Grundlagen der Agenten-KI auf AWS

AWS Präskriptive Leitlinien

AWS Präskriptive Leitlinien: Grundlagen der Agenten-KI auf AWS

Table of Contents

Grundlagen der agentischen KI auf AWS

Aaron Sempf, Amazon Web Services

Juli 2025 (Geschichte der Dokumente)

In einer Welt zunehmend intelligenter, verteilter und autonomer Systeme ist das Konzept eines Agenten — einer Entität, die ihre Umgebung wahrnehmen, über ihren Zustand nachdenken und bewusst handeln kann — grundlegend geworden. Agenten sind nicht nur Programme, die Anweisungen ausführen. Sie sind zielorientierte, kontextsensitive Einheiten, die Entscheidungen im Namen von Benutzern, Systemen oder Organisationen treffen. Ihre Entstehung spiegelt einen Wandel in der Art und Weise wider, wie man Software entwickelt und über sie denkt: eine Verlagerung von prozeduraler Logik und reaktiver Automatisierung hin zu Systemen, die eigenständig und zielgerichtet arbeiten.

An der Schnittstelle von KI, verteilten Systemen und Softwareentwicklung liegt ein mächtiges Paradigma, das als agentische KI bekannt ist. Diese neue Generation intelligenter Systeme besteht aus Softwareagenten, die zu adaptivem Verhalten, komplexer Koordination und delegierter Entscheidungsfindung fähig sind.

Dieser Leitfaden stellt die Prinzipien vor, die moderne Softwareagenten definieren, und skizziert ihre Entwicklung hin zu agentischer KI. Um diesen Wandel zu erklären, bietet der Leitfaden den konzeptionellen Hintergrund und zeichnet anschließend die Entwicklung von Softwareagenten zur agentischen KI nach:

- In der Einführung in Softwareagenten werden Softwareagenten definiert, sie mit herkömmlichen Softwarekomponenten verglichen und anhand etablierter Frameworks die wesentlichen Merkmale vorgestellt, die das Verhalten von Agenten von der herkömmlichen Automatisierung unterscheiden.
- Der Zweck von Softwareagenten untersucht, warum Softwareagenten existieren, welche Rollen sie erfüllen, welche Probleme sie lösen und wie sie intelligentes Delegieren ermöglichen, die kognitive Belastung reduzieren und adaptives Verhalten in dynamischen Umgebungen unterstützen.
- Die Entwicklung von Softwareagenten zeichnet die intellektuellen und technologischen Meilensteine nach, die Softwareagenten geprägt haben, von frühen Konzepten der Autonomie und Parallelität bis hin zur Entstehung von Systemen mit mehreren Agenten und formalen Agentenarchitekturen, die zur Konvergenz mit generativer KI führten.
- Von Softwareagenten hin zu agentischer KI stellt die agentische KI den Höhepunkt jahrzehntelanger Fortschritte dar, bei denen verteilte Agentenmodelle mit Basismodellen,

serverloser Datenverarbeitung und Orchestrierungsprotokollen kombiniert werden. In diesem Abschnitt wird beschrieben, wie diese Konvergenz eine neue Generation intelligenter, Tools verwendender Agenten ermöglicht, die autonom, asynchron und in großem Maßstab agieren können.

Zielgruppe

Dieser Leitfaden richtet sich an Architekten, Entwickler und Technologieführer, die sich mit der Geschichte, den wichtigsten Konzepten und der Entwicklung von Softwareagenten zur agentischen KI vertraut machen möchten, bevor sie diese Technologie für moderne Cloud-Lösungen einsetzen. AWS

Ziele

Die Einführung agentischer Architekturen hilft Unternehmen:

- Schnellere Amortisierung: Automatisieren und skalieren Sie die Wissensarbeit und reduzieren Sie den manuellen Aufwand und die Latenz.
- Verbessern Sie die Kundenbindung: Stellen Sie intelligente Assistenten in allen Bereichen bereit.
- Senken Sie die Betriebskosten: Automatisieren Sie Entscheidungsabläufe, für die bisher menschliches Eingreifen oder Beaufsichtigung erforderlich war.
- Förderung von Innovation und Differenzierung: Entwickeln Sie intelligente Produkte, die sich anpassen, lernen und in Echtzeit konkurrieren.
- Modernisieren Sie veraltete Workflows: Gestalten Sie Skripte und Monolithen in modulare Argumentationsagenten um.

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

Einführung in Softwareagenten

Das Konzept der Softwareagenten hat sich seit seinen Anfängen in autonomen Einheiten in den 1960er Jahren bis hin zu seiner formalen Erforschung in den frühen 1990er Jahren erheblich weiterentwickelt. Da digitale Systeme immer komplexer werden — von deterministischen Skripten bis hin zu adaptiven, intelligenten Anwendungen — sind Softwareagenten zu unverzichtbaren Bausteinen geworden, um autonomes, kontextsensitives und zielorientiertes Verhalten in Computersystemen zu ermöglichen. Im Kontext cloudnativer und KI-gestützter Architekturen, insbesondere mit dem Aufkommen generativer KI, umfangreicher Sprachmodelle (LLMs) und Plattformen wie Amazon Bedrock, werden Softwareagenten durch neue Fähigkeiten und Skalierbarkeit neu definiert.

Diese Einführung basiert auf dem wegweisenden Werk Software Agents: An Overview von Hyacinth S. Nwana (Nwana 1996). Es definiert Softwareagenten, erörtert ihre konzeptionellen Wurzeln und erweitert die Diskussion auf einen aktuellen Rahmen, um drei übergreifende Prinzipien moderner Softwareagenten zu definieren: Autonomie, Asynchronität und Entscheidungsfreiheit. Diese Prinzipien unterscheiden Softwareagenten von anderen Arten von Diensten oder Anwendungen und ermöglichen es diesen Agenten, zielgerichtet, robust und intelligent in verteilten Echtzeitumgebungen zu arbeiten.

In diesem Abschnitt

Von der Autonomie zur verteilten Intelligenz

Bevor der Begriff Softwareagent Einzug hielt, beschäftigte sich die frühe Computerforschung mit der Idee autonomer digitaler Einheiten. Dabei handelt es sich um Systeme, die in der Lage sind, unabhängig zu handeln, auf Eingaben zu reagieren und Entscheidungen auf der Grundlage interner Regeln oder Ziele zu treffen. Diese frühen Ideen legten den konzeptionellen Grundstein für das, was später zum Agenten-Paradigma werden sollte. (Einen historischen Überblick finden Sie im Abschnitt Die Entwicklung von Softwareagenten weiter unten in diesem Handbuch.)

Frühe Konzepte von Autonomie

Die Vorstellung von Maschinen oder Programmen, die unabhängig von menschlichen Bedienern agieren, beschäftigt Systemdesigner seit Jahrzehnten. Frühe Arbeiten in den Bereichen Kybernetik, künstliche Intelligenz und Steuerungssysteme untersuchten, wie Software selbstregulierendes Verhalten zeigen, dynamisch auf Veränderungen reagieren und ohne ständige menschliche Überwachung funktionieren kann.

Diese Ideen führten Autonomie als zentrales Merkmal intelligenter Systeme ein und bereiteten die Voraussetzungen für die Entstehung von Software, die entscheiden und handeln konnte, anstatt nur zu reagieren oder auszuführen.

Das Akteurmodell und die asynchrone Ausführung

In den 1970er Jahren bot das Akteurmodell, das in der paper A Universal Modular ACTOR Formalism for Artificial Intelligence (Hewitt et al. 1973) vorgestellt wurde, einen formalen Rahmen für Überlegungen zu dezentraler, nachrichtengesteuerter Berechnung. In diesem Modell sind Akteure unabhängige Einheiten, die ausschließlich durch die Weitergabe asynchroner Nachrichten kommunizieren und skalierbare, gleichzeitige und fehlertolerante Systeme ermöglichen.

Das Akteurmodell betonte drei Hauptmerkmale, die das moderne Agentendesign weiterhin beeinflussen:

- Isolierung von Zustand und Verhalten
- Asynchrone Interaktion zwischen Entitäten
- Dynamische Erstellung und Delegierung von Aufgaben

Diese Eigenschaften entsprachen den Anforderungen verteilter Systeme und legten die Betriebseigenschaften von Softwareagenten in cloudnativen Umgebungen fest.

Verteilte Intelligenz und Systeme mit mehreren Agenten

Als Computersysteme nach den 1960er Jahren immer stärker miteinander vernetzt wurden, beschäftigten sich Forscher mit verteilter künstlicher Intelligenz (DAI). In diesem Bereich stand die Frage im Mittelpunkt, wie mehrere autonome Einheiten innerhalb eines Systems zusammenarbeiten oder kompetitiv zusammenarbeiten könnten. DAI führte zur Entwicklung von Systemen mit mehreren Agenten, bei denen jeder Akteur lokale Ziele verfolgt, seine eigene Wahrnehmung und Argumentation verfolgt, aber auch in einem breiteren, vernetzten Umfeld tätig ist.

Diese Vision der verteilten Intelligenz, bei der die Entscheidungsfindung dezentralisiert ist und neues Verhalten aus der Interaktion der Agenten resultiert, ist nach wie vor von zentraler Bedeutung für die Konzeption und den Aufbau moderner agentenbasierter Systeme.

Nwanas Typologie und der Aufstieg von Softwareagenten

Die Formalisierung des Softwareagentenkonzepts Mitte der 1990er Jahre markierte einen Wendepunkt in der Entwicklung intelligenter Systeme. Zu den einflussreichsten Beiträgen zu dieser Formalisierung gehört Hyacinth S. Nwanas wegweisender paper Software Agents: An Overview (Nwana 1996), der einen der ersten umfassenden Frameworks zur Kategorisierung und zum Verständnis von Softwareagenten in verschiedenen Dimensionen bot.

In diesem paper Nwana den Stand der Softwareagentenforschung und identifiziert eine zunehmende Divergenz in der Art und Weise, wie Agenten definiert und implementiert wurden. Das paper unterstreicht die Notwendigkeit eines gemeinsamen konzeptionellen Rahmens und schlägt eine Typologie vor, die Agenten nach ihren wichtigsten Fähigkeiten klassifiziert. Es untersucht repräsentative Agentensysteme aus Wissenschaft und Industrie, unterscheidet Agenten von herkömmlichen Programmen und Objekten und skizziert die Herausforderungen und Möglichkeiten des agentenbasierten Rechnens.

Nwana betont, dass es sich bei Softwareagenten nicht um ein monolithisches Konzept handelt, sondern dass es sich um ein breites Spektrum an Raffinesse und Leistungsfähigkeit handelt. Die Typologie dient dazu, diese Landschaft zu verdeutlichen und als Leitfaden für future Gestaltung und Forschung zu dienen.

Nwana definiert einen Softwareagenten als eine Softwareeinheit, die kontinuierlich und autonom in einer bestimmten Umgebung funktioniert, die häufig von anderen Agenten und Prozessen bewohnt wird. Diese Definition hebt zwei Hauptmerkmale hervor:

- Kontinuität: Der Agent arbeitet dauerhaft im Laufe der Zeit, ohne dass ein ständiges menschliches Eingreifen erforderlich ist.
- Autonomie: Der Agent ist in der Lage, auf der Grundlage seiner Wahrnehmung der Umgebung unabhängig Entscheidungen zu treffen und darauf zu reagieren.

Diese Definition, kombiniert mit der Agententypologie von Nwana, betont delegierte Autorität (durch Autonomie) und Proaktivität als grundlegende Merkmale von Agenten. Sie unterscheidet zwischen Agenten und Subroutinen oder Diensten, indem sie die Fähigkeit des Agenten hervorhebt,

unabhängig im Namen einer anderen Entität zu handeln und zielorientiertes Verhalten zu initiieren, anstatt nur auf direkte Befehle zu reagieren.

Nwanas Agententypologie

Zur weiteren Differenzierung zwischen den verschiedenen Agententypen führt Nwana ein Klassifizierungssystem ein, das auf sechs Schlüsselattributen basiert:

- Autonomie: Der Agent arbeitet ohne direkte Intervention von Menschen oder anderen.
- Soziale Fähigkeit: Der Agent interagiert mithilfe von Kommunikationsmechanismen mit anderen Agenten oder Menschen.
- Reaktivität: Der Agent nimmt seine Umgebung wahr und reagiert rechtzeitig.
- Proaktivität: Der Agent zeigt zielgerichtetes Verhalten, indem er die Initiative ergreift.
- Anpassungsfähigkeit und Lernen: Der Agent verbessert seine Leistung im Laufe der Zeit durch Erfahrung.
- Mobilität: Der Agent kann sich zwischen verschiedenen Systemumgebungen oder Netzwerken bewegen.

Von der Typologie zu modernen Agentenprinzipien

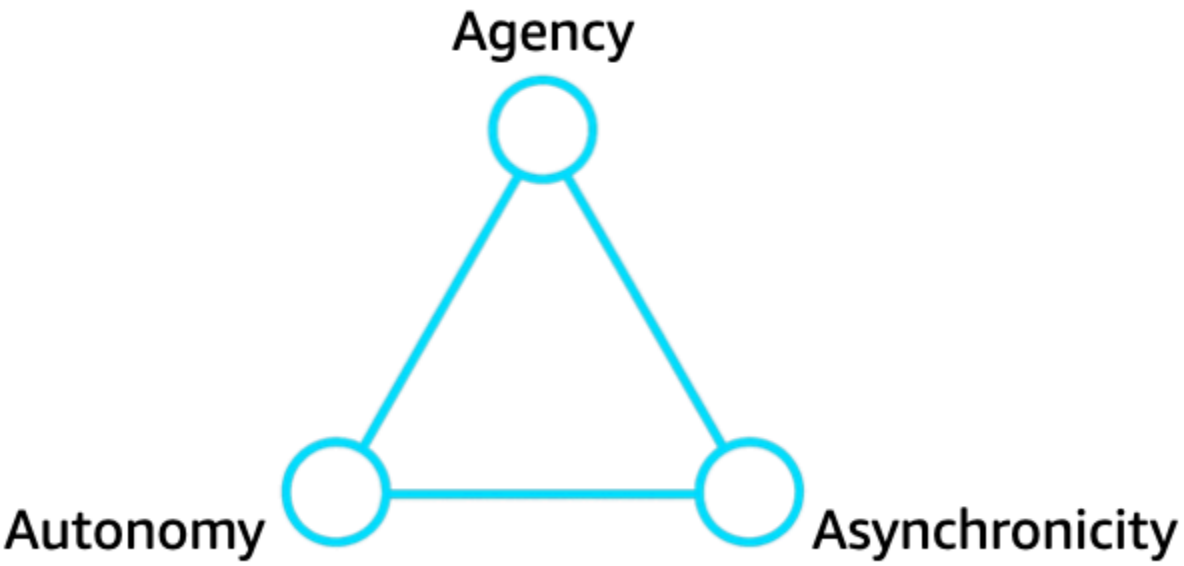
Nwanas Arbeit diente sowohl als Taxonomie als auch als grundlegende Linse, anhand derer die Computer-Community die sich entwickelnden Formen der Entscheidungsfreiheit im Bereich Software bewerten konnte. Seine Betonung von Autonomie, Proaktivität und dem Konzept, im Namen eines Benutzers oder Systems zu handeln, legte den Grundstein für das, was wir heute als agentisches Verhalten bezeichnen.

Obwohl sich die Technologien und Umgebungen verändert haben, insbesondere mit dem Aufkommen generativer KI, serverloser Infrastruktur und Frameworks für die Orchestrierung mehrerer Agenten, sind die grundlegenden Erkenntnisse aus Nwanas Arbeit nach wie vor relevant. Sie bilden eine wichtige Brücke zwischen der Theorie der frühen Agenten und den drei modernen Säulen der Softwareagenten.

Die drei Säulen moderner Softwareagenten

Im Kontext der heutigen KI-gestützten Plattformen, Microservice-Architekturen und ereignisgesteuerten Systeme können Softwareagenten durch drei voneinander abhängige Prinzipien

definiert werden, die sie von Standarddiensten oder Automatisierungsskripten unterscheiden: Autonomie, Asynchronität und Entscheidungsfreiheit. In der folgenden Abbildung und den nachfolgenden Diagrammen stellt das Dreieck diese drei Säulen moderner Softwareagenten dar.



Autonomie

Moderne Agenten arbeiten unabhängig. Sie treffen Entscheidungen auf der Grundlage des internen Zustands und des Umweltkontextes, ohne dass menschliche Eingabeaufforderungen erforderlich sind. Dies ermöglicht es ihnen, in Echtzeit auf Daten zu reagieren, ihren eigenen Lebenszyklus zu verwalten und ihr Verhalten auf der Grundlage von Zielen und situativen Eingaben anzupassen.

Autonomie ist die Grundlage für das Verhalten von Agenten. Es ermöglicht Agenten, ohne kontinuierliche Überwachung oder fest programmierte Kontrollabläufe zu arbeiten.

Asynchronität

Agenten sind grundsätzlich asynchron. Das bedeutet, dass sie auf Ereignisse, Signale und Stimuli reagieren, sobald sie auftreten, ohne sich auf blockierende Anrufe oder lineare Workflows verlassen zu müssen. Diese Eigenschaft ermöglicht eine skalierbare, blockierungsfreie Kommunikation, Reaktionsfähigkeit in verteilten Umgebungen und eine lose Kopplung zwischen Komponenten.

Durch Asynchronität können Agenten an Echtzeitsystemen teilnehmen und sich problemlos und effizient mit anderen Diensten oder Agenten abstimmen.

Agentur als entscheidendes Prinzip

Autonomie und Asynchronität sind notwendig, aber diese Funktionen allein reichen nicht aus, um ein System zu einem echten Softwareagenten zu machen. Das entscheidende Unterscheidungsmerkmal ist die Entscheidungsfreiheit, die Folgendes einführt:

- Zielgerichtetes Verhalten: Die Agenten verfolgen Ziele und bewerten die Fortschritte bei der Erreichung dieser Ziele.
- Entscheidungsfindung: Agenten bewerten Optionen und wählen Maßnahmen auf der Grundlage von Regeln, Modellen oder erlernten Richtlinien aus.
- Delegierte Absicht: Agenten handeln im Namen einer Person, eines Systems oder einer Organisation und haben ein ausgeprägtes Zielbewusstsein.
- Kontextuelles Denken: Agenten integrieren Erinnerungen oder Modelle ihrer Umgebung, um ihr Verhalten intelligent zu steuern.

Ein System, das autonom und asynchron ist, kann immer noch ein reaktiver Dienst sein. Was es zu einem Softwareagenten macht, ist seine Fähigkeit, bewusst und zielgerichtet zu handeln, agentisch zu sein.

Agentur mit Zielstrebigkeit

Die Prinzipien der Autonomie, Asynchronität und Entscheidungsfreiheit ermöglichen es Systemen, intelligent, adaptiv und unabhängig in verteilten Umgebungen zu arbeiten. Diese Prinzipien wurzeln in jahrzehntelanger konzeptioneller und architektonischer Entwicklung und liegen heute vielen der fortschrittlichsten KI-Systeme zugrunde, die heute gebaut werden.

In dieser neuen Ära generativer KI, zielorientierter Orchestrierung und Zusammenarbeit mit mehreren Agenten ist es wichtig zu verstehen, was einen Softwareagenten wirklich agentisch macht. Die Anerkennung der Entscheidungsfreiheit als entscheidendes Merkmal hilft uns, die Automatisierung hinter uns zu lassen und zielgerichtet in den Bereich der autonomen Intelligenz vorzudringen.

Der Zweck von Softwareagenten

Da moderne Systeme immer komplexer, verteilter und intelligenter werden, hat die Rolle von Softwareagenten in allen Bereichen, die von autonomen Operationen bis hin zu benutzerunterstützenden Technologien reichen, an Bedeutung gewonnen. Aber was ist der eigentliche Zweck von Softwareagenten? Warum entwerfen wir Systeme, die über Skripte, Dienste oder statische Modelle hinausgehen und stattdessen Aufgaben an Entitäten delegieren, die in der Lage sind, wahrzunehmen, zu argumentieren und zu handeln?

In diesem Abschnitt wird der grundlegende Zweck von Softwareagenten untersucht: die intelligente Delegierung von Aufgaben in dynamischen Umgebungen zu ermöglichen, wobei der Schwerpunkt auf Autonomie, Anpassungsfähigkeit und zielgerichtetem Handeln liegt. Er stellt die konzeptionellen Grundlagen von Softwareagenten vor, zeichnet ihre kognitive Struktur nach und skizziert die realen Probleme, zu deren Lösung sie in einzigartiger Weise in der Lage sind.

In diesem Abschnitt

Vom Schauspielermodell zur Agentenkognition

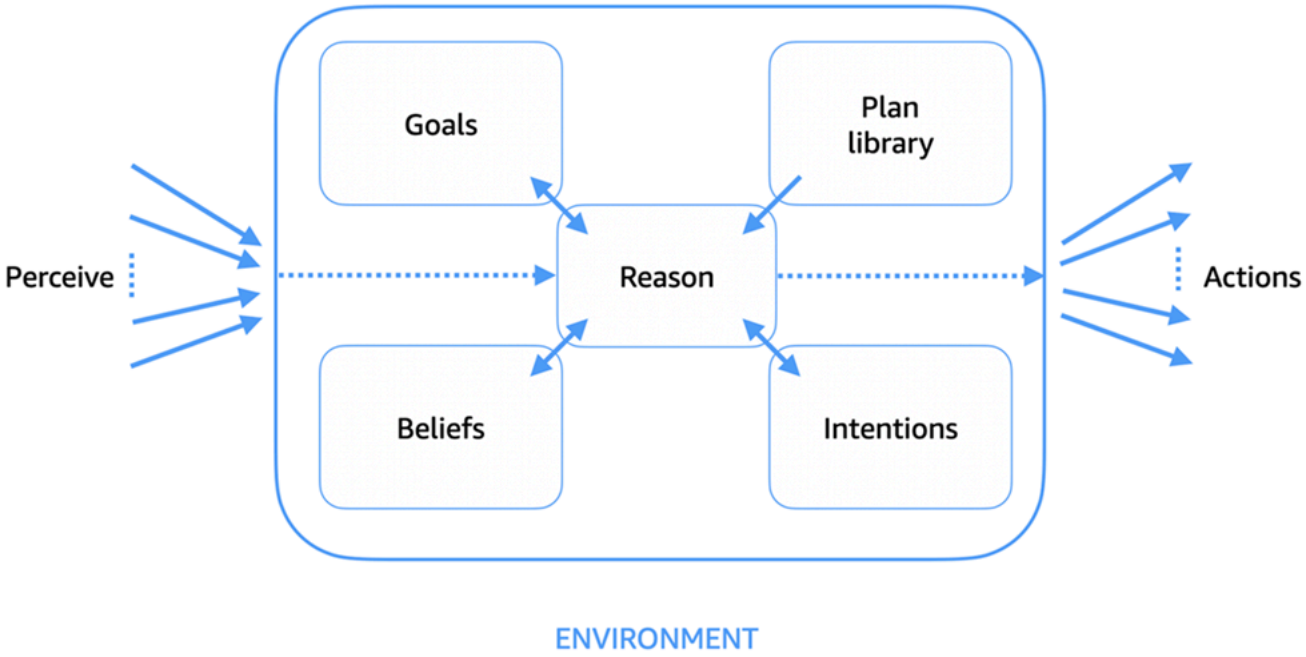
Der Zweck und die Struktur von Softwareagenten basieren auf Ideen, die aus frühen Berechnungsmodellen hervorgegangen sind, insbesondere dem Akteurmodell, das in den 1970er Jahren von Carl Hewitt eingeführt wurde (Hewitt et al. 1973).

Das Akteurmodell behandelt Berechnungen als eine Sammlung unabhängiger, gleichzeitig ausführender Einheiten, die als Akteure bezeichnet werden. Jeder Akteur kapselt seinen eigenen Status, interagiert ausschließlich durch asynchrone Nachrichtenübertragung und kann neue Akteure erstellen und Aufgaben delegieren.

Dieses Modell bildete die konzeptionelle Grundlage für dezentrales Denken, Reaktivität und Isolation — all diese Faktoren bilden die Grundlage für die Verhaltensarchitektur moderner Softwareagenten.

Die Agentenfunktion: wahrnehmen, argumentieren, handeln

Im Mittelpunkt jedes Softwareagenten steht ein kognitiver Zyklus, der oft als Schleife „Wahrnehmung, Vernunft, Handlung“ beschrieben wird. Dieser Prozess wird im folgenden Diagramm veranschaulicht. Er definiert, wie Agenten in dynamischen Umgebungen autonom agieren.



- Wahrnehmen: Agenten sammeln Informationen (z. B. Ereignisse, Sensoreingaben oder API-Signale) aus der Umgebung und aktualisieren ihren internen Zustand oder ihre Überzeugungen.
- Begründung: Agenten analysieren aktuelle Überzeugungen, Ziele und kontextuelles Wissen mithilfe einer Planbibliothek oder eines Logiksystems. Dieser Prozess kann die Priorisierung von Zielen, die Konfliktlösung oder die Auswahl von Absichten beinhalten.
- Handeln: Agenten wählen Aktionen aus und führen sie aus, die sie der Erreichung ihrer delegierten Ziele näher bringen.

Diese Architektur unterstützt die Fähigkeit von Agenten, über starre Programmierung hinaus zu arbeiten, und ermöglicht flexibles, kontextsensitives und zielgerichtetes Verhalten. Sie bildet den mentalen Rahmen, der die umfassenderen Zwecke von Softwareagenten bestimmt.

Autonome Zusammenarbeit und Intentionalität

Der Zweck von Softwareagenten besteht darin, modernen Computern Autonomie, Kontextsensitivität und intelligente Delegierung zu verleihen. Da Agenten auf den Prinzipien des Akteurmodells basieren und in den Zyklus „Wahrnehmen, Vernunftdenken“ und „Handeln“ eingebettet sind, ermöglichen sie Systeme, die nicht nur reaktiv, sondern auch proaktiv und zielgerichtet sind.

Agenten ermöglichen es der Software, in komplexen Umgebungen zu entscheiden, sich anzupassen und zu handeln. Sie repräsentieren Benutzer, interpretieren Ziele und implementieren Aufgaben mit Maschinengeschwindigkeit. Je weiter wir in das Zeitalter der agentischen KI vordringen, desto mehr werden Softwareagenten zur operativen Schnittstelle zwischen menschlicher Absicht und intelligentem digitalem Handeln.

Absicht delegieren

Im Gegensatz zu herkömmlichen Softwarekomponenten agieren Softwareagenten für etwas anderes: für einen Benutzer, ein anderes System oder für einen Dienst auf höherer Ebene. Sie verfolgen eine delegierte Absicht, was bedeutet, dass sie:

- Arbeiten Sie nach der Initiierung selbständig.
- Treffen Sie Entscheidungen, die auf die Ziele des Delegierenden abgestimmt sind.
- Bewältigen Sie Unsicherheiten und Kompromisse bei der Ausführung.

Agenten überbrücken die Lücke zwischen Anweisungen und Ergebnissen, sodass Benutzer ihre Absicht auf einer höheren Abstraktionsebene ausdrücken können, anstatt explizite Anweisungen zu benötigen.

Betrieb in dynamischen, unvorhersehbaren Umgebungen

Softwareagenten sind für Umgebungen konzipiert, in denen sich die Bedingungen ständig ändern, Daten in Echtzeit ankommen und Kontrolle und Kontext verteilt sind.

Im Gegensatz zu statischen Programmen, die exakte Eingaben oder synchrone Ausführung erfordern, passen sich Agenten an ihre Umgebung an und reagieren dynamisch. Dies ist eine wichtige Funktion für Cloud-native Infrastrukturen, Edge-Computing, Netzwerke für das Internet der Dinge (IoT) und Entscheidungssysteme in Echtzeit.

Reduzierung der kognitiven Belastung des Menschen

Einer der Hauptzwecke von Softwareagenten besteht darin, die kognitive und betriebliche Belastung des Menschen zu verringern. Agenten können:

- Systeme und Workflows kontinuierlich überwachen.
- Erkennen Sie vordefinierte oder neu auftretende Bedingungen und reagieren Sie darauf.
- Automatisieren Sie sich wiederholende, umfangreiche Entscheidungen.
- Reagieren Sie mit minimaler Latenz auf Umweltveränderungen.

Wenn die Entscheidungsfindung von den Benutzern auf die Agenten verlagert wird, reagieren die Systeme schneller, sind widerstandsfähiger und stehen stärker auf den Menschen im Mittelpunkt. Sie können sich in Echtzeit an neue Informationen oder Störungen anpassen. Dies ermöglicht eine schnellere Reaktionszeit sowie eine größere Betriebskontinuität in Umgebungen mit hoher Komplexität oder großem Maßstab. Das Ergebnis ist eine Verlagerung des menschlichen Schwerpunkts, weg von der Entscheidungsfindung auf Mikroebene hin zu strategischer Aufsicht und kreativer Problemlösung.

Wir ermöglichen verteilte Intelligenz

Die Fähigkeit von Softwareagenten, einzeln oder gemeinsam zu arbeiten, ermöglicht den Entwurf von Multi-Agentensystemen (MAS), die umgebungs- oder organisationsübergreifend koordiniert werden. Diese Systeme können Aufgaben intelligent verteilen und miteinander verhandeln, zusammenarbeiten oder miteinander konkurrieren, um gemeinsame Ziele zu erreichen.

In einem globalen Lieferkettensystem verwalten beispielsweise einzelne Agenten die Fabriken, den Versand, die Lagerhäuser und die Lieferung auf der letzten Meile. Jeder Vertreter arbeitet mit lokaler Autonomie: Die Werksmitarbeiter optimieren die Produktion auf der Grundlage von Ressourcenbeschränkungen, die Lagermitarbeiter passen die Lagerflüsse in Echtzeit an und die Zusteller leiten Lieferungen auf der Grundlage des Verkehrs und der Kundenverfügbarkeit um.

Diese Agenten kommunizieren und koordinieren dynamisch und passen sich ohne zentrale Steuerung an Störungen wie Hafenverzögerungen oder Lkw-Ausfälle an. Die gesamte Intelligenz des Systems ergibt sich aus diesen Interaktionen und ermöglicht eine belastbare, optimierte Logistik, die die Fähigkeiten einer einzelnen Komponente übersteigt.

In diesem Modell agieren Agenten als Knoten in einem breiteren Informationsgefüge. Sie bilden neue Systeme, die in der Lage sind, Probleme zu lösen, die keine einzelne Komponente alleine bewältigen könnte.

Zielgerichtet handeln, nicht nur reagieren

Automatisierung allein reicht in komplexen Systemen nicht aus. Der grundlegende Zweck eines Softwareagenten besteht darin, zielgerichtet zu handeln und Ziele zu bewerten, den Kontext abzuwägen und fundierte Entscheidungen zu treffen. Das bedeutet, dass Softwareagenten Ziele verfolgen, anstatt nur auf Auslöser zu reagieren. Sie können Überzeugungen und Absichten auf der Grundlage von Erfahrungen oder Feedback revidieren. In diesem Zusammenhang beziehen sich Überzeugungen auf die interne Repräsentation der Umgebung durch den Agenten (z. B. „Paket X befindet sich in Lager A“), die auf seinen Wahrnehmungen (Eingaben und Sensoren) basiert. Intentionen beziehen sich auf die Pläne, die der Mitarbeiter wählt, um ein Ziel zu erreichen (z. B. „Lieferweg B verwenden und den Empfänger benachrichtigen“). Agenten können Aktionen auch nach Bedarf eskalieren, verschieben oder anpassen.

Diese Intentionalität macht Softwareagenten nicht nur zu reaktiven Ausführenden, sondern zu autonomen Kollaborateuren in intelligenten Systemen.

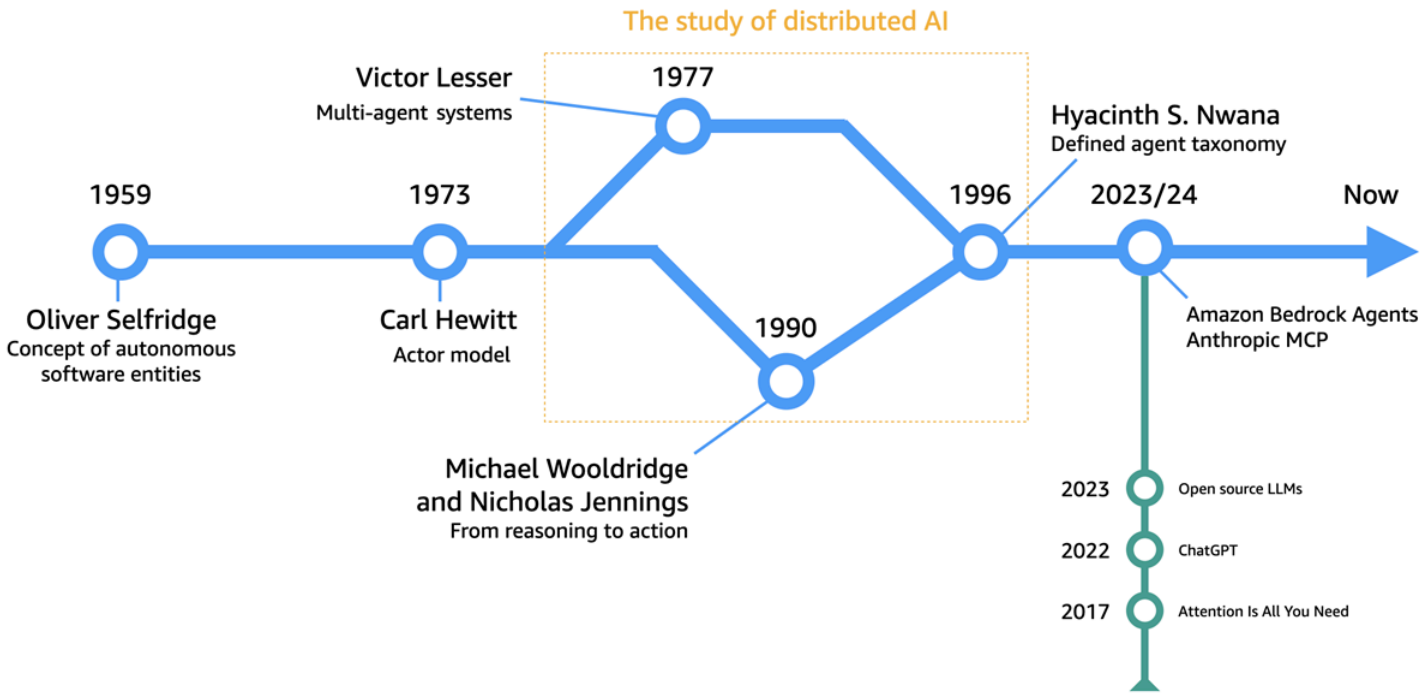
Die Entwicklung von Softwareagenten

Der Weg von einfachen automatisierten Systemen zu intelligenten, autonomen und zielgerichteten Softwareagenten spiegelt die jahrzehntelange Entwicklung in den Bereichen Informatik, künstliche Intelligenz und verteilte Systeme wider.

Auf diese Entwicklung folgte der Aufstieg des maschinellen Lernens, der das Paradigma von handgefertigten Regeln hin zur statistischen Mustererkennung verlagerte. Diese Systeme konnten aus Daten lernen und ermöglichten Fortschritte bei der Wahrnehmung, Klassifizierung und Entscheidungsfindung.

Große Sprachmodelle (LLMs) stehen für eine Konvergenz von Umfang, Architektur und unbeaufsichtigtem Lernen. LLMs kann Aufgaben mit wenig oder keiner aufgabenspezifischen Schulung begründen, generieren und anpassen. Durch die Kombination LLMs mit skalierbarer, cloudnativer Infrastruktur und kombinierbaren Architekturen erreichen wir nun die vollständige Vision von agentischer KI: intelligente Softwareagenten, die auf Unternehmensebene autonom, kontextbewusst und anpassungsfähig arbeiten können.

In diesem Abschnitt wird die Geschichte der Softwareagenten von der grundlegenden Theorie bis zur modernen Praxis untersucht, wie in der folgenden Abbildung dargestellt. Es beleuchtet die Konvergenz von verteilter künstlicher Intelligenz (DAI) und transformatorbasierter generativer KI und identifiziert die wichtigsten Meilensteine, die die Entstehung der agentischen KI geprägt haben.



In diesem Abschnitt

Grundlagen von Softwareagenten

1959 — Oliver Selfridge: Die Geburt der Autonomie in der Software

Die Wurzeln von Softwareagenten gehen auf Oliver Selfridge zurück, der das Konzept autonomer Softwareentitäten (Dämonen) einführte — Programme, die in der Lage sind, ihre Umgebung wahrzunehmen und unabhängig zu handeln (Selfridge 1959). Seine frühen Arbeiten zur maschinellen Wahrnehmung und zum Lernen legten den philosophischen Grundstein für future Vorstellungen von Agenten als unabhängige, intelligente Systeme.

1973 — Carl Hewitt: das Schauspieler-Modell

Ein entscheidender Fortschritt war das Schauspielermodell von Carl Hewitt (Hewitt et al. 1973), ein formales Rechenmodell, das Agenten als unabhängige, gleichzeitige Einheiten beschreibt. In diesem Modell können Agenten ihren eigenen Status und ihr eigenes Verhalten kapseln, mithilfe asynchroner Nachrichtenübergabe kommunizieren und dynamisch andere Akteure erstellen und Aufgaben an sie delegieren.

Das Akteurmodell lieferte sowohl die theoretische Grundlage als auch das architektonische Paradigma für verteilte, agentenbasierte Systeme. Dieses Modell hat moderne Parallelitätsimplementierungen wie die Programmiersprache Erlang und das Akka-Framework vorkonfiguriert.

Das Feld reifer werden lassen: vom Denken zum Handeln

1977 — Victor Lesser: Systeme mit mehreren Agenten

In den späten 1970er Jahren entstand verteilte künstliche Intelligenz (DAI). Sie wurde von Victor Lesser unterstützt, der weithin für seine wegweisenden Multi-Agenten-Systeme (MAS) bekannt ist. Seine Arbeit konzentrierte sich darauf, wie unabhängige Softwareunternehmen zusammenarbeiten,

sich koordinieren und verhandeln könnten (siehe Abschnitt Ressourcen). Diese Entwicklung führte zu Systemen, die in der Lage waren, komplexe Probleme gemeinsam zu lösen — ein entscheidender Schritt beim Aufbau verteilter Intelligenz.

1990er Jahre — Michael Wooldridge und Nicholas Jennings: das Agentenspektrum

In den 1990er Jahren hatte sich der Bereich der verteilten Intelligenz durch Beiträge von Forschern wie Michael Wooldridge und Nicholas Jennings weiterentwickelt. Diese Wissenschaftler kategorisierten Agenten anhand eines Spektrums, das von reaktiven bis deliberativen, von nichtkognitiven Systemen bis hin zu zielorientierten Argumentatoren reichte (Wooldridge und Jennings 1995). In ihrer Arbeit wurde betont, dass Agenten keine abstrakten Ideen mehr sind, sondern dass sie in einer Vielzahl praktischer Bereiche, von der Robotik bis hin zu Unternehmenssoftware, eingesetzt werden.

Diese Forscher führten auch eine Schwerpunktverlagerung ein: weg von zentralisiertem Denken hin zu verteiltem Handeln. Agenten waren nicht mehr nur Denker, sondern auch Macher, die eigenständig und zielgerichtet in Echtzeitumgebungen agierten.

1996 — Hyacinth S. Nwana: Formalisierung des Agentenkonzepts

1996 veröffentlichte Hyacinth S. Nwana den einflussreichen paper Software Agents: An Overview, der die bisher umfassendste Klassifizierung von Agenten enthielt. Seine Typologie umfasste Merkmale wie Autonomie, soziale Fähigkeiten, Reaktivität, Proaktivität, Lernen und Mobilität und unterschied zwischen Softwareagenten und traditionellen Softwarekonstrukten.

Nwana bot auch eine inzwischen allgemein anerkannte, paraphrasierte Definition an: Ein Softwareagent ist ein softwarebasiertes Computerprogramm, das für einen Benutzer oder ein anderes Programm in einem Autoritätsverhältnis agiert, was auf den Begriff der Delegation zurückgeht.

Diese Formalisierung war entscheidend für den Übergang von Softwareagenten von theoretischen Konstrukten zu realen Anwendungen. Sie führte zu einer Generation agentenbasierter Systeme in Bereichen wie Telekommunikation, Workflow-Automatisierung und intelligenten Assistenten.

Nwanas Arbeit steht an der Schnittstelle zwischen der frühen verteilten KI-Forschung und den Betriebsarchitekturen moderner Agenten. Es ist eine entscheidende Brücke zwischen der kognitiven Theorie von Agenten und ihrem praktischen Einsatz in heutigen Systemen.

Eine parallel Zeitleiste: Der Aufstieg großer Sprachmodelle

Während sich die Agenten-Frameworks weiterentwickelten, fand eine parallel und konvergente Revolution in der Verarbeitung natürlicher Sprache und beim maschinellen Lernen statt:

- 2017 — Transformers: In dem paper Attention Is All You Need (Vaswani et al. 2017) wurde die Transformator-Architektur vorgestellt, die die Art und Weise, wie Maschinen Sprache verarbeiten und erzeugen, dramatisch verbessert hat.
- 2022 — ChatGPT: OpenAI veröffentlichte eine Chat-basierte Schnittstelle für GPT-3.5 namens ChatGPT, die eine natürliche, interaktive Konversation mit einem Allzweck-KI-System ermöglichte.
- 2023 — Open Source LLMs: Die Veröffentlichungen von Llama, Falcon und Mistral machten leistungsstarke Modelle allgemein zugänglich und beschleunigten die Entwicklung von Agenten-Frameworks in Open Source- und Unternehmensumgebungen.

Diese Innovationen machten Sprachmodelle zu Argumentationsmechanismen, die in der Lage sind, den Kontext zu analysieren, Maßnahmen zu planen und Antworten zu verketten, und LLMs wurden zu wichtigen Voraussetzungen für intelligente Softwareagenten.

Die Zeitlinien laufen zusammen: das Aufkommen agentischer KI

2023-2024 — Agentenplattformen für Unternehmen

Die Konvergenz von verteilten Softwareagenten-Architekturen und transformatorbasierten Architekturen LLMs gipfelte im Aufstieg der Agenten-KI.

- Amazon Bedrock Agents führte eine vollständig verwaltete Methode zur Erstellung zielorientierter, Tools verwendender Softwareagenten ein, die auf Fundamentmodellen von Amazon Bedrock basieren.
- Das Model Context Protocol (MCP) von Anthropic definierte eine Methode, mit der große Sprachmodelle auf externe Tools, Umgebungen und Speicher zugreifen und mit ihnen interagieren können. Dies ist der Schlüssel für kontextuelles, persistentes und autonomes Verhalten.

Diese beiden Meilensteine stellen die Synthese von Entscheidungsfreiheit und Intelligenz dar. Agenten waren nicht mehr auf statische Workflows oder starre Automatisierung beschränkt. Sie konnten nun in mehreren Schritten argumentieren, sich mit Tools abstimmen und APIs den kontextuellen Status beibehalten sowie im Laufe der Zeit lernen und sich anpassen.

Januar-Juni 2025 — erweiterte Unternehmenskapazitäten

In der ersten Hälfte des Jahres 2025 wurde die KI-Landschaft der Agenturen um neue Unternehmensfunktionen erheblich erweitert. Im Februar 2025 veröffentlichte Anthropic Claude 3.7 Sonnet, das erste hybride Argumentationsmodell auf dem Markt, und die MCP-Spezifikation fand breite Akzeptanz.

KI-Programmierassistenten wie Amazon Q Developer, Cursor und WindSurf integriertes MCP zur Standardisierung von Codegenerierung, Repository-Analyse und Entwicklungsworkflows. Die MCP-Version vom März 2025 führte wichtige Funktionen für Unternehmen ein, darunter die Sicherheitsintegration OAuth 2.1, erweiterte Ressourcentypen für den vielfältigen Datenzugriff und erweiterte Konnektivitätsoptionen über Streamable HTTP. Auf dieser Grundlage wurde im Mai 2025 AWS angekündigt, dem MCP-Lenkungsausschuss beizutreten und zu neuen Kommunikationsmöglichkeiten beizutragen. agent-to-agent Dies stärkt die Position des Protokolls als Industriestandard für agentische KI-Interoperabilität weiter.

Im Mai 2025 wurden die Optionen der Kunden für den Aufbau agentischer KI-Workflows AWS gestärkt, indem das Strands Agents-Framework als Open-Source-Lösung zur Verfügung gestellt wurde. Dieses anbieterunabhängige und modellunabhängige Framework ermöglicht es Entwicklern, Basismodelle plattformübergreifend zu verwenden und gleichzeitig eine tiefe Serviceintegration aufrechtzuerhalten. AWS Wie im AWS Open-Source-Blog hervorgehoben, folgt Strands Agents einer Designphilosophie, bei der das Modell an erster Stelle steht und die Basismodelle in den Mittelpunkt der Agentenintelligenz stellt. Dies erleichtert es Kunden, ausgefeilte KI-Agenten für ihre spezifischen Anwendungsfälle zu entwickeln und einzusetzen.

Emergence — Agentengestützte KI

Die Entwicklung von Softwareagenten, von den frühen Ideen der Autonomie bis hin zur modernen, LLM-fähigen Orchestrierung, war langwierig und vielschichtig. Was mit Oliver Selfridges Vision der Wahrnehmung von Programmen begann, hat sich zu einem robusten Ökosystem intelligenter, kontextsensitiver, zielorientierter Softwareagenten entwickelt, die zusammenarbeiten, sich anpassen und vernünftig denken können.

Die Konvergenz von verteilter künstlicher Intelligenz (DAI) und transformatorbasierter generativer KI markiert den Beginn einer neuen Ära, in der Softwareagenten nicht mehr nur Werkzeuge, sondern autonome Akteure in intelligenten Systemen sind.

Agentic AI stellt die nächste Entwicklung von Softwaresystemen dar. Es bietet eine Klasse intelligenter Agenten, die autonom, asynchron und agentisch sind und mit delegierter Absicht agieren

und zielgerichtet in dynamischen, verteilten Umgebungen arbeiten können. Agentic AI vereint Folgendes:

- Die architektonische Abstammung von Systemen mit mehreren Agenten und das Akteurmodell
- Das kognitive Modell von Wahrnehmen, Vernunft, Handeln
- Die schöpferische Kraft von LLMs und Transformatoren
- Die betriebliche Flexibilität von Cloud-nativem und serverlosem Computing

Von Softwareagenten zu agentischer KI

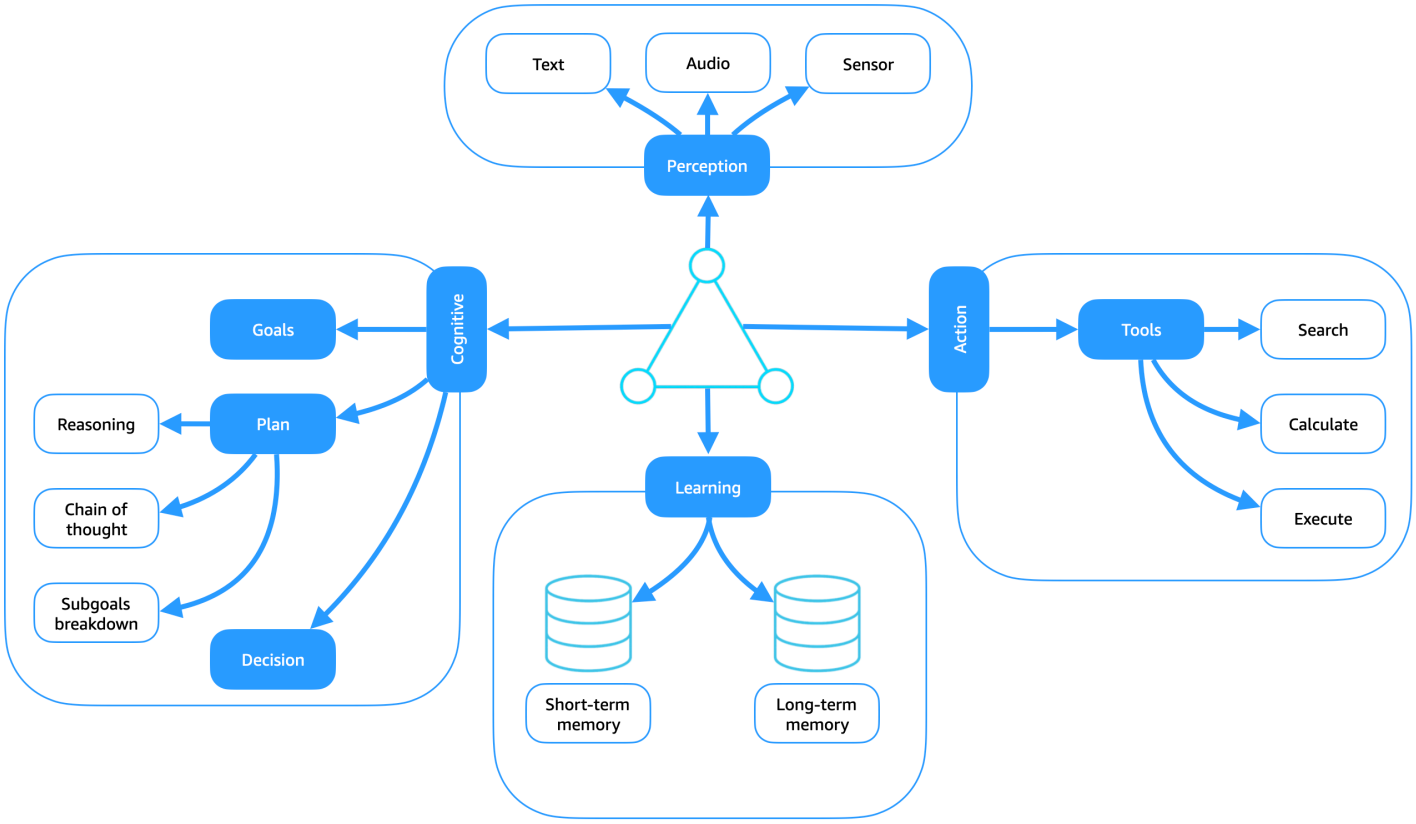
Softwareagenten sind autonome digitale Einheiten, die darauf ausgelegt sind, ihre Umgebung wahrzunehmen, über ihre Ziele nachzudenken und entsprechend zu handeln. Im Gegensatz zu herkömmlichen Softwareprogrammen, die einer festen Logik folgen, passen Agenten ihr Verhalten auf der Grundlage kontextueller Eingaben und Entscheidungsrahmen an. Dadurch eignen sie sich ideal für dynamische, verteilte Umgebungen wie Cloud-native Systeme, Robotik, intelligente Automatisierung und jetzt auch generative KI-Orchestrierung.

In diesem Abschnitt werden die Kernbausteine von Softwareagenten vorgestellt und es wird erklärt, wie diese Komponenten innerhalb herkömmlicher Architekturen auf der Grundlage des Modells „Wahrnehmen, Vernunftdenken“ und „Handeln“ interagieren. Es wird erörtert, wie generative KI, insbesondere große Sprachmodelle (LLMs), die Art und Weise verändert hat, wie Softwareagenten denken und planen. Dies markiert einen grundlegenden Wandel von regelbasierten Systemen hin zur datengesteuerten, erlernten Intelligenz der agentischen KI.

In diesem Abschnitt

Kernbausteine von Softwareagenten

Das folgende Diagramm zeigt die wichtigsten Funktionsmodule, die in den meisten intelligenten Agenten zu finden sind. Jede Komponente trägt dazu bei, dass der Agent in komplexen Umgebungen autonom arbeiten kann.



Im Rahmen der Schleife „Wahrnehmen, Vernunft, Handeln“ verteilt sich das Denkvermögen eines Agenten sowohl auf seine kognitiven als auch auf seine Lernmodule. Durch die Integration von Gedächtnis und Lernen entwickelt der Agent adaptives Denken, das auf früheren Erfahrungen basiert. Wenn der Agent in seiner Umgebung agiert, erzeugt er eine emergente Rückkopplungsschleife: Jede Aktion beeinflusst future Wahrnehmungen, und die daraus resultierende Erfahrung wird durch das Lernmodul in das Gedächtnis und in interne Modelle aufgenommen. Diese kontinuierliche Schleife aus Wahrnehmung, Argumentation und Handlung ermöglicht es dem Agenten, sich im Laufe der Zeit zu verbessern, und schließt den gesamten Zyklus „Wahrnehmen, Denken und Handeln“ ab.

Modul „Wahrnehmung“

Das Wahrnehmungsmodul ermöglicht es dem Agenten, über verschiedene Eingabemodalitäten wie Text, Audio und Sensoren mit seiner Umgebung zu interagieren. Diese Eingaben bilden die Rohdaten, auf denen alle Überlegungen und Handlungen basieren. Texteingaben können Eingabeaufforderungen in natürlicher Sprache, strukturierte Befehle oder Dokumente beinhalten. Zu den Audioeingängen gehören gesprochene Anweisungen oder Umgebungsgeräusche. Zu den Sensoreingängen gehören physische Daten wie visuelle Feeds, Bewegungssignale oder GPS-Koordinaten. Die Kernfunktion der Wahrnehmung besteht darin, aus diesen Rohdaten

aussagekräftige Merkmale und Repräsentationen zu extrahieren. Dies ermöglicht es dem Agenten, ein genaues und umsetzbares Verständnis seines aktuellen Kontextes zu entwickeln. Der Prozess kann die Extraktion von Merkmalen, die Erkennung von Objekten oder Ereignissen und die semantische Interpretation beinhalten und ist der entscheidende erste Schritt in der Schleife „Wahrnehmen, Begründen, Handeln“. Eine effektive Wahrnehmung stellt sicher, dass nachgelagerte Überlegungen und Entscheidungen auf relevantem Situationsbewusstsein beruhen. up-to-date

Kognitives Modul

Das kognitive Modul dient als beratender Kern des Softwareagenten. Es ist dafür verantwortlich, Wahrnehmungen zu interpretieren, Absichten zu entwickeln und zielgerichtetes Verhalten durch zielgerichtete Planung und Entscheidungsfindung zu lenken. Dieses Modul wandelt Eingaben in strukturierte Argumentationsprozesse um, die es dem Agenten ermöglichen, bewusst statt reaktiv zu handeln. Diese Prozesse werden über drei wichtige Untermodule gesteuert: Ziele, Planung und Entscheidungsfindung.

Untermodul „Ziele“

Das Untermodul Ziele definiert die Absicht und Richtung des Agenten. Ziele können explizit (z. B. „zu einem Standort navigieren“ oder „Bericht einreichen“) oder implizit (z. B. „Benutzerinteraktion maximieren“ oder „Latenz minimieren“) sein. Sie sind von zentraler Bedeutung für den Argumentationszyklus des Agenten und bilden die Grundlage für seine Planung und Entscheidungen.

Der Agent bewertet kontinuierlich, welche Fortschritte bei der Erreichung seiner Ziele erzielt wurden, und kann Ziele auf der Grundlage neuer Erkenntnisse oder Erkenntnisse neu priorisieren oder neu definieren. Dieses Zielbewusstsein sorgt dafür, dass der Agent in dynamischen Umgebungen anpassungsfähig bleibt.

Untermodul Planung

Das Submodul Planung entwickelt Strategien, um die aktuellen Ziele des Agenten zu erreichen. Es generiert Aktionssequenzen, zerlegt Aufgaben hierarchisch und wählt aus vordefinierten oder dynamisch generierten Plänen aus.

Um in nicht deterministischen oder sich verändernden Umgebungen effektiv arbeiten zu können, ist Planung nicht statisch. Moderne Agenten können chain-of-thought Sequenzen generieren, Teilziele als Zwischenschritte einführen und Pläne in Echtzeit überarbeiten, wenn sich die Bedingungen ändern.

Dieses Untermodul ist eng mit Gedächtnis und Lernen verknüpft und ermöglicht es dem Agenten, seine Planung im Laufe der Zeit auf der Grundlage früherer Ergebnisse zu verfeinern.

Untermodul „Entscheidungsfindung“

Das Untermodul Entscheidungsfindung bewertet die verfügbaren Pläne und Maßnahmen, um den am besten geeigneten nächsten Schritt auszuwählen. Es integriert Informationen aus der Wahrnehmung, den aktuellen Plan, die Ziele des Agenten und den Umweltkontext.

Die Entscheidungsfindung berücksichtigt:

- Kompromisse zwischen widersprüchlichen Zielen
- Vertrauensschwellen (z. B. Unsicherheit in der Wahrnehmung)
- Folgen von Handlungen
- Die erlernte Erfahrung des Agenten

Je nach Architektur können sich Agenten auf symbolisches Denken, Heuristik, Reinforcement-Learning oder Sprachmodelle (LLMs) verlassen, um fundierte Entscheidungen zu treffen. Dieser Prozess sorgt dafür, dass das Verhalten des Agenten kontextsensitiv, zielgerichtet und anpassungsfähig bleibt.

Aktionsmodul

Das Aktionsmodul ist dafür verantwortlich, die vom Agenten ausgewählten Entscheidungen auszuführen und sich mit der Außenwelt oder internen Systemen zu verbinden, um sinnvolle Effekte zu erzielen. Es stellt die Phase des Handelns in der Schleife „Wahrnehmung, Vernunft, Handlung“ dar, in der Absicht in Verhalten umgewandelt wird.

Wenn das kognitive Modul eine Aktion auswählt, koordiniert das Aktionsmodul die Ausführung mithilfe spezialisierter Submodule, wobei jedes Submodul auf die integrierte Umgebung des Agenten abgestimmt ist:

- Physische Betätigung: Für Agenten, die in Robotersysteme oder IoT-Geräte eingebettet sind, übersetzt dieses Submodul Entscheidungen in reale physische Bewegungen oder Anweisungen auf Hardwareebene.

  Beispiele: Steuerung eines Roboters, Auslösung eines Ventils, Einschalten eines Sensors.
- Integrierte Interaktion: Dieses Untermodul verarbeitet nicht-physische, aber von außen sichtbare Aktionen wie die Interaktion mit Softwaresystemen, Plattformen oder. APIs

Beispiele: Senden eines Befehls an einen Cloud-Dienst, Aktualisieren einer Datenbank, Senden eines Berichts durch Aufrufen einer API.

- Aufruf von Tools: Agenten erweitern ihre Fähigkeiten häufig, indem sie spezielle Tools verwenden, um Unteraufgaben wie die folgenden zu erledigen:
  - Suche: Abfragen strukturierter oder unstrukturierter Wissensquellen
  - Zusammenfassung: Komprimierung großer Texteingaben zu übersichtlichen Übersichten
  - Berechnung: Durchführung logischer, numerischer oder symbolischer Berechnungen

  Der Aufruf von Tools ermöglicht die Zusammenstellung komplexer Verhaltensweisen durch modulare, aufrufbare Fähigkeiten.

Lernmodul

Das Lernmodul ermöglicht es Agenten, sich im Laufe der Zeit auf der Grundlage von Erfahrungen anzupassen, zu verallgemeinern und zu verbessern. Es unterstützt den Argumentationsprozess, indem es die internen Modelle, Strategien und Entscheidungsrichtlinien des Agenten kontinuierlich verfeinert, indem es Feedback aus Wahrnehmung und Handlung nutzt.

Dieses Modul arbeitet in Abstimmung mit dem Kurzzeit- und Langzeitgedächtnis:

- Kurzzeitgedächtnis: Speichert vorübergehenden Kontext wie den Status eines Dialogs, aktuelle Aufgabeninformationen und aktuelle Beobachtungen. Es hilft dem Agenten, die Kontinuität der Interaktionen und Aufgaben aufrechtzuerhalten.
- Langzeitgedächtnis: Kodiert beständiges Wissen aus vergangenen Erfahrungen, einschließlich zuvor erreichter Ziele, Ergebnisse von Maßnahmen und Umweltzuständen. Das Langzeitgedächtnis ermöglicht es dem Agenten, Muster zu erkennen, Strategien wiederzuverwenden und Wiederholungen von Fehlern zu vermeiden.

Lernmodi

Das Lernmodul unterstützt eine Reihe von Paradigmen, wie z. B. überwachtes, unbeaufsichtigtes und verstärkendes Lernen, die unterschiedliche Umgebungen und Agentenrollen unterstützen:

- Überwachtes Lernen: Aktualisiert interne Modelle auf der Grundlage beschrifteter Beispiele, die häufig auf menschlichem Feedback oder Trainingsdatensätzen basieren.

Beispiel: Lernen, Benutzerabsichten auf der Grundlage früherer Konversationen zu klassifizieren.

- Unüberwachtes Lernen: Identifiziert versteckte Muster oder Strukturen in Daten ohne explizite Bezeichnungen.

  Beispiel: Clustering von Umgebungssignalen zur Erkennung von Anomalien.

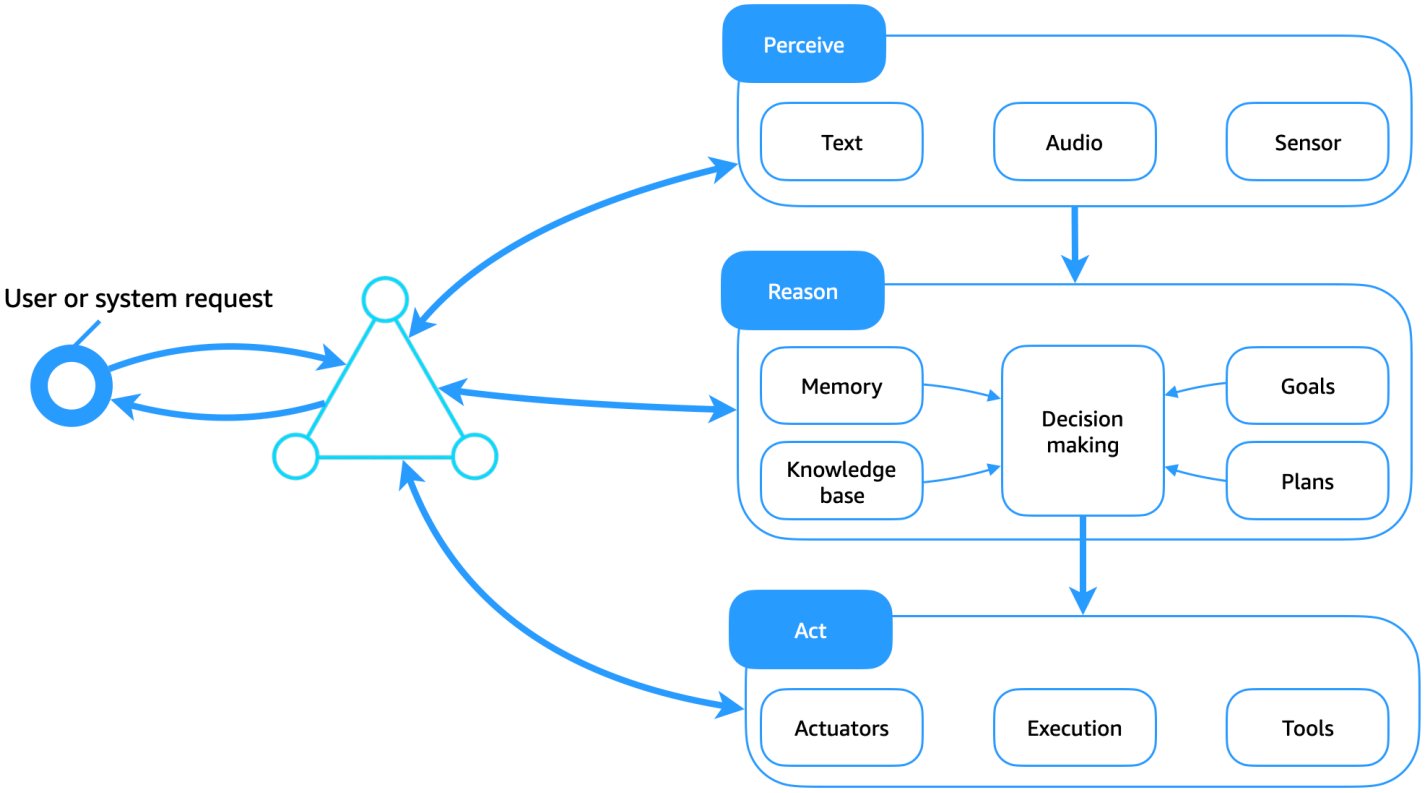
- Verstärktes Lernen: Optimiert das Verhalten durch Versuch und Irrtum, indem die kumulative Belohnung in interaktiven Umgebungen maximiert wird.

  Beispiel: Lernen, welche Strategie zur schnellsten Erledigung der Aufgabe führt.

Das Lernen ist eng mit dem kognitiven Modul des Agenten verknüpft. Es verfeinert die Planungsstrategien auf der Grundlage vergangener Ergebnisse, verbessert die Entscheidungsfindung durch die Bewertung historischer Erfolge und verbessert kontinuierlich die Zuordnung zwischen Wahrnehmung und Handlung. Durch diesen geschlossenen Lern- und Feedbackkreislauf entwickeln sich die Akteure über die reaktive Ausführung hinaus zu sich selbst verbessernden Systemen, die sich im Laufe der Zeit an neue Ziele, Bedingungen und Kontexte anpassen können.

Traditionelle Agentenarchitektur: wahrnehmen, argumentieren, handeln

Das folgende Diagramm veranschaulicht, wie die im vorherigen Abschnitt erörterten Bausteine im Rahmen des Zyklus „Wahrnehmen, Begründen, Handeln“ funktionieren.



Modul Perceive

Das Wahrnehmungsmodul fungiert als sensorische Schnittstelle des Agenten zur Außenwelt. Es wandelt rohe Umwelteinflüsse in strukturierte Repräsentationen um, die als Grundlage für das Denken dienen. Dazu gehört der Umgang mit multimodalen Daten wie Text-, Audio- oder Sensorsignalen.

- Die Texteingabe kann über Benutzerbefehle, Dokumente oder Dialoge erfolgen.
- Die Audioeingabe umfasst gesprochene Anweisungen oder Umgebungsgeräusche.
- Der Sensoreingang erfasst reale Signale wie Bewegung, visuelle Feeds oder GPS.

Wenn die Rohdaten aufgenommen wurden, führt der Wahrnehmungsprozess eine Merkmalsextraktion durch, gefolgt von Objekt- oder Ereigniserkennung und semantischer Interpretation, um ein aussagekräftiges Modell der aktuellen Situation zu erstellen. Diese Ergebnisse bieten einen strukturierten Kontext für nachfolgende Entscheidungen und verankern die Argumentation des Agenten in realen Beobachtungen.

Modul „Grund“

Das Reason-Modul ist der kognitive Kern des Agenten. Es bewertet den Kontext, formuliert die Absicht und legt geeignete Maßnahmen fest. Dieses Modul orchestriert zielorientiertes Verhalten, indem es sowohl erlerntes Wissen als auch Argumentation nutzt.

Das Reason-Modul besteht aus eng integrierten Submodulen:

- Gedächtnis: Behält den Dialogstatus, den Aufgabenkontext und den episodischen Verlauf sowohl kurz- als auch langfristig bei.
- Wissensdatenbank: Bietet Zugriff auf symbolische Regeln, Ontologien oder erlernte Modelle (wie Einbettungen, Fakten und Richtlinien).
- Ziele und Pläne: Definiert die gewünschten Ergebnisse und entwickelt Handlungsstrategien, um diese zu erreichen. Ziele können dynamisch aktualisiert werden und Pläne können auf der Grundlage von Feedback adaptiv geändert werden.
- Entscheidungsfindung: Dient als zentrale Schiedsinstanz, indem es Optionen abwägt, Kompromisse bewertet und die nächste Maßnahme auswählt. Dieses Untermodul berücksichtigt Konfidenzschwellen, Zielausrichtung und kontextuelle Einschränkungen.

Zusammen ermöglichen diese Komponenten dem Agenten, über seine Umgebung nachzudenken, Überzeugungen zu aktualisieren, Pfade auszuwählen und sich kohärent und anpassungsfähig zu verhalten. Das Modul Vernunft schließt die Lücke zwischen Wahrnehmung und Verhalten.

Modul Handeln

Das Act-Modul führt die vom Agenten gewählte Entscheidung aus, indem es entweder eine Schnittstelle zur digitalen oder zur physischen Umgebung herstellt, um Aufgaben auszuführen. Hier wird aus Absicht Handlung.

Dieses Modul umfasst drei Funktionskanäle:

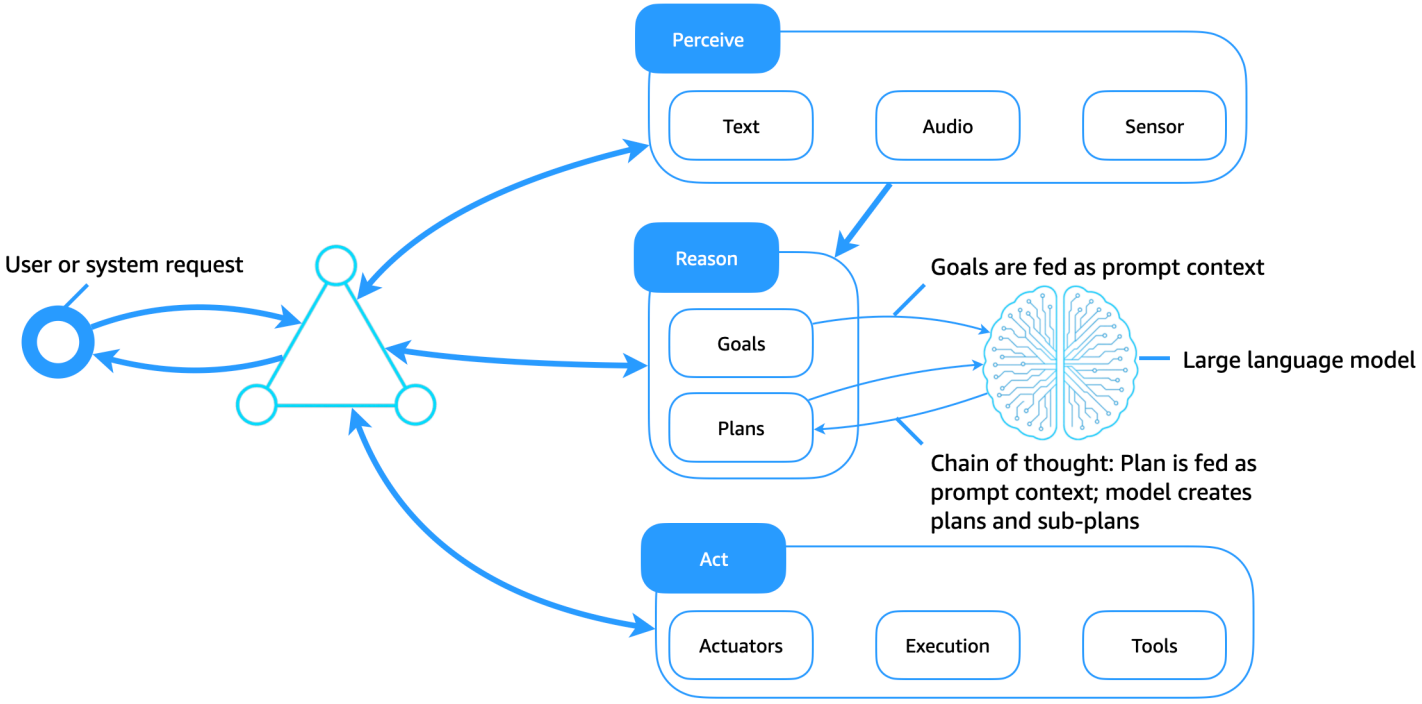
- Aktuatoren: Steuert für Agenten, die physisch präsent sind (wie Roboter und IoT-Geräte), Interaktionen auf Hardwareebene wie Bewegung, Manipulation oder Signalisierung.
- Ausführung: Verwaltet softwarebasierte Aktionen, einschließlich des Aufrufs und Versendens von Befehlen APIs und der Aktualisierung von Systemen.

- Tools: Ermöglicht funktionale Funktionen wie Suche, Zusammenfassung, Codeausführung, Berechnung und Dokumentenverarbeitung. Diese Tools sind oft dynamisch und kontextsensitiv, was den Nutzen des Agenten erweitert.

Die Ausgänge des Act-Moduls werden in die Umgebung zurückgespeist und schließen den Regelkreis. Diese Ergebnisse werden vom Agenten erneut wahrgenommen. Sie aktualisieren den internen Zustand des Agenten und beeinflussen future Entscheidungen, wodurch der Zyklus „Wahrnehmen, Begründen, Handeln“ abgeschlossen wird.

Generative KI-Agenten: Ersetzen symbolischer Logik durch LLMs

Das folgende Diagramm zeigt, wie große Sprachmodelle (LLMs) heute als flexibler und intelligenter kognitiver Kern für Softwareagenten dienen. Im Gegensatz zu herkömmlichen symbolischen Logiksystemen, die auf statischen Planbibliotheken und handcodierten Regeln basieren, LLMs ermöglichen sie adaptives Denken, kontextuelle Planung und dynamischen Einsatz von Tools, die die Art und Weise verändern, wie Agenten wahrnehmen, argumentieren und handeln.



Die wichtigsten Verbesserungen

Diese Architektur erweitert die herkömmliche Agentenarchitektur wie folgt:

- LLMs als kognitive Engines: Ziele, Pläne und Abfragen werden als prompter Kontext an das Modell übergeben. Das LLM generiert Argumentationswege (z. B. Gedankenketten), zerlegt Aufgaben in Unterziele und entscheidet über die nächsten Maßnahmen.
- Einsatz von Tools durch Eingabeaufforderungen: LLMs kann durch Tools oder durch Reasoning and Acting (ReAct) gesteuert werden, die dazu auffordern, Ergebnisse aufzurufen und zu suchen, abzufragen, zu berechnen APIs und zu interpretieren.
- Kontextsensitive Planung: Agenten generieren oder überarbeiten Pläne dynamisch auf der Grundlage des aktuellen Ziels, der Eingabeumgebung und des Feedbacks des Agenten, ohne dass fest codierte Planbibliotheken erforderlich sind.
- Prompt-Kontext als Speicher: Anstatt symbolische Wissensdatenbanken zu verwenden, kodieren Agenten Speicher, Pläne und Ziele als Eingabeaufforderungs-Token, die an das Modell übergeben werden.
- Lernen durch gezieltes, kontextbezogenes Lernen: LLMs Passen Sie Verhaltensweisen durch promptes Engineering an, wodurch die Notwendigkeit expliziter Umschulungen oder starre Planbibliotheken reduziert wird.

Erreichen des Langzeitgedächtnisses bei auf LLM basierenden Wirkstoffen

Im Gegensatz zu herkömmlichen Agenten, die das Langzeitgedächtnis in strukturierten Wissensdatenbanken speicherten, müssen generative KI-Agenten innerhalb der Einschränkungen des Kontextfensters von arbeiten. LLMs Um das Gedächtnis zu erweitern und persistente Intelligenz zu unterstützen, verwenden generative KI-Agenten mehrere sich ergänzende Techniken: Agentenspeicher, Retrieval-Augmented Generation (RAG), kontextinternes Lernen und Prompt-Chaining sowie Vortraining.

Agentenspeicher: externes Langzeitgedächtnis

Agentenstatus, Benutzerverlauf, Entscheidungen und Ergebnisse werden in einem langfristigen Agentenspeicher (z. B. einer Vektordatenbank, einem Objektspeicher oder einem Dokumentenspeicher) gespeichert. Relevante Erinnerungen werden bei Bedarf abgerufen und zur Laufzeit in den LLM-Prompt-Kontext eingefügt. Dadurch entsteht eine persistente Speicherschleife, in der der Agent die Kontinuität zwischen Sitzungen, Aufgaben oder Interaktionen beibehält.

LAPPEN

RAG verbessert die Leistung von LLM, indem es abgerufenes Wissen mit generativen Fähigkeiten kombiniert. Wenn ein Ziel oder eine Abfrage ausgegeben wird, durchsucht der Agent einen

Abrufindex (z. B. durch eine semantische Suche nach Dokumenten, früheren Konversationen oder strukturiertem Wissen). Die abgerufenen Ergebnisse werden an die LLM-Eingabeaufforderung angehängt, wodurch die Generierung auf externen Fakten oder einem personalisierten Kontext basiert. Diese Methode erweitert das effektive Gedächtnis des Agenten und verbessert die Zuverlässigkeit und sachliche Richtigkeit.

Kontextbezogenes Lernen und schnelle Verkettung

Agenten pflegen das Kurzzeitgedächtnis, indem sie den Token-Kontext der Sitzung und die strukturierte Verkettung von Eingabeaufforderungen verwenden. Kontextuelle Elemente wie der aktuelle Plan, die Ergebnisse früherer Aktionen und der Agentenstatus werden zwischen Aufrufen weitergegeben, um das Verhalten zu steuern.

Kontinuierliche Vorschulung und Feinabstimmung

Für domänenspezifische Agenten LLMs kann die Vorschulung mit benutzerdefinierten Sammlungen wie Protokollen, Unternehmensdaten oder Produktdokumentationen fortgesetzt werden. Alternativ kann durch Feinabstimmung von Anweisungen oder Reinforcement-Learning auf Basis von menschlichem Feedback (RLHF) agentenähnliches Verhalten direkt in das Modell integriert werden. Dadurch werden die Argumentationsmuster von der Prompt-Time-Logik in die interne Repräsentation des Modells verlagert, die Länge der Eingabeaufforderungen reduziert und die Effizienz verbessert.

Kombinierte Vorteile agentischer KI

Wenn diese Techniken zusammen verwendet werden, ermöglichen es generativen KI-Agenten:

- Behalten Sie das kontextuelle Bewusstsein im Laufe der Zeit bei.
- Passen Sie das Verhalten auf der Grundlage der Benutzerhistorie oder der Präferenzen an.
- Treffen Sie Entscheidungen up-to-date, indem Sie Faktenwissen oder privates Wissen nutzen.
- Skalieren Sie Ihr Verhalten auf Unternehmensanwendungsfälle mit beständigem, regelkonformem und erklärbarem Verhalten.

Durch die Erweiterung LLMs mit externem Speicher, Abrufebenen und kontinuierlicher Schulung können Agenten ein Maß an kognitiver Kontinuität und Zielsetzung erreichen, das zuvor mit symbolischen Systemen allein nicht erreicht werden konnte.

Vergleich herkömmlicher KI mit Softwareagenten und agentischer KI

Die folgende Tabelle bietet einen detaillierten Vergleich von herkömmlicher KI, Softwareagenten und agentischer KI.

Merkmal	Traditionelle KI	Softwareagenten	Agentische KI
Beispiele	Spam-Filter, Bildklassifikatoren, Empfehlungsmaschinen	Chatbots, Taskplaner, Überwachungsagenten	KI-Assistenten, autonome Entwickleragenten, LLM-Orchestrierungen mit mehreren Agenten
Ausführungsmodell	Batch oder synchron	Ereignisgesteuert oder geplant	Asynchron, ereignisgesteuert und zielgesteuert
Autonomie	Eingeschränkt; erfordert oft menschliche oder externe Orchestrierung	Mittel; arbeitet unabhängig innerhalb vordefinierter Grenzen	Hoch; agiert unabhängig mit adaptiven Strategien
Reaktivität	Reaktiv auf Eingabedaten	Reagiert auf Umgebung und Ereignisse	Reaktiv und proaktiv; antizipiert Maßnahmen und leitet sie ein
Proaktivität	Selten	In einigen Systemen vorhanden	Kernattribut; fördert zielgerichtetes Verhalten
Kommunikation	Minimal; normalerweise eigenständig oder API-gebunden	Nachrichtenübermittlung zwischen Agenten oder zwischen Agenten und Mitarbeitern	Umfangreiche Interaktion mit mehreren Agenten human-in-the-loop

Merkmal	Traditionelle KI	Softwareagenten	Agentische KI
Entscheidungsfindung	Nur Modellinferenz (Klassifizierung, Vorhersage usw.)	Symbolisches Denken oder regelbasierte oder skriptbasierte Entscheidungen	Kontextuelles, zielorientiertes, dynamisches Denken (oft LLM-unterstützt)
Delegierte Absicht	Nein, führt direkt vom Benutzer definierte Aufgaben aus	Teilweise; handelt im Namen von Benutzern oder Systemen mit begrenztem Anwendungsbereich	Ja, handelt mit delegierten Zielen, oft dienst-, benutzer- oder systemübergreifend
Lernen und Anpassung	Oft modellzentriert (z. B. ML-Training)	Manchmal adaptiv	Integriertes Lernen, Gedächtnis oder Argumentation (z. B. Feedback, Selbstkorrektur)
Agentur	Keine; Werkzeuge für Menschen	Implizit oder grundlegend	Explizit; arbeitet zielgerichtet, zielgerichtet und selbstbestimmt
Bewusstsein für den Kontext	Niedrig; zustandslos oder auf Snapshot-Basis	Mäßig; ein gewisses Maß an Statusverfolgung	Hoch; verwendet Speicher-, Situationskontext- und Umgebungsmodelle
Infrastrukturrolle	Eingebettet in Apps oder Analyse-Pipelines	Middleware- oder Service Layer-Komponente	Zusammensetzbares Agent-Mesh, das in Cloud-, Serverless- oder Edge-Systeme integriert ist

Zusammenfassend:

- Herkömmliche KI ist toolzentriert und funktionell eng gefasst. Sie konzentriert sich auf die Vorhersage oder Klassifizierung.
- Herkömmliche Softwareagenten ermöglichen Autonomie und grundlegende Kommunikation, sind aber oft regelgebunden oder statisch.
- Agentic AI vereint Autonomie, Asynchronität und Entscheidungsfreiheit. Sie ermöglicht intelligente, zielorientierte Einheiten, die in komplexen Systemen vernünftig denken, handeln und sich anpassen können. Dies macht agentic AI ideal für die Cloud-native, KI-gesteuerte future.

Nächste Schritte

In diesem Leitfaden wurden die Geschichte und die Grundlagen der agentischen KI erörtert. Dabei handelt es sich um die Entwicklung traditioneller Softwareagenten hin zu autonomen, intelligenten Systemen, die auf generativer KI basieren. Es wurde beschrieben, wie frühe Softwareagenten vordefinierten Regeln und Logiken folgten, um Aufgaben innerhalb fester Grenzen zu automatisieren, und es wurde erklärt, wie agentische KI auf dieser Grundlage aufbaut, indem sie umfangreiche Sprachmodelle einbezieht, die es Agenten ermöglichen, in offenen Umgebungen zu argumentieren, zu lernen und sich dynamisch anzupassen.

Sie können sich eingehend mit agentischer KI befassen, indem Sie die folgenden Veröffentlichungen dieser Reihe lesen:

- Die Operationalisierung von KI auf Agenturebene AWS bietet eine organisatorische Strategie, um die KI von isolierten Experimenten in eine wertschöpfende Infrastruktur auf Unternehmensebene umzuwandeln.
- Agentic AI Patterns and Workflows AWS erläutert die grundlegenden Pläne und modularen Konstrukte, die für die Entwicklung, Zusammenstellung und Orchestrierung zielorientierter KI-Agenten verwendet werden.
- Die KI-Frameworks, -Protokolle und -Tools von Agentic behandeln die Software-Grundlagen, Toolkits und AWS Protokolle, die Sie bei der Entwicklung Ihrer KI-Lösungen für Agenturen berücksichtigen sollten.
- Der Aufbau serverloser Architekturen für agentic AI on AWS behandelt serverlose Architekturen als natürliche Grundlage moderner KI-Workloads und beschreibt, wie Sie KI-native serverlose Architekturen in der erstellen können. AWS Cloud
- Der Aufbau von Mehrmandantenarchitekturen für agentische KI auf AWS beschreibt den Einsatz von KI-Agenten in Umgebungen mit mehreren Mandanten, einschließlich Überlegungen zum Hosting, Bereitstellungsmodellen und Steuerungsebenen.

Ressourcen

Weitere Informationen zu den in diesem Leitfaden erörterten Konzepten finden Sie in den folgenden Leitfäden und Artikeln.

AWS Verweise

- Amazon-Bedrock-Agenten
- Amazon Q Developer
- Strands Agents SDK

Andere Referenzen

- Hewitt, Carl, Peter Bishop und Richard Steiger. „Ein universeller modularer ACTOR-Formalismus für künstliche Intelligenz.“ Tagungsband der 3. Internationalen Gemeinsamen Konferenz über künstliche Intelligenz (1973): 235-245. https://www.ijcai. org/Proceedings/73/Papers/027B.pdf
- Lesser, Victor R., relevante Veröffentlichungen (siehe vollständige Liste):
  - Lesser, Victor R. und Daniel D. Corkill. „Funktionell genaue, kooperative verteilte Systeme.“ IEEE-Transaktionen über Systeme, Mensch und Kybernetik 11, Nr. 1 (1981): 81-96. https://ieeexplore.ieee. org/abstract/document/4308581
  - Decker, Keith S. und Victor R. Lesser. „Kommunikation im Dienste der Koordination.“ AAAI-Workshop zur Planung agentenübergreifender Kommunikation (1994). https://www.researchgate.net/profile/Victor-Lesser/publication/2768884_Communication_in_the_Service_of_Coordination/links/00b7d51cc2a0750cb4000000/Communication-in-the-Service-of-Coordination.pdf
  - Durfee, Edmund H., Victor R. Lesser und Daniel D. Corkill. „Trends bei der kooperativen verteilten Problemlösung.“ IEEE-Transaktionen zu Wissens- und Datentechnik (1989). http://mas.cs.umass. edu/Documents/ieee-tkde89.pdf
  - Durfee, Edmund H., V.R. Lesser und D.D. Corkill, „Verteilte künstliche Intelligenz“. Kooperation durch Kommunikation in einem verteilten Problemlösungsnetzwerk (1987): 29-58. https://www.academia. edu/download/79885643/durf94_1.pdf
  - Lâasri, Brigitte, Hassan Lâasri, Susan Lander und Victor Lesser. „Ein generisches Modell für intelligente Verhandlungsagenten.“ Internationale Zeitschrift für kooperative Informationssysteme 01, Nr. 02 (1992): 291-317. https://doi.org/10.1142/S0218215792000210

  - Lander, Susan E. und Victor R. Lesser. „Die Rolle von Verhandlungen bei der verteilten Suche unter heterogenen Akteuren verstehen“. IJCAI'93: Tagungsband der 13. gemeinsamen internationalen Konferenz über künstliche Intelligenz (1993): 438-444. https://www.ijcai. org/Proceedings/93-1/Papers/062.pdf
  - Lander, Susan, Victor R. Lesser und Margaret E. Connell. „Strategien zur Konfliktlösung für kooperierende Experten“ CKBS'90: Tagungsband der Internationalen Arbeitskonferenz über kooperierende wissensbasierte Systeme (Oktober 1990): 183-200. https://doi.org/10.1007/978-1-4471-1831-2_10
  - Prasad, M. V. Nagendra, Victor Lesser und Susan E. Lander. „Lernexperimente in einem heterogenen Multiagentensystem.“ IJCAI-95-Workshop über Anpassung und Lernen in Systemen mit mehreren Agenten (1995): 59-64. https://www.researchgate.net/publication/2784280_Learning_Experiments_in_a_Heterogeneous_Multi-agent_System
- Nwana, Hyacinth S. „Softwareagenten: Ein Überblick“. Knowledge Engineering Review 11, Nr. 3 (Oktober/November 1996): 205-244. https://teaching.shu.ac. uk/aces/rh1/elearning/multiagents/introduction/nwana.pdf
- Selfridge, Oliver G. „Pandemonium: Ein Paradigma für das Lernen“. Mechanisierung von Denkprozessen: Tagungsband eines Symposiums im National Physical Laboratory 1 (1959): 511—529. https://aitopics. org/download/classics. ----sep----:504E1BAC
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser und Illia Polosukhin. „Aufmerksamkeit ist alles was du brauchst.“ Tagungsband der 31. Konferenz über neuronale Informationsverarbeitungssysteme (NIPS). Fortschritte in neuronalen Informationsverarbeitungssystemen 30 (2017): 5998-6008. https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf
- Wooldridge, Michael und Nicholas R. Jennings. „Intelligente Agenten: Theorie und Praxis.“ Knowledge Engineering Review 10, Nr. 2 (Januar 1995): 115-152. https://www.cs.cmu.edu/ ~ _intelligent_agents.pdf motionplanning/papers/sbp_papers/integrated1/woodridge

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.