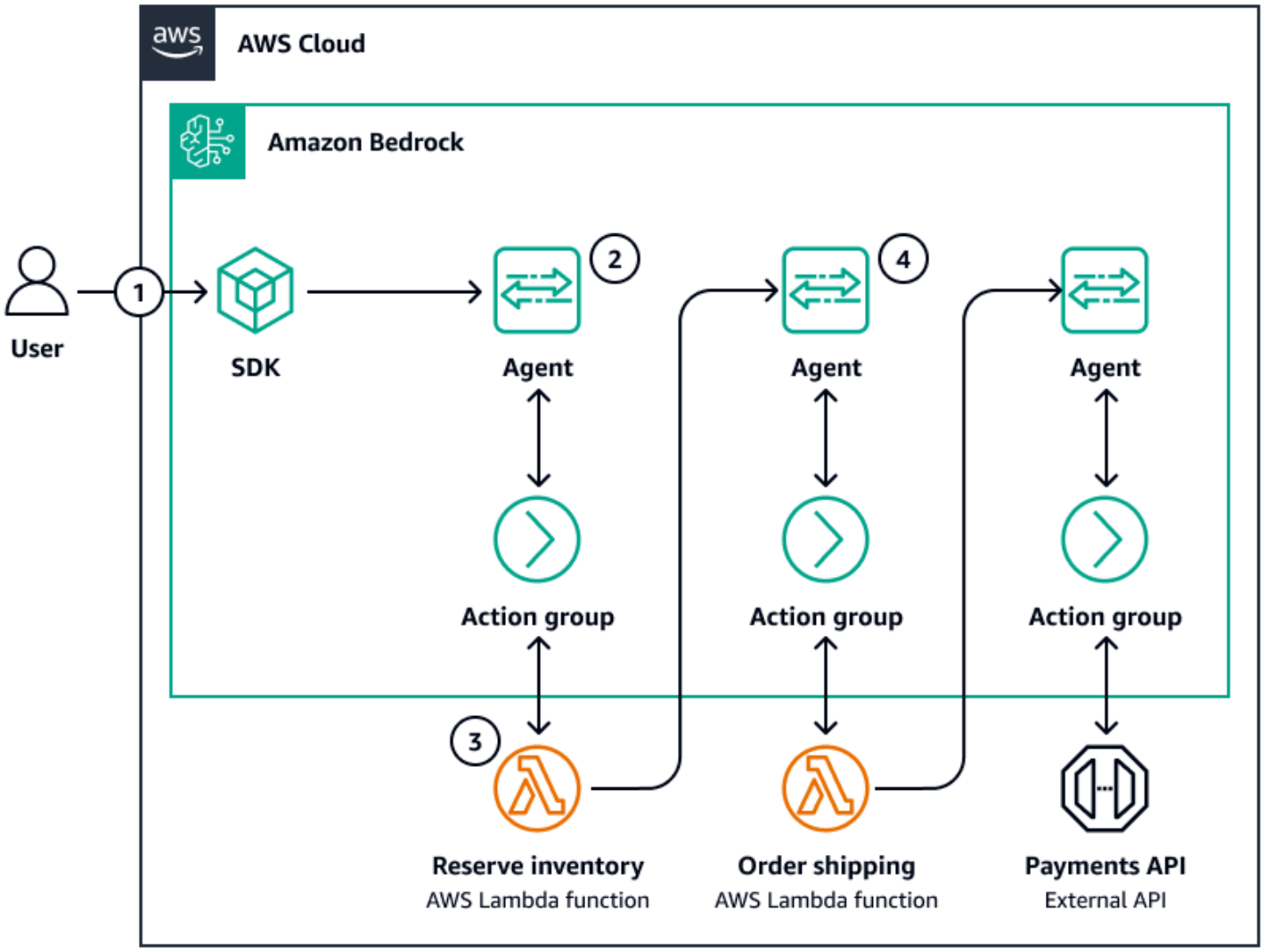
1. LLM interpretiert eine komplexe Benutzerabfrage und generiert eine Hypothese
2. LLM erarbeitet einen Plan zur Lösung der Aufgabe
3. LLM führt eine Unteraufgabe aus (z. B. mithilfe eines Tool-Aufrufs oder durch das Abrufen von Wissen)
4. LLM verfeinert die Ausgabe oder wiederholt einen früheren Schritt, wenn es ein Ergebnis für unbefriedigend hält

Wenn ein Zwischenergebnis fehlerhaft ist, kann das System einen der folgenden Schritte ausführen:

- Wiederholen Sie die Schritte mit einem anderen Ansatz
- Kehren Sie zu einer vorherigen Eingabeaufforderung zurück und planen Sie neu
- Verwenden Sie eine Evaluatorschleife (z. B. anhand des Evaluator-Optimizer-Musters), um Fehler zu erkennen und zu korrigieren

Wie beim Saga-Muster ermöglicht die Verkettung von Eingabeaufforderungen teilweise Fortschritts- und Rollback-Mechanismen. Dies geschieht eher durch iterative Verfeinerung und LLM-gesteuerte Korrektur als durch kompensierende Datenbanktransaktionen.

Das folgende Diagramm ist ein Beispiel für die Choreographie eines Agenten:



1. Ein Benutzer sendet eine Anfrage über ein SDK.
2. Ein Amazon Bedrock-Agent orchestriert die Argumentation wie folgt:
   - Dolmetschen (LLM)
   - Planung (LLM)
   - Ausführung über ein Tool oder eine Wissensdatenbank
   - Konstruktion der Antwort
3. Wenn ein Tool ausfällt oder unzureichende Daten zurückgibt, kann der Agent die Aufgabe dynamisch neu planen oder umformulieren.
4. Speicher (z. B. ein kurzfristiger Vektorspeicher) kann seinen Zustand stufenübergreifend beibehalten

Imbissbuden

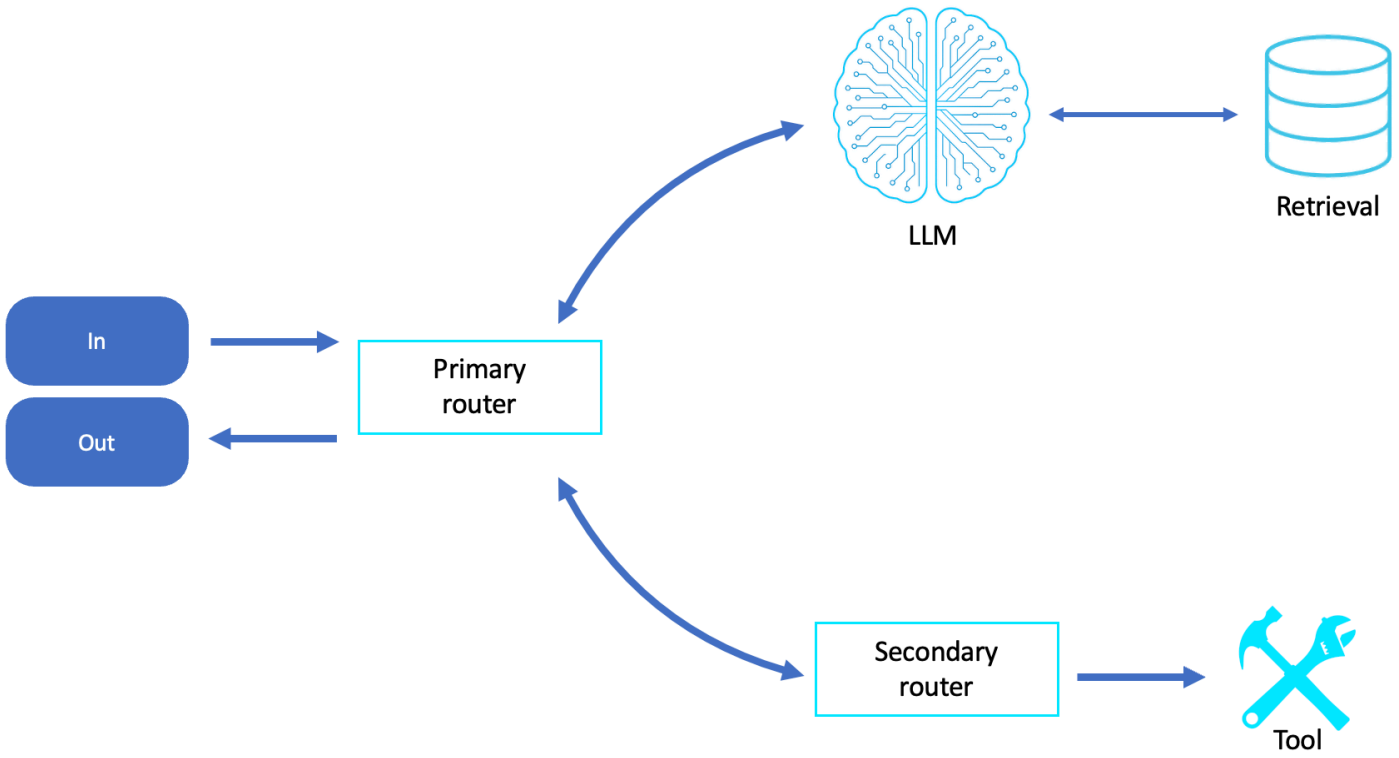
Während das Saga-Muster verteilte Serviceanrufe mit kompensierender Logik verwaltet, erledigt Prompt Chaining Denkaufgaben mit reflektierter Reihenfolge und adaptiver Neuplanung. Beide Systeme ermöglichen inkrementellen Fortschritt, dezentrale Entscheidungspunkte und die Behebung von Fehlern, und das alles durch fundierte Argumentation und nicht durch starres Rollback.

Prompt Chaining führt Transaktionsdenken ein, das kognitive Äquivalent von Sagen ist. Das heißt, jeder „Gedanke“ wird im Rahmen eines umfassenderen zielgerichteten Dialogs neu bewertet, überarbeitet oder aufgegeben.

Weiterleitung dynamischer Versandmuster

In modernen Agentensystemen, in denen die Aufgaben vom Analysieren von Dokumenten bis hin zur autonomen Softwaregenerierung reichen, ist die Fähigkeit, Anfragen dynamisch an das leistungsfähigste Large Language Model (LLM) oder den leistungsfähigsten Agenten weiterzuleiten, von entscheidender Bedeutung. Der statischen Routing-Logik, die häufig in Orchestrierungsskripten oder API-Ebenen eingebettet ist, fehlt die Anpassungsfähigkeit, die für Umgebungen mit mehreren Modellen und Funktionen in Echtzeit erforderlich ist. Um diesem Problem zu begegnen, können LLM-Routing-Workflows in eine ereignisgesteuerte Architektur umgewandelt werden, die ein dynamisches Versandmuster nutzt und LLM-Aufrufe in intelligent geroutete, kontextsensitive Ereignisse umwandelt.

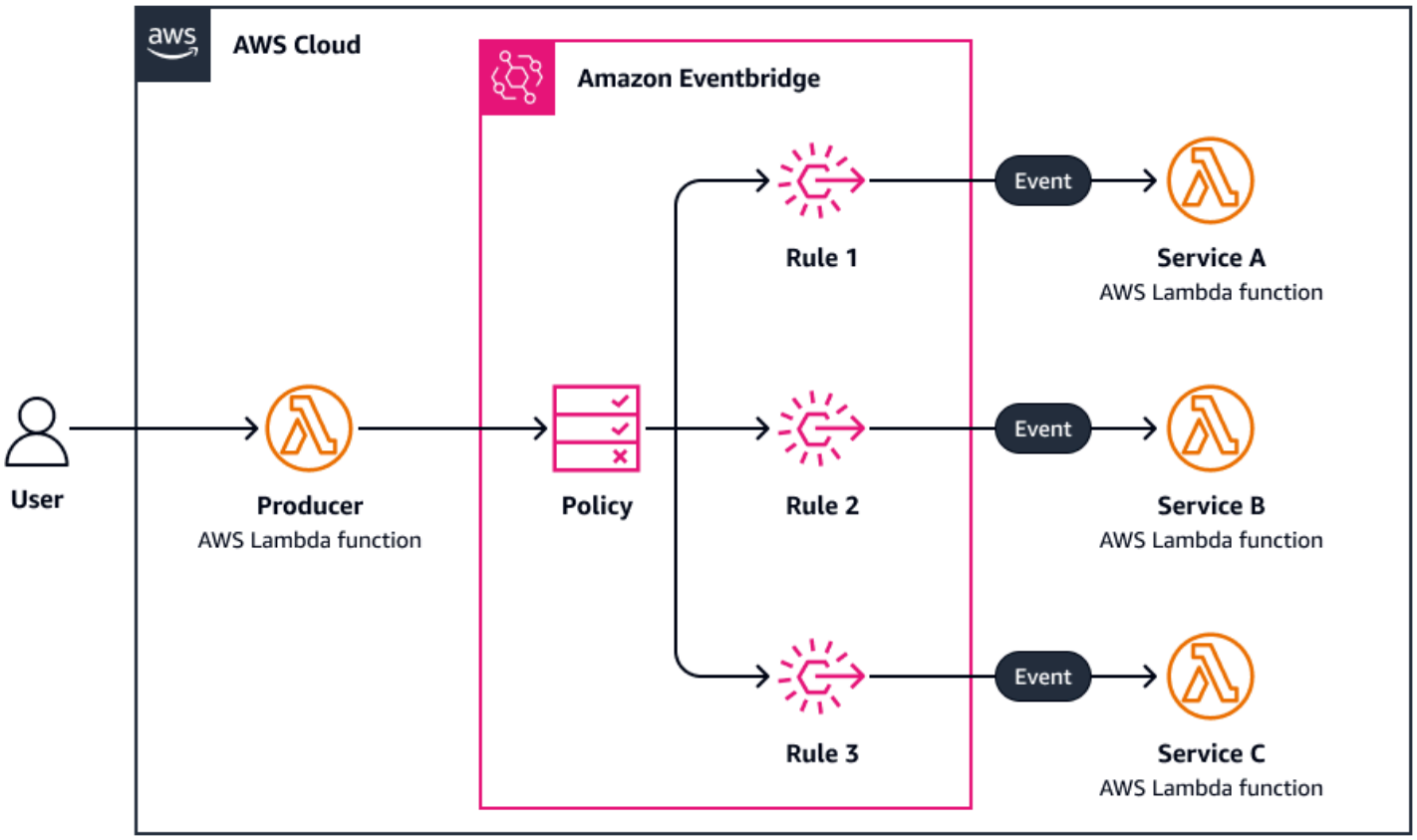
Das folgende Diagramm ist ein Beispiel für LLM-Routing:



Dynamischer Versand

In herkömmlichen verteilten Systemen wählt das dynamische Versandmuster bestimmte Dienste zur Laufzeit aus und ruft sie auf der Grundlage eingehender Ereignisattribute wie Ereignistyp, Quelle und Nutzlast aus. Dies wird üblicherweise mit Amazon implementiert EventBridge, das eingehende Ereignisse auswerten und an geeignete Ziele weiterleiten kann (z. AWS Step Functions B. AWS Lambda Funktionen oder Amazon Elastic Container Service-Aufgaben).

Das folgende Diagramm ist ein Beispiel für dynamischen Versand:



1. Eine Anwendung gibt ein Ereignis aus (z. B. {"type“: „orderCreated“, „priority“: „high"}).
2. Amazon EventBridge bewertet das Ereignis anhand seiner Routing-Regeln.
3. Basierend auf den Attributen eines Ereignisses sendet das System dynamisch an Folgendes:
   - `HighPriorityOrderProcessor`(Dienst A)
   - `StandardOrderProcessor`(Dienst B)
   - `UpdateOrderProcessor`(Dienst C)

Dieses Muster unterstützt lose Kopplung, domänenbasierte Spezialisierung und Laufzeiterweiterbarkeit. Dadurch können Systeme intelligent auf sich ändernde Anforderungen und die Semantik von Ereignissen reagieren.

LLM-basiertes Routing

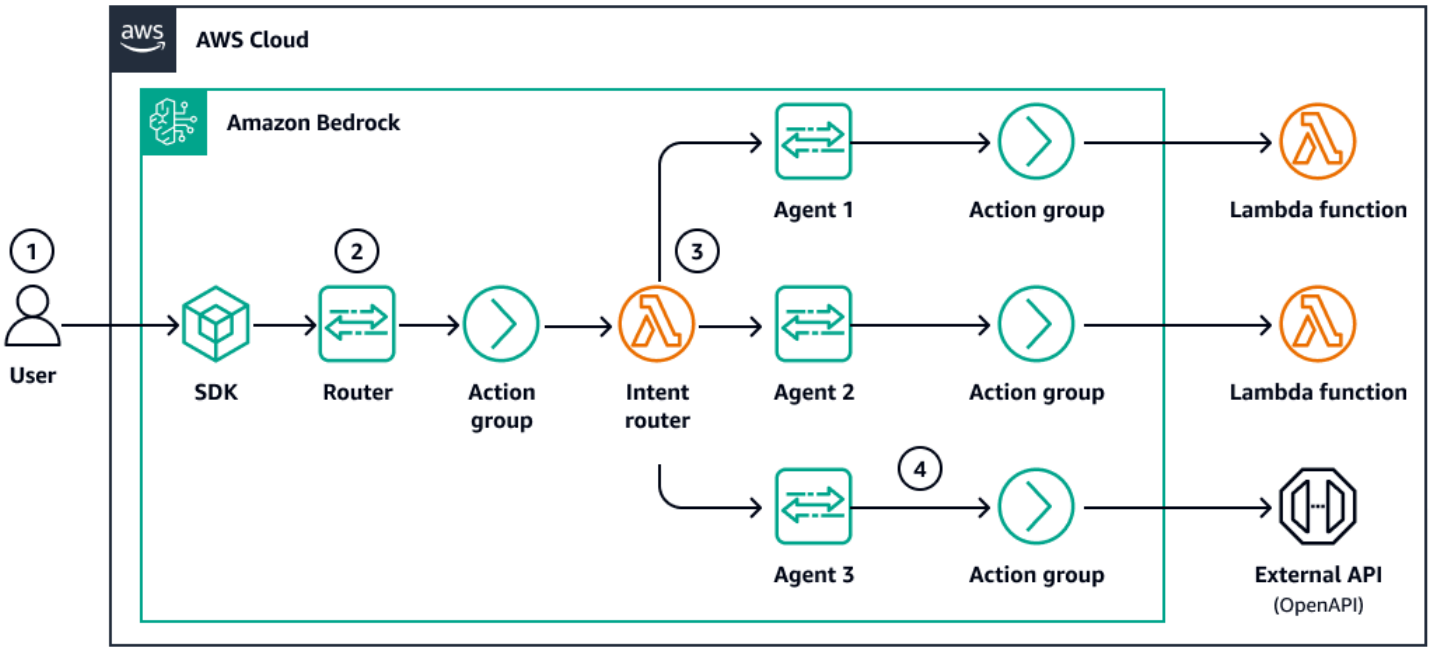
In Agentensystemen führt das Routing auch eine dynamische Aufgabendelegierung durch — aber anstelle von EventBridge Amazon-Regeln oder Metadatenfiltern klassifiziert und interpretiert das LLM die Absicht des Benutzers in natürlicher Sprache. Das Ergebnis ist eine flexible, semantische und anpassungsfähige Form des Dispatchings.

Agenten-Router

Diese Architektur ermöglicht ein funktionsorientiertes Dispatching ohne vordefinierte Schemas oder Ereignistypen, was sich ideal für unstrukturierte Eingaben und komplexe Abfragen eignet.

1. Ein Benutzer sendet die Anfrage „Können Sie mir helfen, meine Vertragsbedingungen zu überprüfen?“
2. Das LLM interpretiert dies als eine Aufgabe für ein rechtliches Dokument.
3. Der Agent leitet die Aufgabe an eine oder mehrere der folgenden Stellen weiter:
   - Vorlage für die Aufforderung zur Vertragsprüfung
   - Subagent für rechtliche Überlegungen
   - Tool zum Analysieren von Dokumenten

Das folgende Diagramm ist ein Beispiel für einen Agent-Router:



1. Ein Benutzer sendet eine Anfrage in natürlicher Sprache über ein SDK.
2. Ein Amazon Bedrock-Agent verwendet ein LLM, um die Aufgabe zu klassifizieren (z. B. rechtlich, technisch oder terminlich).
3. Der Agent leitet die Aufgabe dynamisch über eine Aktionsgruppe weiter, um den erforderlichen Agenten aufzurufen:
   - Domänenspezifischer Agent

- Spezialisierte Werkzeugkette
- Benutzerdefinierte Prompt-Konfiguration

4. Der ausgewählte Handler verarbeitet die Aufgabe und gibt eine maßgeschneiderte Antwort zurück.

Imbissbuden

Während herkömmlicher dynamischer Versand EventBridge Amazon-Regeln für die Weiterleitung auf der Grundlage strukturierter Ereignisattribute verwendet, verwendet LLMs agentisches Routing die semantische Klassifizierung und Weiterleitung von Aufgaben auf der Grundlage von Bedeutung und Absicht. Dies erweitert die Flexibilität des Systems, indem es Folgendes ermöglicht:

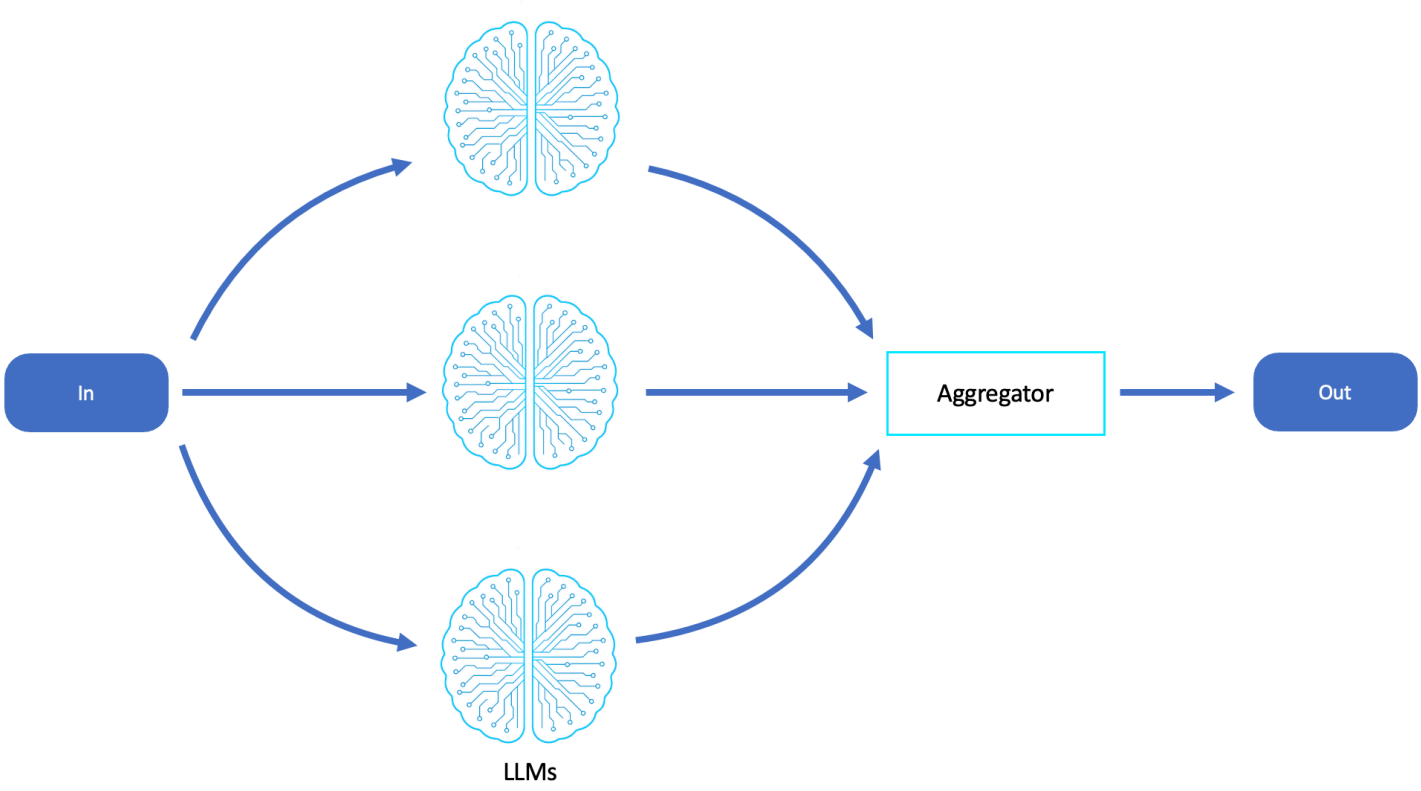
- Umfassenderes Verständnis der
- Intelligenter Fallback und Toolauswahl
- Natürliche Erweiterbarkeit durch neue Agentenrollen oder Eingabeaufforderungsstile

Agentic Routing ersetzt starre Regeln durch dynamisches kognitives Dispatching, das es Systemen ermöglicht, sich mit Sprache und nicht mit Code weiterzuentwickeln.

Parallelisierung und Scatter-Gather-Muster

Viele fortgeschrittene Denk- und Generierungsaufgaben — wie das Zusammenfassen großer Dokumente, die Bewertung mehrerer Lösungswege oder der Vergleich verschiedener Perspektiven — profitieren von der parallel Ausführung von Eingabeaufforderungen. Herkömmliche sequentielle Workflows reichen nicht aus, wenn Skalierbarkeit, Reaktionsfähigkeit und Fehlertoleranz erforderlich sind. Um dieses Problem zu lösen, kann die LLM-basierte Parallelisierung mithilfe eines ereignisgesteuerten Scatter-Gather-Musters neu konzipiert werden, bei dem Aufgaben dynamisch an autonome Agenten verteilt und die Ergebnisse intelligent synthetisiert werden.

Das folgende Diagramm ist ein Beispiel für einen LLM-Parallelisierungs-Workflow:



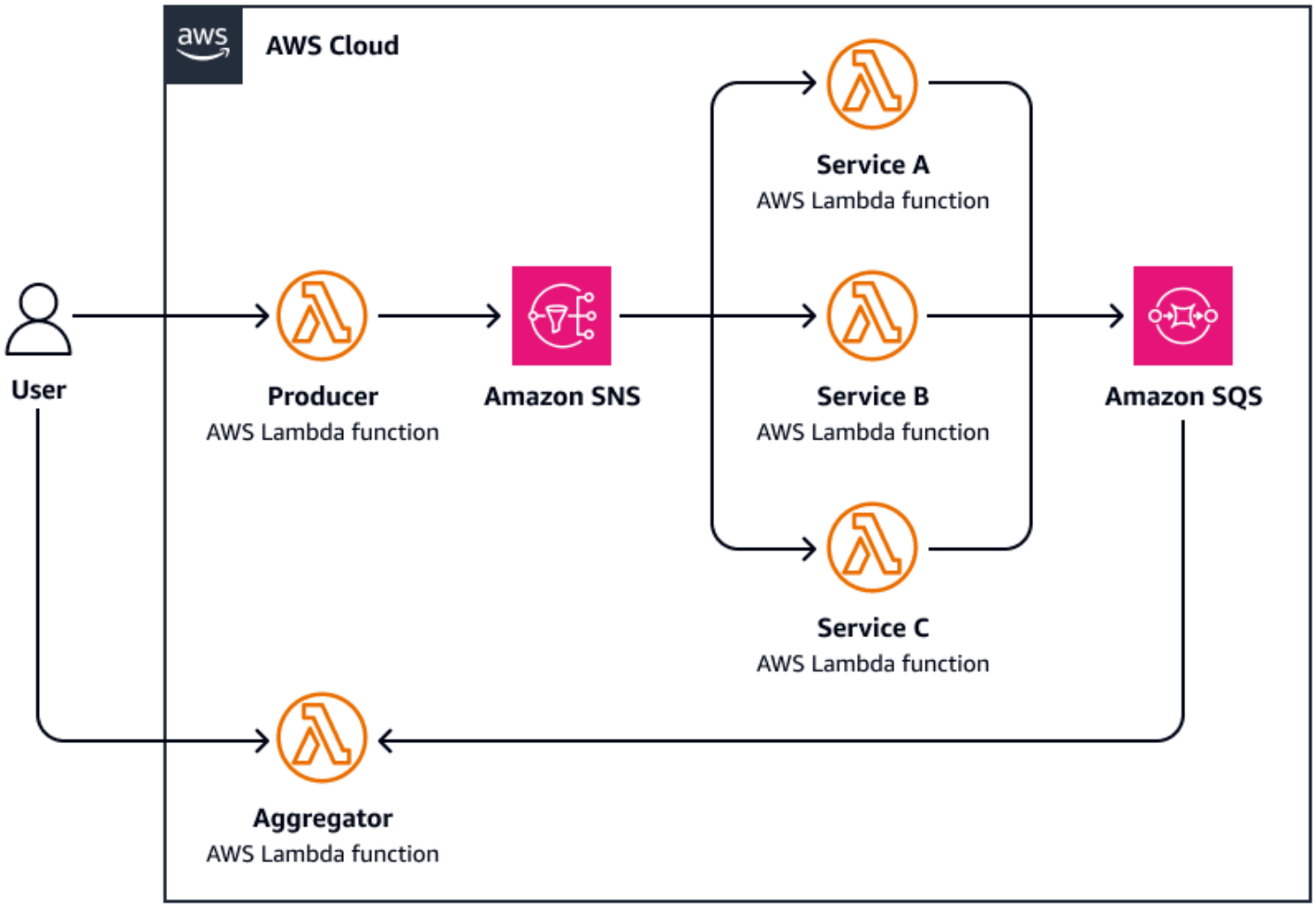
Verstreuen und sammeln

In verteilten Systemen sendet ein Scatter-Gather-Muster Aufgaben parallel an mehrere Dienste oder Verarbeitungseinheiten, wartet auf ihre Antworten und aggregiert dann die Ergebnisse zu einer konsolidierten Ausgabe. Im Gegensatz zu Fan-Out wird Scatter-Gathering koordiniert, weil es Antworten erwartet und in der Regel Logik anwendet, um Ergebnisse zu kombinieren, zu vergleichen und auszuwählen.

Zu den gängigen Implementierungen für Parallelisierung und Scatter-Gather gehören die folgenden:

- AWS Step Functions einen Status für die parallel Aufgabenausführung zuordnen
- AWS Lambda mit Parallelität, Koordination der Ergebnisse mehrerer aufgerufener Funktionen
- Amazon EventBridge mit Korrelations IDs - und Aggregationsworkflows
- Benutzerdefiniertes Controller-Muster zur Verwaltung von Fan-Outs und zur Erfassung von Ergebnissen mithilfe von Amazon Simple Storage Service (Amazon S3), Amazon DynamoDB oder Warteschlangen

Das folgende Diagramm ist ein Beispiel für Scatter-Gather:



1. Ein Benutzer sendet eine Anfrage an eine zentrale Koordinatorfunktion, die die Aufgabe verteilt, indem sie parallel Nachrichten zu einem Amazon Simple Notification Service (Amazon SNS) - Thema veröffentlicht.
2. Jede Nachricht enthält Aufgabenmetadaten und wird an einen Facharbeiter weitergeleitet. AWS Lambda
3. Jeder Worker verarbeitet AWS Lambda unabhängig die ihm zugewiesene Unteraufgabe (z. B. das Abfragen einer externen API, das Verarbeiten eines Dokuments und das Analysieren von Daten).
4. Die Ergebnisse werden auf eine gemeinsame Speicherebene wie Amazon Simple Queue Service (Amazon SQS) geschrieben.
5. Die Aggregatorfunktion wartet, bis alle Antworten abgeschlossen sind, und führt dann Folgendes aus:
   - Sammelt und aggregiert die Ergebnisse (führt beispielsweise Zusammenfassungen zusammen und wählt die besten Treffer aus)

- Sendet eine endgültige Antwort oder löst einen nachgelagerten Workflow aus

Zu den häufigsten Anwendungsfällen für Scatter-Gather-Muster gehören:

- Föderierte Suche
- Suchmaschinen für Preisvergleiche
- Aggregierte Datenanalyse
- Inferenz mit mehreren Modellen

LLM-basierte Parallelisierung (Scatter-Gather-Kognition)

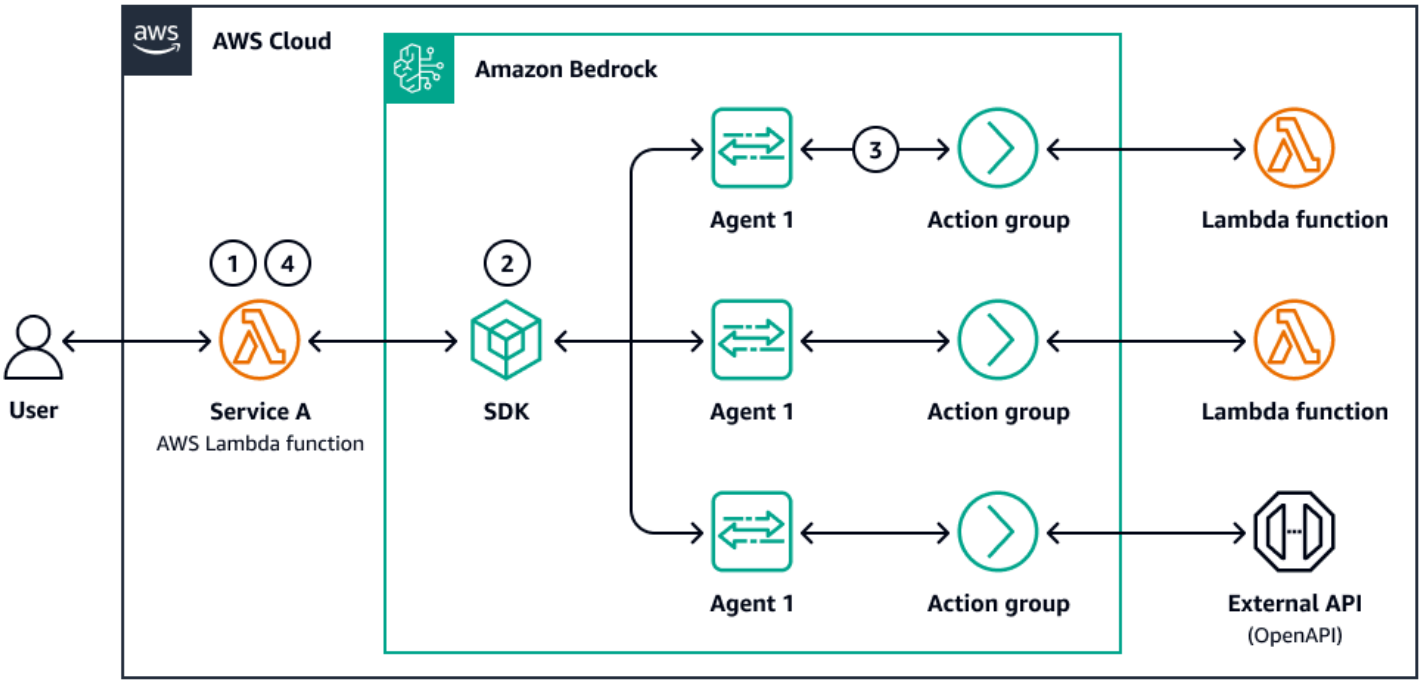
In Agentensystemen spiegelt die Parallelisierung stark das Scatter-Gather-Verfahren wider, indem Teilaufgaben auf mehrere LLM-Aufrufe oder -Agenten verteilt werden, die jeweils unabhängig voneinander einen Teil des Problems lösen. Die zurückgegebenen Ergebnisse werden durch einen Aggregationsprozess gesammelt und synthetisiert, bei dem es sich häufig um einen anderen LLM- oder Controller-Agenten handelt.

Parallelisierung der Agenten

1. Ein Agent reicht die Anfrage „Zusammenfassung der Erkenntnisse aus diesen 10 Berichten“ ein.
2. Es verteilt die Berichte auf 10 parallel LLM-Zusammenfassungsaufgaben.
3. Wenn er alle Zusammenfassungen zurückgibt, geht der Agent wie folgt vor:
   - Fasst Zusammenfassungen zu einem einheitlichen Briefing zusammen
   - Identifiziert Themen oder Widersprüche
   - Sendet die synthetisierte Ausgabe an den Benutzer

Dieser agentische Workflow ermöglicht skalierbares, modulares und adaptives paralleles Denken. Dies ist ideal für Anwendungsfälle, die einen hohen kognitiven Durchsatz erfordern.

Das folgende Diagramm ist ein Beispiel für die Parallelisierung von Agenten:



1. Ein Benutzer reicht eine mehrteilige Abfrage oder einen Dokumentensatz ein.
2. Eine Controller AWS Lambda - oder Schrittfunktion verteilt die Unteraufgaben. Jede Aufgabe ruft einen Amazon Bedrock LLM-Aufruf oder -Subagent mit eigener Aufforderung auf.
3. Wenn die Aufrufe und Unteraufgaben abgeschlossen sind, werden die Ergebnisse gespeichert (z. B. in Amazon S3 oder im Speicherspeicher), und ein Aggregationsschritt führt die Ausgaben zusammen, vergleicht oder filtert sie.
4. Das System sendet die endgültige Antwort an den Benutzer oder den nachgeschalteten Agenten zurück.

Dieses System verfügt über eine verteilte Argumentationsschleife mit Rückverfolgbarkeit, Fehlertoleranz und optionaler Ergebnisgewichtungs- oder Auswahllogik.

Imbissbuden

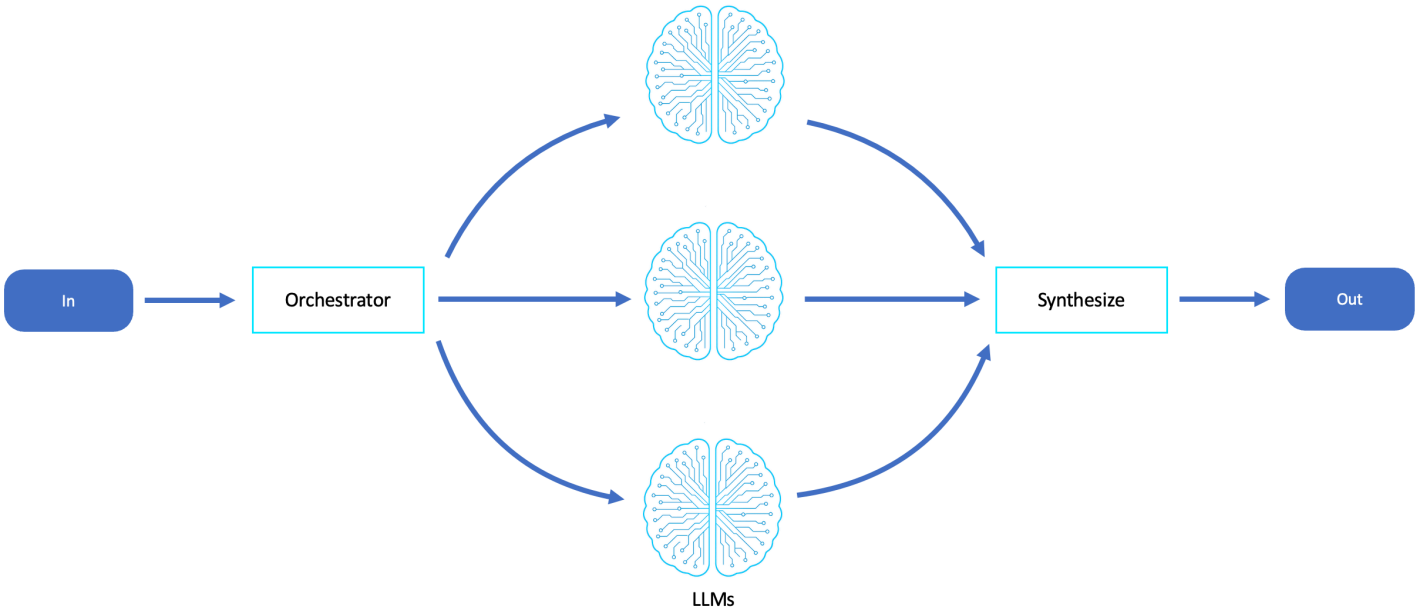
Die Agentenparallelisierung verwendet Scatter-Gather-Muster, um LLM-Aufgaben zu verteilen, was eine parallel Verarbeitung und eine intelligente Ergebnissynthese ermöglicht.

Orchestrierungsmuster von Saga

Da Workflows, die darauf basieren, immer komplexer LLMs werden und Eingabeaufforderungsketten, Datenverarbeitungsschritte, Tool-Aufrufe und Zusammenarbeit mit Agenten umfassen, wird die

Notwendigkeit einer intelligenten Orchestrierung immer wichtiger. Anstatt sich auf eng gekoppelte Skripte oder statische, vorgegebene Ausführungsabläufe zu verlassen, können diese Workflows als ereignisgesteuerte Orchestrierungsmuster implementiert werden, sodass LLM-basierte Systeme mehrstufige Aufgaben zwischen autonomen Agenten dynamisch koordinieren, überwachen und anpassen können.

Das folgende Diagramm ist ein Beispiel für einen Orchestrator:



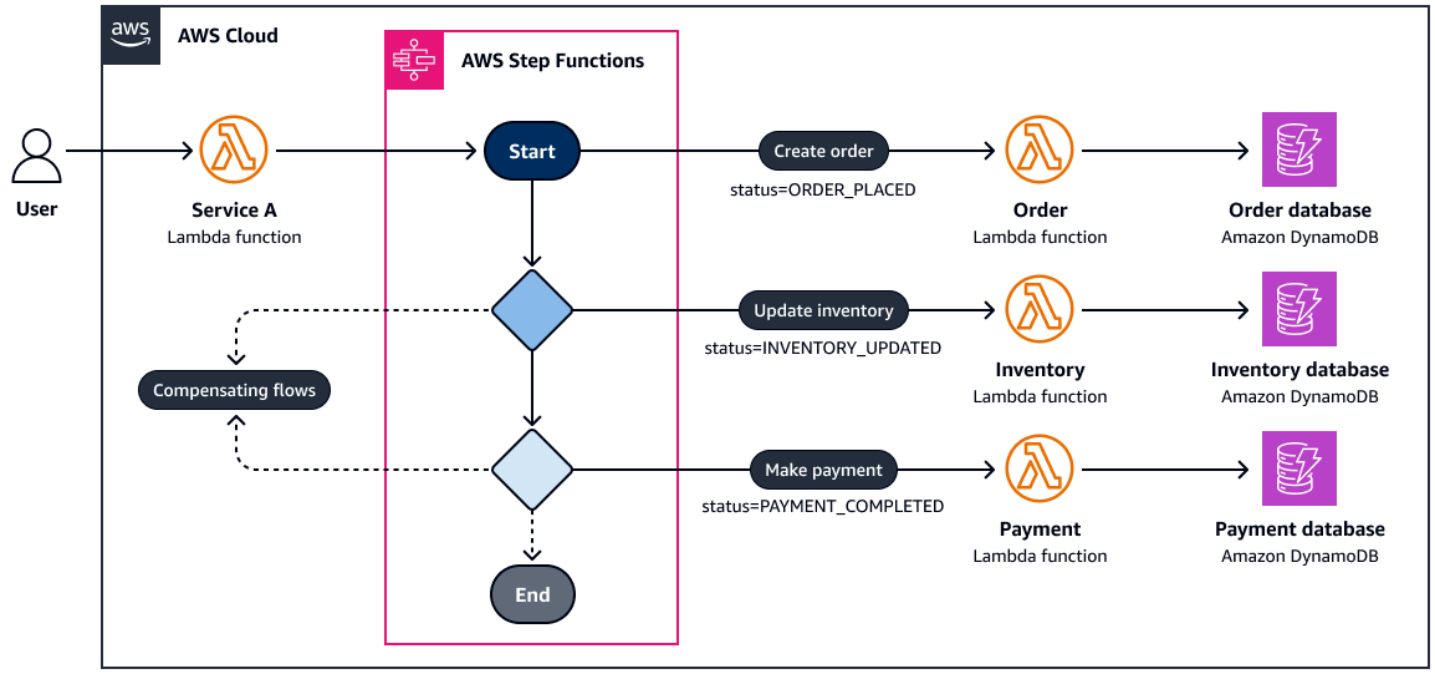
Orchestrierung von Veranstaltungen

In herkömmlichen verteilten Systemen bezieht sich Event Orchestration auf ein Muster, bei dem ein zentraler Koordinator einen komplexen Arbeitsablauf verwaltet, indem er den Kontrollfluss explizit über mehrere Dienste oder Aufgaben hinweg lenkt. Im Gegensatz zur Event-Choreographie (bei der jeder Service unabhängig reagiert) bietet die Orchestrierung eine zentralisierte Logik, Transparenz und Kontrolle über den gesamten Prozess.

Dies wird in der Regel mit den folgenden Tools implementiert:

- AWS Step Functions— Stateful-Workflows definieren und ausführen
- AWS Lambda— Führen Sie diskrete Aufgaben innerhalb des orchestrierten Ablaufs aus
- Amazon SQS oder Amazon EventBridge — Löst asynchrone Schritte oder Antworten aus

Das folgende Diagramm ist ein Beispiel für die Saga-Orchestrierung:



Ein AWS Step Functions Workflow verwaltet einen Kundenbestellvorgang:

1. Bestellung erstellen (AWS Lambda)
2. Inventar aktualisieren (AWS Lambda)
3. Zahlung tätigen (AWS Lambda)

Der Orchestrator koordiniert jeden Schritt, indem er Wiederholungen, parallel Verzweigungen, Timeouts und Fehler verwaltet.

Rollenbasiertes Agentensystem (Orchestrator)

In Agentensystemen spiegelt das Orchestrator-Muster die Orchestrierung von Ereignissen wider, verteilt die Logik jedoch auf mehrere Argumentationsagenten, von denen jeder eine definierte Rolle oder Spezialisierung hat. Ein zentraler Orchestrator-Agent interpretiert die Gesamtaufgabe, zerlegt sie in Unteraufgaben und delegiert diese an Worker Agents, die jeweils für einen bestimmten Bereich optimiert sind (z. B. Recherche, Programmierung, Zusammenfassung, Überprüfung).

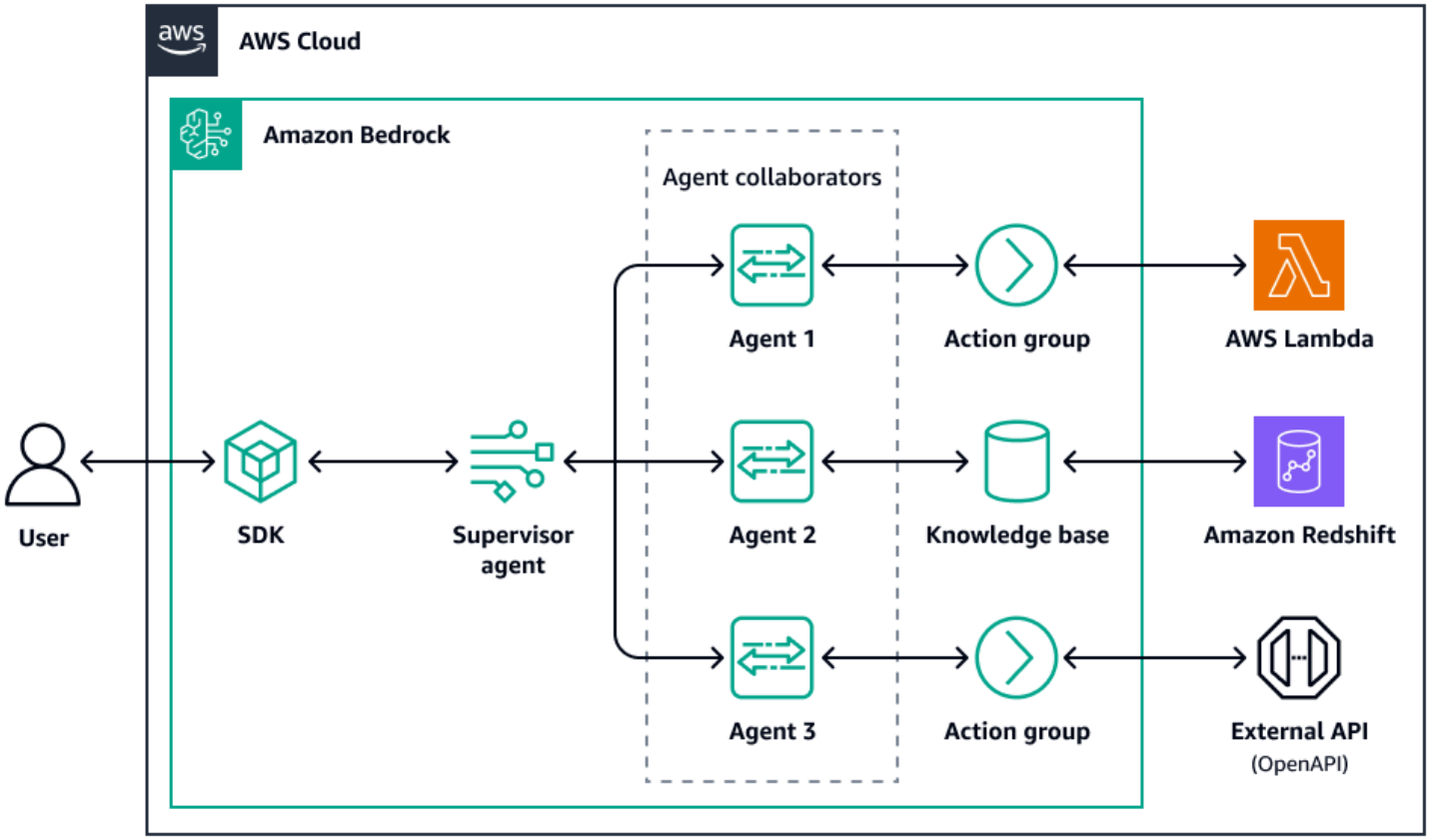
Vorgesetzter

1. Ein Benutzer sendet die Abfrage „Erstellen Sie eine Projektbeschreibung und fassen Sie die fünf wichtigsten Wettbewerber zusammen“.
2. Der Orchestrator-Agent macht Folgendes:

- Weist einen Forschungsagenten mit der Suche nach Konkurrenzdaten zu
- Sendet die Rohergebnisse an einen Zusammenfassungsagenten
- Leitet die Ergebnisse an einen Mitarbeiter weiter, der einen Brief verfasst
- Kompiliert die endgültige Ausgabe für den Benutzer

Jeder Agent arbeitet unabhängig, aber der Orchestrator koordiniert die Aufgaben. Dies ist wie eine Lambda-Funktion, die Workflow-Aufgaben erledigt.

Das folgende Diagramm zeigt ein Beispiel für einen Supervisor:



1. Ein Benutzer reicht eine Aufgabe an einen Amazon Bedrock Supervisor-Mitarbeiter ein.
2. Der Supervisor-Agent unterteilt die Anfrage in Unteraufgaben für jeden Mitarbeitenden.
3. Jede Unteraufgabe wird einem Mitarbeiter-Agenten mit rollenspezifischen Eingabeaufforderungen oder Toolchains zugewiesen.
4. Worker Agents rufen externe Tools APIs oder Tools über eine Aktionsgruppe auf.
5. Jeder Worker-Agent gibt die Ausgabe in einem strukturierten Format zurück.

6. Wenn alle Mitarbeiter ihre Ergebnisse zurückgeben, bewertet und fasst der Supervisor die endgültige Antwort zusammen und gibt sie zurück.

Diese Struktur ermöglicht Modularität, Anpassungsfähigkeit und Selbstbeobachtung bei komplexen, mehrstufigen Arbeitsabläufen für Agenten.

Imbissbuden

Während bei der Eventorchestrierung die zentrale Steuerung (z. B. AWS Step Functions) zur Steuerung der Serviceausführung verwendet wird, verwenden rollenbasierte Agentensysteme einen LLM-gestützten Orchestrator-Agenten, um über das Ziel nachzudenken, Unteraufgaben an Worker Agents zu delegieren und das Endergebnis zu synthetisieren.

In beiden Paradigmen macht der Orchestrator Folgendes:

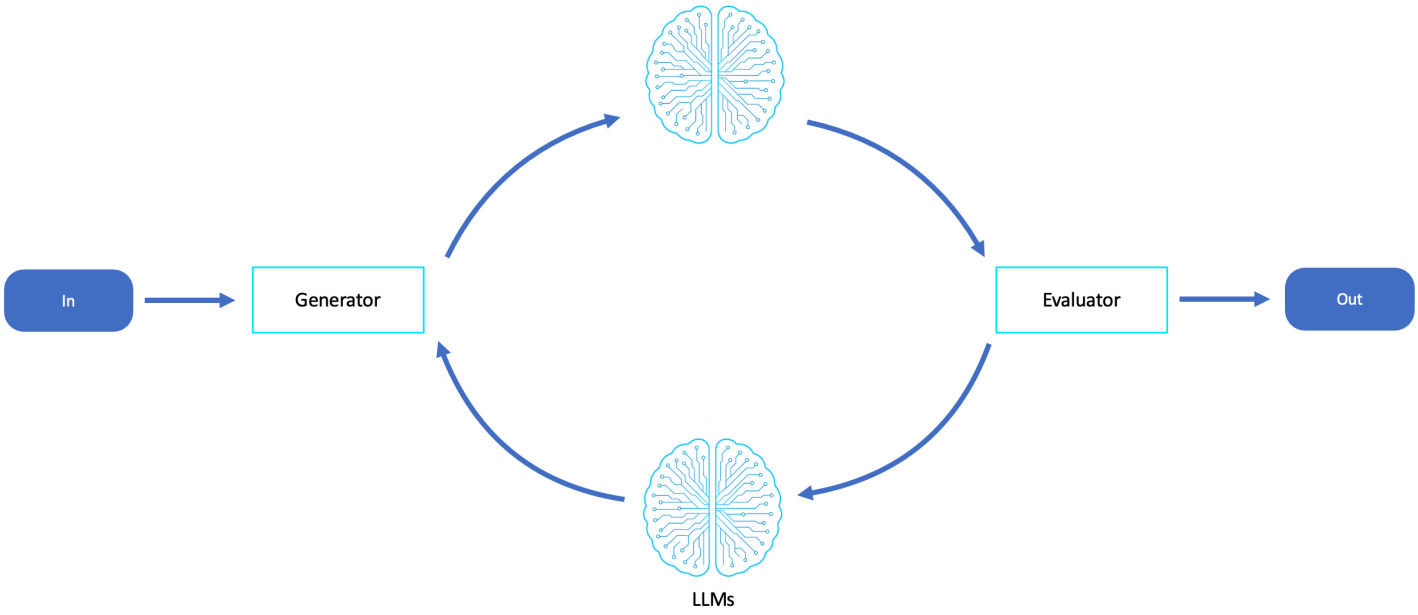
- Behält den Kontext und den Ausführungsablauf bei
- Verwaltet Verzweigungen, Sequenzierung und Fehlerbehandlung
- Erzeugt ein einheitliches Ergebnis aus verteilten Komponenten

Die behördliche Orchestrierung sorgt jedoch für mehr Argumentation, Anpassungsfähigkeit und semantische Delegierung. Dadurch eignet sie sich gut für offene, mehrdeutige und sich ständig weiterentwickelnde Aufgaben.

Schleifenmuster im Evaluator reflektieren/verfeinern

Aufgaben wie Codegenerierung, Zusammenfassung oder autonome Entscheidungsfindung profitieren in hohem Maße vom Feedback zur Laufzeit, sodass sich das System durch Beobachtung und Verfeinerung weiterentwickeln kann. Um dies zu operationalisieren, kann der Reflect-Refine-Zyklus als ereignisgesteuerter Feedback-Regelkreis implementiert werden — ein von der Systemtechnik inspiriertes Muster, das für autonome, intelligente Workflows angepasst ist.

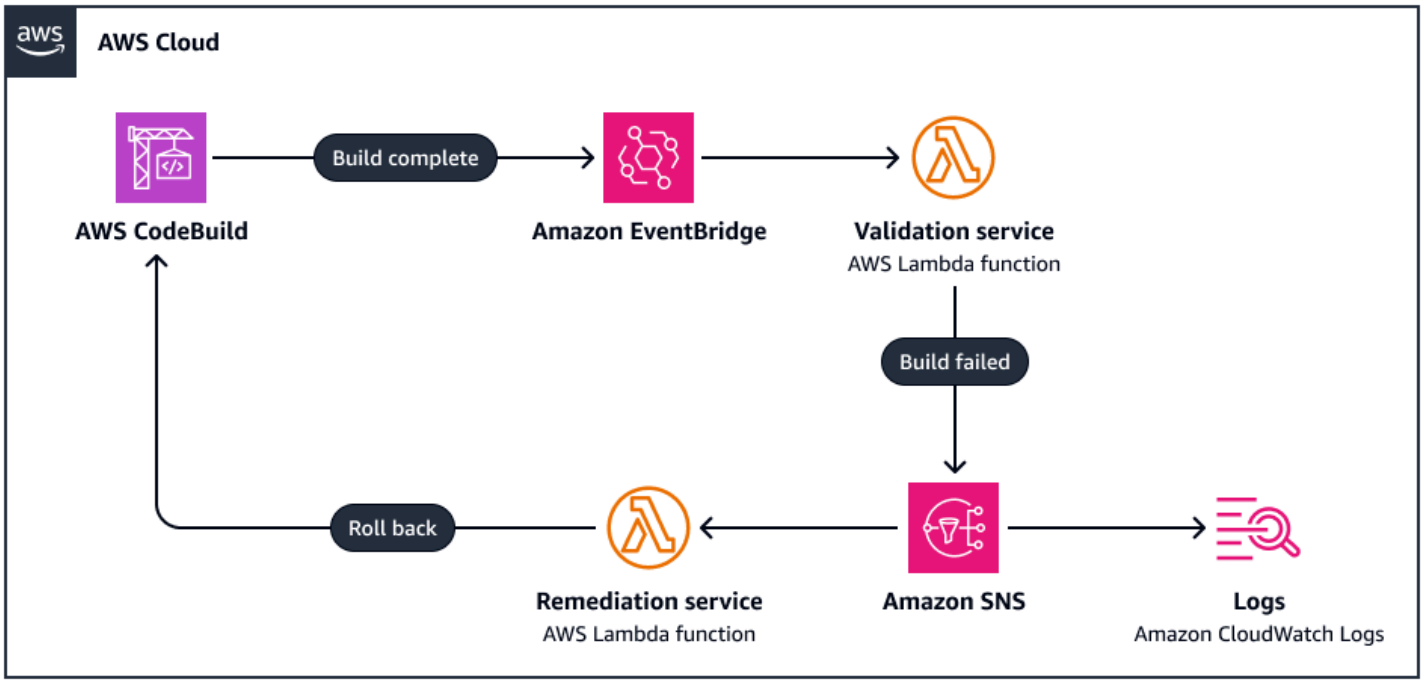
Das folgende Diagramm ist ein Beispiel für eine Reflect-Refine-Feedback-Schleife von Evaluatoren:



Regelkreis für Rückkopplung

Ein Feedback-Regelkreis ist ein Muster, das seine eigenen Leistungen und Verhaltensweisen überwacht, sie anhand definierter Kriterien oder eines gewünschten Zustands bewertet und dann seine Aktionen entsprechend anpasst. Diese Architektur ist von der Regelungstheorie inspiriert und ist grundlegend für Bereiche wie Automatisierung, CI/CD-Pipelines (Continuous Integration and Continuous Delivery) und maschinelles Lernen.

Das folgende Diagramm ist ein Beispiel für einen Feedback-Regelkreis:



1. Eine Bereitstellungspipeline gibt ein BuildComplete-Ereignis aus.
2. Das Ereignis löst einen automatisierten Test- oder Evaluierungsjob aus, der den Build validiert.
3. Schlägt die Validierung fehl (z. B. aufgrund fehlgeschlagener Tests, Sicherheitsprobleme oder eines Richtlinienverstoßes), geht das System wie folgt vor:
   - Gibt ein BuildComplete-Ereignis aus
   - Protokolliert das Problem oder sendet eine Benachrichtigung
   - Löst eine Behebungs- oder Korrekturmaßnahme aus, z. B. ein Rollback, ein Patchen oder ein erneuter Versuch

Die Schleife wird fortgesetzt, bis ein akzeptables Ergebnis oder eine akzeptable Eskalation erzielt wird oder ein Timeout eintritt. Dieses Muster wird häufig für Folgendes verwendet:

- EventBridge Amazon-Regeln für die Weiterleitung von Ereignissen an Evaluierungs- oder Problembehebungsaufgaben
- AWS Step Functions für iterative Wiederholungslogik und Verzweigung der Evaluierungsergebnisse
- Amazon Simple Notification Service (Amazon SNS) oder CloudWatch Amazon-Alarme für Feedback-Auslöser und Benachrichtigungen
- AWS Lambda Funktionen oder Mitarbeiter in Containern, um Abhilfemaßnahmen zu ergreifen

Regelkreis für Rückkopplung (Evaluator)

Ein Evaluator-Workflow ist eine kognitive Feedback-Schleife, die von unseren Argumentatoren angetrieben LLMs wird. Der Prozess besteht aus den folgenden Schritten:

1. Ein Generator-Agent oder LLM erzeugt eine Ausgabe (z. B. einen Plan, eine Antwort oder einen Entwurf).
2. Ein Gutachter überprüft das Ergebnis anhand einer Aufforderung zur Kritik oder einer Bewertungsrubrik.
3. Auf der Grundlage des Feedbacks überarbeitet der ursprüngliche Agent oder ein neuer Optimierer-Agent die Ausgabe.

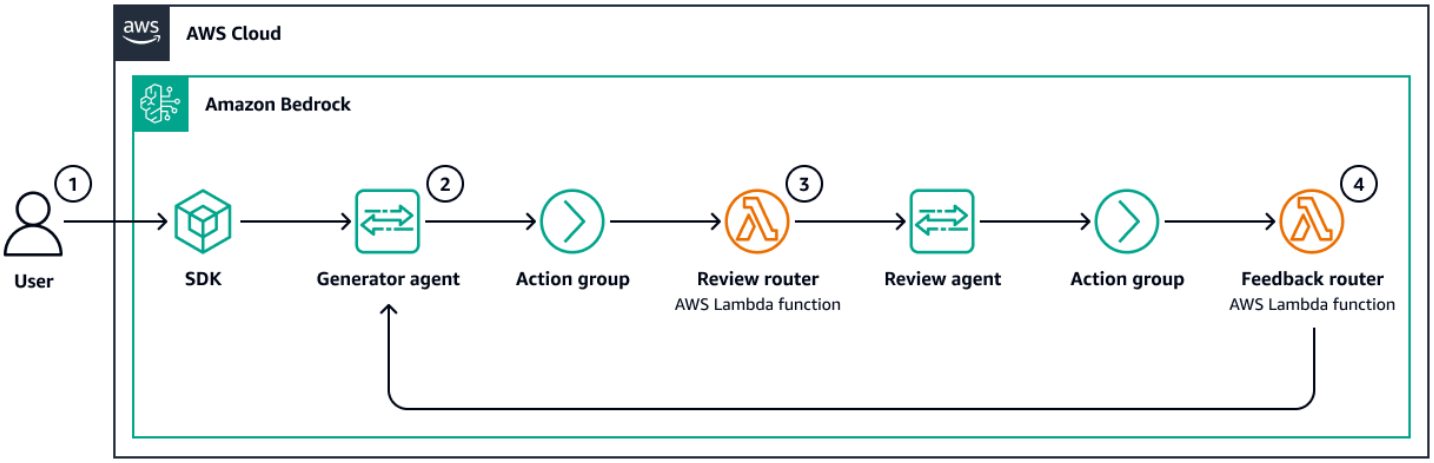
Die Schleife wird wiederholt, bis das Ergebnis eine Reihe von Kriterien erfüllt, genehmigt wurde oder ein Wiederholungslimit erreicht.

Evaluator

1. Ein Benutzer bittet einen Agenten, eine Richtlinienübersicht zu verfassen.
2. Der Generator-Agent entwirft sie.
3. Ein Gutachter überprüft den Umfang, den Ton und die rechtliche Richtigkeit.
4. Wenn die Antwort unzureichend ist, wird sie verfeinert und erneut eingereicht, bis die Feedback-Schleife zusammenläuft.

Dies ermöglicht eine Selbstbeurteilung, iterative Verfeinerung und adaptive Ausgabesteuerung — alles ohne menschliches Zutun.

Das folgende Diagramm ist ein Beispiel für einen Feedback-Regelkreis (Evaluator):



1. Ein Benutzer gibt eine Aufgabe aus (z. B. Entwurf einer Geschäftsstrategie).
2. Ein Amazon Bedrock-Agent generiert mithilfe eines LLM einen ersten Entwurf.
3. Ein zweiter Mitarbeiter (oder eine Folgeaufforderung) führt eine strukturierte Bewertung durch (z. B. „Bewerten Sie dieses Ergebnis nach Klarheit, Vollständigkeit und Tonalität“).
4. Wenn die Bewertung unter einen Schwellenwert fällt, wird die Antwort wie folgt überarbeitet:
   - Den Generator mit einer eingebetteten Kritik erneut aufrufen
   - Senden des Feedbacks an einen spezialisierten Raffineriemitarbeiter
   - Iterieren, bis eine akzeptable Antwort erreicht ist

Optionale Komponenten wie AWS Lambda Controller AWS Step Functions können Feedback-Schwellenwerte, Wiederholungsversuche und Fallback-Strategien verwalten.

Imbissbuden

Während herkömmliche Feedback-Regelkreise Ereignisse, Metriken und Korrekturlogik verwenden, um das Systemverhalten zu validieren und anzupassen, verwenden behördliche Bewertungsschleifen Begründungsagenten, um Ergebnisse dynamisch zu bewerten, zu reflektieren und zu überarbeiten.

In beiden Paradigmen gilt:

- Die Ausgabe wird ausgewertet, nachdem sie generiert wurde
- Korrektur- oder Verbesserungsmaßnahmen werden auf der Grundlage von Feedback ausgelöst
- Das System passt sich kontinuierlich an eine Zielqualität oder ein Ziel an

Die agentische Version wandelt die statische Validierung in eine semantische Reflexion um und ermöglicht es Agenten, sich selbst zu verbessern, ihre eigene Effektivität zu bewerten.

Gestaltung agentischer Workflows auf AWS

Jedes Muster in diesem Handbuch kann mithilfe von erstellt werden. AWS-Services Amazon Bedrock-Agenten bieten Orchestrierung, Datenzugriff und Interaktionskanäle.

Komponente	AWS-Service	Zweck
LLM-Argumentation	Amazon Bedrock	Agentenlogik, Planung, Einsatz von Tools
Ausführung des Tools	AWS Lambda, Amazon ECS, Amazon SageMaker	Hosten Sie externe Tools für Agenten
Speicher und RAG	Amazon Bedrock Wissensda tenbank, Amazon S3, OpenSearch	Persistenter und semantischer Speicher
Orchestrierung	AWS Step Functions	Mehrstufige Aufgaben- und Agentenkoordination
Weiterleitung von Ereignissen	Amazon EventBridge, Amazon SQS	Entkoppeltes interagen tenübergreifendes Messaging
Benutzeroberfläche	Amazon API Gateway AWS AppSync, SDK	Einstiegspunkte für Anwendungen oder Systeme
Überwachung	Amazon CloudWatch AWS X-Ray, AWS Distribution für OpenTelemetry	Beobachtbarkeit und Introspek tion der Agenten

Schlussfolgerung

Agentengestützte Workflow-Muster sind die nächste Evolutionsstufe ereignisgesteuerter Architekturen, bei denen Geschäftslogik nicht statisch definiert, sondern mithilfe eines Large Language Model (LLM) — erweiterter Kognition — dynamisch begründet wird. Durch die Kombination

traditioneller cloudnativer Primitives mit LLM-Workflows und Agentendesignmustern können Unternehmen anpassungsfähige, intelligente und modulare Systeme aufbauen, die zielgerichtet reagieren und aus Erfahrungen lernen.

In diesen Mustern ist Amazon Bedrock das Tor zu agentischer Kognition und ermöglicht es LLM-basierten Agenten, auf Event-Workflows zuzugreifen, mit Tools und Speichern zu interagieren und strukturierte, nachvollziehbare und aufeinander abgestimmte Ergebnisse zu liefern.

Bei der Entwicklung und Bereitstellung agentischer Systeme bieten diese Workflow-Muster Blaupausen für den Aufbau autonomer, zusammensetzbarer KI-Architekturen. Diese Systeme basieren auf bewährten Methoden für serverlose Systeme und werden durch intelligente Basismodelle erweitert.

Dokumentverlauf

In der folgenden Tabelle werden wichtige Änderungen in diesem Leitfaden beschrieben. Um Benachrichtigungen über zukünftige Aktualisierungen zu erhalten, können Sie einen RSS-Feed abonnieren.

Änderung	Beschreibung	Datum
Erste Veröffentlichung	—	14. Juli 2025

AWS Glossar zu präskriptiven Leitlinien

Die folgenden Begriffe werden häufig in Strategien, Leitfäden und Mustern von AWS Prescriptive Guidance verwendet. Um Einträge vorzuschlagen, verwenden Sie bitte den Link Feedback geben am Ende des Glossars.

Zahlen

7 Rs

Sieben gängige Migrationsstrategien für die Verlagerung von Anwendungen in die Cloud. Diese Strategien bauen auf den 5 Rs auf, die Gartner 2011 identifiziert hat, und bestehen aus folgenden Elementen:

- Faktorwechsel/Architekturwechsel – Verschieben Sie eine Anwendung und ändern Sie ihre Architektur, indem Sie alle Vorteile cloudnativer Feature nutzen, um Agilität, Leistung und Skalierbarkeit zu verbessern. Dies beinhaltet in der Regel die Portierung des Betriebssystems und der Datenbank. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank auf die Amazon Aurora PostgreSQL-kompatible Edition.
- Plattformwechsel (Lift and Reshape) – Verschieben Sie eine Anwendung in die Cloud und führen Sie ein gewisses Maß an Optimierung ein, um die Cloud-Funktionen zu nutzen. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank zu Amazon Relational Database Service (Amazon RDS) für Oracle in der. AWS Cloud
- Neukauf (Drop and Shop) – Wechseln Sie zu einem anderen Produkt, indem Sie typischerweise von einer herkömmlichen Lizenz zu einem SaaS-Modell wechseln. Beispiel: Migrieren Sie Ihr CRM-System (Customer Relationship Management) zu Salesforce.com.
- Hostwechsel (Lift and Shift) – Verschieben Sie eine Anwendung in die Cloud, ohne Änderungen vorzunehmen, um die Cloud-Funktionen zu nutzen. Beispiel: Migrieren Sie Ihre lokale Oracle-Datenbank zu Oracle auf einer EC2-Instanz in der. AWS Cloud
- Verschieben (Lift and Shift auf Hypervisor-Ebene) – Verlagern Sie die Infrastruktur in die Cloud, ohne neue Hardware kaufen, Anwendungen umschreiben oder Ihre bestehenden Abläufe ändern zu müssen. Sie migrieren Server von einer lokalen Plattform zu einem Cloud-Dienst für dieselbe Plattform. Beispiel: Migrieren Sie eine Microsoft Hyper-V Anwendung zu AWS.
- Beibehaltung (Wiederaufgreifen) – Bewahren Sie Anwendungen in Ihrer Quellumgebung auf. Dazu können Anwendungen gehören, die einen umfangreichen Faktorwechsel erfordern und

die Sie auf einen späteren Zeitpunkt verschieben möchten, sowie ältere Anwendungen, die Sie beibehalten möchten, da es keine geschäftliche Rechtfertigung für ihre Migration gibt.

- Außerbetriebnahme – Dekommissionierung oder Entfernung von Anwendungen, die in Ihrer Quellumgebung nicht mehr benötigt werden.

A

ABAC

Siehe attributbasierte Zugriffskontrolle.

abstrahierte Dienste

Siehe Managed Services.

ACID

Siehe Atomarität, Konsistenz, Isolierung und Haltbarkeit.

Aktiv-Aktiv-Migration

Eine Datenbankmigrationsmethode, bei der die Quell- und Zieldatenbanken synchron gehalten werden (mithilfe eines bidirektionalen Replikationstools oder dualer Schreibvorgänge) und beide Datenbanken Transaktionen von miteinander verbundenen Anwendungen während der Migration verarbeiten. Diese Methode unterstützt die Migration in kleinen, kontrollierten Batches, anstatt einen einmaligen Cutover zu erfordern. Es ist flexibler, erfordert aber mehr Arbeit als eine aktiv-passive Migration.

Aktiv-Passiv-Migration

Eine Datenbankmigrationsmethode, bei der die Quell- und Zieldatenbanken synchron gehalten werden, aber nur die Quelldatenbank verarbeitet Transaktionen von verbindenden Anwendungen, während Daten in die Zieldatenbank repliziert werden. Die Zieldatenbank akzeptiert während der Migration keine Transaktionen.

Aggregatfunktion

Eine SQL-Funktion, die mit einer Gruppe von Zeilen arbeitet und einen einzelnen Rückgabewert für die Gruppe berechnet. Beispiele für Aggregatfunktionen sind `SUM` und`MAX`.

AI

Siehe künstliche Intelligenz.

AIOps

Siehe Operationen im Bereich künstliche Intelligenz.

Anonymisierung

Der Prozess des dauerhaften Löschens personenbezogener Daten in einem Datensatz. Anonymisierung kann zum Schutz der Privatsphäre beitragen. Anonymisierte Daten gelten nicht mehr als personenbezogene Daten.

Anti-Muster

Eine häufig verwendete Lösung für ein wiederkehrendes Problem, bei dem die Lösung kontraproduktiv, ineffektiv oder weniger wirksam als eine Alternative ist.

Anwendungssteuerung

Ein Sicherheitsansatz, bei dem nur zugelassene Anwendungen verwendet werden können, um ein System vor Schadsoftware zu schützen.

Anwendungsportfolio

Eine Sammlung detaillierter Informationen zu jeder Anwendung, die von einer Organisation verwendet wird, einschließlich der Kosten für die Erstellung und Wartung der Anwendung und ihres Geschäftswerts. Diese Informationen sind entscheidend für den Prozess der Portfoliofindung und -analyse und hilft bei der Identifizierung und Priorisierung der Anwendungen, die migriert, modernisiert und optimiert werden sollen.

künstliche Intelligenz (KI)

Das Gebiet der Datenverarbeitungswissenschaft, das sich der Nutzung von Computertechnologien zur Ausführung kognitiver Funktionen widmet, die typischerweise mit Menschen in Verbindung gebracht werden, wie Lernen, Problemlösen und Erkennen von Mustern. Weitere Informationen finden Sie unter Was ist künstliche Intelligenz?

Operationen mit künstlicher Intelligenz (AIOps)

Der Prozess des Einsatzes von Techniken des Machine Learning zur Lösung betrieblicher Probleme, zur Reduzierung betrieblicher Zwischenfälle und menschlicher Eingriffe sowie zur Steigerung der Servicequalität. Weitere Informationen zur Verwendung in der AWS Migrationsstrategie finden Sie im Operations Integration Guide. AIOps

Asymmetrische Verschlüsselung

Ein Verschlüsselungsalgorithmus, der ein Schlüsselpaar, einen öffentlichen Schlüssel für die Verschlüsselung und einen privaten Schlüssel für die Entschlüsselung verwendet. Sie können den

öffentlichen Schlüssel teilen, da er nicht für die Entschlüsselung verwendet wird. Der Zugriff auf den privaten Schlüssel sollte jedoch stark eingeschränkt sein.

Atomizität, Konsistenz, Isolierung, Haltbarkeit (ACID)

Eine Reihe von Softwareeigenschaften, die die Datenvalidität und betriebliche Zuverlässigkeit einer Datenbank auch bei Fehlern, Stromausfällen oder anderen Problemen gewährleisten.

Attributbasierte Zugriffskontrolle (ABAC)

Die Praxis, detaillierte Berechtigungen auf der Grundlage von Benutzerattributen wie Abteilung, Aufgabenrolle und Teamname zu erstellen. Weitere Informationen finden Sie unter ABAC AWS in der AWS Identity and Access Management (IAM-) Dokumentation.

autoritative Datenquelle

Ein Ort, an dem Sie die primäre Version der Daten speichern, die als die zuverlässigste Informationsquelle angesehen wird. Sie können Daten aus der maßgeblichen Datenquelle an andere Speicherorte kopieren, um die Daten zu verarbeiten oder zu ändern, z. B. zu anonymisieren, zu redigieren oder zu pseudonymisieren.

Availability Zone

Ein bestimmter Standort innerhalb einer AWS-Region , der vor Ausfällen in anderen Availability Zones geschützt ist und kostengünstige Netzwerkkonnektivität mit niedriger Latenz zu anderen Availability Zones in derselben Region bietet.

AWS Framework für die Einführung der Cloud (AWS CAF)

Ein Framework mit Richtlinien und bewährten Verfahren, das Unternehmen bei der Entwicklung eines effizienten und effektiven Plans für die erfolgreiche Umstellung auf die Cloud unterstützt. AWS AWS CAF unterteilt die Leitlinien in sechs Schwerpunktbereiche, die als Perspektiven bezeichnet werden: Unternehmen, Mitarbeiter, Unternehmensführung, Plattform, Sicherheit und Betrieb. Die Perspektiven Geschäft, Mitarbeiter und Unternehmensführung konzentrieren sich auf Geschäftskompetenzen und -prozesse, während sich die Perspektiven Plattform, Sicherheit und Betriebsabläufe auf technische Fähigkeiten und Prozesse konzentrieren. Die Personalperspektive zielt beispielsweise auf Stakeholder ab, die sich mit Personalwesen (HR), Personalfunktionen und Personalmanagement befassen. Aus dieser Perspektive bietet AWS CAF Leitlinien für Personalentwicklung, Schulung und Kommunikation, um das Unternehmen auf eine erfolgreiche Cloud-Einführung vorzubereiten. Weitere Informationen finden Sie auf der AWS -CAF-Webseite und dem AWS -CAF-Whitepaper.

AWS Workload-Qualifizierungsrahmen (AWS WQF)

Ein Tool, das Workloads bei der Datenbankmigration bewertet, Migrationsstrategien empfiehlt und Arbeitsschätzungen bereitstellt. AWS WQF ist in () enthalten. AWS Schema Conversion Tool AWS SCT Es analysiert Datenbankschemas und Codeobjekte, Anwendungscode, Abhängigkeiten und Leistungsmerkmale und stellt Bewertungsberichte bereit.

B

schlechter Bot

Ein Bot, der Einzelpersonen oder Organisationen stören oder ihnen Schaden zufügen soll.

BCP

Siehe Planung der Geschäftskontinuität.

Verhaltensdiagramm

Eine einheitliche, interaktive Ansicht des Ressourcenverhaltens und der Interaktionen im Laufe der Zeit. Sie können ein Verhaltensdiagramm mit Amazon Detective verwenden, um fehlgeschlagene Anmeldeversuche, verdächtige API-Aufrufe und ähnliche Vorgänge zu untersuchen. Weitere Informationen finden Sie unter Daten in einem Verhaltensdiagramm in der Detective-Dokumentation.

Big-Endian-System

Ein System, welches das höchstwertige Byte zuerst speichert. Siehe auch Endianness.

Binäre Klassifikation

Ein Prozess, der ein binäres Ergebnis vorhersagt (eine von zwei möglichen Klassen). Beispielsweise könnte Ihr ML-Modell möglicherweise Probleme wie „Handelt es sich bei dieser E-Mail um Spam oder nicht?“ vorhersagen müssen oder „Ist dieses Produkt ein Buch oder ein Auto?“

Bloom-Filter

Eine probabilistische, speichereffiziente Datenstruktur, mit der getestet wird, ob ein Element Teil einer Menge ist.

Blau/Grün-Bereitstellung

Eine Bereitstellungsstrategie, bei der Sie zwei separate, aber identische Umgebungen erstellen. Sie führen die aktuelle Anwendungsversion in einer Umgebung (blau) und die neue

Anwendungsversion in der anderen Umgebung (grün) aus. Mit dieser Strategie können Sie schnell und mit minimalen Auswirkungen ein Rollback durchführen.

Bot

Eine Softwareanwendung, die automatisierte Aufgaben über das Internet ausführt und menschliche Aktivitäten oder Interaktionen simuliert. Manche Bots sind nützlich oder nützlich, wie z. B. Webcrawler, die Informationen im Internet indexieren. Einige andere Bots, sogenannte bösartige Bots, sollen Einzelpersonen oder Organisationen stören oder ihnen Schaden zufügen.

Botnetz

Netzwerke von Bots, die mit Malware infiziert sind und unter der Kontrolle einer einzigen Partei stehen, die als Bot-Herder oder Bot-Operator bezeichnet wird. Botnetze sind der bekannteste Mechanismus zur Skalierung von Bots und ihrer Wirkung.

branch

Ein containerisierter Bereich eines Code-Repositorys. Der erste Zweig, der in einem Repository erstellt wurde, ist der Hauptzweig. Sie können einen neuen Zweig aus einem vorhandenen Zweig erstellen und dann Feature entwickeln oder Fehler in dem neuen Zweig beheben. Ein Zweig, den Sie erstellen, um ein Feature zu erstellen, wird allgemein als Feature-Zweig bezeichnet. Wenn das Feature zur Veröffentlichung bereit ist, führen Sie den Feature-Zweig wieder mit dem Hauptzweig zusammen. Weitere Informationen finden Sie unter Über Branches (GitHub Dokumentation).

Zugang durch Glasbruch

Unter außergewöhnlichen Umständen und im Rahmen eines genehmigten Verfahrens ist dies eine schnelle Methode für einen Benutzer, auf einen Bereich zuzugreifen AWS-Konto , für den er normalerweise keine Zugriffsrechte besitzt. Weitere Informationen finden Sie unter dem Indikator Implementation break-glass procedures in den AWS Well-Architected-Leitlinien.

Brownfield-Strategie

Die bestehende Infrastruktur in Ihrer Umgebung. Wenn Sie eine Brownfield-Strategie für eine Systemarchitektur anwenden, richten Sie sich bei der Gestaltung der Architektur nach den Einschränkungen der aktuellen Systeme und Infrastruktur. Wenn Sie die bestehende Infrastruktur erweitern, könnten Sie Brownfield- und Greenfield-Strategien mischen.

Puffer-Cache

Der Speicherbereich, in dem die am häufigsten abgerufenen Daten gespeichert werden.

Geschäftsfähigkeit

Was ein Unternehmen tut, um Wert zu generieren (z. B. Vertrieb, Kundenservice oder Marketing). Microservices-Architekturen und Entwicklungsentscheidungen können von den Geschäftskapazitäten beeinflusst werden. Weitere Informationen finden Sie im Abschnitt Organisiert nach Geschäftskapazitäten des Whitepapers Ausführen von containerisierten Microservices in AWS.

Planung der Geschäftskontinuität (BCP)

Ein Plan, der die potenziellen Auswirkungen eines störenden Ereignisses, wie z. B. einer groß angelegten Migration, auf den Betrieb berücksichtigt und es einem Unternehmen ermöglicht, den Betrieb schnell wieder aufzunehmen.

C

CAF

Weitere Informationen finden Sie unter Framework AWS für die Cloud-Einführung.

Bereitstellung auf Kanaren

Die langsame und schrittweise Veröffentlichung einer Version für Endbenutzer. Wenn Sie sich sicher sind, stellen Sie die neue Version bereit und ersetzen die aktuelle Version vollständig.

CCoE

Weitere Informationen finden Sie im Cloud Center of Excellence.

CDC

Siehe Erfassung von Änderungsdaten.

Erfassung von Datenänderungen (CDC)

Der Prozess der Nachverfolgung von Änderungen an einer Datenquelle, z. B. einer Datenbanktabelle, und der Aufzeichnung von Metadaten zu der Änderung. Sie können CDC für verschiedene Zwecke verwenden, z. B. für die Prüfung oder Replikation von Änderungen in einem Zielsystem, um die Synchronisation aufrechtzuerhalten.

Chaos-Technik

Absichtliches Einführen von Ausfällen oder Störungsereignissen, um die Widerstandsfähigkeit eines Systems zu testen. Sie können AWS Fault Injection Service (AWS FIS) verwenden, um Experimente durchzuführen, die Ihre AWS Workloads stress, und deren Reaktion zu bewerten.

CI/CD

Siehe Continuous Integration und Continuous Delivery.

Klassifizierung

Ein Kategorisierungsprozess, der bei der Erstellung von Vorhersagen hilft. ML-Modelle für Klassifikationsprobleme sagen einen diskreten Wert voraus. Diskrete Werte unterscheiden sich immer voneinander. Beispielsweise muss ein Modell möglicherweise auswerten, ob auf einem Bild ein Auto zu sehen ist oder nicht.

clientseitige Verschlüsselung

Lokale Verschlüsselung von Daten, bevor das Ziel sie AWS-Service empfängt.

Cloud-Exzellenzzentrum (CCoE)

Ein multidisziplinäres Team, das die Cloud-Einführung in der gesamten Organisation vorantreibt, einschließlich der Entwicklung bewährter Cloud-Methoden, der Mobilisierung von Ressourcen, der Festlegung von Migrationszeitplänen und der Begleitung der Organisation durch groß angelegte Transformationen. Weitere Informationen finden Sie in den CCoE-Beiträgen im AWS Cloud Enterprise Strategy Blog.

Cloud Computing

Die Cloud-Technologie, die typischerweise für die Ferndatenspeicherung und das IoT-Gerätemanagement verwendet wird. Cloud Computing ist häufig mit Edge-Computing-Technologie verbunden.

Cloud-Betriebsmodell

In einer IT-Organisation das Betriebsmodell, das zum Aufbau, zur Weiterentwicklung und Optimierung einer oder mehrerer Cloud-Umgebungen verwendet wird. Weitere Informationen finden Sie unter Aufbau Ihres Cloud-Betriebsmodells.

Phasen der Einführung der Cloud

Die vier Phasen, die Unternehmen bei der Migration in der Regel durchlaufen AWS Cloud:

- Projekt – Durchführung einiger Cloud-bezogener Projekte zu Machbarkeitsnachweisen und zu Lernzwecken
- Fundament — Tätigen Sie grundlegende Investitionen, um Ihre Cloud-Einführung zu skalieren (z. B. Einrichtung einer landing zone, Definition eines CCo E, Einrichtung eines Betriebsmodells)

- Migration – Migrieren einzelner Anwendungen
- Neuentwicklung – Optimierung von Produkten und Services und Innovation in der Cloud

Diese Phasen wurden von Stephen Orban im Blogbeitrag The Journey Toward Cloud-First & the Stages of Adoption im AWS Cloud Enterprise Strategy-Blog definiert. Informationen darüber, wie sie mit der AWS Migrationsstrategie zusammenhängen, finden Sie im Leitfaden zur Vorbereitung der Migration.

CMDB

Siehe Datenbank für das Konfigurationsmanagement.

Code-Repository

Ein Ort, an dem Quellcode und andere Komponenten wie Dokumentation, Beispiele und Skripts gespeichert und im Rahmen von Versionskontrollprozessen aktualisiert werden. Zu den gängigen Cloud-Repositorys gehören GitHub oderBitbucket Cloud. Jede Version des Codes wird Zweig genannt. In einer Microservice-Struktur ist jedes Repository einer einzelnen Funktionalität gewidmet. Eine einzelne CI/CD-Pipeline kann mehrere Repositorien verwenden.

Kalter Cache

Ein Puffer-Cache, der leer oder nicht gut gefüllt ist oder veraltete oder irrelevante Daten enthält. Dies beeinträchtigt die Leistung, da die Datenbank-Instance aus dem Hauptspeicher oder der Festplatte lesen muss, was langsamer ist als das Lesen aus dem Puffercache.

Kalte Daten

Daten, auf die selten zugegriffen wird und die in der Regel historisch sind. Bei der Abfrage dieser Art von Daten sind langsame Abfragen in der Regel akzeptabel. Durch die Verlagerung dieser Daten auf leistungsschwächere und kostengünstigere Speicherstufen oder -klassen können Kosten gesenkt werden.

Computer Vision (CV)

Ein Bereich der KI, der maschinelles Lernen nutzt, um Informationen aus visuellen Formaten wie digitalen Bildern und Videos zu analysieren und zu extrahieren. Amazon SageMaker AI bietet beispielsweise Bildverarbeitungsalgorithmen für CV.

Drift in der Konfiguration

Bei einer Arbeitslast eine Änderung der Konfiguration gegenüber dem erwarteten Zustand. Dies kann dazu führen, dass der Workload nicht mehr richtlinienkonform wird, und zwar in der Regel schrittweise und unbeabsichtigt.

Verwaltung der Datenbankkonfiguration (CMDB)

Ein Repository, das Informationen über eine Datenbank und ihre IT-Umgebung speichert und verwaltet, inklusive Hardware- und Softwarekomponenten und deren Konfigurationen. In der Regel verwenden Sie Daten aus einer CMDB in der Phase der Portfolioerkennung und -analyse der Migration.

Konformitätspaket

Eine Sammlung von AWS Config Regeln und Abhilfemaßnahmen, die Sie zusammenstellen können, um Ihre Konformitäts- und Sicherheitsprüfungen individuell anzupassen. Mithilfe einer YAML-Vorlage können Sie ein Conformance Pack als einzelne Entität in einer AWS-Konto AND-Region oder unternehmensweit bereitstellen. Weitere Informationen finden Sie in der Dokumentation unter Conformance Packs. AWS Config

Kontinuierliche Bereitstellung und kontinuierliche Integration (CI/CD)

Der Prozess der Automatisierung der Quell-, Build-, Test-, Staging- und Produktionsphasen des Softwareveröffentlichungsprozesses. CI/CD wird allgemein als Pipeline beschrieben. CI/CD kann Ihnen helfen, Prozesse zu automatisieren, die Produktivität zu steigern, die Codequalität zu verbessern und schneller zu liefern. Weitere Informationen finden Sie unter Vorteile der kontinuierlichen Auslieferung. CD kann auch für kontinuierliche Bereitstellung stehen. Weitere Informationen finden Sie unterKontinuierliche Auslieferung im Vergleich zu kontinuierlicher Bereitstellung.

CV

Siehe Computer Vision.

D

Daten im Ruhezustand

Daten, die in Ihrem Netzwerk stationär sind, z. B. Daten, die sich im Speicher befinden.

Datenklassifizierung

Ein Prozess zur Identifizierung und Kategorisierung der Daten in Ihrem Netzwerk auf der Grundlage ihrer Kritikalität und Sensitivität. Sie ist eine wichtige Komponente jeder Strategie für das Management von Cybersecurity-Risiken, da sie Ihnen hilft, die geeigneten Schutz- und Aufbewahrungskontrollen für die Daten zu bestimmen. Die Datenklassifizierung ist ein Bestandteil

der Sicherheitssäule im AWS Well-Architected Framework. Weitere Informationen finden Sie unter Datenklassifizierung.

Datendrift

Eine signifikante Abweichung zwischen den Produktionsdaten und den Daten, die zum Trainieren eines ML-Modells verwendet wurden, oder eine signifikante Änderung der Eingabedaten im Laufe der Zeit. Datendrift kann die Gesamtqualität, Genauigkeit und Fairness von ML-Modellvorhersagen beeinträchtigen.

Daten während der Übertragung

Daten, die sich aktiv durch Ihr Netzwerk bewegen, z. B. zwischen Netzwerkressourcen.

Datennetz

Ein architektonisches Framework, das verteilte, dezentrale Dateneigentum mit zentraler Verwaltung und Steuerung ermöglicht.

Datenminimierung

Das Prinzip, nur die Daten zu sammeln und zu verarbeiten, die unbedingt erforderlich sind. Durch Datenminimierung im AWS Cloud können Datenschutzrisiken, Kosten und der CO2-Fußabdruck Ihrer Analysen reduziert werden.

Datenperimeter

Eine Reihe präventiver Schutzmaßnahmen in Ihrer AWS Umgebung, die sicherstellen, dass nur vertrauenswürdige Identitäten auf vertrauenswürdige Ressourcen von erwarteten Netzwerken zugreifen. Weitere Informationen finden Sie unter Aufbau eines Datenperimeters auf. AWS

Vorverarbeitung der Daten

Rohdaten in ein Format umzuwandeln, das von Ihrem ML-Modell problemlos verarbeitet werden kann. Die Vorverarbeitung von Daten kann bedeuten, dass bestimmte Spalten oder Zeilen entfernt und fehlende, inkonsistente oder doppelte Werte behoben werden.

Herkunft der Daten

Der Prozess der Nachverfolgung des Ursprungs und der Geschichte von Daten während ihres gesamten Lebenszyklus, z. B. wie die Daten generiert, übertragen und gespeichert wurden.

betroffene Person

Eine Person, deren Daten gesammelt und verarbeitet werden.

Data Warehouse

Ein Datenverwaltungssystem, das Business Intelligence wie Analysen unterstützt. Data Warehouses enthalten in der Regel große Mengen historischer Daten und werden in der Regel für Abfragen und Analysen verwendet.

Datenbankdefinitionssprache (DDL)

Anweisungen oder Befehle zum Erstellen oder Ändern der Struktur von Tabellen und Objekten in einer Datenbank.

Datenbankmanipulationssprache (DML)

Anweisungen oder Befehle zum Ändern (Einfügen, Aktualisieren und Löschen) von Informationen in einer Datenbank.

DDL

Siehe Datenbankdefinitionssprache.

Deep-Ensemble

Mehrere Deep-Learning-Modelle zur Vorhersage kombinieren. Sie können Deep-Ensembles verwenden, um eine genauere Vorhersage zu erhalten oder um die Unsicherheit von Vorhersagen abzuschätzen.

Deep Learning

Ein ML-Teilbereich, der mehrere Schichten künstlicher neuronaler Netzwerke verwendet, um die Zuordnung zwischen Eingabedaten und Zielvariablen von Interesse zu ermitteln.

defense-in-depth

Ein Ansatz zur Informationssicherheit, bei dem eine Reihe von Sicherheitsmechanismen und -kontrollen sorgfältig in einem Computernetzwerk verteilt werden, um die Vertraulichkeit, Integrität und Verfügbarkeit des Netzwerks und der darin enthaltenen Daten zu schützen. Wenn Sie diese Strategie anwenden AWS, fügen Sie mehrere Steuerelemente auf verschiedenen Ebenen der AWS Organizations Struktur hinzu, um die Ressourcen zu schützen. Ein defense-in-depth Ansatz könnte beispielsweise Multi-Faktor-Authentifizierung, Netzwerksegmentierung und Verschlüsselung kombinieren.

delegierter Administrator

In AWS Organizations kann ein kompatibler Dienst ein AWS Mitgliedskonto registrieren, um die Konten der Organisation und die Berechtigungen für diesen Dienst zu verwalten. Dieses Konto

wird als delegierter Administrator für diesen Service bezeichnet. Weitere Informationen und eine Liste kompatibler Services finden Sie unter Services, die mit AWS Organizations funktionieren in der AWS Organizations -Dokumentation.

Einsatz

Der Prozess, bei dem eine Anwendung, neue Feature oder Codekorrekturen in der Zielumgebung verfügbar gemacht werden. Die Bereitstellung umfasst das Implementieren von Änderungen an einer Codebasis und das anschließende Erstellen und Ausführen dieser Codebasis in den Anwendungsumgebungen.

Entwicklungsumgebung

Siehe Umgebung.

Detektivische Kontrolle

Eine Sicherheitskontrolle, die darauf ausgelegt ist, ein Ereignis zu erkennen, zu protokollieren und zu warnen, nachdem ein Ereignis eingetreten ist. Diese Kontrollen stellen eine zweite Verteidigungslinie dar und warnen Sie vor Sicherheitsereignissen, bei denen die vorhandenen präventiven Kontrollen umgangen wurden. Weitere Informationen finden Sie unter Detektivische Kontrolle in Implementierung von Sicherheitskontrollen in AWS.

Abbildung des Wertstroms in der Entwicklung (DVSM)

Ein Prozess zur Identifizierung und Priorisierung von Einschränkungen, die sich negativ auf Geschwindigkeit und Qualität im Lebenszyklus der Softwareentwicklung auswirken. DVSM erweitert den Prozess der Wertstromanalyse, der ursprünglich für Lean-Manufacturing-Praktiken konzipiert wurde. Es konzentriert sich auf die Schritte und Teams, die erforderlich sind, um durch den Softwareentwicklungsprozess Mehrwert zu schaffen und zu steigern.

digitaler Zwilling

Eine virtuelle Darstellung eines realen Systems, z. B. eines Gebäudes, einer Fabrik, einer Industrieanlage oder einer Produktionslinie. Digitale Zwillinge unterstützen vorausschauende Wartung, Fernüberwachung und Produktionsoptimierung.

Maßtabelle

In einem Sternschema eine kleinere Tabelle, die Datenattribute zu quantitativen Daten in einer Faktentabelle enthält. Bei Attributen von Dimensionstabellen handelt es sich in der Regel um Textfelder oder diskrete Zahlen, die sich wie Text verhalten. Diese Attribute werden häufig zum Einschränken von Abfragen, zum Filtern und zur Kennzeichnung von Ergebnismengen verwendet.

Katastrophe

Ein Ereignis, das verhindert, dass ein Workload oder ein System seine Geschäftsziele an seinem primären Einsatzort erfüllt. Diese Ereignisse können Naturkatastrophen, technische Ausfälle oder das Ergebnis menschlichen Handelns sein, wie z. B. unbeabsichtigte Fehlkonfigurationen oder ein Malware-Angriff.

Notfallwiederherstellung (DR)

Die Strategie und der Prozess, mit denen Sie Ausfallzeiten und Datenverluste aufgrund einer Katastrophe minimieren. Weitere Informationen finden Sie unter Disaster Recovery von Workloads unter AWS: Wiederherstellung in der Cloud im AWS Well-Architected Framework.

DML

Siehe Sprache zur Datenbankmanipulation.

Domainorientiertes Design

Ein Ansatz zur Entwicklung eines komplexen Softwaresystems, bei dem seine Komponenten mit sich entwickelnden Domains oder Kerngeschäftszielen verknüpft werden, denen jede Komponente dient. Dieses Konzept wurde von Eric Evans in seinem Buch Domaingesteuertes Design: Bewältigen der Komplexität im Herzen der Software (Boston: Addison-Wesley Professional, 2003) vorgestellt. Informationen darüber, wie Sie domaingesteuertes Design mit dem Strangler-Fig-Muster verwenden können, finden Sie unter Schrittweises Modernisieren älterer Microsoft ASP.NET (ASMX)-Webservices mithilfe von Containern und Amazon API Gateway.

DR

Siehe Disaster Recovery.

Erkennung von Driften

Verfolgung von Abweichungen von einer Basiskonfiguration. Sie können es beispielsweise verwenden, AWS CloudFormation um Abweichungen bei den Systemressourcen zu erkennen, oder Sie können AWS Control Tower damit Änderungen in Ihrer landing zone erkennen, die sich auf die Einhaltung von Governance-Anforderungen auswirken könnten.

DVSM

Siehe Abbildung des Wertstroms in der Entwicklung.

E

EDA

Siehe explorative Datenanalyse.

EDI

Siehe elektronischer Datenaustausch.

Edge-Computing

Die Technologie, die die Rechenleistung für intelligente Geräte an den Rändern eines IoT-Netzwerks erhöht. Im Vergleich zu Cloud Computing kann Edge Computing die Kommunikationslatenz reduzieren und die Reaktionszeit verbessern.

elektronischer Datenaustausch (EDI)

Der automatisierte Austausch von Geschäftsdokumenten zwischen Organisationen. Weitere Informationen finden Sie unter Was ist elektronischer Datenaustausch.

Verschlüsselung

Ein Rechenprozess, der Klartextdaten, die für Menschen lesbar sind, in Chiffretext umwandelt.

Verschlüsselungsschlüssel

Eine kryptografische Zeichenfolge aus zufälligen Bits, die von einem Verschlüsselungsalgorithmus generiert wird. Schlüssel können unterschiedlich lang sein, und jeder Schlüssel ist so konzipiert, dass er unvorhersehbar und einzigartig ist.

Endianismus

Die Reihenfolge, in der Bytes im Computerspeicher gespeichert werden. Big-Endian-Systeme speichern das höchstwertige Byte zuerst. Little-Endian-Systeme speichern das niedrigwertigste Byte zuerst.

Endpunkt

Siehe Service-Endpunkt.

Endpunkt-Services

Ein Service, den Sie in einer Virtual Private Cloud (VPC) hosten können, um ihn mit anderen Benutzern zu teilen. Sie können einen Endpunktdienst mit anderen AWS-Konten oder AWS Identity and Access Management (IAM AWS PrivateLink -) Prinzipalen erstellen und diesen

Berechtigungen gewähren. Diese Konten oder Prinzipale können sich privat mit Ihrem Endpunktservice verbinden, indem sie Schnittstellen-VPC-Endpunkte erstellen. Weitere Informationen finden Sie unter Einen Endpunkt-Service erstellen in der Amazon Virtual Private Cloud (Amazon VPC)-Dokumentation.

Unternehmensressourcenplanung (ERP)

Ein System, das wichtige Geschäftsprozesse (wie Buchhaltung, MES und Projektmanagement) für ein Unternehmen automatisiert und verwaltet.

Envelope-Verschlüsselung

Der Prozess der Verschlüsselung eines Verschlüsselungsschlüssels mit einem anderen Verschlüsselungsschlüssel. Weitere Informationen finden Sie unter Envelope-Verschlüsselung in der AWS Key Management Service (AWS KMS) -Dokumentation.

Umgebung

Eine Instance einer laufenden Anwendung. Die folgenden Arten von Umgebungen sind beim Cloud-Computing üblich:

- Entwicklungsumgebung – Eine Instance einer laufenden Anwendung, die nur dem Kernteam zur Verfügung steht, das für die Wartung der Anwendung verantwortlich ist. Entwicklungsumgebungen werden verwendet, um Änderungen zu testen, bevor sie in höhere Umgebungen übertragen werden. Diese Art von Umgebung wird manchmal als Testumgebung bezeichnet.
- Niedrigere Umgebungen – Alle Entwicklungsumgebungen für eine Anwendung, z. B. solche, die für erste Builds und Tests verwendet wurden.
- Produktionsumgebung – Eine Instance einer laufenden Anwendung, auf die Endbenutzer zugreifen können. In einer CI/CD Pipeline ist die Produktionsumgebung die letzte Bereitstellungsumgebung.
- Höhere Umgebungen – Alle Umgebungen, auf die auch andere Benutzer als das Kernentwicklungsteam zugreifen können. Dies kann eine Produktionsumgebung, Vorproduktionsumgebungen und Umgebungen für Benutzerakzeptanztests umfassen.

Epics

In der agilen Methodik sind dies funktionale Kategorien, die Ihnen helfen, Ihre Arbeit zu organisieren und zu priorisieren. Epics bieten eine allgemeine Beschreibung der Anforderungen und Implementierungsaufgaben. Zu den Sicherheitsepen AWS von CAF gehören beispielsweise Identitäts- und Zugriffsmanagement, Detektivkontrollen, Infrastruktursicherheit, Datenschutz und

Reaktion auf Vorfälle. Weitere Informationen zu Epics in der AWS -Migrationsstrategie finden Sie im Leitfaden zur Programm-Implementierung.

ERP

Siehe Enterprise Resource Planning.

Explorative Datenanalyse (EDA)

Der Prozess der Analyse eines Datensatzes, um seine Hauptmerkmale zu verstehen. Sie sammeln oder aggregieren Daten und führen dann erste Untersuchungen durch, um Muster zu finden, Anomalien zu erkennen und Annahmen zu überprüfen. EDA wird durchgeführt, indem zusammenfassende Statistiken berechnet und Datenvisualisierungen erstellt werden.

F

Faktentabelle

Die zentrale Tabelle in einem Sternschema. Sie speichert quantitative Daten über den Geschäftsbetrieb. In der Regel enthält eine Faktentabelle zwei Arten von Spalten: Spalten, die Kennzahlen enthalten, und Spalten, die einen Fremdschlüssel für eine Dimensionstabelle enthalten.

schnell scheitern

Eine Philosophie, die häufige und inkrementelle Tests verwendet, um den Entwicklungslebenszyklus zu verkürzen. Dies ist ein wichtiger Bestandteil eines agilen Ansatzes.

Grenze zur Fehlerisolierung

Dabei handelt es sich um eine Grenze AWS Cloud, z. B. eine Availability Zone AWS-Region, eine Steuerungsebene oder eine Datenebene, die die Auswirkungen eines Fehlers begrenzt und die Widerstandsfähigkeit von Workloads verbessert. Weitere Informationen finden Sie unter Grenzen zur AWS Fehlerisolierung.

Feature-Zweig

Siehe Zweig.

Features

Die Eingabedaten, die Sie verwenden, um eine Vorhersage zu treffen. In einem Fertigungskontext könnten Feature beispielsweise Bilder sein, die regelmäßig von der Fertigungslinie aus aufgenommen werden.

Bedeutung der Feature

Wie wichtig ein Feature für die Vorhersagen eines Modells ist. Dies wird in der Regel als numerischer Wert ausgedrückt, der mit verschiedenen Techniken wie Shapley Additive Explanations (SHAP) und integrierten Gradienten berechnet werden kann. Weitere Informationen finden Sie unter Interpretierbarkeit von Modellen für maschinelles Lernen mit AWS.

Featuretransformation

Daten für den ML-Prozess optimieren, einschließlich der Anreicherung von Daten mit zusätzlichen Quellen, der Skalierung von Werten oder der Extraktion mehrerer Informationssätze aus einem einzigen Datenfeld. Das ermöglicht dem ML-Modell, von den Daten profitieren. Wenn Sie beispielsweise das Datum „27.05.2021 00:15:37“ in „2021“, „Mai“, „Donnerstag“ und „15“ aufschlüsseln, können Sie dem Lernalgorithmus helfen, nuancierte Muster zu erlernen, die mit verschiedenen Datenkomponenten verknüpft sind.

Eingabeaufforderung mit wenigen Klicks

Bereitstellung einer kleinen Anzahl von Beispielen, die die Aufgabe und das gewünschte Ergebnis veranschaulichen, bevor das LLM aufgefordert wird, eine ähnliche Aufgabe auszuführen. Bei dieser Technik handelt es sich um eine Anwendung des kontextbezogenen Lernens, bei der Modelle anhand von Beispielen (Aufnahmen) lernen, die in Eingabeaufforderungen eingebettet sind. Bei Aufgaben, die spezifische Formatierungs-, Argumentations- oder Fachkenntnisse erfordern, kann die Eingabeaufforderung mit wenigen Handgriffen effektiv sein. Siehe auch Zero-Shot Prompting.

FGAC

Siehe detaillierte Zugriffskontrolle.

Feinkörnige Zugriffskontrolle (FGAC)

Die Verwendung mehrerer Bedingungen, um eine Zugriffsanfrage zuzulassen oder abzulehnen.

Flash-Cut-Migration

Eine Datenbankmigrationsmethode, bei der eine kontinuierliche Datenreplikation durch Erfassung von Änderungsdaten verwendet wird, um Daten in kürzester Zeit zu migrieren, anstatt einen schrittweisen Ansatz zu verwenden. Ziel ist es, Ausfallzeiten auf ein Minimum zu beschränken.

FM

Siehe Fundamentmodell.

Fundamentmodell (FM)

Ein großes neuronales Deep-Learning-Netzwerk, das mit riesigen Datensätzen generalisierter und unbeschrifteter Daten trainiert wurde. FMs sind in der Lage, eine Vielzahl allgemeiner Aufgaben zu erfüllen, z. B. Sprache zu verstehen, Text und Bilder zu generieren und Konversationen in natürlicher Sprache zu führen. Weitere Informationen finden Sie unter Was sind Foundation-Modelle.

G

Generative KI

Eine Untergruppe von KI-Modellen, die mit großen Datenmengen trainiert wurden und mit einer einfachen Textaufforderung neue Inhalte und Artefakte wie Bilder, Videos, Text und Audio erstellen können. Weitere Informationen finden Sie unter Was ist Generative KI.

Geoblocking

Siehe geografische Einschränkungen.

Geografische Einschränkungen (Geoblocking)

Bei Amazon eine Option CloudFront, um zu verhindern, dass Benutzer in bestimmten Ländern auf Inhaltsverteilungen zugreifen. Sie können eine Zulassungsliste oder eine Sperrliste verwenden, um zugelassene und gesperrte Länder anzugeben. Weitere Informationen finden Sie in der Dokumentation unter Beschränkung der geografischen Verteilung Ihrer Inhalte. CloudFront

Gitflow-Workflow

Ein Ansatz, bei dem niedrigere und höhere Umgebungen unterschiedliche Zweige in einem Quellcode-Repository verwenden. Der Gitflow-Workflow gilt als veraltet, und der Trunk-basierte Workflow ist der moderne, bevorzugte Ansatz.

goldenes Bild

Ein Snapshot eines Systems oder einer Software, der als Vorlage für die Bereitstellung neuer Instanzen dieses Systems oder dieser Software verwendet wird. In der Fertigung kann ein Golden Image beispielsweise zur Bereitstellung von Software auf mehreren Geräten verwendet werden und trägt zur Verbesserung der Geschwindigkeit, Skalierbarkeit und Produktivität bei der Geräteherstellung bei.

Greenfield-Strategie

Das Fehlen vorhandener Infrastruktur in einer neuen Umgebung. Bei der Einführung einer Neuausrichtung einer Systemarchitektur können Sie alle neuen Technologien ohne Einschränkung der Kompatibilität mit der vorhandenen Infrastruktur auswählen, auch bekannt als Brownfield. Wenn Sie die bestehende Infrastruktur erweitern, könnten Sie Brownfield- und Greenfield-Strategien mischen.

Integritätsschutz

Eine allgemeine Regel, die dazu beiträgt, Ressourcen, Richtlinien und die Einhaltung von Vorschriften in allen Unternehmenseinheiten zu regeln (OUs). Präventiver Integritätsschutz setzt Richtlinien durch, um die Einhaltung von Standards zu gewährleisten. Sie werden mithilfe von Service-Kontrollrichtlinien und IAM-Berechtigungsgrenzen implementiert. Detektivischer Integritätsschutz erkennt Richtlinienverstöße und Compliance-Probleme und generiert Warnmeldungen zur Abhilfe. Sie werden mithilfe von AWS Config, AWS Security Hub CSPM, Amazon GuardDuty AWS Trusted Advisor, Amazon Inspector und benutzerdefinierten AWS Lambda Prüfungen implementiert.

H

HEKTAR

Siehe Hochverfügbarkeit.

Heterogene Datenbankmigration

Migrieren Sie Ihre Quelldatenbank in eine Zieldatenbank, die eine andere Datenbank-Engine verwendet (z. B. Oracle zu Amazon Aurora). Eine heterogene Migration ist in der Regel Teil einer Neuarchitektur, und die Konvertierung des Schemas kann eine komplexe Aufgabe sein. AWS bietet AWS SCT, welches bei Schemakonvertierungen hilft.

hohe Verfügbarkeit (HA)

Die Fähigkeit eines Workloads, im Falle von Herausforderungen oder Katastrophen kontinuierlich und ohne Eingreifen zu arbeiten. HA-Systeme sind so konzipiert, dass sie automatisch ein Failover durchführen, gleichbleibend hohe Leistung bieten und unterschiedliche Lasten und Ausfälle mit minimalen Leistungseinbußen bewältigen.

historische Modernisierung

Ein Ansatz zur Modernisierung und Aufrüstung von Betriebstechnologiesystemen (OT), um den Bedürfnissen der Fertigungsindustrie besser gerecht zu werden. Ein Historian ist eine Art von Datenbank, die verwendet wird, um Daten aus verschiedenen Quellen in einer Fabrik zu sammeln und zu speichern.

Daten zurückhalten

Ein Teil historischer, beschrifteter Daten, der aus einem Datensatz zurückgehalten wird, der zum Trainieren eines Modells für maschinelles Lernen verwendet wird. Sie können Holdout-Daten verwenden, um die Modellleistung zu bewerten, indem Sie die Modellvorhersagen mit den Holdout-Daten vergleichen.

Homogene Datenbankmigration

Migrieren Sie Ihre Quelldatenbank zu einer Zieldatenbank, die dieselbe Datenbank-Engine verwendet (z. B. Microsoft SQL Server zu Amazon RDS für SQL Server). Eine homogene Migration ist in der Regel Teil eines Hostwechsels oder eines Plattformwechsels. Sie können native Datenbankserviceprogramme verwenden, um das Schema zu migrieren.

heiße Daten

Daten, auf die häufig zugegriffen wird, z. B. Echtzeitdaten oder aktuelle Translationsdaten. Für diese Daten ist in der Regel eine leistungsstarke Speicherebene oder -klasse erforderlich, um schnelle Abfrageantworten zu ermöglichen.

Hotfix

Eine dringende Lösung für ein kritisches Problem in einer Produktionsumgebung. Aufgrund seiner Dringlichkeit wird ein Hotfix normalerweise außerhalb des typischen DevOps Release-Workflows erstellt.

Hypercare-Phase

Unmittelbar nach dem Cutover, der Zeitraum, in dem ein Migrationsteam die migrierten Anwendungen in der Cloud verwaltet und überwacht, um etwaige Probleme zu beheben. In der Regel dauert dieser Zeitraum 1–4 Tage. Am Ende der Hypercare-Phase überträgt das Migrationsteam in der Regel die Verantwortung für die Anwendungen an das Cloud-Betriebsteam.

I

IaC

Sehen Sie Infrastruktur als Code.

Identitätsbasierte Richtlinie

Eine Richtlinie, die einem oder mehreren IAM-Prinzipalen zugeordnet ist und deren Berechtigungen innerhalb der AWS Cloud Umgebung definiert.

Leerlaufanwendung

Eine Anwendung mit einer durchschnittlichen CPU- und Arbeitsspeicherauslastung zwischen 5 und 20 Prozent über einen Zeitraum von 90 Tagen. In einem Migrationsprojekt ist es üblich, diese Anwendungen außer Betrieb zu nehmen oder sie On-Premises beizubehalten.

IIoT

Siehe Industrielles Internet der Dinge.

unveränderliche Infrastruktur

Ein Modell, das eine neue Infrastruktur für Produktionsworkloads bereitstellt, anstatt die bestehende Infrastruktur zu aktualisieren, zu patchen oder zu modifizieren. Unveränderliche Infrastrukturen sind von Natur aus konsistenter, zuverlässiger und vorhersehbarer als veränderliche Infrastrukturen. Weitere Informationen finden Sie in der Best Practice Deploy using immutable infrastructure im AWS Well-Architected Framework.

Eingehende (ingress) VPC

In einer Architektur AWS mit mehreren Konten ist dies eine VPC, die Netzwerkverbindungen von außerhalb einer Anwendung akzeptiert, überprüft und weiterleitet. Die AWS Security Reference Architecture empfiehlt, Ihr Netzwerkkonto mit eingehendem und ausgehendem Datenverkehr und Inspektion einzurichten, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

Inkrementelle Migration

Eine Cutover-Strategie, bei der Sie Ihre Anwendung in kleinen Teilen migrieren, anstatt eine einziges vollständiges Cutover durchzuführen. Beispielsweise könnten Sie zunächst nur einige Microservices oder Benutzer auf das neue System umstellen. Nachdem Sie sich vergewissert haben, dass alles ordnungsgemäß funktioniert, können Sie weitere Microservices oder Benutzer

schrittweise verschieben, bis Sie Ihr Legacy-System außer Betrieb nehmen können. Diese Strategie reduziert die mit großen Migrationen verbundenen Risiken.

Industrie 4.0

Ein Begriff, der 2016 von Klaus Schwab eingeführt wurde und sich auf die Modernisierung von Fertigungsprozessen durch Fortschritte in den Bereichen Konnektivität, Echtzeitdaten, Automatisierung, Analytik und KI/ML bezieht.

Infrastruktur

Alle Ressourcen und Komponenten, die in der Umgebung einer Anwendung enthalten sind.

Infrastructure as Code (IaC)

Der Prozess der Bereitstellung und Verwaltung der Infrastruktur einer Anwendung mithilfe einer Reihe von Konfigurationsdateien. IaC soll Ihnen helfen, das Infrastrukturmanagement zu zentralisieren, Ressourcen zu standardisieren und schnell zu skalieren, sodass neue Umgebungen wiederholbar, zuverlässig und konsistent sind.

industrielles Internet der Dinge (T) IIo

Einsatz von mit dem Internet verbundenen Sensoren und Geräten in Industriesektoren wie Fertigung, Energie, Automobilindustrie, Gesundheitswesen, Biowissenschaften und Landwirtschaft. Weitere Informationen finden Sie unter Aufbau einer digitalen Transformationsstrategie für das industrielle Internet der Dinge (IIoT).

Inspektions-VPC

In einer Architektur AWS mit mehreren Konten eine zentralisierte VPC, die Inspektionen des Netzwerkverkehrs zwischen VPCs (in demselben oder unterschiedlichen AWS-Regionen), dem Internet und lokalen Netzwerken verwaltet. In der AWS Security Reference Architecture wird empfohlen, Ihr Netzwerkkonto mit eingehendem und ausgehendem Datenverkehr sowie Inspektionen einzurichten, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

Internet of Things (IoT)

Das Netzwerk verbundener physischer Objekte mit eingebetteten Sensoren oder Prozessoren, das über das Internet oder über ein lokales Kommunikationsnetzwerk mit anderen Geräten und Systemen kommuniziert. Weitere Informationen finden Sie unter Was ist IoT?

Interpretierbarkeit

Ein Merkmal eines Modells für Machine Learning, das beschreibt, inwieweit ein Mensch verstehen kann, wie die Vorhersagen des Modells von seinen Eingaben abhängen. Weitere Informationen finden Sie unter Interpretierbarkeit des Modells für maschinelles Lernen mit. AWS

IoT

Siehe Internet der Dinge.

IT information library (ITIL, IT-Informationsbibliothek)

Eine Reihe von bewährten Methoden für die Bereitstellung von IT-Services und die Abstimmung dieser Services auf die Geschäftsanforderungen. ITIL bietet die Grundlage für ITSM.

T service management (ITSM, IT-Servicemanagement)

Aktivitäten im Zusammenhang mit der Gestaltung, Implementierung, Verwaltung und Unterstützung von IT-Services für eine Organisation. Informationen zur Integration von Cloud-Vorgängen mit ITSM-Tools finden Sie im Leitfaden zur Betriebsintegration.

BIS

Siehe IT-Informationsbibliothek.

ITSM

Siehe IT-Servicemanagement.

L

Labelbasierte Zugangskontrolle (LBAC)

Eine Implementierung der Mandatory Access Control (MAC), bei der den Benutzern und den Daten selbst jeweils explizit ein Sicherheitslabelwert zugewiesen wird. Die Schnittmenge zwischen der Benutzersicherheitsbeschriftung und der Datensicherheitsbeschriftung bestimmt, welche Zeilen und Spalten für den Benutzer sichtbar sind.

Landing Zone

Eine landing zone ist eine gut strukturierte AWS Umgebung mit mehreren Konten, die skalierbar und sicher ist. Dies ist ein Ausgangspunkt, von dem aus Ihre Organisationen Workloads und Anwendungen schnell und mit Vertrauen in ihre Sicherheits- und Infrastrukturumgebung starten

und bereitstellen können. Weitere Informationen zu Landing Zones finden Sie unter Einrichtung einer sicheren und skalierbaren AWS -Umgebung mit mehreren Konten..

großes Sprachmodell (LLM)

Ein Deep-Learning-KI-Modell, das anhand einer riesigen Datenmenge vorab trainiert wurde. Ein LLM kann mehrere Aufgaben ausführen, z. B. Fragen beantworten, Dokumente zusammenfassen, Text in andere Sprachen übersetzen und Sätze vervollständigen. Weitere Informationen finden Sie unter Was sind. LLMs

Große Migration

Eine Migration von 300 oder mehr Servern.

SCHWARZ

Siehe Labelbasierte Zugriffskontrolle.

Geringste Berechtigung

Die bewährte Sicherheitsmethode, bei der nur die für die Durchführung einer Aufgabe erforderlichen Mindestberechtigungen erteilt werden. Weitere Informationen finden Sie unter Geringste Berechtigungen anwenden in der IAM-Dokumentation.

Lift and Shift

Siehe 7 Rs.

Little-Endian-System

Ein System, welches das niedrigwertigste Byte zuerst speichert. Siehe auch Endianness.

LLM

Siehe großes Sprachmodell.

Niedrigere Umgebungen

Siehe Umgebung.

M

Machine Learning (ML)

Eine Art künstlicher Intelligenz, die Algorithmen und Techniken zur Mustererkennung und zum Lernen verwendet. ML analysiert aufgezeichnete Daten, wie z. B. Daten aus dem Internet der

Dinge (IoT), und lernt daraus, um ein statistisches Modell auf der Grundlage von Mustern zu erstellen. Weitere Informationen finden Sie unter Machine Learning.

Hauptzweig

Siehe Filiale.

Malware

Software, die entwickelt wurde, um die Computersicherheit oder den Datenschutz zu gefährden. Malware kann Computersysteme stören, vertrauliche Informationen durchsickern lassen oder sich unbefugten Zugriff verschaffen. Beispiele für Malware sind Viren, Würmer, Ransomware, Trojaner, Spyware und Keylogger.

verwaltete Dienste

AWS-Services für die die Infrastrukturebene, das Betriebssystem und die Plattformen AWS betrieben werden, und Sie greifen auf die Endgeräte zu, um Daten zu speichern und abzurufen. Amazon Simple Storage Service (Amazon S3) und Amazon DynamoDB sind Beispiele für Managed Services. Diese werden auch als abstrakte Dienste bezeichnet.

Manufacturing Execution System (MES)

Ein Softwaresystem zur Verfolgung, Überwachung, Dokumentation und Steuerung von Produktionsprozessen, bei denen Rohstoffe in der Fertigung zu fertigen Produkten umgewandelt werden.

MAP

Siehe Migration Acceleration Program.

Mechanismus

Ein vollständiger Prozess, bei dem Sie ein Tool erstellen, die Akzeptanz des Tools vorantreiben und anschließend die Ergebnisse überprüfen, um Anpassungen vorzunehmen. Ein Mechanismus ist ein Zyklus, der sich im Laufe seiner Tätigkeit selbst verstärkt und verbessert. Weitere Informationen finden Sie unter Aufbau von Mechanismen im AWS Well-Architected Framework.

Mitgliedskonto

Alle AWS-Konten außer dem Verwaltungskonto, die Teil einer Organisation in sind. AWS Organizations Ein Konto kann jeweils nur Mitglied einer Organisation sein.

MES

Siehe Manufacturing Execution System.

Message Queuing-Telemetrietransport (MQTT)

Ein leichtes machine-to-machine (M2M) -Kommunikationsprotokoll, das auf dem Publish/Subscribe-Muster für IoT-Geräte mit beschränkten Ressourcen basiert.

Microservice

Ein kleiner, unabhängiger Dienst, der über genau definierte Kanäle kommuniziert APIs und in der Regel kleinen, eigenständigen Teams gehört. Ein Versicherungssystem kann beispielsweise Microservices beinhalten, die Geschäftsfunktionen wie Vertrieb oder Marketing oder Subdomains wie Einkauf, Schadenersatz oder Analytik zugeordnet sind. Zu den Vorteilen von Microservices gehören Agilität, flexible Skalierung, einfache Bereitstellung, wiederverwendbarer Code und Ausfallsicherheit. Weitere Informationen finden Sie unter Integration von Microservices mithilfe serverloser Dienste. AWS

Microservices-Architekturen

Ein Ansatz zur Erstellung einer Anwendung mit unabhängigen Komponenten, die jeden Anwendungsprozess als Microservice ausführen. Diese Microservices kommunizieren mithilfe von Lightweight über eine klar definierte Schnittstelle. APIs Jeder Microservice in dieser Architektur kann aktualisiert, bereitgestellt und skaliert werden, um den Bedarf an bestimmten Funktionen einer Anwendung zu decken. Weitere Informationen finden Sie unter Implementierung von Microservices auf. AWS

Migration Acceleration Program (MAP)

Ein AWS Programm, das Beratung, Unterstützung, Schulungen und Services bietet, um Unternehmen dabei zu unterstützen, eine solide betriebliche Grundlage für die Umstellung auf die Cloud zu schaffen und die anfänglichen Kosten von Migrationen auszugleichen. MAP umfasst eine Migrationsmethode für die methodische Durchführung von Legacy-Migrationen sowie eine Reihe von Tools zur Automatisierung und Beschleunigung gängiger Migrationsszenarien.

Migration in großem Maßstab

Der Prozess, bei dem der Großteil des Anwendungsportfolios in Wellen in die Cloud verlagert wird, wobei in jeder Welle mehr Anwendungen schneller migriert werden. In dieser Phase werden die bewährten Verfahren und Erkenntnisse aus den früheren Phasen zur Implementierung einer Migrationsfabrik von Teams, Tools und Prozessen zur Optimierung der Migration von Workloads durch Automatisierung und agile Bereitstellung verwendet. Dies ist die dritte Phase der AWS - Migrationsstrategie.

Migrationsfabrik

Funktionsübergreifende Teams, die die Migration von Workloads durch automatisierte, agile Ansätze optimieren. Zu den Teams in der Migrationsabteilung gehören in der Regel Betriebsabläufe, Geschäftsanalysten und Eigentümer, Migrationsingenieure, Entwickler und DevOps Experten, die in Sprints arbeiten. Zwischen 20 und 50 Prozent eines Unternehmensanwendungsportfolios bestehen aus sich wiederholenden Mustern, die durch einen Fabrik-Ansatz optimiert werden können. Weitere Informationen finden Sie in Diskussion über Migrationsfabriken und den Leitfaden zur Cloud-Migration-Fabrik in diesem Inhaltssatz.

Migrationsmetadaten

Die Informationen über die Anwendung und den Server, die für den Abschluss der Migration benötigt werden. Für jedes Migrationsmuster ist ein anderer Satz von Migrationsmetadaten erforderlich. Beispiele für Migrationsmetadaten sind das Zielsubnetz, die Sicherheitsgruppe und AWS das Konto.

Migrationsmuster

Eine wiederholbare Migrationsaufgabe, in der die Migrationsstrategie, das Migrationsziel und die verwendete Migrationsanwendung oder der verwendete Migrationsservice detailliert beschrieben werden. Beispiel: Rehost-Migration zu Amazon EC2 mit AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

Ein Online-Tool, das Informationen zur Validierung des Geschäftsszenarios für die Migration auf das bereitstellt. AWS Cloud MPA bietet eine detaillierte Portfoliobewertung (richtige Servergröße, Preisgestaltung, Gesamtbetriebskostenanalyse, Migrationskostenanalyse) sowie Migrationsplanung (Anwendungsdatenanalyse und Datenerfassung, Anwendungsgruppierung, Migrationspriorisierung und Wellenplanung). Das MPA-Tool (Anmeldung erforderlich) steht allen AWS Beratern und APN-Partnerberatern kostenlos zur Verfügung.

Migration Readiness Assessment (MRA)

Der Prozess, bei dem mithilfe des AWS CAF Erkenntnisse über den Cloud-Bereitschaftsstatus eines Unternehmens gewonnen, Stärken und Schwächen identifiziert und ein Aktionsplan zur Schließung festgestellter Lücken erstellt wird. Weitere Informationen finden Sie im Benutzerhandbuch für Migration Readiness. MRA ist die erste Phase der AWS - Migrationsstrategie.

Migrationsstrategie

Der Ansatz, der verwendet wurde, um einen Workload auf den AWS Cloud zu migrieren. Weitere Informationen finden Sie im Eintrag 7 Rs in diesem Glossar und unter Mobilisieren Sie Ihr Unternehmen, um umfangreiche Migrationen zu beschleunigen.

ML

Siehe maschinelles Lernen.

Modernisierung

Umwandlung einer veralteten (veralteten oder monolithischen) Anwendung und ihrer Infrastruktur in ein agiles, elastisches und hochverfügbares System in der Cloud, um Kosten zu senken, die Effizienz zu steigern und Innovationen zu nutzen. Weitere Informationen finden Sie unter Strategie zur Modernisierung von Anwendungen in der AWS Cloud.

Bewertung der Modernisierungsfähigkeit

Eine Bewertung, anhand derer festgestellt werden kann, ob die Anwendungen einer Organisation für die Modernisierung bereit sind, Vorteile, Risiken und Abhängigkeiten identifiziert und ermittelt wird, wie gut die Organisation den zukünftigen Status dieser Anwendungen unterstützen kann. Das Ergebnis der Bewertung ist eine Vorlage der Zielarchitektur, eine Roadmap, in der die Entwicklungsphasen und Meilensteine des Modernisierungsprozesses detailliert beschrieben werden, sowie ein Aktionsplan zur Behebung festgestellter Lücken. Weitere Informationen finden Sie unter Evaluierung der Modernisierungsbereitschaft von Anwendungen in der AWS Cloud.

Monolithische Anwendungen (Monolithen)

Anwendungen, die als ein einziger Service mit eng gekoppelten Prozessen ausgeführt werden. Monolithische Anwendungen haben verschiedene Nachteile. Wenn ein Anwendungs-Feature stark nachgefragt wird, muss die gesamte Architektur skaliert werden. Das Hinzufügen oder Verbessern der Feature einer monolithischen Anwendung wird ebenfalls komplexer, wenn die Codebasis wächst. Um diese Probleme zu beheben, können Sie eine Microservices-Architektur verwenden. Weitere Informationen finden Sie unter Zerlegen von Monolithen in Microservices.

MPA

Siehe Bewertung des Migrationsportfolios.

MQTT

Siehe Message Queuing-Telemetrietransport.

Mehrklassen-Klassifizierung

Ein Prozess, der dabei hilft, Vorhersagen für mehrere Klassen zu generieren (wobei eines von mehr als zwei Ergebnissen vorhergesagt wird). Ein ML-Modell könnte beispielsweise fragen: „Ist dieses Produkt ein Buch, ein Auto oder ein Telefon?“ oder „Welche Kategorie von Produkten ist für diesen Kunden am interessantesten?“

veränderbare Infrastruktur

Ein Modell, das die bestehende Infrastruktur für Produktionsworkloads aktualisiert und modifiziert. Für eine verbesserte Konsistenz, Zuverlässigkeit und Vorhersagbarkeit empfiehlt das AWS Well-Architected Framework die Verwendung einer unveränderlichen Infrastruktur als bewährte Methode.

O

OAC

Siehe Origin Access Control.

EICHE

Siehe Zugriffsidentität von Origin.

COM

Siehe organisatorisches Change-Management.

Offline-Migration

Eine Migrationsmethode, bei der der Quell-Workload während des Migrationsprozesses heruntergefahren wird. Diese Methode ist mit längeren Ausfallzeiten verbunden und wird in der Regel für kleine, unkritische Workloads verwendet.

OI

Siehe Betriebsintegration.

OLA

Siehe Vereinbarung auf operativer Ebene.

Online-Migration

Eine Migrationsmethode, bei der der Quell-Workload auf das Zielsystem kopiert wird, ohne offline genommen zu werden. Anwendungen, die mit dem Workload verbunden sind, können während

der Migration weiterhin funktionieren. Diese Methode beinhaltet keine bis minimale Ausfallzeit und wird in der Regel für kritische Produktionsworkloads verwendet.

OPC-UA

Siehe Open Process Communications — Unified Architecture.

Offene Prozesskommunikation — Einheitliche Architektur (OPC-UA)

Ein machine-to-machine (M2M) -Kommunikationsprotokoll für die industrielle Automatisierung. OPC-UA bietet einen Interoperabilitätsstandard mit Datenverschlüsselungs-, Authentifizierungs- und Autorisierungsschemata.

Vereinbarung auf Betriebsebene (OLA)

Eine Vereinbarung, in der klargestellt wird, welche funktionalen IT-Gruppen sich gegenseitig versprechen zu liefern, um ein Service Level Agreement (SLA) zu unterstützen.

Überprüfung der Betriebsbereitschaft (ORR)

Eine Checkliste mit Fragen und zugehörigen bewährten Methoden, die Ihnen helfen, Vorfälle und mögliche Ausfälle zu verstehen, zu bewerten, zu verhindern oder deren Umfang zu reduzieren. Weitere Informationen finden Sie unter Operational Readiness Reviews (ORR) im AWS Well-Architected Framework.

Betriebstechnologie (OT)

Hardware- und Softwaresysteme, die mit der physischen Umgebung zusammenarbeiten, um industrielle Abläufe, Ausrüstung und Infrastruktur zu steuern. In der Fertigung ist die Integration von OT- und Informationstechnologie (IT) -Systemen ein zentraler Schwerpunkt der Industrie 4.0-Transformationen.

Betriebsintegration (OI)

Der Prozess der Modernisierung von Abläufen in der Cloud, der Bereitschaftsplanung, Automatisierung und Integration umfasst. Weitere Informationen finden Sie im Leitfaden zur Betriebsintegration.

Organisationspfad

Ein Pfad, der von erstellt wird und in AWS CloudTrail dem alle Ereignisse für alle AWS-Konten in einer Organisation protokolliert werden. AWS Organizations Diese Spur wird in jedem AWS-Konto , der Teil der Organisation ist, erstellt und verfolgt die Aktivität in jedem Konto. Weitere Informationen finden Sie in der CloudTrail Dokumentation unter Einen Trail für eine Organisation erstellen.

Organisatorisches Veränderungsmanagement (OCM)

Ein Framework für das Management wichtiger, disruptiver Geschäftstransformationen aus Sicht der Mitarbeiter, der Kultur und der Führung. OCM hilft Organisationen dabei, sich auf neue Systeme und Strategien vorzubereiten und auf diese umzustellen, indem es die Akzeptanz von Veränderungen beschleunigt, Übergangsprobleme angeht und kulturelle und organisatorische Veränderungen vorantreibt. In der AWS Migrationsstrategie wird dieses Framework aufgrund der Geschwindigkeit des Wandels, der bei Projekten zur Cloud-Einführung erforderlich ist, als Mitarbeiterbeschleunigung bezeichnet. Weitere Informationen finden Sie im OCM-Handbuch.

Ursprungszugriffskontrolle (OAC)

In CloudFront, eine erweiterte Option zur Zugriffsbeschränkung, um Ihre Amazon Simple Storage Service (Amazon S3) -Inhalte zu sichern. OAC unterstützt alle S3-Buckets insgesamt AWS-Regionen, serverseitige Verschlüsselung mit AWS KMS (SSE-KMS) sowie dynamische PUT und `DELETE` Anfragen an den S3-Bucket.

Ursprungszugriffsidentität (OAI)

In CloudFront, eine Option zur Zugriffsbeschränkung, um Ihre Amazon S3 S3-Inhalte zu sichern. Wenn Sie OAI verwenden, CloudFront erstellt es einen Principal, mit dem sich Amazon S3 authentifizieren kann. Authentifizierte Principals können nur über eine bestimmte Distribution auf Inhalte in einem S3-Bucket zugreifen. CloudFront Siehe auch OAC, das eine detailliertere und verbesserte Zugriffskontrolle bietet.

ORR

Weitere Informationen finden Sie unter Überprüfung der Betriebsbereitschaft.

NICHT

Siehe Betriebstechnologie.

Ausgehende (egress) VPC

In einer Architektur AWS mit mehreren Konten eine VPC, die Netzwerkverbindungen verarbeitet, die von einer Anwendung aus initiiert werden. Die AWS Security Reference Architecture empfiehlt die Einrichtung Ihres Netzwerkkontos mit eingehendem und ausgehendem Datenverkehr sowie Inspektion, VPCs um die bidirektionale Schnittstelle zwischen Ihrer Anwendung und dem Internet im weiteren Sinne zu schützen.

P

Berechtigungsgrenze

Eine IAM-Verwaltungsrichtlinie, die den IAM-Prinzipalen zugeordnet ist, um die maximalen Berechtigungen festzulegen, die der Benutzer oder die Rolle haben kann. Weitere Informationen finden Sie unter Berechtigungsgrenzen für IAM-Entitys in der IAM-Dokumentation.

persönlich identifizierbare Informationen (PII)

Informationen, die, wenn sie direkt betrachtet oder mit anderen verwandten Daten kombiniert werden, verwendet werden können, um vernünftige Rückschlüsse auf die Identität einer Person zu ziehen. Beispiele für personenbezogene Daten sind Namen, Adressen und Kontaktinformationen.

Personenbezogene Daten

Siehe persönlich identifizierbare Informationen.

Playbook

Eine Reihe vordefinierter Schritte, die die mit Migrationen verbundenen Aufgaben erfassen, z. B. die Bereitstellung zentraler Betriebsfunktionen in der Cloud. Ein Playbook kann die Form von Skripten, automatisierten Runbooks oder einer Zusammenfassung der Prozesse oder Schritte annehmen, die für den Betrieb Ihrer modernisierten Umgebung erforderlich sind.

PLC

Siehe programmierbare Logiksteuerung.

PLM

Siehe Produktlebenszyklusmanagement.

policy

Ein Objekt, das Berechtigungen definieren (siehe identitätsbasierte Richtlinie), Zugriffsbedingungen spezifizieren (siehe ressourcenbasierte Richtlinie) oder die maximalen Berechtigungen für alle Konten in einer Organisation definieren kann AWS Organizations (siehe Dienststeuerungsrichtlinie).

Polyglotte Beharrlichkeit

Unabhängige Auswahl der Datenspeichertechnologie eines Microservices auf der Grundlage von Datenzugriffsmustern und anderen Anforderungen. Wenn Ihre Microservices über dieselbe Datenspeichertechnologie verfügen, kann dies zu Implementierungsproblemen oder zu

Leistungseinbußen führen. Microservices lassen sich leichter implementieren und erzielen eine bessere Leistung und Skalierbarkeit, wenn sie den Datenspeicher verwenden, der ihren Anforderungen am besten entspricht.

Portfoliobewertung

Ein Prozess, bei dem das Anwendungsportfolio ermittelt, analysiert und priorisiert wird, um die Migration zu planen. Weitere Informationen finden Sie in Bewerten der Migrationsbereitschaft.

predicate

Eine Abfragebedingung, die `true` oder zurückgibt`false`, was üblicherweise in einer Klausel vorkommt. `WHERE`

Prädikat Pushdown

Eine Technik zur Optimierung von Datenbankabfragen, bei der die Daten in der Abfrage vor der Übertragung gefiltert werden. Dadurch wird die Datenmenge reduziert, die aus der relationalen Datenbank abgerufen und verarbeitet werden muss, und die Abfrageleistung wird verbessert.

Präventive Kontrolle

Eine Sicherheitskontrolle, die verhindern soll, dass ein Ereignis eintritt. Diese Kontrollen stellen eine erste Verteidigungslinie dar, um unbefugten Zugriff oder unerwünschte Änderungen an Ihrem Netzwerk zu verhindern. Weitere Informationen finden Sie unter Präventive Kontrolle in Implementierung von Sicherheitskontrollen in AWS.

Prinzipal

Eine Entität AWS , die Aktionen ausführen und auf Ressourcen zugreifen kann. Diese Entität ist in der Regel ein Root-Benutzer für eine AWS-Konto, eine IAM-Rolle oder einen Benutzer. Weitere Informationen finden Sie unter Prinzipal in Rollenbegriffe und -konzepte in der IAM-Dokumentation.

Datenschutz von Natur aus

Ein systemtechnischer Ansatz, der den Datenschutz während des gesamten Entwicklungsprozesses berücksichtigt.

Privat gehostete Zonen

Ein Container, der Informationen darüber enthält, wie Amazon Route 53 auf DNS-Abfragen für eine Domain und deren Subdomains innerhalb einer oder mehrerer VPCs Domains antworten soll. Weitere Informationen finden Sie unter Arbeiten mit privat gehosteten Zonen in der Route-53-Dokumentation.

proaktive Steuerung

Eine Sicherheitskontrolle, die den Einsatz nicht richtlinienkonformer Ressourcen verhindern soll. Diese Steuerelemente scannen Ressourcen, bevor sie bereitgestellt werden. Wenn die Ressource nicht der Kontrolle entspricht, wird sie nicht bereitgestellt. Weitere Informationen finden Sie im Referenzhandbuch zu Kontrollen in der AWS Control Tower Dokumentation und unter Proaktive Kontrollen unter Implementierung von Sicherheitskontrollen am AWS.

Produktlebenszyklusmanagement (PLM)

Das Management von Daten und Prozessen für ein Produkt während seines gesamten Lebenszyklus, vom Design, der Entwicklung und Markteinführung über Wachstum und Reife bis hin zur Markteinführung und Markteinführung.

Produktionsumgebung

Siehe Umgebung.

Speicherprogrammierbare Steuerung (SPS)

In der Fertigung ein äußerst zuverlässiger, anpassungsfähiger Computer, der Maschinen überwacht und Fertigungsprozesse automatisiert.

schnelle Verkettung

Verwendung der Ausgabe einer LLM-Eingabeaufforderung als Eingabe für die nächste Aufforderung, um bessere Antworten zu generieren. Diese Technik wird verwendet, um eine komplexe Aufgabe in Unteraufgaben zu unterteilen oder um eine vorläufige Antwort iterativ zu verfeinern oder zu erweitern. Sie trägt dazu bei, die Genauigkeit und Relevanz der Antworten eines Modells zu verbessern und ermöglicht detailliertere, personalisierte Ergebnisse.

Pseudonymisierung

Der Prozess, bei dem persönliche Identifikatoren in einem Datensatz durch Platzhalterwerte ersetzt werden. Pseudonymisierung kann zum Schutz der Privatsphäre beitragen. Pseudonymisierte Daten gelten weiterhin als personenbezogene Daten.

publish/subscribe (pub/sub)

Ein Muster, das asynchrone Kommunikation zwischen Microservices ermöglicht, um die Skalierbarkeit und Reaktionsfähigkeit zu verbessern. In einem auf Microservices basierenden MES kann ein Microservice beispielsweise Ereignismeldungen in einem Kanal veröffentlichen, den andere Microservices abonnieren können. Das System kann neue Microservices hinzufügen, ohne den Veröffentlichungsservice zu ändern.

Q

Abfrageplan

Eine Reihe von Schritten, wie Anweisungen, die für den Zugriff auf die Daten in einem relationalen SQL-Datenbanksystem verwendet werden.

Abfrageplanregression

Wenn ein Datenbankserviceoptimierer einen weniger optimalen Plan wählt als vor einer bestimmten Änderung der Datenbankumgebung. Dies kann durch Änderungen an Statistiken, Beschränkungen, Umgebungseinstellungen, Abfrageparameter-Bindungen und Aktualisierungen der Datenbank-Engine verursacht werden.

R

RACI-Matrix

Siehe verantwortlich, rechenschaftspflichtig, konsultiert, informiert (RACI).

RAG

Siehe Erweiterte Generierung beim Abrufen.

Ransomware

Eine bösartige Software, die entwickelt wurde, um den Zugriff auf ein Computersystem oder Daten zu blockieren, bis eine Zahlung erfolgt ist.

RASCI-Matrix

Siehe verantwortlich, rechenschaftspflichtig, konsultiert, informiert (RACI).

RCAC

Siehe Zugriffskontrolle für Zeilen und Spalten.

Read Replica

Eine Kopie einer Datenbank, die nur für Lesezwecke verwendet wird. Sie können Abfragen an das Lesereplikat weiterleiten, um die Belastung auf Ihrer Primärdatenbank zu reduzieren.

neu strukturieren

Siehe 7 Rs.

Recovery Point Objective (RPO)

: Die maximal zulässige Zeitspanne seit dem letzten Datenwiederherstellungspunkt. Damit wird festgelegt, was als akzeptabler Datenverlust zwischen dem letzten Wiederherstellungspunkt und der Serviceunterbrechung gilt.

Wiederherstellungszeitziel (RTO)

: Die maximal zulässige Verzögerung zwischen der Betriebsunterbrechung und der Wiederherstellung des Dienstes.

Refaktorierung

: Siehe 7 Rs.

Region

: Eine Sammlung von AWS Ressourcen in einem geografischen Gebiet. Jeder AWS-Region ist isoliert und unabhängig von den anderen, um Fehlertoleranz, Stabilität und Belastbarkeit zu gewährleisten. Weitere Informationen finden Sie unter Geben Sie an, was AWS-Regionen Ihr Konto verwenden kann.

Regression

: Eine ML-Technik, die einen numerischen Wert vorhersagt. Zum Beispiel, um das Problem „Zu welchem Preis wird dieses Haus verkauft werden?“ zu lösen Ein ML-Modell könnte ein lineares Regressionsmodell verwenden, um den Verkaufspreis eines Hauses auf der Grundlage bekannter Fakten über das Haus (z. B. die Quadratmeterzahl) vorherzusagen.

rehosten

: Siehe 7 Rs.

Veröffentlichung

: In einem Bereitstellungsprozess der Akt der Förderung von Änderungen an einer Produktionsumgebung.

umziehen

: Siehe 7 Rs.

neue Plattform

: Siehe 7 Rs.

Rückkauf

Siehe 7 Rs.

Ausfallsicherheit

Die Fähigkeit einer Anwendung, Störungen zu widerstehen oder sich von ihnen zu erholen. Hochverfügbarkeit und Notfallwiederherstellung sind häufig Überlegungen bei der Planung der Ausfallsicherheit in der. AWS Cloud Weitere Informationen finden Sie unter AWS Cloud Resilienz.

Ressourcenbasierte Richtlinie

Eine mit einer Ressource verknüpfte Richtlinie, z. B. ein Amazon-S3-Bucket, ein Endpunkt oder ein Verschlüsselungsschlüssel. Diese Art von Richtlinie legt fest, welchen Prinzipalen der Zugriff gewährt wird, welche Aktionen unterstützt werden und welche anderen Bedingungen erfüllt sein müssen.

RACI-Matrix (verantwortlich, rechenschaftspflichtig, konsultiert, informiert)

Eine Matrix, die die Rollen und Verantwortlichkeiten aller an Migrationsaktivitäten und Cloud-Operationen beteiligten Parteien definiert. Der Matrixname leitet sich von den in der Matrix definierten Zuständigkeitstypen ab: verantwortlich (R), rechenschaftspflichtig (A), konsultiert (C) und informiert (I). Der Unterstützungstyp (S) ist optional. Wenn Sie Unterstützung einbeziehen, wird die Matrix als RASCI-Matrix bezeichnet, und wenn Sie sie ausschließen, wird sie als RACI-Matrix bezeichnet.

Reaktive Kontrolle

Eine Sicherheitskontrolle, die darauf ausgelegt ist, die Behebung unerwünschter Ereignisse oder Abweichungen von Ihren Sicherheitsstandards voranzutreiben. Weitere Informationen finden Sie unter Reaktive Kontrolle in Implementieren von Sicherheitskontrollen in AWS.

Beibehaltung

Siehe 7 Rs.

zurückziehen

Siehe 7 Rs.

Retrieval Augmented Generation (RAG)

Eine generative KI-Technologie, bei der ein LLM auf eine maßgebliche Datenquelle verweist, die sich außerhalb seiner Trainingsdatenquellen befindet, bevor eine Antwort generiert wird. Ein RAG-Modell könnte beispielsweise eine semantische Suche in der Wissensdatenbank oder in

benutzerdefinierten Daten einer Organisation durchführen. Weitere Informationen finden Sie unter Was ist RAG.

Drehung

Der Vorgang, bei dem ein Geheimnis regelmäßig aktualisiert wird, um es einem Angreifer zu erschweren, auf die Anmeldeinformationen zuzugreifen.

Zugriffskontrolle für Zeilen und Spalten (RCAC)

Die Verwendung einfacher, flexibler SQL-Ausdrücke mit definierten Zugriffsregeln. RCAC besteht aus Zeilenberechtigungen und Spaltenmasken.

RPO

Siehe Recovery Point Objective.

RTO

Siehe Ziel für die Erholungszeit.

Runbook

Eine Reihe manueller oder automatisierter Verfahren, die zur Ausführung einer bestimmten Aufgabe erforderlich sind. Diese sind in der Regel darauf ausgelegt, sich wiederholende Operationen oder Verfahren mit hohen Fehlerquoten zu rationalisieren.

S

SAML 2.0

Ein offener Standard, den viele Identitätsanbieter (IdPs) verwenden. Diese Funktion ermöglicht föderiertes Single Sign-On (SSO), sodass sich Benutzer bei den API-Vorgängen anmelden AWS-Managementkonsole oder die AWS API-Operationen aufrufen können, ohne dass Sie einen Benutzer in IAM für alle in Ihrer Organisation erstellen müssen. Weitere Informationen zum SAML-2.0.-basierten Verbund finden Sie unter Über den SAML-2.0-basierten Verbund in der IAM-Dokumentation.

SCADA

Siehe Aufsichtskontrolle und Datenerfassung.

SCP

Siehe Richtlinie zur Dienstkontrolle.

Secret

Interne AWS Secrets Manager, vertrauliche oder eingeschränkte Informationen, wie z. B. ein Passwort oder Benutzeranmeldeinformationen, die Sie in verschlüsselter Form speichern. Es besteht aus dem geheimen Wert und seinen Metadaten. Der geheime Wert kann binär, eine einzelne Zeichenfolge oder mehrere Zeichenketten sein. Weitere Informationen finden Sie unter Was ist in einem Secrets Manager Manager-Geheimnis? in der Secrets Manager Manager-Dokumentation.

Sicherheit durch Design

Ein systemtechnischer Ansatz, der die Sicherheit während des gesamten Entwicklungsprozesses berücksichtigt.

Sicherheitskontrolle

Ein technischer oder administrativer Integritätsschutz, der die Fähigkeit eines Bedrohungsakteurs, eine Schwachstelle auszunutzen, verhindert, erkennt oder einschränkt. Es gibt vier Haupttypen von Sicherheitskontrollen: präventiv, detektiv, reaktionsschnell und proaktiv.

Härtung der Sicherheit

Der Prozess, bei dem die Angriffsfläche reduziert wird, um sie widerstandsfähiger gegen Angriffe zu machen. Dies kann Aktionen wie das Entfernen von Ressourcen, die nicht mehr benötigt werden, die Implementierung der bewährten Sicherheitsmethode der Gewährung geringster Berechtigungen oder die Deaktivierung unnötiger Feature in Konfigurationsdateien umfassen.

System zur Verwaltung von Sicherheitsinformationen und Ereignissen (security information and event management – SIEM)

Tools und Services, die Systeme für das Sicherheitsinformationsmanagement (SIM) und das Management von Sicherheitsereignissen (SEM) kombinieren. Ein SIEM-System sammelt, überwacht und analysiert Daten von Servern, Netzwerken, Geräten und anderen Quellen, um Bedrohungen und Sicherheitsverletzungen zu erkennen und Warnmeldungen zu generieren.

Automatisierung von Sicherheitsreaktionen

Eine vordefinierte und programmierte Aktion, die darauf ausgelegt ist, automatisch auf ein Sicherheitsereignis zu reagieren oder es zu beheben. Diese Automatisierungen dienen als detektive oder reaktionsschnelle Sicherheitskontrollen, die Sie bei der Implementierung bewährter AWS Sicherheitsmethoden unterstützen. Beispiele für automatisierte Antwortaktionen sind das Ändern einer VPC-Sicherheitsgruppe, das Patchen einer Amazon EC2 EC2-Instance oder das Rotieren von Anmeldeinformationen.

Serverseitige Verschlüsselung

Verschlüsselung von Daten am Zielort durch denjenigen AWS-Service , der sie empfängt.

Service-Kontrollrichtlinie (SCP)

Eine Richtlinie, die eine zentrale Steuerung der Berechtigungen für alle Konten in einer Organisation in ermöglicht AWS Organizations. SCPs Definieren Sie Leitplanken oder legen Sie Grenzwerte für Aktionen fest, die ein Administrator an Benutzer oder Rollen delegieren kann. Sie können sie SCPs als Zulassungs- oder Ablehnungslisten verwenden, um festzulegen, welche Dienste oder Aktionen zulässig oder verboten sind. Weitere Informationen finden Sie in der AWS Organizations Dokumentation unter Richtlinien zur Dienststeuerung.

Service-Endpunkt

Die URL des Einstiegspunkts für einen AWS-Service. Sie können den Endpunkt verwenden, um programmgesteuert eine Verbindung zum Zielservice herzustellen. Weitere Informationen finden Sie unter AWS-Service -Endpunkte in der Allgemeine AWS-Referenz.

Service Level Agreement (SLA)

Eine Vereinbarung, in der klargestellt wird, was ein IT-Team seinen Kunden zu bieten verspricht, z. B. in Bezug auf Verfügbarkeit und Leistung der Services.

Service-Level-Indikator (SLI)

Eine Messung eines Leistungsaspekts eines Dienstes, z. B. seiner Fehlerrate, Verfügbarkeit oder Durchsatz.

Service-Level-Ziel (SLO)

Eine Zielkennzahl, die den Zustand eines Dienstes darstellt, gemessen anhand eines Service-Level-Indikators.

Modell der geteilten Verantwortung

Ein Modell, das die Verantwortung beschreibt, mit der Sie gemeinsam AWS für Cloud-Sicherheit und Compliance verantwortlich sind. AWS ist für die Sicherheit der Cloud verantwortlich, während Sie für die Sicherheit in der Cloud verantwortlich sind. Weitere Informationen finden Sie unter Modell der geteilten Verantwortung.

SIEM

Siehe Sicherheitsinformations- und Event-Management-System.

Single Point of Failure (SPOF)

Ein Fehler in einer einzelnen, kritischen Komponente einer Anwendung, der das System stören kann.

SLA

Siehe Service Level Agreement.

SLI

Siehe Service-Level-Indikator.

ALSO

Siehe Service-Level-Ziel.

split-and-seed Modell

Ein Muster für die Skalierung und Beschleunigung von Modernisierungsprojekten. Sobald neue Features und Produktversionen definiert werden, teilt sich das Kernteam auf, um neue Produktteams zu bilden. Dies trägt zur Skalierung der Fähigkeiten und Services Ihrer Organisation bei, verbessert die Produktivität der Entwickler und unterstützt schnelle Innovationen. Weitere Informationen finden Sie unter Schrittweiser Ansatz zur Modernisierung von Anwendungen in der AWS Cloud.

SPOTTEN

Siehe Single Point of Failure.

Sternschema

Eine Datenbank-Organisationsstruktur, die eine große Faktentabelle zum Speichern von Transaktions- oder Messdaten und eine oder mehrere kleinere dimensionale Tabellen zum Speichern von Datenattributen verwendet. Diese Struktur ist für die Verwendung in einem Data Warehouse oder für Business Intelligence-Zwecke konzipiert.

Strangler-Fig-Muster

Ein Ansatz zur Modernisierung monolithischer Systeme, bei dem die Systemfunktionen schrittweise umgeschrieben und ersetzt werden, bis das Legacy-System außer Betrieb genommen werden kann. Dieses Muster verwendet die Analogie einer Feigenrebe, die zu einem etablierten Baum heranwächst und schließlich ihren Wirt überwindet und ersetzt. Das Muster wurde eingeführt von Martin Fowler als Möglichkeit, Risiken beim Umschreiben monolithischer Systeme zu managen. Ein Beispiel für die Anwendung dieses Musters finden Sie

unter Schrittweises Modernisieren älterer Microsoft ASP.NET (ASMX)-Webservices mithilfe von Containern und Amazon API Gateway.

Subnetz

Ein Bereich von IP-Adressen in Ihrer VPC. Ein Subnetz muss sich in einer einzigen Availability Zone befinden.

Aufsichtskontrolle und Datenerfassung (SCADA)

In der Fertigung ein System, das Hardware und Software zur Überwachung von Sachanlagen und Produktionsabläufen verwendet.

Symmetrische Verschlüsselung

Ein Verschlüsselungsalgorithmus, der denselben Schlüssel zum Verschlüsseln und Entschlüsseln der Daten verwendet.

synthetisches Testen

Testen eines Systems auf eine Weise, die Benutzerinteraktionen simuliert, um potenzielle Probleme zu erkennen oder die Leistung zu überwachen. Sie können Amazon CloudWatch Synthetics verwenden, um diese Tests zu erstellen.

Systemaufforderung

Eine Technik, mit der einem LLM Kontext, Anweisungen oder Richtlinien zur Verfügung gestellt werden, um sein Verhalten zu steuern. Systemaufforderungen helfen dabei, den Kontext festzulegen und Regeln für Interaktionen mit Benutzern festzulegen.

T

tags

Schlüssel-Wert-Paare, die als Metadaten für die Organisation Ihrer Ressourcen dienen. AWS Mit Tags können Sie Ressourcen verwalten, identifizieren, organisieren, suchen und filtern. Weitere Informationen finden Sie unter Markieren Ihrer AWS -Ressourcen.

Zielvariable

Der Wert, den Sie in überwachtem ML vorhersagen möchten. Dies wird auch als Ergebnisvariable bezeichnet. In einer Fertigungsumgebung könnte die Zielvariable beispielsweise ein Produktfehler sein.

Aufgabenliste

Ein Tool, das verwendet wird, um den Fortschritt anhand eines Runbooks zu verfolgen. Eine Aufgabenliste enthält eine Übersicht über das Runbook und eine Liste mit allgemeinen Aufgaben, die erledigt werden müssen. Für jede allgemeine Aufgabe werden der geschätzte Zeitaufwand, der Eigentümer und der Fortschritt angegeben.

Testumgebungen

Siehe Umgebung.

Training

Daten für Ihr ML-Modell bereitstellen, aus denen es lernen kann. Die Trainingsdaten müssen die richtige Antwort enthalten. Der Lernalgorithmus findet Muster in den Trainingsdaten, die die Attribute der Input-Daten dem Ziel (die Antwort, die Sie voraussagen möchten) zuordnen. Es gibt ein ML-Modell aus, das diese Muster erfasst. Sie können dann das ML-Modell verwenden, um Voraussagen für neue Daten zu erhalten, bei denen Sie das Ziel nicht kennen.

Transit-Gateway

Ein Netzwerk-Transit-Hub, über den Sie Ihre Netzwerke VPCs und Ihre lokalen Netzwerke miteinander verbinden können. Weitere Informationen finden Sie in der Dokumentation unter Was ist ein Transit-Gateway. AWS Transit Gateway

Stammbasierter Workflow

Ein Ansatz, bei dem Entwickler Feature lokal in einem Feature-Zweig erstellen und testen und diese Änderungen dann im Hauptzweig zusammenführen. Der Hauptzweig wird dann sequentiell für die Entwicklungs-, Vorproduktions- und Produktionsumgebungen erstellt.

Vertrauenswürdiger Zugriff

Gewährung von Berechtigungen für einen Dienst, den Sie angeben, um Aufgaben in Ihrer Organisation AWS Organizations und in deren Konten in Ihrem Namen auszuführen. Der vertrauenswürdige Service erstellt in jedem Konto eine mit dem Service verknüpfte Rolle, wenn diese Rolle benötigt wird, um Verwaltungsaufgaben für Sie auszuführen. Weitere Informationen finden Sie in der AWS Organizations Dokumentation unter Verwendung AWS Organizations mit anderen AWS Diensten.

Optimieren

Aspekte Ihres Trainingsprozesses ändern, um die Genauigkeit des ML-Modells zu verbessern. Sie können das ML-Modell z. B. trainieren, indem Sie einen Beschriftungssatz generieren,

Beschriftungen hinzufügen und diese Schritte dann mehrmals unter verschiedenen Einstellungen wiederholen, um das Modell zu optimieren.

Zwei-Pizzen-Team

Ein kleines DevOps Team, das Sie mit zwei Pizzen ernähren können. Eine Teamgröße von zwei Pizzen gewährleistet die bestmögliche Gelegenheit zur Zusammenarbeit bei der Softwareentwicklung.

U

Unsicherheit

Ein Konzept, das sich auf ungenaue, unvollständige oder unbekannte Informationen bezieht, die die Zuverlässigkeit von prädiktiven ML-Modellen untergraben können. Es gibt zwei Arten von Unsicherheit: Epistemische Unsicherheit wird durch begrenzte, unvollständige Daten verursacht, wohingegen aleatorische Unsicherheit durch Rauschen und Randomisierung verursacht wird, die in den Daten liegt. Weitere Informationen finden Sie im Leitfaden Quantifizieren der Unsicherheit in Deep-Learning-Systemen.

undifferenzierte Aufgaben

Diese Arbeit wird auch als Schwerstarbeit bezeichnet. Dabei handelt es sich um Arbeiten, die zwar für die Erstellung und den Betrieb einer Anwendung erforderlich sind, aber dem Endbenutzer keinen direkten Mehrwert bieten oder keinen Wettbewerbsvorteil bieten. Beispiele für undifferenzierte Aufgaben sind Beschaffung, Wartung und Kapazitätsplanung.

höhere Umgebungen

Siehe Umgebung.

V

Vacuuming

Ein Vorgang zur Datenbankwartung, bei dem die Datenbank nach inkrementellen Aktualisierungen bereinigt wird, um Speicherplatz zurückzugewinnen und die Leistung zu verbessern.

Versionskontrolle

Prozesse und Tools zur Nachverfolgung von Änderungen, z. B. Änderungen am Quellcode in einem Repository.

VPC-Peering

Eine Verbindung zwischen zwei VPCs , die es Ihnen ermöglicht, den Verkehr mithilfe privater IP-Adressen weiterzuleiten. Weitere Informationen finden Sie unter Was ist VPC-Peering? in der Amazon-VPC-Dokumentation.

Schwachstelle

Ein Software- oder Hardwarefehler, der die Sicherheit des Systems beeinträchtigt.

W

Warmer Cache

Ein Puffer-Cache, der aktuelle, relevante Daten enthält, auf die häufig zugegriffen wird. Die Datenbank-Instance kann aus dem Puffer-Cache lesen, was schneller ist als das Lesen aus dem Hauptspeicher oder von der Festplatte.

warme Daten

Daten, auf die selten zugegriffen wird. Bei der Abfrage dieser Art von Daten sind mäßig langsame Abfragen in der Regel akzeptabel.

Fensterfunktion

Eine SQL-Funktion, die eine Berechnung für eine Gruppe von Zeilen durchführt, die sich in irgendeiner Weise auf den aktuellen Datensatz beziehen. Fensterfunktionen sind nützlich für die Verarbeitung von Aufgaben wie die Berechnung eines gleitenden Durchschnitts oder für den Zugriff auf den Wert von Zeilen auf der Grundlage der relativen Position der aktuellen Zeile.

Workload

Ein Workload ist eine Sammlung von Ressourcen und Code, die einen Unternehmenswert bietet, wie z. B. eine kundenorientierte Anwendung oder ein Backend-Prozess.

Workstream

Funktionsgruppen in einem Migrationsprojekt, die für eine bestimmte Reihe von Aufgaben verantwortlich sind. Jeder Workstream ist unabhängig, unterstützt aber die anderen Workstreams

im Projekt. Der Portfolio-Workstream ist beispielsweise für die Priorisierung von Anwendungen, die Wellenplanung und die Erfassung von Migrationsmetadaten verantwortlich. Der Portfolio-Workstream liefert diese Komponenten an den Migrations-Workstream, der dann die Server und Anwendungen migriert.

WURM

Sehen Sie einmal schreiben, viele lesen.

WQF

Siehe AWS Workload-Qualifizierungsrahmen.

einmal schreiben, viele lesen (WORM)

Ein Speichermodell, das Daten ein einziges Mal schreibt und verhindert, dass die Daten gelöscht oder geändert werden. Autorisierte Benutzer können die Daten so oft wie nötig lesen, aber sie können sie nicht ändern. Diese Datenspeicherinfrastruktur gilt als unveränderlich.

Z

Zero-Day-Exploit

Ein Angriff, in der Regel Malware, der eine Zero-Day-Sicherheitslücke ausnutzt.

Zero-Day-Sicherheitslücke

Ein unfehlbarer Fehler oder eine Sicherheitslücke in einem Produktionssystem. Bedrohungsakteure können diese Art von Sicherheitslücke nutzen, um das System anzugreifen. Entwickler werden aufgrund des Angriffs häufig auf die Sicherheitsanfälligkeit aufmerksam.

Eingabeaufforderung ohne Zwischenfälle

Bereitstellung von Anweisungen für die Ausführung einer Aufgabe an einen LLM, jedoch ohne Beispiele (Schnappschüsse), die ihm als Orientierungshilfe dienen könnten. Der LLM muss sein vortrainiertes Wissen einsetzen, um die Aufgabe zu bewältigen. Die Effektivität von Zero-Shot Prompting hängt von der Komplexität der Aufgabe und der Qualität der Aufforderung ab. Siehe auch Few-Shot-Prompting.

Zombie-Anwendung

Eine Anwendung, deren durchschnittliche CPU- und Arbeitsspeichernutzung unter 5 Prozent liegt. In einem Migrationsprojekt ist es üblich, diese Anwendungen außer Betrieb zu nehmen.

Die vorliegende Übersetzung wurde maschinell erstellt. Im Falle eines Konflikts oder eines Widerspruchs zwischen dieser übersetzten Fassung und der englischen Fassung (einschließlich infolge von Verzögerungen bei der Übersetzung) ist die englische Fassung maßgeblich.