Strategie Model Context Protocol su AWS

AWS Linee guida prescrittive

AWS Linee guida prescrittive: Strategie Model Context Protocol su AWS

Table of Contents

Strategie Model Context Protocol su AWS

Amazon Web Services (collaboratori)

Marzo 2026 (cronologia dei documenti)

Questa guida può aiutarti a sviluppare e implementare strategie MCP (Model Context Protocol) in tutta l'organizzazione per supportare il tuo percorso verso l'intelligenza artificiale agentica. Poiché gli agenti e i modelli linguistici diventano sempre più centrali nelle operazioni aziendali, la definizione di una strategia MCP è fondamentale per soluzioni agentiche di successo.

Questa guida esplora tre pilastri fondamentali per la creazione di una strategia MCP: progettazione di strumenti MCP, hosting di server MCP e governance MCP. Affrontando questi componenti interconnessi, le organizzazioni possono creare sistemi scalabili, sicuri ed efficaci per la gestione del contesto del modello in tutte le loro implementazioni di intelligenza artificiale. Queste linee guida forniscono informazioni utili e linee guida strategiche per le organizzazioni in qualsiasi fase del percorso di intelligenza artificiale di un'organizzazione, dalla sperimentazione iniziale alle implementazioni di produzione su vasta scala. Questo le aiuta a sviluppare soluzioni MCP personalizzate in linea con le loro esigenze e obiettivi specifici.

Queste best practice derivano da implementazioni reali di organizzazioni che implementano MCP su scala aziendale, da un'analisi degli attuali standard di specifica MCP e dalle lezioni apprese dalle applicazioni personalizzate Large Language Model (LLM) in produzione.

I sistemi di intelligenza artificiale utilizzano sistemi sempre più sofisticati e robusti LLMs in un'ampia varietà di casi d'uso. LLMs eccellono nella comprensione del linguaggio naturale, nella generazione di risposte simili a quelle umane e nel ragionamento su informazioni complesse. Tuttavia, per passare LLMs da interfacce conversazionali a sistemi in grado di svolgere in modo autonomo attività complesse, le organizzazioni stanno adottando architetture di intelligenza artificiale agentiche, sistemi di intelligenza artificiale in grado di percepire l'ambiente, ragionare sugli obiettivi, prendere decisioni autonome, orchestrare in più fasi e intraprendere azioni per raggiungere gli obiettivi per conto degli utenti. Questo approccio agentico aiuta le organizzazioni a creare sistemi di intelligenza artificiale in grado di comprendere l'intento dell'utente attraverso il linguaggio naturale, coordinarsi autonomamente tra più fonti di dati e strumenti e fornire esperienze personalizzate su una scala che non era possibile con i tradizionali modelli di richiesta-risposta. Per rendere questi agenti più capaci, le organizzazioni devono fornire l'accesso agli strumenti e ai dati esistenti per arricchire la comprensione contestuale dell'agente e consentirgli di agire per conto dell'utente.

MCP fornisce un protocollo standardizzato per l'integrazione di strumenti di intelligenza artificiale, che consente una comunicazione coerente tra agenti e risorse esterne. Sebbene MCP stesso definisca lo standard di comunicazione, la sua implementazione efficace richiede un'attenta considerazione dei modelli architettonici, dei modelli di sicurezza, delle pratiche operative e delle strategie di ottimizzazione delle prestazioni per ottenere soluzioni scalabili, sicure e gestibili.

Questa guida sintetizza le lezioni apprese dalle implementazioni MCP aziendali, fornendo consigli pratici in linea con il Well-Architected Framework.AWS Descrive le strategie per la progettazione di strumenti MCP, l'hosting di server MCP e la governance MCP, essenziali per creare soluzioni MCP personalizzate. Le raccomandazioni contenute in questa guida si riferiscono ai seguenti cinque pilastri del AWS Well-Architected Framework:

- Sicurezza: isolamento dei token, credenziali ridotte, autorizzazione separata read/write
- Eccellenza operativa: metriche di precisione nella selezione degli strumenti, set di dati eccezionali per i test di regressione
- Affidabilità: limitazione della velocità per utente e per utensile, riduzione del carico
- Efficienza delle prestazioni: strumenti basati sul flusso di lavoro, filtraggio degli strumenti, ricerca semantica per ridurre l'utilizzo delle finestre contestuali
- Ottimizzazione dei costi: server MCP riutilizzabili tra i team, riduzione dei costi dei token per richiesta grazie al filtraggio degli strumenti

Destinatari principali

Questa guida è destinata ad architetti, sviluppatori e leader tecnologici che implementano soluzioni di intelligenza artificiale agentica nelle loro organizzazioni. Per comprendere i concetti di questa guida, è necessario comprendere come LLMs funziona e avere conoscenze di base su MCP, strumenti e progettazione rapida.

Obiettivi

Costruire sistemi di intelligenza artificiale Agentic pronti per la produzione significa risolvere insieme i problemi di governance, ottimizzazione e sicurezza per supportare le politiche dell'organizzazione. Di seguito viene spiegato in che modo questa guida affronta questi obiettivi:

- Governance: senza una governance centralizzata, non è possibile rispondere alle domande di audit sui carichi di lavoro di intelligenza artificiale, tra cui quali agenti hanno avuto accesso a quali

dati, con quali autorizzazioni e quando. Inoltre, non è possibile imporre il controllo delle versioni. La sezione sulla strategia di hosting MCP di questa guida spiega come gli utenti potrebbero utilizzare server MCP locali obsoleti con vulnerabilità note a causa della mancanza di applicazioni sistematiche.

Per i settori regolamentati, la governance è fondamentale. I revisori vogliono vedere l'applicazione delle politiche e il monitoraggio dell'utilizzo degli strumenti tra tutti gli agenti da un unico pannello. La governance MCP lo prevede.

Seguendo i consigli di questa guida, è possibile migliorare la precisione delle attività del 28-32% nei benchmark sottoposti a revisione paritaria. Per ulteriori informazioni, consulta MARCO: Multi-Agent Real-time Chat Orchestration (sito web ACL Anthology). La governance non riguarda solo la conformità, ma migliora anche le prestazioni del sistema di intelligenza artificiale agentica.

- Ottimizzazione: i tuoi team potrebbero creare le stesse integrazioni più di una volta. Ad esempio, quando cinque team diversi scrivono il proprio script di interrogazione del database per consentire alla loro applicazione di intelligenza artificiale di comunicare con i propri database, si tratta di cinque volte il costo di sviluppo e cinque serie di bug da gestire. MCP ti consente di crearlo una sola volta e condividerlo con l'intera comunità di ingegneri. I risparmi aumentano man mano che aumenta il numero di agenti.

  C'è anche un problema di costo per richiesta che la maggior parte dei team non nota all'inizio. Ogni definizione di strumento utilizza i token della finestra contestuale. Con 20 strumenti, spendi 5.000-10.000 token per invocazione solo per le descrizioni, oltre alle richieste degli utenti. Ciò aumenta la latenza e i costi di inferenza LLM e riduce la precisione poiché il modello fatica a scegliere lo strumento giusto dall'elenco degli strumenti disponibili.

  Gli agenti che utilizzano strumenti wrapper strutturati sono circa tre volte più accurati nelle attività del database rispetto agli agenti che accedono APIs direttamente (per ulteriori informazioni, vedere Middleware for LLMs: Gli strumenti sono strumentali per gli agenti linguistici in ambienti complessi). Il modo in cui si progettano e si presentano gli strumenti per un modello di intelligenza artificiale è importante. Questa guida consiglia di fornire agli strumenti schemi chiari, di applicarli ai flussi di lavoro effettivi anziché agli endpoint grezzi e di limitare le informazioni nella finestra contestuale. La sezione sulla strategia di progettazione degli strumenti MCP di questa guida approfondisce questi aspetti.
- Sicurezza e conformità: immaginate un sistema di intelligenza artificiale agentico che allucina una fase di pulizia e tenta di eliminare un database di produzione. Se l'agente ha ereditato le credenziali di amministratore complete dell'utente, l'eliminazione potrebbe andare a buon fine. Con

l'isolamento dei token e le credenziali ridotte che garantiscono solo l'accesso in lettura e creazione, l'operazione fallisce in modo sicuro.

I flussi di lavoro regolamentati migliorano ulteriormente questa situazione. La guida fornisce alcuni esempi (pipeline sanitarie che richiedono la convalida HIPAA e l'anonimizzazione delle informazioni di identificazione personale prima di elaborare i dati dei pazienti). L'integrazione di tale logica negli strumenti MCP significa che la conformità avviene in modo deterministico ogni volta.

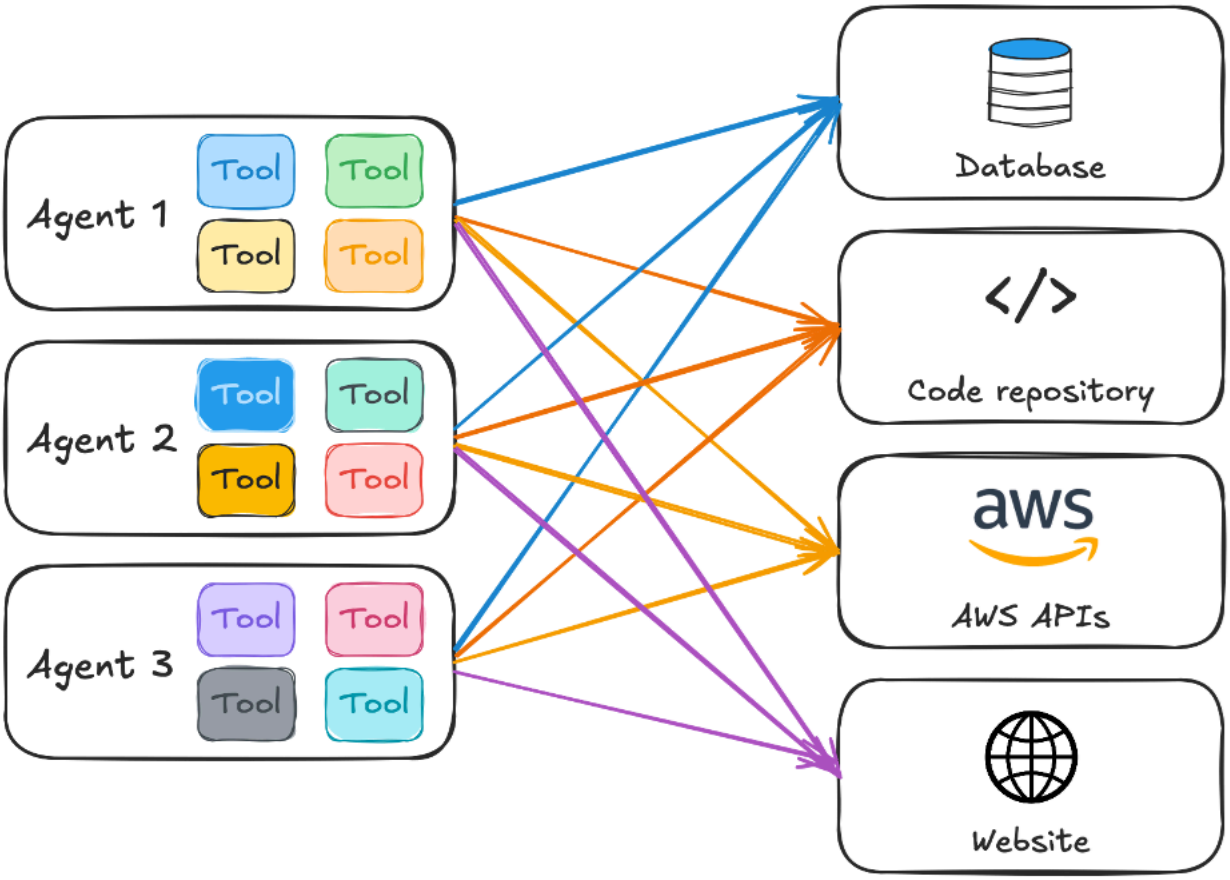
Che cos'è l'MCP?

I LLM funzionano prevedendo una risposta a un prompt in base ai dati di addestramento. Ciò significa che l'LLM può fornire solo risposte su dati ed eventi che ha già visto. Metodi come Retrieval Augmented Generation (RAG) e le knowledge base consentono di includere dati contestuali. Tuttavia, se chiedeste a un LLM quali saranno le previsioni del tempo di domani o quanti clienti ci sono nel vostro database, probabilmente vi sarebbero allucinazioni o non sarebbe in grado di fornire una risposta, perché queste non rientrano nelle conoscenze pre-addestrate del LLM. Per poter rispondere a questo tipo di domande, un agente deve accedere a funzionalità, dati e API esterne al di fuori del contesto nativo del LLM.

Strumenti di comprensione

Possiamo fornire all'LLM l'accesso a sistemi e contesti aggiuntivi tramite strumenti.Gli strumenti sono funzioni fornite al LLM per raggiungere un obiettivo chiaro. Uno strumento può richiamare un'API, interrogare un database, eseguire operazioni con la calcolatrice, gestire una sandbox di codice, eseguire una ricerca sul Web e persino richiamare un altro sistema di intelligenza artificiale o agent-as-a-tool. Ogni strumento dovrebbe includere una descrizione che indichi all'LLM cosa fa lo strumento, quando usarlo e quali parametri accetta. Ciò consente all'LLM di prendere decisioni dettagliate su quale strumento o combinazione di strumenti invocare in base all'input dell'utente. L'LLM viene informato sugli strumenti a disposizione dell'agente, consentendogli di generare risposte che indicano all'agente di richiamare lo strumento. Ad esempio, quando chiedi all'LLM quanti clienti ci sono nel tuo database, l'LLM invierà una risposta all'agente richiedendo di eseguire lo strumento con parametri di input specifici. `query_database` L'LLM determina quale strumento richiamare e gli input per la chiamata allo strumento. L'agente esegue quindi lo strumento, che converte l'input in linguaggio naturale in una chiamata di funzione sintatticamente corretta ed esegue la query. L'agente richiama lo strumento o gli strumenti in base alle istruzioni del LLM e tali risultati vengono restituiti all'LLM. Ciò sfrutta la capacità dell'LLM di ragionare utilizzando input testuali e di selezionare gli strumenti appropriati per il lavoro.

L'immagine seguente mostra come ogni agente gestisce il proprio set di strumenti per ogni destinazione.



La scalabilità dell'accesso agli strumenti può presentare sfide per le soluzioni di intelligenza artificiale agentica:

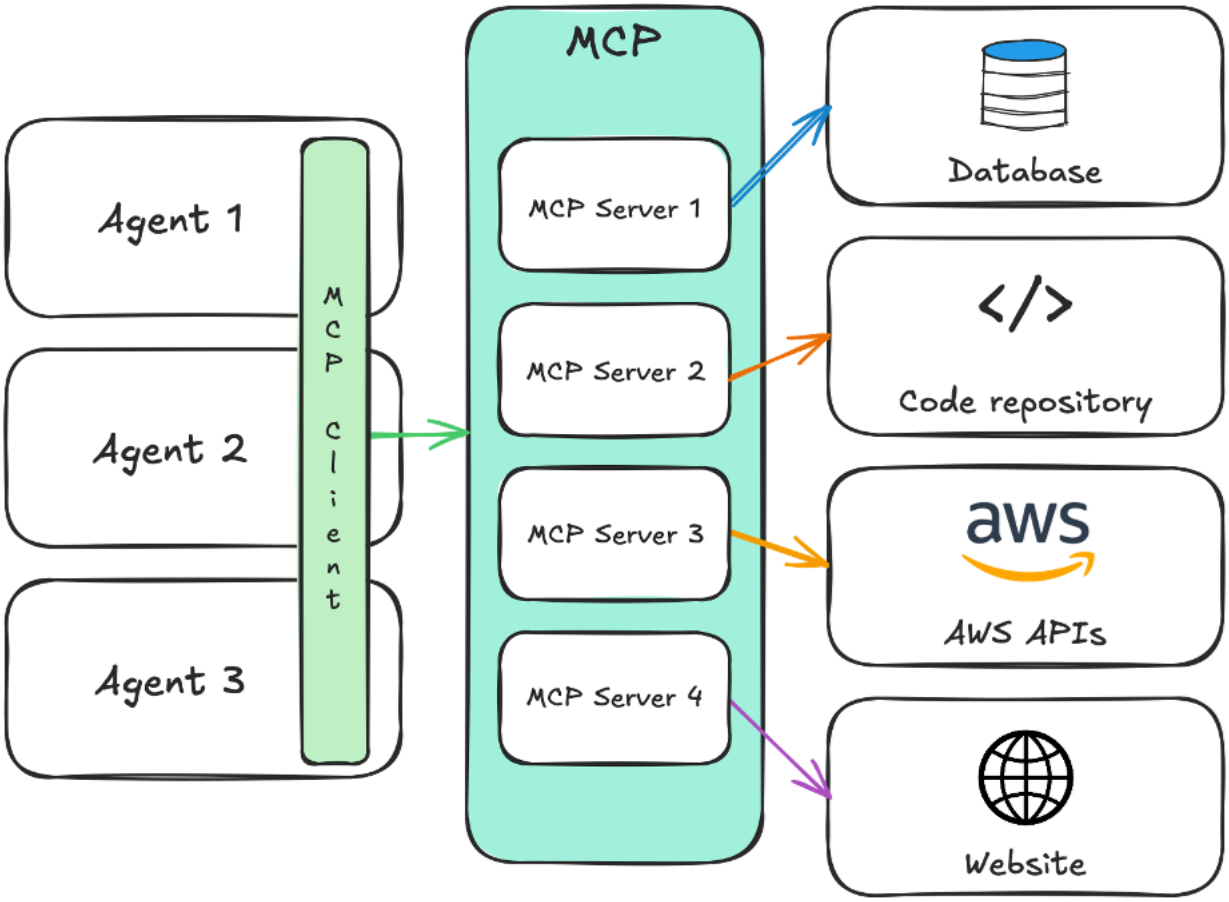
- Se ogni sviluppatore crea il proprio strumento per le stesse funzionalità esterne, si assiste a molti sforzi duplicati e a modi non standardizzati di interagire con queste funzionalità esterne. Ciò produce implementazioni incoerenti tra i vostri agenti. Sebbene sia possibile risolvere il problema sviluppando strumenti standard nelle librerie e distribuendoli, ciò manca di una governance centralizzata. Ciò rende difficile applicare le politiche di sicurezza, tenere traccia dell'utilizzo degli strumenti, gestire il controllo delle versioni tra i team o garantire la conformità agli standard organizzativi. Inoltre, quando incorpori gli strumenti direttamente con l'agente, devi ridistribuirlo ogni volta che viene creato un nuovo strumento o ne viene aggiornato uno esistente.
- La fornitura di strumenti a un LLM utilizza la relativa finestra contestuale. La finestra contestuale è il numero di token (unità di testo elaborate dai LLM, che in genere rappresentano parole, parti di parole o punteggiatura) che un modello può prendere in considerazione in qualsiasi momento. I LLM hanno dei limiti relativi alla finestra contestuale. Gli strumenti e la relativa documentazione utilizzano quella finestra a contesto limitato insieme ai prompt di sistema e ai prompt degli utenti. Man mano che la finestra contestuale si riempie, i LLM possono subire un peggioramento delle

prestazioni a causa di molteplici fattori: difficoltà nell'identificare le informazioni pertinenti, maggiore complessità di elaborazione e ridotta capacità di ragionamento. La sfida è aggravata quando le definizioni degli strumenti, i prompt di sistema e la cronologia delle conversazioni competono per lo spazio limitato nella finestra contestuale, poiché vengono forniti in ogni chiamata LLM.

Pertanto, il numero di strumenti e il modo in cui vengono documentati hanno un impatto diretto sulle prestazioni del LLM, in termini di tempi di risposta e precisione.

MCP stabilisce uno standard universale per connettere gli agenti a funzionalità esterne. Viene comunemente definito «USB-C per applicazioni AI». Invece di registrare gli strumenti direttamente con gli agenti, i server MCP fungono da intermediari per ospitare strumenti che vengono scoperti e richiamati nella versione 2.0. JSON-RPC Invece di aggiungere decine o centinaia di strumenti diversi all'agente e mantenerli nel tempo, MCP consente di registrare server MCP che incapsulano gli strumenti a cui l'agente può accedere. Questo approccio standardizza il modo in cui gli strumenti vengono impacchettati, presentati e richiamati. Questo può aiutare ad affrontare le sfide di scalabilità e governance legate all'utilizzo degli strumenti all'interno degli agenti. Inoltre, separa lo sviluppo e le operazioni degli agenti dagli strumenti utilizzati per le funzionalità esterne.

La figura seguente mostra gli agenti che utilizzano MCP per accedere a risorse esterne.



Tuttavia, lo standard MCP non risolve tutte le sfide di scalabilità e governance. L'implementazione dei server MCP deve essere combinata con strategie efficaci di progettazione degli strumenti, hosting e governance aziendale. Questa guida fornisce le migliori pratiche per ogni strategia per aiutarti a creare e utilizzare MCP come parte delle tue soluzioni di intelligenza artificiale agentica.

Quando usare MCP

MCP fornisce un'infrastruttura strategica per scalare le tue iniziative di intelligenza artificiale agentica. Centralizzando la gestione e la governance degli strumenti, i server MCP riducono il costo cumulativo della creazione e del mantenimento di integrazioni personalizzate tra più agenti. Ciò offre rendimenti crescenti man mano che l'ecosistema degli agenti si espande.

È probabile che MCP diventi parte della tua strategia quando:

- È necessaria una governance centralizzata per il modo in cui gli agenti accedono ai sistemi e ai servizi aziendali, come database, API, strumenti interni e integrazioni di terze parti.

- Gli sviluppatori dedicano troppo tempo a scrivere integrazioni personalizzate che non sono coerenti tra le implementazioni.
- Disponi di strumenti duplicati che potrebbero offrire funzionalità comuni.
- Desiderate offrire i vostri strumenti o dati proprietari a consumatori esterni o sistemi di agenti di terze parti attraverso interfacce MCP standardizzate e gestite, sbloccando nuovi flussi di entrate mantenendo al contempo sicurezza e controllo.

Dopo aver deciso che i server MCP faranno parte della vostra strategia, valutate se le implementazioni dei server MCP open source esistenti soddisfano le vostre esigenze, se richiedono miglioramenti o se è necessario creare server personalizzati. Molte implementazioni predefinite di server MCP sono disponibili negli archivi pubblici e coprono funzionalità comuni come l'accesso al file system, la navigazione sul Web, le sandbox di codice, l'accesso al database e le integrazioni API.

In molti casi, i server MCP preesistenti sono sufficienti. Ad esempio, AWS fornisce il Server AWS MCP, un server MCP remoto gestito che fornisce agli assistenti e agli agenti AI un accesso sicuro e autenticato tramite interazioni in linguaggio naturale. Servizi AWS È possibile utilizzarlo Server AWS MCP per eseguire AWS attività complesse e in più fasi combinando accesso in tempo reale alla AWS documentazione, chiamate API sintatticamente corrette e flussi di lavoro predefiniti denominati Agent SOP che seguono le migliori pratiche. AWS AWS li testa continuamente Server AWS MCP per assicurarsi che gli agenti del cliente possano utilizzarli con successo.

Dovresti testare questi server MCP esistenti con i tuoi agenti per determinare se soddisfano i tuoi casi d'uso. Se un agente non riesce a completare i flussi di lavoro, genera risposte errate o non ottimali, non riesce a gestire processi complessi in più fasi o non rispetta importanti best practice o considerazioni sulla sicurezza specifiche del dominio, dovreste prendere in considerazione miglioramenti in diverse dimensioni.

Quando i server MCP esistenti non soddisfano appieno le vostre esigenze e fanno fatica a utilizzare correttamente gli strumenti esistenti o a produrre risposte accurate, prendete in considerazione questi approcci di miglioramento prima di creare server personalizzati:

- Arricchisci il contesto degli agenti: se il tuo agente ha difficoltà a utilizzare in modo corretto o efficiente gli strumenti di un server MCP esistente, valuta la possibilità di integrare tali definizioni degli strumenti con documentazione o esempi aggiuntivi. Questo aiuta a fornire un contesto aggiuntivo all'LLM.

- Aggiungi strumenti complementari: estendi i server MCP esistenti con strumenti che accedono a dati o contesti organizzativi aggiuntivi di cui gli agenti hanno bisogno per completare con successo i flussi di lavoro.
- Migliora le API sottostanti: semplifica le API di servizio per migliorarle riducendo la complessità dei parametri, fornendo messaggi di errore più chiari e offrendo impostazioni predefinite ragionevoli utilizzabili LLM-friendly dagli agenti.

Sebbene l'utilizzo delle implementazioni di server MCP esistenti acceleri lo sviluppo di funzionalità comuni, la creazione di server MCP personalizzati è una necessità quando il caso d'uso richiede funzionalità specializzate. I server MCP personalizzati consentono di incapsulare le competenze del settore, applicare gli standard organizzativi, migliorare l'affidabilità degli agenti per flussi di lavoro complessi e supportare la conformità ai requisiti di sicurezza. Prendi in considerazione la creazione di un server MCP personalizzato nelle seguenti situazioni:

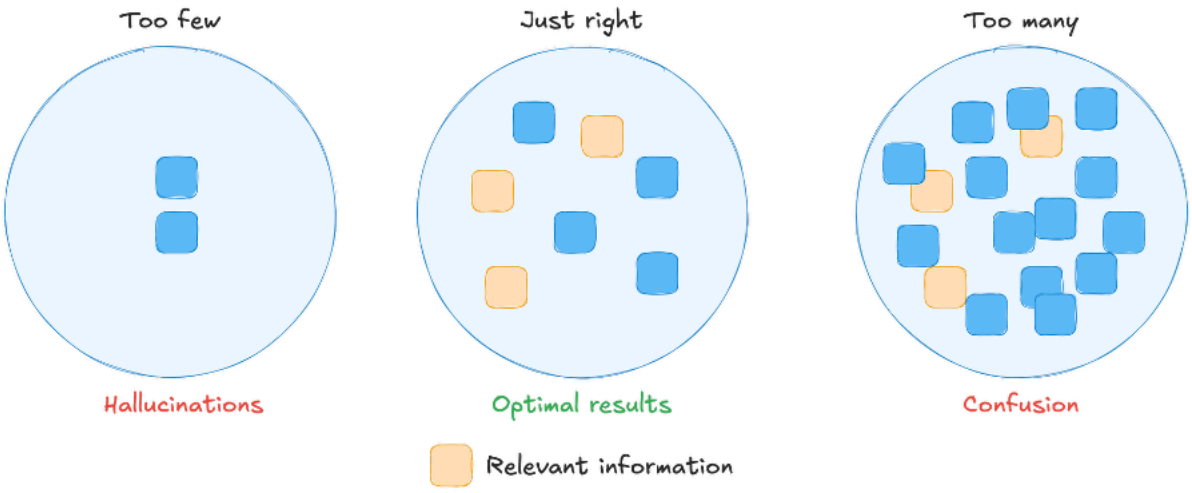
- Domain-specific flussi di lavoro: i Multi-step flussi di lavoro che richiedono competenze di settore devono essere incapsulati in strumenti MCP personalizzati quando le conoscenze necessarie non vengono acquisite nella documentazione delle API. Ad esempio, anziché consentire agli agenti di orchestrare complesse pipeline di dati sanitari che devono convalidare la conformità all'Health Insurance Portability and Accountability Act (HIPAA), rendere anonime le PII e passare al formato HL7 FHIR, offri uno strumento che incorpori direttamente l'esperienza del settore. `process_patient_data` Ciò elimina la dipendenza dall'LLM per orchestrare ed eseguire correttamente le fasi del flusso di lavoro, migliorando la coerenza e la conformità.
- Astrazioni del percorso privilegiato: gli agenti potrebbero avere difficoltà a implementare approcci ottimali perché mancano di contesto organizzativo e si basano su modelli di base piuttosto che sulle migliori pratiche organizzative. In questi scenari, è possibile applicare standard prescrittivi in termini di costi, prestazioni o sicurezza incapsulando questi percorsi fondamentali in strumenti MCP personalizzati. Ad esempio, anziché consentire agli agenti di implementare un'infrastruttura con impostazioni predefinite che potrebbero essere non sicure o inefficienti, fornite uno strumento che incorpori direttamente gli standard della vostra organizzazione. `deploy_secure_infrastructure`
- Orchestrazione multiservizio complessa: anziché far orchestrare all'agente flussi di lavoro complessi cercando di dedurre la sequenza e il set di servizi corretti da utilizzare in ogni fase, puoi creare tale logica in modo deterministico all'interno di uno strumento MCP. Potresti anche voler fornire competenze sui modelli ottimali di integrazione dei servizi di cui l'agente potrebbe non essere a conoscenza. Ciò può anche migliorare la precisione e l'efficienza dei tuoi agenti.

- Service-specific best practice: ciò è comune per gli strumenti incentrati sulla sicurezza che aiutano gli agenti a implementare politiche di crittografia, controlli di accesso e modelli di conformità specifici per il servizio a cui si accede tramite lo strumento dell'agente. Inoltre, se esistono best practice operative specifiche per un servizio che non sono ovvie, l'utilizzo di un server MCP può aiutarvi ad assicurarvi che vengano implementate e non lasciate che sia un agente a ragionare.

Strategia di progettazione degli strumenti MCP

Il compito principale del client e del server MCP è scoprire e presentare gli strumenti all'LLM in modo che possa utilizzarli per migliorare le sue risposte. Ciò rende la progettazione degli strumenti MCP una delle strategie più importanti per creare soluzioni MCP efficaci. Dal punto di vista del modello, gli strumenti sono una funzione che possono richiamare secondo necessità per fornire risposte più accurate e complete. L'interfaccia funzionale riassume l'implementazione sottostante di uno strumento, che può spaziare da un wrapper per una singola chiamata API a una logica di flusso di lavoro complessa.

Tuttavia, è necessario trovare un equilibrio con la quantità di strumenti forniti al LLM. Se gli strumenti sono troppo pochi, l'LLM potrebbe non essere in grado di raccogliere il contesto e le informazioni corretti, quindi farà l'ipotesi migliore utilizzando le informazioni disponibili all'interno del modello. Se gli strumenti sono troppi, l'LLM potrebbe confondersi sulla scelta e sulla sequenza corrette degli strumenti, con conseguenti allucinazioni. Il tuo obiettivo è ottenere il numero giusto di strumenti. L'immagine seguente mostra le problematiche legate al numero insufficiente e al numero eccessivo di strumenti.



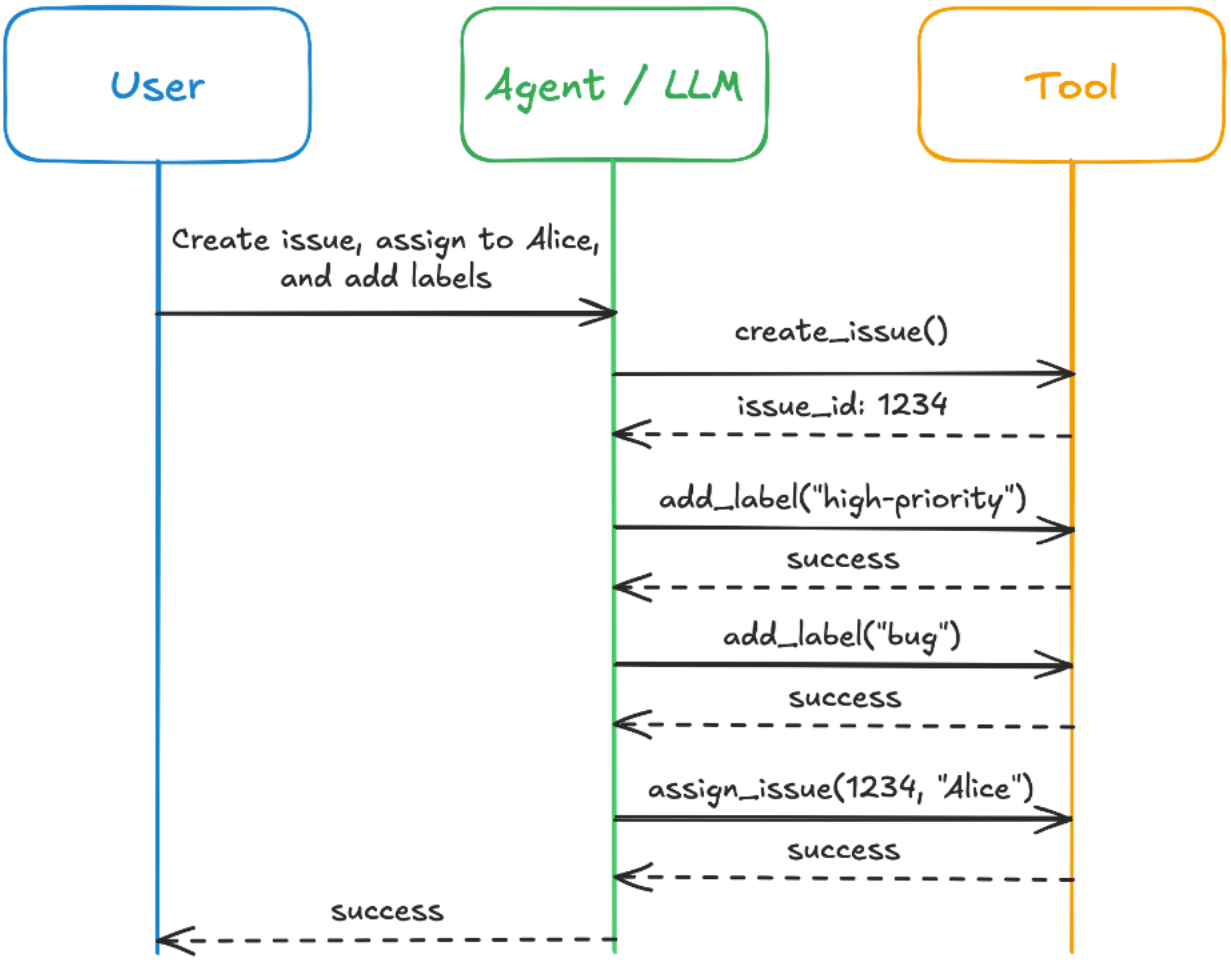
La soluzione richiede la comprensione del numero di strumenti da fornire e dell'ambito di ciascun strumento. La granularità degli strumenti, indipendentemente dal fatto che si riferiscano a singole chiamate API o a flussi di lavoro completi, influisce direttamente sul numero totale di strumenti di cui gli agenti hanno bisogno e sull'efficacia con cui possono utilizzarli. Questa sezione fornisce le migliori pratiche per definire gli strumenti MCP, creare definizioni degli strumenti, scoprirli e organizzarli.

Ambito dello strumento

Esistono due approcci per lo sviluppo di strumenti: granulari e a grana grossa.

Granulare

Con un approccio granulare, creeresti uno strumento per API, azione o query. Ad esempio, puoi crearecreate_issue, get_issue add_labelassign_issue, e close_issue strumenti per il tuo repository Git. Ciò consentirebbe all'LLM di effettuare chiamate granulari a ciascuna API e di orchestrarle ciascuna secondo necessità. Considerate la seguente richiesta: «Create un problema per il servizio di prodotto chiamato 'Query restituisce solo risultati parziali', etichettatelo come bug e con priorità alta e assegnatelo ad Alice». L'immagine seguente mostra come un approccio Tool-per-API risponderebbe a questa richiesta.

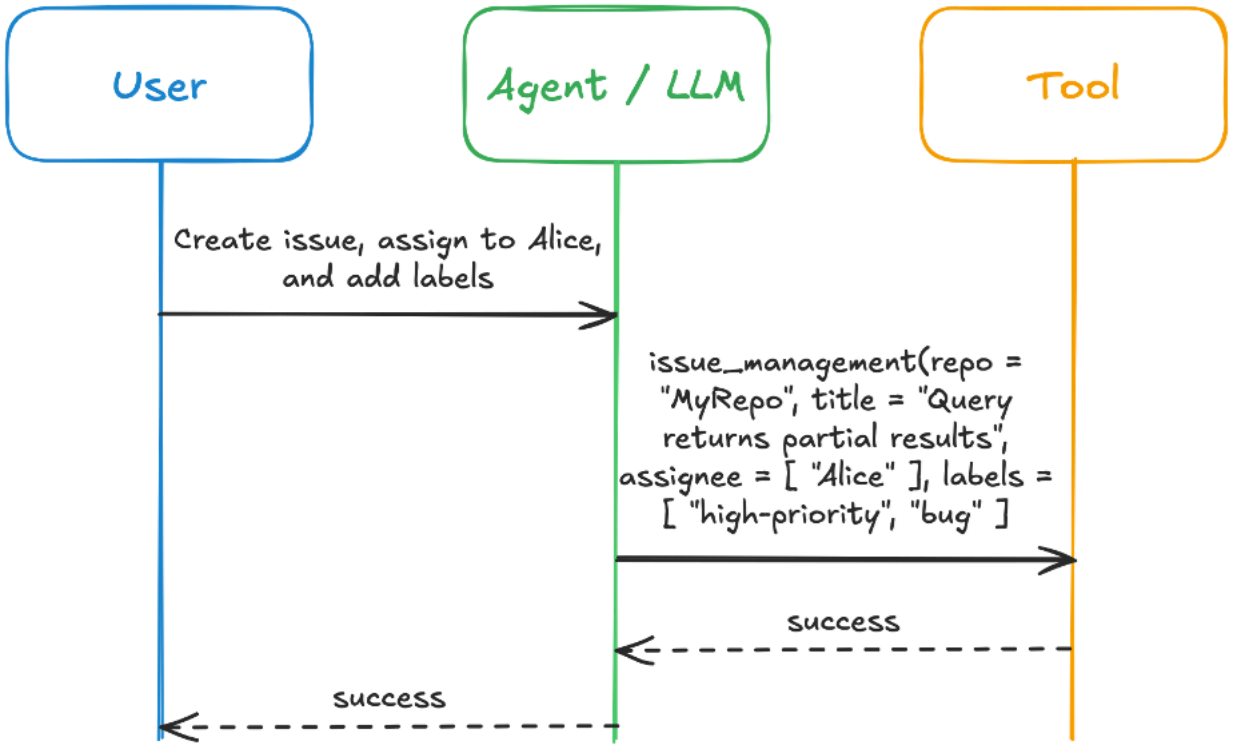


In questo approccio, il prompt di sistema e ogni definizione di strumento registrata vengono fornite all'LLM a ogni chiamata. Ciò consuma ulteriore contesto e comporta una penalità di latenza

perché ogni chiamata allo strumento rappresenta una singola chiamata al LLM. Inoltre, aumenta la complessità della gestione degli errori all'interno del flusso di lavoro.

Coarse-grained

Un approccio generico o basato sul flusso di lavoro sarebbe costituito da strumenti orientati al flusso di lavoro. Lo strumento si concentra sull'intento dell'utente dall'inizio alla fine rispetto alla struttura delle API. Invece di uno strumento per API, hai uno strumento che chiama in modo deterministico molte API. Utilizzando il precedente esempio di repository Git, è possibile creare uno `create_and_setup_issue` strumento che viene chiamato una sola volta dall'agente. L'implementazione dello strumento crea il problema, aggiunge etichette e lo assegna a un utente, in base ai parametri forniti allo strumento. L'immagine seguente mostra come un approccio a grana grossa elaborerebbe lo stesso prompt.



Questo approccio mostra come tutta la complessità rimanga nascosta al livello LLM. Quando la logica di orchestrazione è incorporata nell'implementazione dello strumento, tutti i passaggi sequenziali, la registrazione, la logica dei tentativi, gli interruttori automatici e la limitazione della velocità vengono eseguiti in modo deterministico nello strumento. L'approccio basato sul flusso di lavoro rende più semplice per l'LLM richiamare lo strumento corretto con i parametri corretti. È importante notare che alcune API potrebbero già fornire l'intento del flusso di lavoro, come l'API Amazon `RunInstances`

EC2. In questi casi, uno strumento per API potrebbe fornire il design orientato al flusso di lavoro che desideri.

Tuttavia, anche gli strumenti possono avere una grana troppo grossa. Se un unico strumento di flusso di lavoro tenta di fare troppe cose e ha molti parametri possibili, l'LLM può avere difficoltà a ragionare su come utilizzare correttamente lo strumento. Può anche creare problemi nella selezione dei parametri e nella gestione degli errori. Pertanto, lo sviluppo degli strumenti deve raggiungere un equilibrio che sia in linea con le intenzioni dell'utente ed evitare funzionalità insufficienti o eccessive in un singolo strumento. Ti consigliamo di progettare strumenti basati su flussi di lavoro utente completi, raggruppando le operazioni che di solito avvengono insieme (come tre o più chiamate API). Ti consigliamo inoltre di scomporre gli strumenti che superano otto o più parametri o di gestire più intenti utente distinti. Esegui il test con istruzioni reali per verificare che gli agenti possano utilizzare correttamente ogni strumento.

Se disponi di flussi di lavoro complessi e dinamici che non possono essere facilmente incapsulati come strumento deterministico, potresti prendere in considerazione l'utilizzo del modello agent-as-tool. Invece che l'agente principale cerchi di orchestrare attività complesse in un flusso di lavoro, un agente specializzato può fungere da strumento. Questi tipi di strumenti possono implementare processi decisionali e ramificazioni avanzati, gestire errori e riprovare logiche che non possono essere gestite facilmente nel codice deterministico. È simile, ma distinto dal protocollo Agent2Agent (A2A). Il protocollo A2A è complementare e fornisce interoperabilità e collaborazione tra agenti in qualsiasi framework agentico.

Ti consigliamo di iniziare con l'analisi del flusso di lavoro mappando i flussi di lavoro degli utenti più comuni per identificare le funzionalità principali di cui ogni agente ha bisogno. Questo stabilisce il set di strumenti minimo utilizzabile. Sulla base della nostra esperienza nello sviluppo di server MCP su larga scala, consigliamo le seguenti pratiche. Quando queste pratiche sono in conflitto, dai la priorità all'intento e al flusso di lavoro dell'utente.

Le migliori pratiche per la definizione degli strumenti MCP

- Pensate alle storie degli utenti e raggruppate le operazioni comuni: gli strumenti devono essere mappati direttamente per completare le interazioni degli utenti anziché richiedere l'orchestrazione di più operazioni. Se i flussi di lavoro richiedono in genere tre o più chiamate separate, combinale in un unico strumento. Ciò riduce il carico cognitivo sull'LLM, minimizza il numero di chiamate agli strumenti, riduce il consumo di contesto e la latenza necessari per completare le attività e migliora la precisione e la latenza.

- Limita i parametri a otto o meno: se uno strumento supera gli otto parametri, scomponilo in più strumenti. I LLM hanno difficoltà a selezionare i parametri man mano che la complessità aumenta.

> **Note**
>
> Se le operazioni di raggruppamento richiedono più di otto parametri, dai la priorità al raggruppamento rispetto al conteggio dei parametri, perché la semplificazione del flusso di lavoro è più importante dei rigidi limiti dei parametri.

- Operazioni di lettura e scrittura separate: fornisce diversi strumenti per leggere i dati e modificarli. Questa separazione rende esplicito quando gli agenti eseguono operazioni potenzialmente distruttive, abilita politiche di autorizzazione diverse e riduce il rischio di modifiche involontarie durante la raccolta delle informazioni.
- Fornisci impostazioni predefinite ragionevoli: progetta strumenti in modo che l'LLM debba specificare solo i parametri specifici della singola richiesta. Le impostazioni predefinite riducono la complessità dei parametri e migliorano la precisione della selezione degli strumenti riducendo al minimo le informazioni su cui l'LLM deve ragionare.
- Preferisci l'esecuzione deterministica: rendi deterministici l'esecuzione degli utensili e l'output quando possibile. Gli strumenti deterministici sono più affidabili e facili da testare. Per flussi di lavoro complessi che richiedono un'orchestrazione intelligente, una logica di ramificazione o una gestione avanzata degli errori che non possono essere facilmente gestite nel codice deterministico, prendi in considerazione l'utilizzo di agenti specializzati come strumenti. Tuttavia, utilizzate questo modello in modo selettivo perché aggiunge complessità.

Definizioni degli strumenti

Quando un LLM riceve una richiesta che non può gestire direttamente, esaminerà gli strumenti disponibili per aiutarlo a completare la richiesta. L'LLM seleziona gli strumenti in base alla sua comprensione semantica dei nomi e delle descrizioni degli strumenti forniti e delle eventuali istruzioni fornite nel prompt. Quindi creerà un input basato sullo schema di input definito e si aspetterà un output basato sullo schema di output. Pertanto, la creazione di definizioni descrittive degli strumenti e schemi di input e output convalidati è fondamentale per aiutare l'LLM a selezionare gli strumenti in modo efficace. Esistono generalmente due approcci per creare questa documentazione: l'approccio alla specificazione degli strumenti e l'approccio docstring.

Approccio alla specifica degli strumenti

L'approccio consigliato consiste nel seguire direttamente le specifiche dell'utensile MCP durante la definizione dell'utensile. L'esempio seguente viene mostrato utilizzando il decoratore di strumenti Strands Agent:

```
@tool(
  name = "search_website",
  description = "This tool searches the provided website for semantic matches to the
 query provided",
  inputSchema = {
    "json": {
      "type": "object",
      "properties": {
        "url": {
          "type": "string",
          "description": "The url of the website to load and search."
        },
        "query": {
          "type": "string",
          "description": "The content you want to try and match in the website."
      }
    },
    "required": ["url", "query"]
  },
  outputSchema = {
    "json": {
      "type": "object",
      "properties": {
      "results": {
          "type": "array",
          "items": {
            "type": "string"
          }
        }
      }
    }
  }
)
def search_website:
  …
```

Utilizza campi standard, come, `name` `description`, `inputSchema`, e `outputSchema` assicura che ogni strumento abbia una documentazione coerente che sia l'LLM che gli umani possano comprendere. Ogni strumento dovrebbe definire questi campi come minimo e, facoltativamente, fornire un titolo e delle annotazioni, che sono suggerimenti opzionali sul comportamento dello strumento. Quando possibile, utilizzate le enumerazioni per i valori dei parametri per facilitare la selezione delle opzioni corrette da parte dell'LLM. Le enumerazioni funzionano meglio per insiemi finiti, come valori di status o priorità, ma non sono adatte per testo in formato libero, valori dinamici, numeri arbitrari o identificatori di risorse. In questi casi, fornisci invece descrizioni ed esempi chiari. Includi anche un valore predefinito, quando possibile, in modo che LLM non debba indovinare quale sia l'opzione corretta. Tieni presente che le definizioni degli strumenti sono incluse nel prompt LLM di ogni chiamata e occupano spazio nella finestra contestuale insieme alle istruzioni di sistema e alla cronologia delle conversazioni.

Approccio Docstring

Un altro approccio, se stai scrivendo i tuoi strumenti in Python, consiste nell'usare le docstring per fornire la descrizione, l'utilizzo e l'output dello strumento. Di seguito è riportato un esempio di questo approccio:

```
def search_website(url: str, query: str) -> list:

    """
    This tool loads the specified website and then attempts to find content that
 matches the provided query through semantic search. It provides back a list of strings
 that are the sentences that match the query.
    Args:
      url: the website url to load
      query: the content you want to semantically match in the website
    """
```

Le docstring non applicano uno schema o un formato standardizzato. L'utilizzo di questo approccio potrebbe produrre risultati incoerenti in base al modo in cui gli sviluppatori di strumenti scelgono di documentare ogni strumento. La definizione e l'applicazione di uno standard a livello di organizzazione sono essenziali se si segue questo approccio.

Le migliori pratiche per le definizioni degli strumenti MCP

- Segui le specifiche dello strumento MCP: fornisci`name`, `description`, `inputSchema`, e `outputSchema` campi per ogni strumento. Per le implementazioni Python, usa i [modelli Pydantic]

per fornire documentazione in linea tramite descrizioni dei campi, convalida automatica dei tipi e valori vincolati tramite enumerazioni. Ciò rende gli schemi autodocumentabili e migliora la comprensione LLM delle opzioni di parametro valide.

- Scrivi le descrizioni come istruzioni: le descrizioni degli strumenti sono istruzioni che guidano il processo decisionale LLM. Includi i componenti essenziali dello scopo dello strumento (cosa fa lo strumento), quando utilizzarlo (modelli o scenari delle intenzioni dell'utente), il contesto dell'output (a cosa serve l'output), i parametri e le condizioni di errore.
- Fornisci esempi concreti: includere esempi di flusso di lavoro con valori effettivi è il modo più efficace per guidare i LLM nell'utilizzo corretto degli strumenti.
- Documenta le dipendenze in modo esplicito: includi prerequisiti, sequenze numerate, modifiche di stato e azioni successive.

Scoperta degli strumenti

Esistono tre approcci per scoprire e registrare gli strumenti nell'agente con i server MCP: definizione statica, rilevamento dinamico e funzione di ricerca.

Definizione statica

Innanzitutto, è possibile definire staticamente gli strumenti disponibili direttamente nel codice dell'agente. In questo approccio si definisce uno strumento remoto (un oggetto di riferimento sul lato client in un framework come Strands Agent SDK) per ogni strumento fornito dal server MCP a cui accede un client MCP. L'esempio seguente utilizza un trasporto HTTP ottimizzato:

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

reverse_text = RemoteTool(
    name="reverseText",
    client=streamable_http_mcp_client
)

agent = Agent(tools=[reverse_text])
```

La registrazione individuale degli strumenti ti aiuta a essere molto selettivo riguardo agli strumenti che metti a disposizione del LLM, il che riduce al minimo la quantità di finestra contestuale utilizzata. Il compromesso è che richiede la conoscenza dei nomi degli strumenti disponibili e può essere fragile se gli strumenti disponibili cambiano nel server MCP.

Scoperta dinamica

L'approccio successivo prevede l'utilizzo del rilevamento dinamico e la registrazione di tutti gli strumenti disponibili con l'agente. Questo approccio utilizza il contesto in modo lineare man mano che vengono aggiunti altri strumenti al server MCP. Di seguito è riportato un esempio di questo approccio:

```
from mcp.client.streamable_http import streamablehttp_client
from strands import Agent
from strands.tools.mcp import MCPClient

streamable_http_mcp_client = MCPClient(
    lambda: streamablehttp_client("https://mcp1:8000/mcp")
)

with streamable_http_mcp_client:
    tools = streamable_http_mcp_client.list_tools_sync()
    agent = Agent(tools=tools)
```

Consideriamo uno scenario in cui una tipica definizione di utensile utilizza circa 250-500 token (inclusi nome, descrizione e schema). La registrazione di 20 strumenti consumerebbe da 5.000 a 10.000 token della finestra contestuale. Quando si dispone di un numero limitato di server MCP e si ha il controllo sul numero di strumenti, questa opzione è la più semplice da implementare. Tuttavia, se si prevede che l'elenco degli strumenti aumenti, è possibile che si verifichino problemi di gestione silenziosa del contesto negli agenti. Una variante alternativa di questo approccio consiste nell'utilizzare un parametro di filtro degli strumenti durante la chiamata`list_tools`, come quello fornito dall'SDK di Strands Agents, per ridurre il numero di strumenti registrati con l'agente.

Funzione di ricerca

La terza opzione consiste nell'utilizzare una funzione di ricerca per trovare gli strumenti pertinenti durante l'esecuzione. Elenca tutti gli strumenti disponibili dal server MCP e quindi esegue una ricerca semantica su tali strumenti in base al prompt dell'utente. Quindi, gli strumenti risultanti vengono registrati presso il vostro agente. Amazon Bedrock AgentCore Gateway offre una funzionalità di ricerca semantica nativa che può semplificare l'implementazione di questo tipo di soluzione.

Le migliori pratiche per la scoperta di strumenti MCP

- Conservazione della finestra contestuale: scegli un approccio di scoperta e registrazione degli strumenti che conservi la maggior parte possibile della finestra contestuale.
- Utilizza funzionalità di filtraggio degli strumenti o di ricerca semantica: fornisci dinamicamente al LLM un set di strumenti ristretto tra cui scegliere, il che ne migliora la precisione e l'efficacia nella scelta dello strumento giusto. Il filtraggio degli strumenti può operare sui nomi degli strumenti (corrispondenza o modelli esatti), sulle descrizioni degli strumenti (corrispondenza semantica) o sui tag di dominio o di categoria. La ricerca semantica è particolarmente efficace per abbinare le intenzioni degli utenti alle descrizioni degli strumenti. Entrambi gli approcci riducono l'uso delle finestre contestuali.

Organizzazione degli strumenti

Scoprire gli strumenti giusti e assicurarsi che l'LLM possa utilizzarli in modo efficace è una delle parti più importanti dello sviluppo di strumenti efficaci. Quando si inizia a sviluppare server MCP, è necessaria una strategia che determini:

- Quanti strumenti sono inclusi in un server MCP
- Quali strumenti non devono essere inseriti nello stesso server MCP
- Come assegnare un nome agli strumenti per renderli ricercabili e prevenire le collisioni tra nomi (strumenti diversi con lo stesso nome)
- Come documentare gli strumenti e il server MCP per renderli facili da usare da parte dell'LLM

L'organizzazione dei namespace è un modello di progettazione che impedisce le collisioni tra i nomi degli strumenti, raggruppa le funzionalità correlate e facilita l'identificazione efficiente degli strumenti da parte degli LLM. Il modello stabilisce una categorizzazione strutturata analoga ai sistemi di storage organizzati piuttosto che all'accumulo non strutturato. Consigliamo il pattern domain-noun-verb per la denominazione degli strumenti. Ad esempio,,,,, o. `github_issue_create` `github_issue_list` `github_issue_update` `github_pullrequest_create` `github_pullrequest_list` `github_pullrequest_merge` Il vantaggio di questo modello è evidente quando si esamina il comportamento di ordinamento alfabetico. Quando gli strumenti sono elencati in ordine alfabetico, tutte le operazioni relative ai problemi si raggruppano (`create`,,`update`)`list`, seguite dalle operazioni di pull request (,,). `create` `list` `merge` Il sostantivo (tipo di risorsa) funge da confine

organizzativo. Questa struttura facilita sia la scansione degli strumenti LLM che la navigazione nella documentazione umana perché le funzionalità correlate si raggruppano naturalmente.

Il server MCP deve essere limitato a livello di dominio, ma può essere suddiviso in base alla separazione dei compiti in base alle funzionalità che fornisce. Ad esempio, potreste disporre di server MCP separati per le operazioni di scrittura e le operazioni di lettura su un database. Per applicare questa separazione, si consiglia di implementare dei guardrail a livello di agente che limitino l'accesso ai server MCP in base alle intenzioni e alle autorizzazioni dell'utente. Ciò può essere ottenuto mediante una combinazione dei seguenti elementi:

- Caricamento condizionale del server: carica il server MCP di sola lettura solo quando l'agente rileva operazioni di lettura nell'input dell'utente.
- Permission-based filtraggio: utilizza l'autorizzazione dell'utente per concedere l'accesso solo ai server MCP appropriati.

Infine, è necessario creare un limite superiore al numero di strumenti forniti da un server MCP. Non fate supposizioni su come gli agenti utilizzeranno il vostro server MCP. Potrebbero elencare ingenuamente tutti gli strumenti disponibili e fornirli tutti all'LLM. Se hai più di 50 strumenti in un singolo server, dovresti prendere in considerazione la possibilità di suddividerlo in più server.

Le migliori pratiche per l'organizzazione degli strumenti MPC

- Utilizza lo standard di denominazione dominio-sostantivo-verbo per gli strumenti: implementa strategie per prevenire le collisioni di nomi sia nei server MCP che negli agenti.
- Imposta un limite superiore: limita il numero di strumenti in un singolo server MCP.
- Dividi i server MCP: utilizza la separazione dei compiti per dividere i server MCP in gruppi logici.

Strategia di hosting MCP

L'astrazione degli strumenti disponibili nei server MCP separa lo sviluppo degli agenti dagli strumenti disponibili. Questo introduce le sfide legate al luogo in cui si ospita il server MCP e al modo in cui gli strumenti sono organizzati all'interno di tali server.

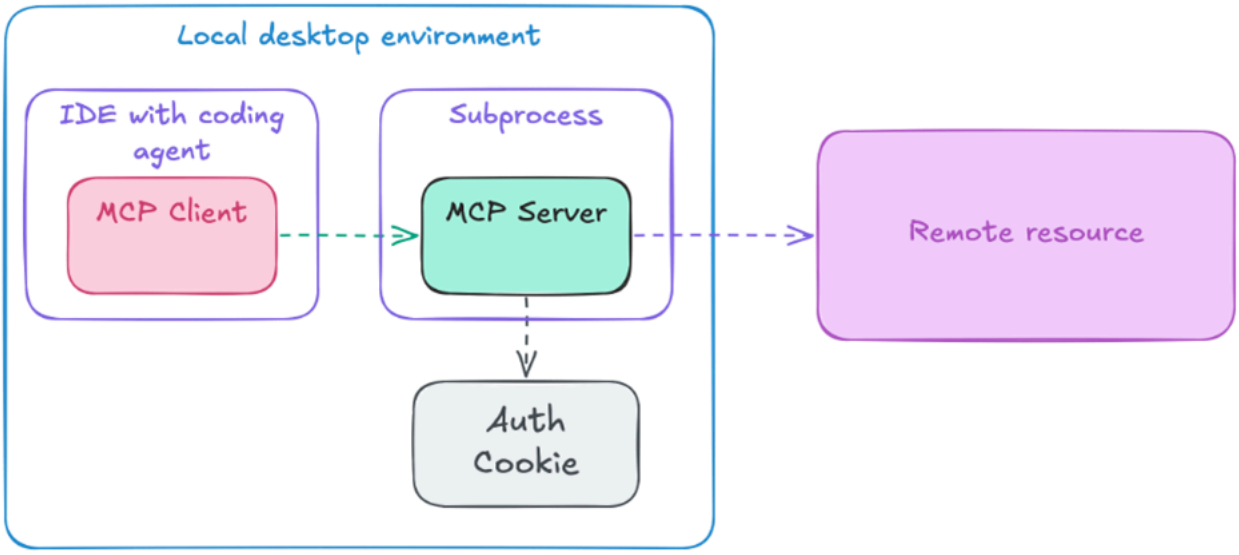
Approcci di hosting

Esistono tre opzioni per ospitare i server MCP: eseguirli localmente su un computer dell'utente finale, ospitarli in remoto o ospitarli tramite un gateway MCP. Ogni opzione presenta vantaggi e compromessi.

Hosting locale

L'hosting locale esegue il server MCP come sottoprocesso sul computer locale insieme all'agente che comunica con il server utilizzando JSON-RPC flussi di input e output standard. Questo approccio non richiede l'autenticazione tra il client e il server. Gli strumenti possono interagire con applicazioni e file locali, utilizzare credenziali archiviate localmente ed ereditare l'accesso alla rete del computer locale dell'utente. Questo è il modello di hosting più semplice e presenta diversi vantaggi.

Molti clienti iniziano a usare MCP utilizzando server locali. Consentono agli ingegneri di iterare e risolvere rapidamente una serie di problemi dal loro ambiente locale. Prendiamo in considerazione un server MCP che si connette a un repository Git utilizzato dall'assistente di programmazione di un ingegnere. Mantenere il server MCP locale ha molto senso perché può utilizzare le credenziali univoche del tecnico per accedere al repository e non aggiunge un'ulteriore chiamata di rete a un server MCP remoto. L'immagine seguente mostra un server MCP ospitato localmente utilizzato con un agente di codifica in un IDE.



Per questi tipi di implementazioni, è necessario considerare come vengono sviluppati e distribuiti i server MCP. La maggior parte dei clienti sviluppa un registro MCP in cui i server possono essere registrati e scaricati dagli utenti finali. È molto simile a un registro di contenitori in cui un utente può cercare funzionalità specifiche e trovare i server MCP adatti alle proprie esigenze.

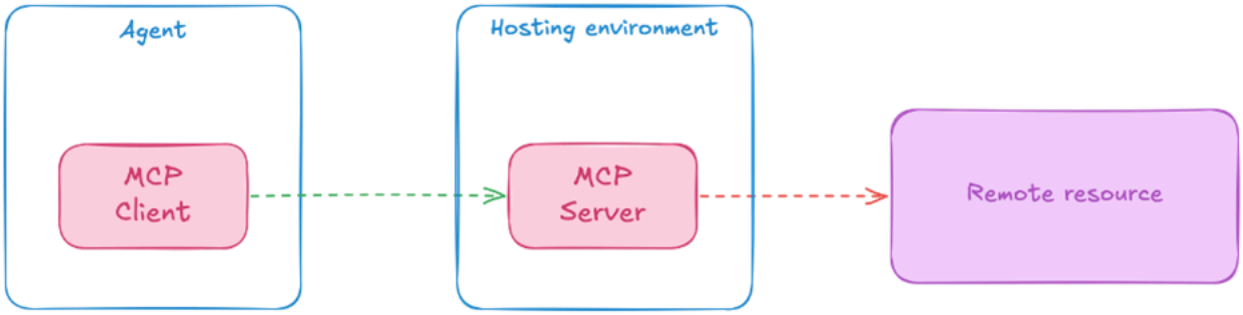
Esistono registri MCP pubblici, come il registro MCP ufficiale, e registri ospitati privatamente. Le organizzazioni in genere allineano la propria strategia di registro MCP alle politiche esistenti relative alla distribuzione del software open source, ai registri dei container e alla gestione interna dei pacchetti. È necessario prendere in considerazione fattori come la scansione di sicurezza, i flussi di lavoro di approvazione e i requisiti di conformità.

Tuttavia, l'hosting locale introduce sfide operative che le organizzazioni dovrebbero prendere in considerazione. Innanzitutto, gli utenti finali devono scoprire, scaricare e configurare i server MCP in modo indipendente. Ciò può aumentare la complessità necessaria per iniziare a utilizzare ogni singolo server MCP che utilizzano localmente. In secondo luogo, non è possibile controllare il ciclo di vita del server MCP, il che significa che gli utenti possono continuare a eseguire versioni obsolete localmente con vulnerabilità di sicurezza o funzionalità mancanti. Ciò può complicare il rispetto dei requisiti di conformità. Alcuni IDE e strumenti CLI, come Kiro, consentono alle organizzazioni di gestire e controllare gli strumenti MCP disponibili, garantendo coerenza e sicurezza tra i team.

Hosting remoto

La seconda opzione consiste nell'ospitare server MCP remoti a cui si accede tramite HTTP o HTTPS. Ciò fornisce l'accesso a qualsiasi client connesso alla rete. L'utilizzo dell'hosting remoto consente di controllare centralmente l'accesso alle risorse e alle funzionalità MCP, implementare l'autenticazione

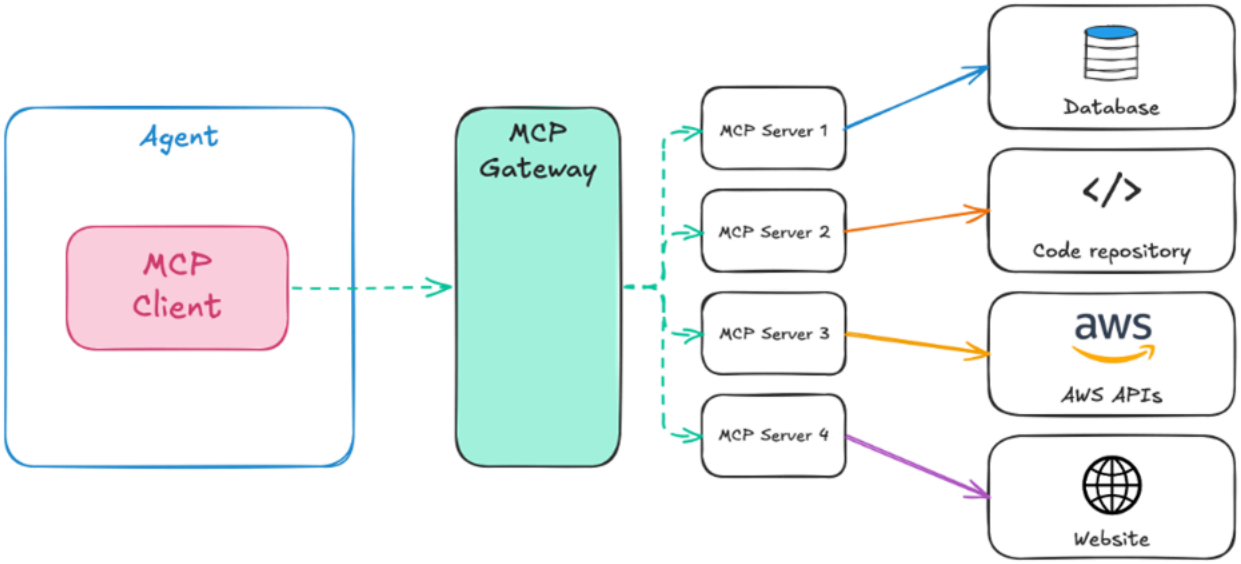
e l'autorizzazione e controllare il controllo delle versioni e gli aggiornamenti della logica del server MCP. L'hosting remoto richiede comunque l'uso di un registro MCP in modo che gli utenti finali possano scoprire i server MCP che desiderano utilizzare con il proprio agente. L'immagine seguente mostra l'approccio all'hosting remoto.



Dal punto di vista dello sviluppo degli agenti, l'esperienza è simile indipendentemente dal fatto che il server MCP sia locale o remoto. La modifica più importante riguarda l'implementazione dell'autenticazione e dell'autorizzazione, che includono sia l'accesso dell'agente al server MCP sia l'accesso del server alle risorse esterne. Le implementazioni dei server MCP remoti devono essere pianificate attentamente per prendere in considerazione l'accesso multi-tenant e la gestione dei privilegi. Il capitolo sulla strategia di governance MCP contiene ulteriori informazioni sulle considerazioni relative all'autenticazione e all'autorizzazione.

Gateway MCP

L'ultima opzione è l'utilizzo di un gateway MCP. I gateway MCP fungono da proxy centralizzato tra client e server MCP e orchestrano l'accesso ai server MCP registrati. Senza un gateway, ogni agente deve registrare tutti i server MCP remoti che potrebbe voler utilizzare. Un gateway consente all'agente di connettersi a un singolo endpoint che gestisce l'autenticazione, l'autorizzazione, il routing e la traduzione del protocollo. È possibile aggiungere dinamicamente nuovi server e strumenti MCP e renderli immediatamente disponibili all'agente. L'immagine seguente mostra l'approccio del gateway MCP.



Alcune soluzioni gateway, come Docker MCP Gateway, gestiscono anche il ciclo di vita dei server MCP, avviando i server su richiesta in base alle esigenze. I gateway MCP, come Amazon Bedrock AgentCore Gateway, possono anche aiutare a gestire l'individuazione degli strumenti fornendo funzionalità di ricerca semantica native. Ciò fornisce agli agenti un unico endpoint per connettersi con un client MCP e aiuta a ottimizzare l'utilizzo della finestra contestuale. Il risultato sono agenti semplici in grado di scegliere e utilizzare gli strumenti MCP in modo efficace. Tuttavia, presenta sfide legate all'identità simili a quelle dell'approccio al server MCP remoto.

Le migliori pratiche per l'hosting di server MCP

- La gamma di opzioni di hosting non è valida per tutti. Gran parte dell'utilizzo dei server MCP oggi è locale.
- Quando iniziate a utilizzare server MCP remoti, la vostra considerazione principale è l'autenticazione e l'autorizzazione coerenti verso il server MCP e il modo in cui il server MCP esegue l'autenticazione e l'autorizzazione sulle risorse a valle.
- I gateway MCP semplificano la connettività, l'autenticazione e l'autorizzazione per l'hosting di più server MCP remoti. Forniscono inoltre funzionalità per migliorare la gestione delle finestre contestuali mediante la ricerca di strumenti applicabili.

Strategia di governance MCP

L'altra funzionalità fondamentale che MCP offre alle organizzazioni è il supporto per la governance centralizzata. La vostra strategia di governance MCP dovrebbe riguardare l'autenticazione e l'autorizzazione sia per i server MCP che per le risorse a cui accedono. Dovrebbe inoltre riguardare la limitazione della velocità per proteggere le risorse a valle, le metriche operative per il monitoraggio dell'utilizzo e delle prestazioni degli strumenti e la gestione delle implementazioni e della distribuzione dei server MCP.

Autenticazione e autorizzazione

Una delle parti più importanti della strategia di autenticazione e autorizzazione è la gestione dell'accesso alle risorse a valle dai server MCP. Quando un utente chiama un agente, vengono eseguite l'autenticazione e l'autorizzazione per garantire che l'utente disponga delle autorizzazioni per chiamare l'agente. Quindi, l'agente orchestra la chiamata di strumenti specifici nei server MCP. È necessario decidere come autorizzare l'accesso in base allo strumento.

Un'opzione è l'machine-to-machine autorizzazione, in cui non è richiesto il consenso o l'interazione dell'utente. Ad esempio, una chiamata di un agente basata sul tempo utilizza un server MCP per raccogliere i log da un'applicazione e analizzarli. In questo scenario, l'agente è preautorizzato ad accedere ai dati specificati. La seconda opzione è l'accesso delegato dall'utente, in cui un utente fornisce il proprio consenso all'accesso a dati e risorse specifici dell'utente.

La tabella seguente mostra i modelli di autenticazione e autorizzazione.

Fattore	Accesso delegato dall'utente	Machine-to-machine
Proprietà dei dati	Autorizzazione specifica dell'utente ai dati	Dati a livello di sistema o organizzazione
Interazione con l'utente	L'utente è presente e può acconsentire	Nessuna interazione con l'utente
Tempistica delle operazioni	Interattivo o in tempo reale	In background, pianificato o in batch
Ambito dell'autorizzazione	Le autorizzazioni variano in base all'utente	Autorizzazioni coerenti a livello di agente

L'accesso delegato dall'utente richiede un'implementazione attenta e deve essere sviluppato con il team di sicurezza. Gli agenti devono essere in grado di valutare quali strumenti ha selezionato un LLM e se richiedono un'autorizzazione aggiuntiva. Gli strumenti MCP devono includere descrizioni per indicare i requisiti di autenticazione e autorizzazione e dove recuperare i token di accesso. Le applicazioni client devono supportare richieste di autenticazione intermedie e il client MCP deve fornire le credenziali recuperate all'agente per ogni chiamata allo strumento.

È necessario assicurarsi che gli strumenti MCP dispongano sempre dei propri token per accedere alle funzionalità esterne e che l'accesso sia registrato e verificato. Le credenziali utente non devono essere propagate attraverso il sistema agentic. Ad esempio, i server MCP non devono utilizzare lo stesso token per accedere ai dati utilizzati per richiamare l'agente. Le chiamate downstream devono utilizzare token con ambito esplicito e generati appositamente. Questo aiuta a fornire barriere aggiuntive per impedire l'accesso involontario ai dati per conto di azioni. Può anche aiutare a prevenire che le allucinazioni producano risultati indesiderati. Immaginate che un utente con autorizzazioni amministrative complete chieda a un agente di clonare un database di produzione da utilizzare in fase di pre-produzione. A tal fine, l'utente necessita solo delle autorizzazioni e dei permessi necessari`READ`. `CREATE` Supponiamo che l'LLM abbia allucinazioni e creda di dover ripulire il vecchio database come parte di questa richiesta. Se riutilizza le credenziali dell'utente, probabilmente avrà successo perché le credenziali originali dell'utente dispongono delle autorizzazioni. `DELETE` Invece, se il server MCP utilizza un token intenzionalmente ridotto per la richiesta con solo `READ` e `CREATE` autorizzazioni, il tentativo di eliminare il database di produzione fallirebbe.

Puoi utilizzare Amazon Bedrock AgentCore Identity per contribuire all'implementazione di questi modelli. Assicurati di scegliere intenzionalmente se le autorizzazioni per elencare e richiamare gli strumenti ospitati da un server MCP implichino l'autorizzazione alle funzionalità esterne esposte dal server MCP. Questo flusso di identità dal server MCP alla risorsa e viceversa all'utente dipende dal tipo di servizio di autenticazione e autorizzazione utilizzato. È necessario decidere come gestirlo su larga scala per i server MCP.

Durante la progettazione dei modelli di autenticazione e autorizzazione, implementate meccanismi di isolamento dei token che recuperino token di accesso diversi per ogni strumento a cui si accede. Non riutilizzate i token tra strumenti e server. AgentCore L'identità fornisce questa funzionalità di isolamento dei token. Gestisce automaticamente sia i token del carico di lavoro (per machine-to-machine l'autenticazione) che i token utente (per l'accesso delegato dall'utente) per garantire una separazione adeguata e prevenire l'aumento delle autorizzazioni. Ciò è particolarmente importante quando si incorporano server MCP remoti o gateway MCP.

Le migliori pratiche per l'autenticazione e l'autorizzazione MCP

- Separazione dei token: non trasferite i token portatori dai chiamanti ai servizi a valle. Convalida che il campo aud (audience) corrisponda al server che riceve il token. L'indicazione audience specifica a quale servizio è destinato il token, impedendo il riutilizzo non autorizzato del token su diversi server MCP.
- Seleziona un approccio di accesso: scegli tra machine-to-machine l'accesso delegato dall'utente per ogni strumento fornito dai server MCP. Prendi in considerazione la possibilità di raggruppare gli strumenti nello stesso server MCP che utilizzano lo stesso modello di autenticazione.

Controllo del carico

Come con qualsiasi sistema distribuito, è necessario considerare come controllare il carico nel parco server MCP. Innanzitutto, valutate se implementare la limitazione della velocità nei server MCP e dove implementarla. Se scegliete di non implementare la limitazione della velocità, trasferite qualsiasi limitazione di velocità eseguita dalle risorse a valle. Molti sistemi scelgono di limitare la velocità in base agli attributi della richiesta, come l'ID utente o l'account. Verifica che le richieste inviate ai servizi downstream contengano gli stessi attributi in modo che più utenti non siano influenzati dal carico generato da un altro utente.

Se scegli di implementare la limitazione della velocità, l'approccio consigliato consiste nell'implementare la limitazione della velocità principale a livello del server MCP, con servizi di backend che forniscono una protezione secondaria e agenti che adattano il loro comportamento in base al feedback sul limite di velocità. Valuta se i limiti di velocità sono per server MCP o per strumento. I limiti di velocità per server MCP aiutano a proteggere la flotta e i servizi di server MCP in un ambiente multi-tenant. Tuttavia, ciò può essere molto complicato. I limiti di velocità per utensile sono stati progettati per evitare che le risorse a valle vengano sovraccaricate, che potrebbero non essere sufficientemente limitate. Se uno strumento effettua chiamate multiple APIs, è necessario impostare il limite di velocità in modo che corrisponda alla tariffa più bassa consentita da tali strumenti. APIs

L'inserimento delle informazioni sui limiti di velocità nelle intestazioni HTTP può inoltre essere una metrica utile per gli utenti e i sistemi automatizzati per gestire la propria strategia relativa alla frequenza di richieste e ai tentativi. Ad esempio, è possibile inviare queste intestazioni all'agente dal server MCP, come illustrato nell'esempio seguente:

```
X-RateLimit-Limit: 100
```

```
X-RateLimit-Remaining: 45
X-RateLimit-Reset: 1640995200
```

Inoltre, prendete in considerazione la riduzione del carico per proteggere l'intero servizio quando nessun cliente supera un limite di velocità, ma il carico influisce sulle prestazioni del sistema.

Le migliori pratiche per il controllo del carico

- Scegliete un approccio basato sulla limitazione della velocità: pianificate di limitare la velocità dei singoli utenti in base all'uso delle risorse downstream o all'uso del server e degli strumenti MCP.
- Prendi in considerazione la riduzione del carico: proteggi la tua flotta di server MCP da un sovraccarico generale che non sia causato da un singolo o da una manciata di clienti.

Parametri operativi

Le metriche chiave da acquisire per le implementazioni MCP devono concentrarsi sull'esperienza del cliente che offrono. Queste metriche includono in genere l'utilizzo dei token, l'accuratezza della selezione degli strumenti, il numero di strumenti registrati con l'agente e la latenza degli strumenti. Ad esempio, il monitoraggio dei token di output restituiti da ogni strumento consente di impostare allarmi quando gli strumenti superano una soglia per l'utilizzo della finestra contestuale. Quando uno strumento supera tale soglia, potresti voler esaminare il comportamento dello strumento. Ciò si collega anche alla strategia di progettazione degli strumenti MCP. Le metriche di precisione nella selezione degli strumenti indicano in che misura gli agenti scelgono gli strumenti appropriati per determinate attività, mentre la velocità di esecuzione e le percentuali di successo evidenziano problemi di prestazioni e affidabilità.

Ad esempio, per valutare le metriche di selezione e precisione relative all'uso degli strumenti, i team hanno creato set di dati pregiati per i test di regressione. AWS I set di dati sono stati generati sinteticamente utilizzando i log di invocazione delle API storici sulle query degli utenti. LLMs Utilizzando le metriche predefinite di selezione e utilizzo degli strumenti (come la precisione della selezione degli strumenti, l'accuratezza dei parametri degli strumenti e l'accuratezza delle chiamate alle funzioni multigiro), i AWS team potevano valutare oggettivamente la capacità dell'agente AI di identificare correttamente gli strumenti appropriati, compilare i propri parametri con valori accurati e mantenere sequenze di invocazione degli strumenti coerenti durante i turni di conversazione.

La misurazione delle metriche relative al numero di strumenti registrati con un agente può aiutarti a identificare potenziali problemi di gestione delle finestre di contesto, nonché le modifiche agli

strumenti disponibili presentati dai server MCP. È necessario rivedere regolarmente le metriche operative che indicano l'esperienza utente con il server e gli strumenti MCP.

Collaboratori

Creazione

- Alex Torres, architetto senior delle soluzioni, AWS
- Saikat Gomes, responsabile senior delle soluzioni per i clienti, AWS
- Mike Haken, responsabile delle soluzioni senior, AWS
- Sreeja Das, ingegnere principale, AWS

Revisione

- Ted Swinyar, responsabile dell'architettura delle soluzioni, AWS
- Raju Patil, esperto di dati, AWS

Scrittura tecnica

- Lilly AbouHarb, scrittrice tecnica senior, AWS

Cronologia dei documenti

La tabella seguente descrive le modifiche significative apportate a questa guida. Per ricevere notifiche sugli aggiornamenti futuri, puoi abbonarti a un feed RSS.

Modifica	Descrizione	Data
Pubblicazione iniziale	—	16 marzo 2026

Le traduzioni sono generate tramite traduzione automatica. In caso di conflitto tra il contenuto di una traduzione e la versione originale in Inglese, quest'ultima prevarrà.