Azure AI-Foundry

Die Reise durch das KI-Orchester!

1. Zielbild und Positionierung

Azure AI Foundry ist ein plattformbasierter Dienst (PaaS) für Enterprise-KI mit einem klaren Fokus auf agentenbasierten KI-Systemen und generativen KI-Anwendungen. Das Zielbild beschreibt eine integrierte Entwicklungs- und Betriebsumgebung, die schnelle Iterationszyklen ermöglicht und gleichzeitig die strengen Anforderungen an Produktionstauglichkeit, Sicherheit und Governance umfassend erfüllt. Im Gegensatz zu isolierten Modell-APIs deckt AI Foundry den vollständigen Lebenszyklus agentenbasierter KI-Anwendungen ab: von Entwurf, Entwicklung und Evaluierung über Validierung und Absicherung bis hin zum nahtlosen Monitoring im produktiven Betrieb – und zwar durchgängig innerhalb einer einheitlichen, konsistenten Plattform.

Positionierung im Azure-Ökosystem

Azure AI Foundry ist die konsequente Weiterentwicklung und zugleich das Rebranding der zuvor bekannten Plattform Azure AI Studio. Im Zentrum steht die Zusammenführung bislang getrennter Dienste wie Azure OpenAI Service, Azure AI Search, Azure AI Content Safety sowie weiterer spezialisierter KI-Komponenten in einer integrierten Oberfläche. Dabei werden alle Dienste über einheitliche und klar definierte APIs zugänglich gemacht, was eine konsistente Nutzung und einfache Integration in bestehende IT-Landschaften ermöglicht.

Der umfassende Modellkatalog von Azure AI Foundry umfasst eine breite Palette etablierter KI-Modelle sowohl aus dem Microsoft-eigenen Portfolio als auch aus dem Open-Source-Ökosystem. Darunter fallen namhafte Modelle und Anbieter wie OpenAI (z.B. GPT-Modelle), Meta (z.B. Llama-Modelle), Mistral AI sowie Microsoft-eigene Entwicklungen wie Phi. Ergänzt wird dieses Angebot durch Modelle ausgewählter Partner und aktiver Communities. Zu beachten ist, dass die konkrete Verfügbarkeit einzelner Modelle je nach Region, Anbieter und regulatorischen Anforderungen variieren kann.

Unabhängig von der jeweiligen Herkunft der Modelle (Azure-gehostet oder externe Partner) stellt Azure AI Foundry einheitliche Schnittstellen bereit, was den Entwicklungs- und Wartungsaufwand deutlich reduziert und eine unkomplizierte Skalierung sowie Portabilität der Lösungen ermöglicht. Durch diesen integrierten Ansatz adressiert Azure AI Foundry gezielt die Bedürfnisse von Unternehmen, die eine robuste, skalierbare und zugleich flexible KI-Infrastruktur für den produktiven Einsatz benötigen.

Sicherheits- und Identitätsfunktionen sind in Azure AI Foundry tief mit zentralen Diensten wie Microsoft Entra ID, rollenbasierter Zugriffskontrolle (RBAC), Netzwerkisolation und Azure Policy integriert. Die konsistente Verwaltung aller Dienste über Azure Resource Manager sowie einheitliche Richtlinienmodelle reduziert Komplexität und erleichtert sowohl den Betrieb als auch die Governance.

Azure AI Foundry lässt sich nahtlos in bestehende Unternehmens-Workflows integrieren, insbesondere auch in Microsoft-365-Anwendungen. Darüber hinaus können individuelle Lösungen und KI-Funktionen über klar definierte APIs, den Microsoft Graph sowie geeignete Konnektoren direkt in etablierte Arbeitsabläufe eingebunden werden. Dadurch gelangen maßgeschneiderte KI-Funktionalitäten unmittelbar in vertraute Arbeitsumgebungen – bei gleichzeitiger Einhaltung bestehender Identitäts- und Berechtigungsstrukturen.

Ein strategisch bedeutsamer Bestandteil von Azure AI Foundry ist das Microsoft Agent Framework, das seit dem 1. Oktober 2025 als Preview verfügbar ist. Dieses Framework kombiniert zentrale Elemente von Semantic Kernel und AutoGen zu einer konsolidierten, produktionsnahen Agenten-Plattform. Ergänzend hierzu unterstützt der Azure AI Foundry Agent Service offene Standards und Interoperabilität, beispielsweise durch die Integration des Model Context Protocol (MCP), das seit Juni 2025 unterstützt wird. Zudem ermöglicht die Plattform die systemübergreifende Koordination verteilter Agenten durch Agent-zu-Agent-Zusammenarbeit (Agent-to-Agent, A2A).

Azure AI Foundry ist aktuell noch sehr dynamisch, es entwickelt sich kontinuierlich weiter und es werden regelmäßig neue innovative Funktionen bereitgestellt. Entwicklungsteams profitieren dadurch frühzeitig von neuen Technologien und Konzepten, die sie unter kontrollierten Rahmenbedingungen evaluieren und gezielt einsetzen können.

Schnelle Iteration und sichere Produktivsetzung

Mit Azure AI Foundry beschleunigen Teams signifikant den Weg von der Idee über den Prototyp bis hin zur produktiven Anwendung. Die integrierten Experimentierumgebungen („Playgrounds“) ermöglichen schnelles Prototyping und das Erstellen von POC ohne zusätzlichen Einrichtungsaufwand und erleichtern eine unmittelbare API-Erkundung und Nutzung. Die Responses-API bündelt dialogorientierte Interaktionen, Tool-Aufrufe und strukturierte Ausgaben in einem konsistenten und einheitlichen Schema.

Der Model Router (zu den später ausführlicher) wählt zur Laufzeit automatisch das am besten geeignete Modell auf Grundlage konfigurierbarer Qualitäts- und Kostenziele aus. Damit entsteht eine leistungsfähige und gleichzeitig kosteneffiziente Anwendungslösung, die flexibel auf unterschiedliche Anforderungen reagiert.

Umfassende SDKs für Python, .NET, JavaScript/TypeScript sowie Java decken den gesamten Lebenszyklus von Anwendungen und Agenten ab. Eine spezielle Erweiterung für VSCode erleichtert die Verwaltung von Projekten, deren Bereitstellungen sowie Konfigurationen und ermöglicht eine direkte Veröffentlichung aus der Entwicklungsumgebung heraus. Dies garantiert einen nahtlosen Workflow von der Entwicklung bis zur finalen Produktivsetzung.

Sicherheit und Qualität

Azure AI Content Safety bietet mehrschichtige Schutzmaßnahmen, darunter Inhaltsfilter und Klassifizierer, die toxische, gewaltverherrlichende oder anderweitig sensible Inhalte zuverlässig identifizieren und blockieren. Zudem schützen speziell entwickelte Mechanismen vor Prompt-Injection-Angriffen. Diese Schutzebene lässt sich nahtlos in Azure AI Foundry-Workflows integrieren und unterstützt Unternehmen bei der Einhaltung umfangreicher Sicherheits- und Compliance-Vorgaben.

Beobachtbarkeit und Betrieb

Azure AI Foundry ermöglicht umfassende Transparenz und Überwachung durch integrierte Ablaufverfolgung und Metriken, die über Azure Monitor und Application Insights bereitgestellt werden. Die Observability-Funktionen bieten kontinuierliche Laufzeitanalysen hinsichtlich Qualität, Sicherheit und Leistung. Zusätzlich gewährleistet Microsoft Defender for Cloud mit einem spezialisierten AI-Workloads-Plan effektiven Schutz, indem KI-spezifische Bedrohungen erkannt und Sicherheitslücken frühzeitig identifiziert werden.

Wissen und Datenanbindung

Azure AI Foundry-Lösungen lassen sich flexibel und sicher mit privaten, lizenzierten sowie öffentlichen Datenquellen verbinden. Wesentliche Komponenten sind unter anderem Azure AI Search für Retrieval-Augmented Generation (RAG)-Szenarien und die Integration mit bestehenden Datenplattformen wie Microsoft Fabric oder SharePoint. Sensible Zugangsdaten und Geheimnisse werden über Azure Key Vault geschützt und sicher verwaltet, wodurch eine Speicherung im Klartext ausgeschlossen wird.

Fazit

Azure AI Foundry fungiert als umfassende, integrierte Entwicklungs- und Betriebsplattform für KI-Lösungen. Die Plattform begleitet den gesamten Wertschöpfungsprozess von der Datenbereitstellung und Modellauswahl über Entwicklung, Evaluierung sowie umfassende Sicherheits- und Governance-Maßnahmen bis hin zum gehärteten Betrieb durch Monitoring, Richtliniensteuerung und aktive Schutzmaßnahmen. Ergebnis sind leistungsfähige, interoperable und robuste agentenbasierte KI-Systeme, die optimal in bestehende Unternehmensabläufe und Kontrollmechanismen integriert werden können und sowohl technischen als auch regulatorischen Anforderungen gerecht werden.

2. Architektur und Ressourcenmodell

Azure AI Foundry etabliert ein konsistentes und strukturiertes Ressourcen- und Architekturmodell, das sowohl organisatorische Trennung als auch technische Flexibilität unterstützt und unterschiedlichste Sicherheitsanforderungen berücksichtigt. Kernkomponenten sind die Azure-AI-Foundry-Ressource sowie einzelne Foundry-Projekte. Parallel dazu existiert ein Hub-basiertes Modell mit Azure AI Hubs und Hub-Projekten, das aus der Weiterentwicklung der Azure Machine Learning Plattform entstanden ist. Diese Architektur verbindet Governance-Anforderungen mit Entwicklerfreiheit, ohne die Agilität der einzelnen Teams einzuschränken.

Modernes Modell: Foundry-Ressource und Foundry-Projekte

Die Foundry-Ressource bildet die zentrale Verwaltungsinstanz für KI-Workloads und definiert den Geltungsbereich für wesentliche Plattformaspekte. Hierüber erfolgt die Verwaltung von Identitäten und Zugriffen mittels Microsoft Entra ID und verwaltete Identitäten (Managed Identities). Zudem werden Netzwerkrichtlinien durch Private Endpoints sowie verwaltete virtuelle Netzwerke konsequent umgesetzt. Die Foundry-Ressource regelt zudem Sicherheitskonfigurationen und Abrechnung und wird technisch als Azure-Ressource vom Typ „`Microsoft.CognitiveServices accounts`“ mit der Art-Kennzeichnung „`AIServices`“ realisiert. Innerhalb dieses Rahmens stehen zentrale Dienste wie Modelle, Agenten, Evaluationen sowie spezifische Azure-AI-Services für Sprach-, Bild- und Textverarbeitung gebündelt zur Verfügung.

Foundry-Projekte stellen Unterressourcen der zentralen Foundry-Ressource dar und dienen als isolierte Arbeitsbereiche einzelner Entwicklungsteams. Sie übernehmen grundlegende Vorgaben der übergeordneten Ressource, erlauben jedoch projektspezifische Anpassungen. Diese Struktur gewährleistet eine einheitliche Durchsetzung zentraler Sicherheitsrichtlinien, während Entwicklungsteams gleichzeitig über maximale Autonomie innerhalb ihrer Projekte verfügen. Zu typischen Assets eines Projekts gehören Modelle aus dem Foundry-Katalog, standardisierte Bereitstellungen von Azure OpenAI-Modellen, Agenten einschließlich Workflows mit Unterstützung des Model Context Protocol (MCP), Evaluationen mit integrierten Metriken sowie REST-basierte Endpunkte und Bereitstellungen.

Die klare Trennung von Management- und Entwicklungsoperationen folgt dem Prinzip der sicherheitsgetriebenen Zuständigkeitstrennung. Identitäten, Netzwerkisolation, Richtlinien und Konnektivität werden zentral auf Ebene der Foundry-Ressource verwaltet, während alle Entwicklungsaktivitäten eigenständig innerhalb der einzelnen Projekte erfolgen. Praktisch bedeutet dies, dass IT-Abteilungen Sicherheitsvorgaben zentral definieren und überwachen, während Entwicklerteams ungestört an ihren spezifischen KI-Lösungen arbeiten können. Standardmäßig

werden Secrets sicher in einem von Microsoft verwalteten Azure Key Vault gespeichert. Organisationen mit individuellen Compliance-Anforderungen können optional einen eigenen Azure Key Vault anbinden, um die volle Kontrolle über ihre sensiblen Zugangsdaten zu gewährleisten.

Paralleles Modell: Azure AI Hubs

Zusätzlich zum Foundry-Ressourcenmodell besteht ein paralleles Hub-basiertes Modell, das auf dem Azure-Ressourcentyp „`Microsoft.MachineLearningServices workspaces`" basiert. Ein Azure AI Hub vereint mehrere Hub-Projekte und ermöglicht gemeinsame Konfigurationen und zentrale Verwaltung. Üblicherweise arbeiten Hubs zusammen mit einer Foundry-Ressource, um den Zugriff auf den zentralen Modellkatalog sowie auf eine einheitliche Inferenz-API zu gewährleisten.

Für den Betrieb eines Azure AI Hubs sind zusätzliche Ressourcen erforderlich, insbesondere ein Storage Account und ein Azure Key Vault. Optional kann zudem eine Azure Container Registry angebunden werden. Hub-Strukturen eignen sich besonders für Szenarien, in denen zahlreiche Teams unter einheitlichen Vorgaben und standardisierten Rahmenbedingungen arbeiten sollen oder spezifische Anforderungen erfüllt werden müssen. Beispiele hierfür sind erweitertes Hosting von Open-Source-Modellen mit dedizierten Rechenressourcen oder umfassende Verwaltung komplexer Prompt-Flow-Orchestrierungsszenarien.

Microsoft betont jedoch, dass das Funktionsspektrum von Foundry-Projekten kontinuierlich erweitert wird und das Hub-Modell perspektivisch ablösen soll. Somit wird empfohlen, zukünftige Implementierungen bevorzugt auf Basis von Foundry-Projekten zu realisieren, um von langfristiger Kontinuität und erweiterten Funktionalitäten zu profitieren.

Empfehlung und Migrationspfad

Für neue Implementierungen empfiehlt sich die Verwendung der Foundry-Ressource mit dazugehörigen Foundry-Projekten als primäres und vereinfachtes Architekturmodell. Dieser Ansatz reduziert Infrastrukturkomplexität deutlich, da weniger externe Abhängigkeiten erforderlich sind und die Integration nativer erfolgt. Bestehende Hub-Installationen können weiterhin genutzt werden und erfüllen unverändert ihre Funktion. Jedoch eröffnet eine schrittweise Migration auf das Foundry-Modell den Zugang zu fortschrittlichen Funktionen wie agentenbasierten Anwendungen mit MCP-Unterstützung, erweiterten Evaluationswerkzeugen sowie einheitlicheren und umfassenderen Monitoring-Optionen.

Ein direkter Wechsel der Art-Kennzeichnung bestehender Azure OpenAI-Ressourcen ist technisch nicht möglich. Stattdessen wird empfohlen, eine neue Azure-Ressource vom Typ

„Microsoft.CognitiveServices accounts“ mit der spezifischen Art-Kennzeichnung „AIServices“ zu erstellen. Die Migration umfasst dabei die geplante Übertragung vorhandener Bereitstellungen sowie Anpassungen der Client-Konfigurationen. Dieser Prozess sollte sorgfältig mit ausreichend zeitlichem Vorlauf geplant und durchgeführt werden, um eine möglichst unterbrechungsfreie und reibungslose Übergangsphase sicherzustellen.

Endpunkte in Azure AI Foundry

Endpunkte fungieren als zentrale Schnittstellen, über die Anwendungen, Agenten und Nutzer auf Modelle zugreifen können. Sie abstrahieren dabei die zugrunde liegende Infrastruktur und bieten standardisierte APIs für eine konsistente und einfache Nutzung. Azure AI Foundry stellt hierzu zwei wesentliche Endpunkttypen bereit, die jeweils spezifisch für unterschiedliche Anwendungsfälle optimiert sind.

Azure AI Inference Endpoint (Einheitlicher Inferenz-Endpunkt)

Der Azure AI Inference Endpoint bietet ein einheitliches API-Schema für eine Vielzahl von Modellen an. Ziel dieses Endpunkts ist es, modellübergreifende Aufrufe mit minimalen Anpassungen am Code zu ermöglichen. Das Routing auf das jeweilige Zielmodell erfolgt hierbei konfigurationsbasiert, etwa über spezifische Modellbezeichner im API-Request. Die Authentifizierung ist flexibel gestaltet und erfolgt entweder mittels API-Schlüsseln oder schlüssellos über Microsoft Entra und Managed Identities. Unterstützt werden insbesondere Anwendungsfälle wie Text Embeddings, Image Embeddings und Chat Completions.

Azure OpenAI Endpoint

Der Azure OpenAI Endpoint fokussiert sich speziell auf OpenAI-Modelle und stellt erweiterte API-Funktionen zur Verfügung, die über die standardmäßigen Inferenzfähigkeiten hinausgehen. Zu den wesentlichen erweiterten Funktionen zählen die Assistants-API für dialogorientierte Anwendungen, Batch-Verarbeitung für umfangreiche Datensätze sowie Echtzeit-Kommunikationsschnittstellen (Realtime APIs). Wichtig ist hierbei, dass Batch-Inferenz und Echtzeit-APIs explizit nicht über den generischen Inference Endpoint verfügbar sind, sondern jeweils eigene, spezifische Deployment-URLs des Azure OpenAI Endpoints benötigen. Alle Nutzungen erfolgen im selben Projektkontext und unterliegen konsistent den etablierten Governance- und Monitoring-Regelungen.

Einordnung und Auswahl der Endpunkttypen

Der Azure AI Inference Endpoint empfiehlt sich als Standard für die meisten Modellaufrufe, insbesondere wenn eine breite und allgemeine Modellunterstützung gefordert ist. Der Azure OpenAI Endpoint sollte genutzt werden, wenn spezifische OpenAI-Funktionen benötigt werden, die über reine Standardinferenz hinausgehen – beispielsweise bei Einsatz der Assistants-API, umfangreicher Batch-

Verarbeitung oder Echtzeit-Kommunikation. Agenten werden ebenfalls über die Assistants-API bereitgestellt, welche exklusiv über den Azure OpenAI Endpoint verfügbar ist. Trotz dieser Spezialisierungen bleibt das API-Erlebnis konsistent und transparent, was die Entwicklung komplexer KI-Lösungen maßgeblich erleichtert.

Bereitstellungsoptionen

Bei der Modellbereitstellung bietet Azure AI Foundry drei primäre Optionen, deren Auswahl von Kriterien wie Latenz, Kosten, Kontrolle und Compliance abhängt. Jede Option adressiert spezifische Anforderungen und Szenarien:

- **Standard-Bereitstellung in Foundry-Ressourcen:** Diese Option nutzt die verwaltete Multi-Tenant-Infrastruktur der Azure AI Foundry-Ressource und unterstützt sowohl regionale Bereitstellungen als auch spezifische Datenresidenz-Vorgaben (Regional, Data-Zone oder Global) zur Einhaltung regulatorischer Compliance-Anforderungen. Das Abrechnungsmodell ist Pay-per-Call und eignet sich für Workloads mit geringem bis mittlerem Volumen und hoher Spitzenlast. Für garantierten Durchsatz und planbare Latenz bei konsistentem, hohem Volumen können stattdessen Bereitgestellte Durchsatzeinheiten (Provisioned Throughput Units, PTUs) mit dedizierten Kapazitäten gebucht werden. Standard-Bereitstellungen sind optimal für produktive Workloads, da sie schlüssellose Authentifizierung mittels Microsoft Entra ID, Private Endpoints, maßgeschneiderte Inhaltsfilterung, tiefgehendes Monitoring, flexible Routing-Optionen und umfassende Sicherheitsfunktionen bieten. Sie gelten als bevorzugte Option für neue und produktionskritische Projekte.
- **Serverlose API-Bereitstellung:** Diese Option steht ausschließlich in hub-basierten Projekten zur Verfügung und bietet vollständig verwaltetes Model-as-a-Service-Hosting ohne eigene Rechenressourcenverwaltung. Die Abrechnung erfolgt nutzungsabhängig, wodurch sie besonders für Experimente, Entwicklungsumgebungen oder Workloads mit stark variierender Last attraktiv ist. Wesentliche Einschränkungen bestehen bei der Authentifizierung (ausschließlich API-Schlüssel, keine Microsoft Entra ID-Unterstützung). Content Filtering ist standardmäßig aktiviert und für Foundry Models optional konfigurierbar. Die Netzwerkkonfiguration (einschließlich Private Endpoints) wird vom übergeordneten Hub vererbt. Zudem gelten spezifische Nutzungslimits pro Modell und Bereitstellung (200.000 Tokens/Min., 1.000 API-Anfragen/Min., eine Bereitstellung pro Modell und Projekt).
- **Verwaltete Bereitstellung auf dedizierten Rechenressourcen:** Diese Option ist ebenfalls ausschließlich in hub-basierten Projekten verfügbar und führt Modelle auf speziell reservierten Rechenressourcen mit expliziten Abonnementquoten aus. Sie ist zwingend erforderlich für spezielle Modellkollektionen wie von Hugging Face, NVIDIA NIMs, Industry Models sowie

benutzerdefinierte Modelle und bietet sich besonders bei spezifischen Hardware-Anforderungen oder zusätzlichem Bedarf an Isolation an. Die Abrechnung erfolgt pro Compute-Uptime plus ggf. modellspezifische GPU-Stundengebühren des Anbieters. Für neue, produktiv eingesetzte Foundry-Projekte empfiehlt sich dennoch primär die Standard-Bereitstellung, da diese optimal in das moderne Ressourcen- und Sicherheitsmodell integriert ist und die umfangreichsten Funktionen bietet.

Praxisempfehlung

Für neue Projekte wird die Standard-Bereitstellung in Foundry-Ressourcen empfohlen. Sie bietet die meisten Funktionen, beste Integration und ist die zukunftsorientierte Wahl. Produktionsmodelle sollten Standard-Bereitstellungen nutzen, optional mit PTU für kritische Anwendungen mit hohen Durchsatzanforderungen. Experimente und Szenarien mit unvorhersehbaren Lasten können von serverloser Bereitstellung im Hub-Modell profitieren, müssen aber die Einschränkung akzeptieren, dass nur API-Schlüssel-Authentifizierung verfügbar ist (keine Microsoft Entra ID). Spezielle Anforderungen wie Hugging Face oder NVIDIA NIMs erfordern verwaltete Rechenressourcen im Hub-Modell. Der Azure AI Inference Endpoint ist die Standard-Wahl für die meisten Aufrufe. Der Azure OpenAI Endpoint ist erforderlich für Assistants-API, Batch-Verarbeitung oder Realtime-Funktionen. Anwendungen und Agenten bleiben dadurch weitgehend entkoppelt von der konkreten Bereitstellungsart, was Flexibilität in der Infrastrukturentwicklung ermöglicht, ohne Anwendungscode ändern zu müssen.

3. Modellkatalog und Routing

Azure AI Foundry bietet einen umfassenden Modellkatalog, der einen zentralen Zugriff auf verschiedenste Anbieter ermöglicht. Für OpenAI-Modelle steht zusätzlich der Model Router (Preview) zur Verfügung, der zur Laufzeit dynamisch das geeignetste Modell aus der OpenAI-Familie auswählt. Der Modellkatalog ist direkt über das Azure-Portal verfügbar und wird stetig erweitert. Aufgrund dieser Vielfalt können Entwicklungsteams für jeden Anwendungsfall das jeweils optimale Modell auswählen, ohne dabei zwischen verschiedenen Plattformen wechseln zu müssen.

Modellkatalog: Umfang und Struktur

Der Katalog umfasst Modelle führender Anbieter wie OpenAI, DeepSeek, Mistral, xAI, Cohere, Meta, NVIDIA und weitere. Er fungiert als zentrale Anlaufstelle für Auswahl, Test und Bereitstellung von KI-Modellen innerhalb der Azure-Infrastruktur. Die Bündelung mehrerer Anbieter unter einer einzigen Oberfläche vereinfacht die Evaluierung erheblich und unterstützt die effiziente Modellauswahl.

Microsoft unterteilt den Katalog in zwei wesentliche Kategorien:

1. **Azure-gehostete Modelle (Foundry Models Sold Directly by Azure):** Diese Modelle werden direkt von Azure bereitgestellt, durch Azure-Servicevereinbarungen unterstützt und vollständig

in die Sicherheits- und Compliance-Infrastruktur von Azure integriert. Dies umfasst Azure OpenAI-Modelle sowie ausgewählte Modelle von Top-Anbietern wie DeepSeek, xAI und Black Forest Labs.

2. **Partner- und Community-Modelle:** Hierbei handelt es sich um Modelle von vertrauenswürdigen Drittanbietern, Partnerorganisationen, Forschungslaboren und Community-Beitragsleistenden. Diese Modelle unterliegen den Lizenzbedingungen der jeweiligen Anbieter, was bei der Architektur- und Projektplanung berücksichtigt werden sollte.

Benutzer haben zudem die Möglichkeit, über das Azure-Portal Bedarf an weiteren Modellen zu melden. Diese Feedback-Funktion ermöglicht eine stetige Anpassung des Angebots an die tatsächlichen Anforderungen und Bedürfnisse der Anwender.

Zusätzlich stehen zahlreiche offene Modelle in kuratierten Sammlungen, beispielsweise von Hugging Face, bereit. Diese Sammlungen erleichtern die Entdeckung und Nutzung verwandter Modelle und Varianten. Spezialisierte Modellgruppen wie NVIDIA NIM-Modelle oder branchenspezifische Lösungen (Industry Models von Saifr, Rockwell, Bayer, Cerence, Sight Machine, Page AI, SDAIA) erfordern verwaltete Rechenkapazitäten innerhalb eines Hub-Projekts. Dies begründet sich durch besondere Hardware- oder spezifische Laufzeitanforderungen, welche die Möglichkeiten der Standardbereitstellung überschreiten.

Azure AI Foundry unterstützt unterschiedliche Deployment-Optionen, um diversen Anforderungen gerecht zu werden: Standard-Deployments für regionale Verfügbarkeit, Global-Deployments für weltweite Abdeckung sowie Data-Zone-Deployments für Szenarien mit spezifischen Datenresidenz-Anforderungen. Diese flexiblen Optionen erlauben es Entwicklungsteams, die optimale Balance zwischen Verfügbarkeit, Latenz und Compliance sicherzustellen.

Einheitliche Inferenz-API und SDKs

Die Azure AI Model Inference API bildet das Fundament für den vereinheitlichten Zugriff auf Modell-Bereitstellungen in Standard-Deployments. Anwendungen können unterschiedliche Modelle über ein konsistentes Schema ansprechen und zwischen Modellen wechseln, indem sie einfach den Deployment-Namen im `model`-Parameter ändern. Diese Abstraktion ist aus architektonischer Sicht besonders wertvoll, da sie eine lose Kopplung zwischen Anwendungslogik und der konkreten Modellauswahl ermöglicht. Entwickler können Modelle austauschen, ohne Änderungen am Anwendungscode vornehmen zu müssen, was die langfristige Wartbarkeit erheblich verbessert. Zu beachten ist aber, dass einige spezialisierte Modelle anbieterspezifische APIs erfordern.

Das Azure AI Foundry SDK deckt die gängigen Entwicklungssprachen ab und bietet Bibliotheken für Python, .NET (C#), JavaScript/TypeScript und Java (Preview). Der Projects-Client bündelt dabei

Zugriffe auf Modelle, Agenten, Evaluierungen und Verbindungen über einen einheitlichen Projekt-Endpunkt. Diese Konsolidierung reduziert die Komplexität der Integration, da Teams nicht mehr mit verschiedenen API-Oberflächen für unterschiedliche Aspekte der Plattform arbeiten müssen.

Modell-Router (Preview)

Der Modell-Router ist eine innovative Komponente, die als einzelnes Modell bereitgestellt werden kann und eingehende Prompts in Echtzeit analysiert. Basierend auf dieser Analyse routet der Router die Anfrage an ein passendes Modell aus seinem verfügbaren Modell-Satz. Das primäre Ziel dieser Architektur ist die Optimierung von Kosten und Latenz bei gleichbleibender Qualität der Antworten. Die Idee dahinter ist elegant: Nicht jede Anfrage benötigt die Rechenleistung des größten verfügbaren Modells. Einfache Anfragen können von kleineren, schnelleren und kostengünstigeren Modellen beantwortet werden, während komplexe Aufgaben automatisch an leistungsfähigere Modelle weitergeleitet werden.

Jede Version des Routers umfasst einen fest definierten Satz unterliegender Modelle. Eigene Änderungen an diesem Modellsatz sind in der Regel nicht vorgesehen. Die Modellauswahl wird von Microsoft kuratiert, um ein ausgewogenes Verhältnis zwischen Kosten, Qualität und Latenz zu gewährleisten. Der konkrete Modellsatz und dessen Verfügbarkeit richten sich nach dem jeweils dokumentierten Produktstand.

Die Schnittstelle des Routers ist bewusst einfach gehalten: Er wird wie ein reguläres Chat-/Responses-Modell über die standardisierte Inferenz-API angesprochen. Das Antwortschema entspricht der Standard-Antwort, ggf. mit Metadaten (z. B. zum ausgewählten Basismodell). Diese Transparenz ermöglicht es Entwicklern, das Routing-Verhalten zu verstehen und zu überwachen, ohne die Einfachheit der Schnittstelle zu beeinträchtigen.

Bei den Parametern und ihren Grenzen gibt es einige wichtige Aspekte zu beachten. Nicht alle Parameter gelten für alle Zielmodelle im Router. Parameter, die ein Zielmodell nicht unterstützt, können ignoriert werden oder zu Fehlern führen – die jeweils gültigen Parameter richten sich nach dem ausgewählten Basismodell und der offiziellen Dokumentation. Das effektiv nutzbare Kontextfenster pro Anfrage hängt ebenfalls vom tatsächlich gewählten Modell ab, bei Überschreitung des unterstützten Kontextfensters ist mit Validierungsfehlern zu rechnen. Multi-Modalität (z. B. Bild- oder Audio-Eingaben) ist abhängig von der Unterstützung des Zielmodells und des jeweiligen Endpunkts.

Für die Abrechnung gilt grundsätzlich: Die Kosten entsprechen den genutzten Inferenz-Diensten und deren Preismodellen. Seit 1. September 2025 werden zusätzlich Router-Gebühren berechnet. Die Details finden sie in der offiziellen Preisdokumentation (inkl. Zuordnung in der Azure-Kostenanalyse).

Für den Betrieb stehen Telemetrie und Monitoring (z. B. über Azure Monitor/Log Analytics) zur Verfügung. Transparenz über das gewählte Basismodell und Metriken wie Latenz, Fehlerquoten und Verteilung der Modellauswahl sind wesentlich, um das Routing-Verhalten zu verstehen und anzupassen. Automatische Aktualisierungen können den Modellmix – und damit Verhalten und Kostenprofil – verändern und sollten in produktiven Umgebungen sorgfältig abgewogen werden.

Aktualisierungsstrategien für Modellversionen

Für viele Modelle lassen sich Aktualisierungsrichtlinien pro Bereitstellung definieren. Die verfügbaren Strategien umfassen die automatische Aktualisierung auf die Standardversion, die Aktualisierung bei Ablauf der aktuellen Version oder den vollständigen Verzicht auf automatische Aktualisierungen. Diese Flexibilität ermöglicht es Teams, präzise zu steuern, ob und wann auf neue Standard- oder Ablöseversionen umgestellt wird. In produktiven Umgebungen empfiehlt sich oft ein konservativer Ansatz ohne automatische Aktualisierungen, um unerwartete Verhaltensänderungen zu vermeiden. Für Entwicklungs- und Testumgebungen können automatische Aktualisierungen hingegen sinnvoll sein, um stets mit den neuesten Modellversionen zu arbeiten.

Strategische Bedeutung für Architektur und Teams

Die Kombination aus umfassendem Modellkatalog, einheitlicher API und intelligentem Routing schafft mehrere strategische Vorteile für Unternehmensarchitekturen. Die einheitliche Inferenz-Schnittstelle reduziert die Abhängigkeit von einzelnen Modellanbietern erheblich. Wenn sich Anwendungen gegen eine abstrakte API-Schicht entwickeln lassen, bleiben Modellwechsel lokal auf die Bereitstellungsreferenz beschränkt. Dies bedeutet in der Praxis, dass Teams ihre Modellstrategie flexibel anpassen können, ohne umfangreiche Änderungen in der Anwendungslogik vornehmen zu müssen.

Der Modell-Router ermöglicht eine ausgeklügelte Kosten- und Qualitätssteuerung, indem er kostengünstige und hochwertige Modelle hinter einem einzelnen Endpunkt bündelt. Dies vereinfacht die Client-Integration erheblich, erfordert aber auch sorgfältige Tests vor der Produktivsetzung. Die Parametergrenzen und das Aktualisierungsverhalten sollten in einer kontrollierten Umgebung gründlich validiert werden, um Überraschungen im Produktivbetrieb zu vermeiden.

Aus Governance-Perspektive ist die Unterscheidung zwischen von Azure direkt bereitgestellten Modellen und Partnermodellen relevant. Modelle aus der ersten Kategorie unterliegen den Azure-Richtlinien für Sicherheit, Datenschutz und Compliance und sind vollständig in die Azure-Governance-Strukturen integriert. Sie unterliegen internationalen Standards wie ISO 27001, SOC 2 Type II, HIPAA, FedRAMP und GDPR. Partnermodelle unterliegen zusätzlich den Konditionen und Lizenzbedingungen

der jeweiligen Anbieter, was bei der Architekturplanung und insbesondere bei der Auswahl von Modellen für regulierte Branchen berücksichtigt werden muss.

Zusammenfassung

Der Modellkatalog von Azure AI Foundry liefert die notwendige Breite an Optionen, verschiedene Deployment-Optionen ermöglichen Flexibilität bei Verfügbarkeit und Compliance, die einheitliche Inferenz-API sorgt für die technische Vereinheitlichung, und der Modell-Router ermöglicht eine intelligente Laufzeitoptimierung. Zusammen bilden diese Komponenten ein durchdachtes System, das robuste und anpassbare Lösungen ermöglicht. Die klare Trennung zwischen Anwendungslogik und Modellwahl ist dabei ein architektonisches Grundprinzip, das langfristige Flexibilität und Wartbarkeit sicherstellt. Teams können ihre KI-Strategie evolutionär entwickeln, neue Modelle evaluieren und austauschen, ohne dabei ihre bestehenden Anwendungen grundlegend umbauen zu müssen.

4. Entwicklungsfluss

Die Entwicklung von KI-Anwendungen mit großen Sprachmodellen und Agenten verlangt andere Werkzeuge und Abläufe als klassische Softwareentwicklung. Während traditionelle Software durch deterministische Logik und vorhersagbare Verzweigungen charakterisiert ist, arbeiten KI-Systeme mit probabilistischen Modellen, die durch natürliche Sprache gesteuert werden. Azure AI Foundry bietet für diese neue Art der Entwicklung einen durchgängigen Entwicklungsfluss, der von der Ideenfindung über Tests und Evaluierung bis zur Bereitstellung für den produktiven Betrieb reicht – stets mit konsistenten Schnittstellen, integrierten Sicherheitsmechanismen und umfassender Beobachtbarkeit. In diesem Kapitel stehen drei zentrale Bausteine im Fokus: Prompt Flow als Werkbank für ablaufzentrierte Orchestrierung, der Azure AI Foundry Agent Service als verwaltete Laufzeitumgebung für agentenbasierte KI-Systeme und die RAG-Unterstützung für die Anreicherung von Modellantworten mit aktuellem Wissen.

Prompt Flow: Werkbank für prompt-zentrierte Abläufe

Prompt Flow ist eine visuelle Entwicklungsumgebung zum Entwerfen, Ausführen, Testen und Bereitstellen von Abläufen, die auf großen Sprachmodellen basieren. Diese Umgebung ermöglicht es Entwicklern, komplexe Orchestrierungen grafisch zu modellieren und dabei einzelne Komponenten wie Modellaufrufe, Python-Funktionen und Datenverarbeitungsschritte miteinander zu verknüpfen.

Historischer Kontext und Architekturentscheidung

Um die aktuelle Positionierung von Prompt Flow zu verstehen, ist ein Blick auf die Entwicklungsgeschichte hilfreich: Prompt Flow hat seine Wurzeln in Azure Machine Learning und wurde ursprünglich als Teil der ML-Plattform entwickelt. Hub-basierte Projekte in Azure AI Foundry sind technisch Azure Machine Learning Workspaces – sie bringen die volle ML-Infrastruktur mit Storage Accounts, Key Vaults, Container Registries und Managed Compute mit. Diese Architektur war für klassische Machine-Learning-Szenarien mit Modelltraining, Experimenten und MLOps konzipiert.

Mit dem Aufkommen generativer KI und agentenbasierter KI-Systemen hat Microsoft jedoch erkannt, dass die Anforderungen grundlegend anders sind: Entwickler benötigen schnellen Zugriff auf Modelle, flexible API-Integration und agile Agentenentwicklung – nicht die Komplexität einer vollständigen ML-Infrastruktur. Die Antwort darauf sind die modernen Foundry-Projekte: eine vereinfachte Platform-as-a-Service-Architektur, die ohne den Azure ML-Unterbau auskommt und direkt auf die Bedürfnisse der generativen KI-Entwicklung zugeschnitten ist.

Daraus ergibt sich die heutige Feature-Segregation: Prompt Flow verbleibt ausschließlich in Hub-basierten Projekten, da es tief in die Azure ML-Architektur integriert ist. Microsoft entwickelt hingegen alle neuen generativen KI-Features – Agent Service, Foundry API, native MCP-Unterstützung – ausschließlich für Foundry-Projekte. Diese strategische Trennung ist bewusst gewählt: Sie ermöglicht Microsoft, die moderne Plattform ohne Legacy-Ballast weiterzuentwickeln, während bestehende ML-basierte Workflows vollständig unterstützt bleiben.

Kernfunktionen von Prompt Flow

Die Kernfunktionen von Prompt Flow decken den gesamten Entwicklungszyklus ab. Im Bereich des Ablauf-Designs orchestrieren Flows verschiedene Komponenten zu einem zusammenhängenden Ganzen. Diese Komponenten umfassen LLM-Aufrufe für die eigentliche Modellinteraktion, Python-Schritte für Datenverarbeitung und Geschäftslogik sowie verschiedene Hilfsfunktionen für spezialisierte Aufgaben. Ein besonders wertvolles Feature ist die systematische Variantenvergleichsfunktion, die es ermöglicht, unterschiedliche Prompt-Formulierungen parallel zu testen und zu evaluieren. Prompts werden dabei über Jinja-Vorlagen definiert, was eine flexible Parametrisierung und Wiederverwendung ermöglicht.

Die Ausführung von Flows erfolgt in verwalteten Compute-Sitzungen, die auf vorgefertigten Docker-Umgebungen basieren. Diese können wahlweise serverlos oder als dedizierte Compute-Instanz bereitgestellt werden. Die serverlose Variante eignet sich besonders für Entwicklung und Experimente, bei denen Ressourcen nur bei Bedarf genutzt werden. Dedizierte Instanzen bieten dagegen vorhersagbare Leistung und sind für produktive Szenarien mit konstanter Last besser geeignet.

Nach erfolgreichem Test und Validierung können Flows als verwaltete Online-Endpunkte bereitgestellt werden, die dann in Echtzeit per API aufgerufen werden können. Diese Bereitstellung erfolgt mit allen

notwendigen Infrastrukturkomponenten, sodass Entwickler sich nicht um Skalierung, Verfügbarkeit oder Monitoring kümmern müssen – diese Aspekte werden von der Plattform übernommen.

Werkzeugbibliothek

Die Werkzeugbibliothek von Prompt Flow umfasst neben den grundlegenden LLM-, Prompt- und Python-Werkzeugen auch spezialisierte Komponenten. Besonders hervorzuheben ist der Index-Lookup für RAG-Szenarien, der die früheren separaten Werkzeuge für Vector Index Lookup, Vector DB Lookup und Faiss Index Lookup konsolidiert und eine einheitliche Schnittstelle für verschiedene Vektorsuche-Backends bietet.

Praktische Implikationen für die Projektplanung

In der Praxis bedeutet dies: Bestehende Hub-basierte Prompt-Flow-Assets können selbstverständlich weitergeführt und gewartet werden – die Investitionen in diese Lösungen bleiben vollständig geschützt.

Für neue Vorhaben stellt sich jedoch die grundsätzliche Architekturfrage:

Wählen Sie Hub-basierte Projekte, wenn Sie:

- Prompt Flow für visuelle LLM-Orchestrierung benötigen
- Managed Compute oder dedizierte Compute-Instanzen einsetzen möchten
- Azure Machine Learning-Integration oder -Kompatibilität benötigen
- Aus der klassischen ML-Welt kommen und die vertraute Infrastruktur nutzen möchten

Setzen Sie auf Foundry-Projekte für:

- Alle neuen generativen AI- oder agentenbasierten Vorhaben
- Moderne Agentenentwicklung mit dem Agent Service
- API-first Entwicklung über die Foundry API
- Schnelles Prototyping ohne ML-Infrastruktur-Overhead
- Zugang zu aktuellen und zukünftigen generativen AI-Features

Microsoft hat klar signalisiert, dass Foundry-Projekte die strategische Plattform darstellen: Neue generative KI-Funktionen werden ausschließlich für diese Architektur entwickelt. Der Agent Service in Kombination mit der VS Code-Erweiterung ist dabei nicht nur die empfohlene, sondern die einzige Option für moderne Agentenentwicklung – diese Funktionen sind in Hub-basierten Projekten technisch nicht verfügbar.

Die klare Feature-Segregation zwischen beiden Projekttypen sollte bei der Architekturplanung von Anfang an berücksichtigt werden, da eine spätere Migration zwischen den Architekturen mit erheblichem Aufwand verbunden ist.

Azure AI Foundry Agent Service

Der Agent Service bildet das Herzstück der agentenbasierten KI-Entwicklung in Azure AI Foundry und fungiert als verwaltete Laufzeitumgebung für KI-Agenten. Der Service verbindet Modelle, Werkzeuge und Governance-Mechanismen zu einem produktionsfähigen Gesamtsystem. Dies bedeutet einen fundamentalen Wandel in der Art und Weise, wie agentenbasierte KI-Systeme entwickelt werden: Statt eine eigene Orchestrierungsschicht mit Frameworks wie LangChain oder Semantic Kernel aufzubauen, stellt der Agent Service diese Funktionalität als vollständig verwalteten Dienst bereit.

Architekturmerkmale und Zustandsverwaltung

Ein zentrales Architekturelement ist die zustandsbehaftete Ausführung von Agenten. Im Gegensatz zu einfachen, zustandslosen (Stateless) API-Aufrufen können Agenten komplexe Mehrrunden-Dialoge führen und dabei Kontext über mehrere Interaktionen hinweg bewahren. In der Standardkonfiguration werden diese Zustände in einem kundeneigenen, Single-Tenant-Cosmos-DB-Konto gespeichert. Diese Architekturentscheidung bietet mehrere Vorteile: Sie ermöglicht nicht nur den Wiederanlauf unterbrochener Sitzungen, sondern auch regionsübergreifende Resilienz für hochverfügbare Szenarien, sofern entsprechende Konfigurationen vom Kunden vorgenommen werden. Die Tatsache, dass die Datenbank im Kundenkonto liegt, gibt Organisationen zudem volle Kontrolle über ihre Daten und erfüllt strenge Compliance-Anforderungen.

Werkzeuge und Integrationsmöglichkeiten

Die Leistungsfähigkeit eines Agenten wird maßgeblich durch die Werkzeuge bestimmt, die ihm zur Verfügung stehen. Der Agent Service bietet hier ein breites Spektrum an Integrationsmöglichkeiten, das sich in mehrere Kategorien gliedert.

Im Bereich der Wissens-Werkzeuge steht Azure AI Search für leistungsstarke Suche in strukturierten und unstrukturierten Daten zur Verfügung. File Search ermöglicht den direkten Zugriff auf Dokumente und deren Inhalte. Die Integration mit Microsoft Fabric öffnet den Zugang zu Data-Warehouse-Infrastrukturen und analytischen Datensätzen. Für Webinhalte bietet Grounding mit Bing Search Zugriff auf aktuelle Online-Informationen, während Grounding mit Bing Custom Search spezialisierte Suchdomänen ermöglicht. Unternehmensquellen wie SharePoint lassen sich über entsprechende Integrationen einbinden.

Aktions-Werkzeuge erweitern die Fähigkeiten von Agenten über reine Informationsabfrage hinaus. Über die Logic Apps Integration steht eine breite Auswahl an Connectors für Geschäftssysteme zur Verfügung. Azure Functions ermöglichen die Integration benutzerdefinierter Geschäftslogik, während OpenAPI-Spezifikationen den standardisierten Zugriff auf eigene Dienste erlauben. Darüber hinaus sind bidirektionale Integrationsmuster mit Geschäftsprozessen möglich.

Spezialisierte Werkzeuge runden das Portfolio ab. Der Code Interpreter bietet eine sichere Python-Ausführungsumgebung, in der Agenten Berechnungen durchführen, Daten analysieren oder Visualisierungen erstellen können, ohne dass Code auf den Produktionssystemen ausgeführt werden muss. Browser Automation ermöglicht strukturierten Zugriff auf Webinhalte und deren automatisierte Interaktion durch natürlichsprachliche Anweisungen. Computer Use erweitert diese Fähigkeiten noch weiter, indem es Agenten ermöglicht, mit Betriebssystemen und Anwendungen über deren Benutzeroberflächen zu interagieren – sie können Anwendungen öffnen, Formulare ausfüllen und mehrstufige Workflows ausführen. Deep Research mit o3-deep-research automatisiert mehrschrittige Web-Recherchen und synthetisiert Informationen aus verschiedenen Quellen zu kohärenten Berichten.

Ein besonders zukunftsweisendes Feature ist die Unterstützung des Model Context Protocol (MCP), das seit Juli 2025 im Preview verfügbar ist. Dieses offene Protokoll ermöglicht die Integration externer Tool-Server und schafft damit ein Ökosystem, in dem Drittanbieter spezialisierte Werkzeuge bereitstellen können, die nahtlos in Agenten integriert werden.

Multi-Agent-Orchestrierung

Komplexe Aufgaben lassen sich oft am besten durch die Zusammenarbeit mehrerer spezialisierter Agenten lösen. Das Konzept der „Connected Agents" ermöglicht die Delegation von Teilaufgaben an spezialisierte Agenten, die jeweils für spezifische Domänen optimiert sind. Multi-Agent-Workflows koordinieren komplexe, mehrstufige Prozesse, bei denen verschiedene Agenten sequenziell oder parallel arbeiten und ihre Ergebnisse zu einem Gesamtresultat zusammenführen. Die Umgebung unterstützt dabei agentenübergreifende Kommunikation (zum Beispiel Agent-to-Agent) und bietet mit dem Azure AI Agents SDK ein konsistentes Programmiermodell für die Entwicklung.

Entwicklungswerkzeuge und Beobachtbarkeit

Die VS-Code-Erweiterung für Azure AI Foundry bietet eine integrierte Entwicklungsumgebung für den gesamten Agent-Lebenszyklus. Entwickler können Agenten direkt in ihrer vertrauten IDE erstellen, lesen, aktualisieren und löschen. Die YAML-Unterstützung mit IntelliSense macht die Konfiguration von Agenten fehlerresistent und beschleunigt die Entwicklung. Multi-Agent-Workflows lassen sich lokal modellieren und testen, bevor sie in die Cloud-Umgebung übertragen werden. Das direkte Deployment aus der IDE heraus eliminiert Medienbrüche und beschleunigt den Iterationszyklus erheblich.

Beobachtbarkeit und Qualitätssicherung sind tief in die Plattform integriert. Tracing und Metriken basieren auf OpenTelemetry, einem offenen Standard, der mit Application Insights integriert ist. Dies ermöglicht detaillierte Einblicke in das Laufzeitverhalten von Agenten, von einzelnen Modellaufrufen bis zu komplexen Workflow-Ausführungen. Die „Continuous-Evaluation-Funktionen" liefern Qualitäts-, Sicherheits- und Performance-Signale nahezu in Echtzeit, sodass Probleme frühzeitig erkannt und behoben werden können, bevor sie Auswirkungen auf Endnutzer haben.

Die strategische Bedeutung des Agent Service liegt darin, dass er Funktionalität als verwalteten Dienst bereitstellt, die zuvor mühsam selbst implementiert werden musste. Identitätsverwaltung, Netzwerkisolation, Sicherheitsmechanismen, Monitoring und Skalierung sind integraler Bestandteil der Plattform und müssen nicht mehr separat aufgebaut und gewartet werden. Dies beschleunigt nicht nur die Entwicklung, sondern erhöht auch die Zuverlässigkeit und Sicherheit der resultierenden Systeme.

RAG-Unterstützung (Retrieval-Augmented Generation)

Das Konzept der Retrieval-Augmented Generation adressiert eine fundamentale Herausforderung großer Sprachmodelle: Während diese Modelle über umfangreiches Weltwissen verfügen, fehlt ihnen der Zugriff auf aktuelle Ereignisse, unternehmensspezifische Informationen oder private Daten. RAG löst dies, indem es Modellantworten mit relevanten Inhalten aus externen Wissensquellen anreichert. Der Agent fragt zunächst eine Wissensquelle nach relevanten Informationen ab und fügt diese dann dem Prompt hinzu, bevor das Modell seine Antwort generiert.

In Azure AI Foundry stehen für RAG-Szenarien zwei komplementäre Pfade zur Verfügung. Der agent-zentrierte Ansatz integriert Retrieval direkt in den Agent Service. Agenten binden Azure AI Search und File Search als Wissens-Werkzeuge ein und können während einer Sitzung dynamisch relevante Inhalte abrufen. Dies geschieht transparent und adaptiv – der Agent entscheidet basierend auf der Konversation, wann eine Wissensabfrage notwendig ist, und formuliert die Suchanfrage entsprechend. Die abgerufenen Inhalte fließen dann nahtlos in die Prompt-Konstruktion ein, ohne dass der Entwickler jeden Abrufschritt explizit orchestrieren muss.

Der zweite Pfad nutzt Vektorindizes in hub-basierten Projekten. Im Foundry-Portal lassen sich Vektorindizes für verschiedene Datenquellen erstellen und verwalten. Prompt-Flow-basierte RAG-Abläufe verwenden hierfür den Index-Lookup als Werkzeug, um strukturiert auf diese Indizes zuzugreifen. Dieser Ansatz eignet sich besonders, wenn die RAG-Pipeline explizit gesteuert werden soll oder wenn spezifische Verarbeitungsschritte zwischen Retrieval und Generation eingefügt werden müssen. Es ist zu beachten, dass diese Funktionalität aktuell nur in hub-basierten Projekten zur Verfügung steht, nicht in den neueren Azure AI Foundry-Projekten.

Technische Überlegungen zur Umsetzung

Azure AI Search bildet das technische Rückgrat für viele RAG-Szenarien. Der Dienst unterstützt klassische Volltext-Suche, vektorbasierte Semantiksuche und hybride Abfragen, die beide Ansätze kombinieren. Diese Flexibilität macht ihn zu einem skalierbaren und leistungsfähigen Retriever für agentenbasierte Szenarien. Die Integration ist tief genug, dass Agenten automatisch die jeweils beste Suchstrategie wählen können. Eine bedeutende Weiterentwicklung ist die Agentic Retrieval-Funktion (seit Mai 2025 im Preview), die speziell für RAG optimiert wurde. Sie nutzt Large Language Models für

intelligentes Query-Planning, zerlegt komplexe Fragen in fokussierte Subqueries und führt diese parallel aus, was die Antwortrelevanz um bis zu 40 % steigern kann.

Eine wichtige Compliance-Überlegung betrifft das Grounding mit Bing Search. Da diese Funktion auf externe Dienste zugreift, gelten eigene Datenverarbeitungsbedingungen, die außerhalb der Azure-Compliance-Grenze liegen. Bei regulierten Workloads, etwa im Finanz- oder Versicherungssektor, muss dies sorgfältig geprüft werden. In solchen Szenarien sollten bevorzugt Azure-interne Wissensquellen wie Azure AI Search oder andere Quellen innerhalb des eigenen Tenants verwendet werden, um die Kontrolle über alle Datenflüsse zu behalten.

Die eigentliche Stärke von RAG im agentenbasierten Kontext liegt in der Möglichkeit, iterative Recherchepfade zu realisieren. Ein Agent kann einen ersten Abruf durchführen, die Ergebnisse prüfen, seine Suchstrategie verfeinern und einen zweiten, präziseren Abruf starten. Werkzeuge wie Deep Research automatisieren solche mehrstufigen Recherchepfade auf Webdaten und synthetisieren Informationen aus verschiedenen Quellen zu kohärenten Antworten. Dies geht weit über einfaches Retrieval hinaus und nähert sich der Art und Weise an, wie Menschen komplexe Recherchen durchführen würden.

Sicherheit und Governance

Sicherheit ist nicht ein nachträglicher Zusatz, sondern fundamental in Azure AI Foundry verankert. Identitätsverwaltung erfolgt über Microsoft Entra ID mit Managed Identities, wodurch Secrets nicht in Code oder Konfiguration gespeichert werden müssen. Rollenbasierte Zugriffskontrolle (RBAC) ermöglicht granulare Berechtigungen auf Hub-, Projekt- und Ressourcenebene, wodurch der Zugriff auf Agenten, Modelle und Datenquellen präzise gesteuert werden kann.

Netzwerkisolation schützt vor unbefugtem Zugriff. Private Endpoints ermöglichen es, dass Agenten und Datenquellen nur über private Netzwerkverbindungen kommunizieren – nicht über das öffentliche Internet. Virtual Networks und Firewall-Regeln können zusätzliche Segmentierung erzwingen.

Datenverschlüsselung erfolgt auf zwei Ebenen: At-Rest (Daten in Speicher) und In-Transit (Daten während Übertragung). Standardmäßig verwendet Azure die Microsoft-verwalteten Schlüssel. Für höchste Sicherheitsanforderungen können Organisationen Customer-Managed Keys (CMK) verwenden.

Seit 2025 ist PII Detection als integrierter Content Filter verfügbar, der automatisch personenbezogene Informationen (Namen, E-Mail-Adressen, Kreditkartennummern, etc.) in Agenten-Eingaben und -Ausgaben erkennt. Dies ermöglicht automatische Maskierung, Logging oder Blockierung sensibler Daten – kritisch für GDPR, HIPAA und andere Datenschutzgesetze.

Audit und Compliance: Alle Aktionen werden protokolliert – wer hat welchen Agenten erstellt, wann wurde er bereitgestellt, welche Daten wurden verarbeitet. Diese Audit-Logs basieren auf einem append-only-Prinzip und können nicht nachträglich verändert werden, was eine zuverlässige Nachvollziehbarkeit für Compliance-Audits gewährleistet. Azure AI Foundry basiert auf der Azure-Plattform, die Standards wie SOC 2, ISO 27001, HIPAA und GDPR erfüllt. Die tatsächliche Compliance-Verantwortung folgt dem Shared Responsibility Model zwischen Microsoft und dem Kunden.

Zusammenführung der Entwicklungsansätze

Die drei vorgestellten Bausteine fügen sich zu einem kohärenten Entwicklungsfluss zusammen, der verschiedene Anwendungsfälle und Reifegrade bedient. Prompt Flow eignet sich weiterhin für bestehende, Hub-basierte Flows und bietet einen visuellen Einstieg für Teams, die mit ablaufzentrierter Orchestrierung vertraut sind. Die grafische Modellierung kann besonders in frühen Prototyping-Phasen wertvoll sein, wenn Nicht-Entwickler in den Design-Prozess eingebunden werden sollen.

Für neue Agent-Lösungen ist der Agent Service der empfohlene und zukunftssichere Pfad. Er bietet eine verwaltete Laufzeitumgebung, die alle notwendigen Infrastrukturaspekte abdeckt. Die breite Werkzeugintegration ermöglicht es, nahezu jedes Integrationsszenario abzudecken, ohne externe Orchestrierungsframeworks einbinden zu müssen. Seit Mai 2025 ermöglichen Multi-Agent-Fähigkeiten die Modellierung komplexer Prozesse durch Komposition spezialisierter Agenten: Connected Agents unterstützen Punkt-zu-Punkt-Interaktionen für Aufgabendelegation, während Multi-Agent Workflows eine zustandsgesteuerte Orchestrierungsebene für langläufige, mehrstufige Prozesse bieten. Umfassende Observability und integrierte Evaluierung stellen sicher, dass die Qualität von Agenten kontinuierlich überwacht und verbessert werden kann.

RAG ist nicht als separates Feature zu verstehen, sondern als nativ verankertes Konzept, das sowohl über Agent-Wissenswerkzeuge als auch über projektseitige Indizes realisiert werden kann. Die Wahl des Ansatzes hängt von den spezifischen Anforderungen ab: Soll der Agent autonom entscheiden, wann Wissen abgerufen wird, eignet sich der agent-zentrierte Ansatz mit Wissensdateien oder -indizes. Soll die RAG-Pipeline explizit gesteuert werden, bietet sich der Index-basierte Ansatz mit Azure AI Search an.

Multimodal-Fähigkeiten und Vision

Azure AI Foundry unterstützt nicht nur Text, sondern auch Bilder, Audio und Video – echte Multimodal-Fähigkeiten. Vision-Fähigkeiten ermöglichen es Agenten, Bilder zu verstehen: Bildanalyse kann Objekte erkennen, Text extrahieren (OCR), Diagramme interpretieren oder Szenen beschreiben. Praktische Anwendungsfälle: Ein Agent analysiert Rechnungen (Bildverarbeitung + OCR), extrahiert

Betrag und Datum, und bucht diese automatisch. Ein Quality-Control-Agent inspiziert Produktfotos auf Mängel.

Audio-Fähigkeiten ermöglichen Spracherkennung und Transkription. Agenten können Sprachnachrichten verarbeiten, diese transkribieren und dann, wie Text-Eingaben behandeln. Sentiment-Analyse auf Audio ermöglicht es, die emotionale Stimmung des Sprechers zu erkennen – wertvoll für Support-Szenarien.

Video-Fähigkeiten ermöglichen Frame-Extraktion und Szenenanalyse. Agenten können Videos analysieren, wichtige Frames extrahieren und diese wie Bilder verarbeiten. Mit Sora steht zudem ein leistungsfähiges Tool für Video-Generierung und Video-zu-Video-Transformation zur Verfügung. Dies ist wertvoll für Überwachung, Qualitätskontrolle, Content-Erstellung oder Marketing-Kampagnen.

Structured Outputs (seit August 2024 verfügbar) ermöglichen zuverlässiges Parsing durch JSON-Schema-Definition. Statt unstrukturiertem Text kann ein Agent JSON-Ausgaben mit definierten Feldern generieren. Dies ist entscheidend für Downstream-Verarbeitung: Wenn Sie garantiert ein JSON-Objekt mit Feldern „name", „email", „phone" erhalten, können Sie dies direkt in Ihre Datenbank schreiben, ohne zusätzlich zu parsen oder zu validieren.

Zusammenfassung

Das übergeordnete Ziel all dieser Werkzeuge ist es, den Weg von der Idee zur produktionsreifen agentenbasierter KI-Systeme möglichst schnell und effizient zu gestalten. Durch die enge Verzahnung von Entwicklung, Test, Evaluierung und Bereitstellung innerhalb einer integrierten Plattform werden aufwendige Integrationsschritte vermieden, die typischerweise bei heterogenen Toolchains entstehen würden.

Die nahtlose Integration in bestehende Azure-Sicherheits- und Governance-Strukturen gewährleistet, dass selbst komplexe agentenbasierte KI-Systeme die hohen Anforderungen geschäftskritischer Szenarien zuverlässig erfüllen. Kontinuierliche Evaluierung und umfassende Monitoring-Funktionen bieten eine verlässliche Grundlage, um die Qualität und Sicherheit der entwickelten Systeme transparent nachzuweisen. Damit erhalten Entwicklungsteams die notwendige Sicherheit und Zuversicht, agentenbasierter KI-Systeme verantwortungsbewusst und stabil in den produktiven Betrieb zu überführen.

5. Datenzugriff und RAG-Bausteine

Der erfolgreiche Einsatz generativer KI in Unternehmen steht und fällt mit verlässlichem, berechtigungskonformem Datenzugriff. Während öffentlich trainierte Sprachmodelle über breites

Weltwissen verfügen, fehlt ihnen naturgemäß der Zugang zu unternehmensspezifischem Wissen, aktuellen Entwicklungen oder vertraulichen Informationen. Azure AI Foundry stellt für diese Herausforderung durchgängige Bausteine bereit: projektseitige Vektorindizes für den schnellen Einstieg, Azure AI Search als leistungsfähiges Such- und Abrufsystem und standardisierte Pfade zum Einbinden eigener Datenquellen. Das übergeordnete Ziel ist präzises Retrieval für Retrieval-Augmented Generation bei klaren Betriebs- und Governance-Leitplanken, die sicherstellen, dass sensible Informationen geschützt bleiben und Berechtigungsstrukturen respektiert werden.

Vektorindizes im Projekt

Ein Vektorindex ist eine Ressource im Azure AI Foundry Projekt, in der Dokumentfragmente als hochdimensionale Vektoren abgelegt werden. Um das Konzept zu verstehen, hilft es, sich die Funktionsweise vorzustellen: Texte werden durch Einbettungsmodelle in mathematische Repräsentationen transformiert, wobei semantisch ähnliche Texte ähnliche Vektorrepräsentationen erhalten. Diese Vektoren werden in einem spezialisierten Index gespeichert, der effiziente Ähnlichkeitssuchen ermöglicht. Ein Vektorindex ist nicht zwingend erforderlich für RAG-Szenarien – man könnte theoretisch auch mit einfacher Keyword-Textsuche arbeiten – erhöht aber die Treffergenauigkeit bei natürlichsprachlichen Anfragen erheblich, da er semantische Zusammenhänge versteht, die über reine Stichwortübereinstimmungen hinausgehen. Die Anlage und Nutzung erfolgen direkt im Foundry-Portal, was den Einstieg erheblich vereinfacht.

Der Weg zum einsatzbereiten Vektorindex

Der Aufbau eines Vektorindex folgt einem systematischen Prozess, der mehrere aufeinander aufbauende Schritte umfasst. Im ersten Schritt müssen die Daten bereitgestellt werden, die durchsuchbar gemacht werden sollen. Dies können beispielsweise Handbücher, Unternehmensrichtlinien, FAQ-Dokumente oder andere textuelle Informationsquellen sein. Die Bereitstellung erfolgt entweder durch direktes Hochladen der Dateien über die Portal-Oberfläche oder durch Anbinden vorhandener Speicherquellen, sodass Dokumente aus bestehenden Quellen übernommen werden können.

Der zweite Schritt besteht in der Auswahl eines geeigneten Einbettungsmodells aus dem Foundry-Katalog. Diese Wahl ist kritisch, denn das Einbettungsmodell transformiert Texte in ihre Vektorrepräsentationen. Wichtig ist hierbei die Konsistenz: Das gleiche Modell muss sowohl für die initiale Indexierung der Dokumente als auch später für die Abfragen verwendet werden. Ein Wechsel des Einbettungsmodells würde bedeuten, dass die Vektorrepräsentationen der Dokumente und der Suchanfragen nicht mehr im gleichen mathematischen Raum liegen, was die Suchqualität erheblich beeinträchtigen würde.

Im dritten Schritt erfolgt die Segmentierung und Schemadefinition. Längere Texte werden in sinnvolle Segmente zerlegt, wobei sich ein Überlapp (Overflow) zwischen aufeinanderfolgenden Segmenten als vorteilhaft erwiesen hat. Dieser Überlapp stellt sicher, dass zusammenhängende Informationen, die eine Segmentgrenze überschreiten, nicht verloren gehen. Parallel dazu werden Metadatenfelder definiert, die zusätzliche Informationen zu jedem Segment speichern. Typische Metadaten umfassen Titel, Quellenangaben und Berechtigungsinformationen. Diese Metadaten dienen später als Filter, um Suchergebnisse einzuschränken oder berechtigungsbasierte Zugriffskontrolle zu implementieren.

Nach Abschluss dieser Vorbereitungen wird der Index erstellt und steht für RAG-Abfragen bereit. Bei einer Abfrage werden die ähnlichsten Segmente identifiziert und zusammen mit ihren Quellenangaben in den Prompt des Sprachmodells eingebunden. Diese Quellenangaben sind nicht nur für die Nachvollziehbarkeit wichtig, sondern ermöglichen es dem Nutzer auch, die Originalquellen zu konsultieren, wenn er die Informationen vertiefen möchte.

Für ablaufbasierte Szenarien in Prompt Flow steht das Index-Lookup-Werkzeug zur Verfügung. Dieses Werkzeug erkennt gängige Vektorindizes automatisch und konsolidiert die Funktionalität, die zuvor auf separate Werkzeuge verteilt war. Die früheren spezialisierten Werkzeuge für Vector Index Lookup, Vector DB Lookup und Faiss Index Lookup wurden durch diese einheitliche Schnittstelle ersetzt, was die Entwicklung erheblich vereinfacht.

Azure AI Search als leistungsfähiges Such- und Abrufsystem

Azure AI Search ist eines der zentralen Such- und Abrufsysteme in Foundry für RAG-Retrieval und bietet Funktionen, die weit über einfache Vektorsuche hinausgehen. Das Herzstück bildet die Hybridsuche, die Vektor- und Stichwortsuche in einer einzigen Anfrage kombiniert. Während die Vektorsuche semantische Ähnlichkeiten erfasst, sorgt die Stichwortsuche dafür, dass exakte Begriffe, Produktnamen oder Akronyme zuverlässig gefunden werden. Die Ergebnisse beider Suchverfahren werden mittels „Reciprocal Rank Fusion" zusammengeführt, einem Algorithmus, der die Rankings beider Methoden intelligent kombiniert, sodass Dokumente, die in beiden Rankings gut abschneiden, bevorzugt werden.

Optional kann der Semantic Ranker (Reranker) als zusätzliche Ebene eingesetzt werden, der die zusammengeführten Ergebnisse nochmals neu bewertet. Dieser Ranker nutzt tiefere semantische Modelle, um die Relevanz jedes Dokuments im Kontext der spezifischen Anfrage zu bewerten. Dies verbessert die Relevanz und Präzision erheblich, insbesondere bei unternehmensspezifischer Terminologie oder komplexen fachlichen Anfragen, bei denen subtile semantische Nuancen den Unterschied zwischen einem hilfreichen und einem irrelevanten Ergebnis ausmachen.

Anbindungswege an Azure AI Search

Die Integration von Azure AI Search in Foundry-Agenten erfolgt über zwei komplementäre Wege, die unterschiedliche Anwendungsfälle adressieren. Der erste Weg nutzt das Agent-Werkzeug „Azure AI Search", das Agenten direkten Zugriff auf bestehende Suchindizes gewährt. Dies eignet sich besonders, wenn bereits etablierte Azure-AI-Search-Instanzen existieren, die möglicherweise von mehreren Anwendungen genutzt werden. Agenten können diese Indizes durchsuchen und die gefundenen Inhalte zur Fundierung ihrer Antworten nutzen, wobei die volle Flexibilität der Azure-AI-Search-Funktionen erhalten bleibt.

Der zweite Weg verwendet das Agent-Werkzeug „File Search", das einen stärker integrierten Ansatz verfolgt. Bei der Standard-Setup-Konfiguration werden hochgeladene Dateien im verbundenen Blob-Speicher des Projekts gespeichert. Die zugehörige Azure-AI-Search-Ressource übernimmt dann automatisch die Vektorisierung und Indizierung dieser Dateien. Der wesentliche Vorteil dieses Ansatzes liegt darin, dass Ingestion und Aufbereitung vollständig vom Dienst übernommen werden. Entwickler müssen sich nicht um die technischen Details der Dokumentverarbeitung, Segmentierung oder Vektorisierung kümmern, sondern können sich auf die eigentliche Agentenlogik konzentrieren.

Eigene Datenquellen einbinden

Foundry unterstützt explizit die Nutzung eigener Unternehmensdaten und bietet dafür mehrere komplementäre Ansätze. Daten lassen sich durch direktes Hochladen in das Portal einbringen, was sich besonders für kleinere Dokumentmengen oder Ad-hoc-Tests eignet. Für größere und regelmäßig aktualisierte Datenbestände können Azure-Speicherdienste angebunden werden, sodass die Dokumente an ihrem ursprünglichen Speicherort verbleiben können. Über sogenannte Indexierungsprogramme in Azure AI Search lassen sich Datenquellen automatisiert durchsuchen und indizieren, wobei Änderungen in den Quelldaten kontinuierlich erkannt und in den Index übernommen werden.

Das Spektrum unterstützter Datenquellen ist breit und deckt die gängigen Unternehmensszenarien ab. Dazu gehören Blob Storage für unstrukturierte Dateien, Azure SQL für strukturierte Datenbankinhalte, Cosmos DB für dokumentenbasierte NoSQL-Daten und Data Lake Storage Gen2 für analytische Datensätze. SharePoint-Dokumentbibliotheken stehen als Vorschaufunktion zur Verfügung, was besonders wertvoll ist, da SharePoint in vielen Organisationen als zentrales Dokumentenmanagement-System dient.

Für unstrukturierte Inhalte wie gescannte Dokumente oder Bilder stehen Dokumentextraktion und optische Zeichenerkennung (OCR) als Anreicherungs-Bausteine bereit. Diese werden in Azure AI Search als „Skills" bezeichnet und können in Verarbeitungspipelines eingebunden werden. Ein typisches Szenario wäre ein gescanntes PDF-Dokument, das zunächst durch OCR in

maschinenlesbaren Text transformiert wird, bevor es segmentiert und vektorisiert in den Index aufgenommen wird.

Im Foundry-Kontext existieren zudem vereinfachte Pfade unter der Bezeichnung „On Your Data", die speziell für RAG-Szenarien optimiert sind. Diese Pfade legen Dateien in Blob Storage ab und indizieren sie automatisch in Azure AI Search, sodass schnell ein durchsuchbarer Wissensspeicher entsteht. Der entscheidende Aspekt dabei ist, dass proprietäre Daten die Azure-Compliance-Grenze nicht verlassen müssen. Alle Verarbeitungsschritte erfolgen innerhalb der Azure-Umgebung unter den gleichen Sicherheits- und Compliance-Richtlinien wie andere Azure-Dienste.

Sicherheit, Berechtigungen und ergebnisbasierte Zugriffskontrolle

Die Verwaltung von Dienst- und Datenzugriffen erfolgt über die etablierten Azure-Mechanismen. Microsoft Entra-Authentifizierung stellt sicher, dass nur autorisierte Identitäten auf die Systeme zugreifen können. Rollenbasierte Zugriffskontrolle definiert, welche Operationen verschiedene Nutzer und Dienste ausführen dürfen. Netzwerkkontrollen über Private Endpoints und verwaltete virtuelle Netzwerke schirmen sensible Daten von öffentlichen Netzwerken ab.

Eine besondere Herausforderung in RAG-Szenarien ist die ergebnisseitige Sichtbarkeitskontrolle, häufig als „Security Trimming" bezeichnet. Das Konzept dahinter ist wichtig zu verstehen: Während die oben genannten Mechanismen kontrollieren, wer überhaupt auf das Suchsystem zugreifen darf, adressiert Security Trimming die Frage, welche spezifischen Dokumente ein authentifizierter Nutzer sehen darf. In traditionellen Dateisystemen wird dies durch Dateiberechtigungen gelöst. In einem Suchindex, der Inhalte aus verschiedenen Quellen konsolidiert, muss dieser Berechtigungskontext auf andere Weise erhalten bleiben.

Klassisches Security Trimming über Filter

Azure AI Search implementiert Security Trimming traditionell über Filter, die in den Dokument-Metadaten hinterlegte, Gruppen- oder Benutzerkennungen berücksichtigen. Beim Indizieren werden jedem Dokument Metadaten hinzugefügt, die angeben, welche Nutzer oder Gruppen Zugriff haben dürfen. Bei einer Suchanfrage muss die Anwendung explizit einen Filter mitgeben, der nur Dokumente zurückgibt, für die der anfragende Nutzer berechtigt ist. Diese Filter simulieren im Suchlayer die Dokumentberechtigungen, die in den Originalsystemen existieren.

Ein kritischer Aspekt ist dabei zu beachten: Eine automatische, implizite Berechtigungsprüfung im Suchdienst findet bei diesem klassischen Ansatz nicht statt – der Dienst wendet lediglich die Filter an, die ihm von der Anwendung explizit mitgegeben werden. Die Anwendung muss die Identität des Nutzers ermitteln, dessen Gruppenmitgliedschaften über Microsoft Entra ID abrufen und daraus die

passende Filterlogik konstruieren. **Die vollständige Verantwortung für die korrekte Umsetzung dieser Berechtigungslogik liegt damit beim Anwendungsentwickler.**

In „On Your Data"-Szenarien mit Azure OpenAI kann die Anwendung diese Filter ebenfalls anhand von Entra-Gruppen konstruieren, indem sie die bestehenden Gruppenmitgliedschaften aus dem Identitätssystem abruft und in entsprechende Suchfilter übersetzt. Auch hier gilt: Die Anwendung trägt die Verantwortung für die korrekte Implementierung der Filterlogik.

Neue native Entra-Unterstützung (Preview)

Seit Mai 2025 bietet Azure AI Search in der Preview-API eine neue Funktionalität mit nativer Microsoft Entra ACL/RBAC-Unterstützung. Bei diesem Ansatz validiert Azure AI Search den Entra-Token des Aufrufers zur Abfragezeit und wendet die Berechtigungsprüfung automatisch an, ohne dass Entwickler manuelle Filterlogik implementieren müssen. Diese Funktionalität ist aktuell für Inhalte aus Azure Data Lake Storage Gen2 (ADLS Gen2) und Azure Blob Storage verfügbar, bei denen die Berechtigungsmetadaten bereits bei der Indizierung aus der Datenquelle übernommen werden. Für andere Datenquellen bleibt das klassische Pattern mit manuellen Filtern die bevorzugte Methode.

Einordnung der verschiedenen Ansätze

Die verschiedenen Bausteine für Datenzugriff und RAG fügen sich zu einem gestuften System zusammen, das unterschiedliche Anforderungen und Reifegrade bedient. Projekt-Vektorindizes bieten einen schnellen und unkomplizierten Einstieg in RAG-Szenarien. Sie sind unmittelbar im Portal verwaltbar, ohne dass zusätzliche Azure-Ressourcen konfiguriert werden müssen. Für Prototypen, Experimente oder kleinere Produktionsworkloads mit überschaubaren Datenmengen stellen sie eine ausgezeichnete Wahl dar.

Azure AI Search liefert die Funktionen, die für anspruchsvolle Produktionsszenarien erforderlich sind: Skalierbarkeit für große Datenmengen und hohe Abfragevolumen, Hybridsuche für optimale Treffergenauigkeit, Semantic Ranking für verfeinerte Relevanzbewertung und tiefe Integration in den Agent Service. Für produktive Szenarien, die auf Zuverlässigkeit, Leistung und erweiterte Funktionen angewiesen sind, ist Azure AI Search das Standard-Such- und Abrufsystem. Die Investition in die etwas komplexere Konfiguration zahlt sich durch die gewonnenen Fähigkeiten aus.

Die verschiedenen Datenwege – direktes Hochladen, Anbindung von Azure-Datenquellen und automatisierte Indexierungsprogramme – decken die gängigen Unternehmensquellen ab, von statischen Dokumentsammlungen bis zu dynamischen, sich kontinuierlich ändernden Datenbeständen. Die Verfügbarkeit von SharePoint-Bibliotheken, auch wenn noch im Vorschaustatus, schließt eine wichtige Lücke für Organisationen, die SharePoint als primäres Dokumentenmanagement nutzen. Die

Anreicherungs-Bausteine für Dokumentextraktion und OCR verbessern die Auffindbarkeit und Qualität, indem sie auch Inhalte erschließen, die nicht von vornherein in strukturierter Form vorliegen.

Best Practices

Indexdesign und Retrieval

Ein wirksames RAG-System steht und fällt mit sauberem Indexdesign und durchdachten Retrieval-Strategien. Folgende Gestaltungsprinzipien haben sich bewährt:

- **Kohärente Segmentierung:** Inhalte entlang natürlicher Dokumentstruktur (Überschriften, Absätze, Listen) in kleine, sinnvolle Abschnitte zerlegen. Bewährte Richtwerte sind 512 Tokens (circa 2000 Zeichen) mit einem Überlapp von 25 % (128 Tokens), um Kontext über Segmentgrenzen hinweg zu erhalten. Strukturbasiertes Chunking mit dem „Document Layout Skill" verbessert die semantische Kohärenz der Segmente.
- **Aussagekräftige Metadaten:** Quelle, Dokumenttyp, Gültigkeits- oder Versionsinformationen sowie Berechtigungen als Metadaten pflegen. Metadaten als Filter zur Eingrenzung von Suchergebnissen und zur Nachvollziehbarkeit nutzen. Parent-Child-Beziehungen zwischen Chunks und Originaldokumenten wahren, um Kontext und Quellenangaben zu ermöglichen.
- **Modellkonsistenz:** Das Einbettungsmodell für Indexierung und Abfragen identisch halten – dies ist absolut kritisch für die Qualität der Suchergebnisse. Azure AI Search prüft nicht automatisch auf Modellkonsistenz, sondern liefert bei unterschiedlichen Modellen schlechte Ergebnisse ohne Fehlermeldung. Ein Modellwechsel erfordert vollständige Re-Indexierung, um Embedding-Räume nicht zu vermischen.
- **Multilinguale Inhalte:** Für mehrsprachige Szenarien multilinguale Embedding-Modelle verwenden, die semantisch äquivalente Inhalte über Sprachgrenzen hinweg im selben Vektorraum repräsentieren. Spracherkennung als Metadatenfeld für optionale Filterung bereitstellen. Bei Hybridsuche (Kombination von Vektor- und Keyword-Suche) zusätzlich sprachspezifische Analyse für die Textnormalisierung einsetzen.
- **Zitierfähigkeit:** Für jedes Segment kanonische Quell-URLs oder eindeutige Identifikatoren speichern, um Antworten mit belastbaren Quellenangaben zu unterlegen und Follow-up-Queries zu ermöglichen.

Abfrage- und Ranking-Tuning

Retrieval-Qualität entsteht aus dem Zusammenspiel von Abfrageformulierung, Hybrid-Suche und Ranking:

- **Hybrid-Suche standardmäßig einsetzen**: Vektor- und Keyword-Suche kombiniert nutzen; die Stärken beider Verfahren (semantisches vs. lexikalisches Matching) ergänzen sich und erhöhen die Treffergenauigkeit.
- **Query-Aufbereitung**: Nutzeranfragen bei Bedarf um Synonyme, Fachbegriffe oder alternative Formulierungen erweitern; Prompt-basierte Umschreibung zur Verbesserung der Suchfreundlichkeit einsetzen.
- **Filter gezielt verwenden**: Zeiträume, Dokumenttypen, Verantwortungsbereiche und Berechtigungen als Filterkriterien modellieren; irrelevante Treffer früh ausschließen und Suchraum eingrenzen.
- **Reranking**: Semantisches Reranking (z.B. Cross-Encoder-Modelle) für anspruchsvolle, fachliche Anfragen aktivieren und Auswirkung auf Precision und Relevanz evaluieren.
- **Umfang der Kontexteingabe steuern**: Anzahl und Länge der übernommenen Segmente an das Kontextfenster des Modells anpassen; Redundanz vermeiden, aber notwendigen Kontext für kohärente Antworten sicherstellen.
- **Retrieval-Qualität messen**: Regelmäßig mit Metriken wie NDCG, MRR oder Precision@K evaluieren und Tuning-Parameter iterativ optimieren.

Evaluation und Monitoring

Systematische Qualitätssicherung macht RAG-Systeme belastbar und reproduzierbar:

- **Datensätze kuratieren**: Repräsentative Frage-Antwort-Paare mit erwarteten Referenzdokumenten definieren; Domänenexpertise einbinden und Ground-Truth-Datensätze pflegen.
- **Retrieval-Qualität messen**: Recall und Precision der abgerufenen Dokumente bewerten; Context Relevance (Relevanz der gefundenen Segmente) und Citation Accuracy (Korrektheit der Quellenverweise) prüfen.
- **Antwortqualität bewerten**: Faithfulness/Groundedness (faktische Treue zu den Quellen), Answer Relevance (Passung zur Frage) und Hallucination Rate systematisch erfassen.
- **Evaluationsläufe automatisieren**: In Azure AI Foundry mit orchestrierten Flows Evaluationsmetriken regelmäßig berechnen, Regressionstests durchführen und Ergebnisse über Versionen hinweg vergleichen.
- **Tracing und Telemetrie**: Abfrage, angewandte Filter, zurückgelieferte Segmente, Ranking-Scores und Modellaufrufe vollständig protokollieren; strukturierte Logs für Ursachenanalyse bei Fehlverhalten bereitstellen.
- **Human-in-the-Loop**: Fachliche Stichprobenprüfung etablieren, qualitative Rückmeldungen sammeln und Erkenntnisse systematisch in Indexdesign, Chunk-Strategie und Prompts zurückspielen.

Betrieb, Skalierung und Kostensteuerung

Produktionsbetrieb erfordert planbare Skalierung und klare Betriebsprozesse:

- **Indexaktualität sicherstellen**: Inkrementelle Indexierung mit Indexer-Zeitplänen (Schedule-basiert oder Change-Tracking) nutzen; Änderungsereignisse aus Quellsystemen (z.B. via Event Grid, Azure Functions) verarbeiten, um Staleness zu minimieren und Datenfrische zu gewährleisten.
- **Kapazitätsplanung**: Suchreplikate (für Verfügbarkeit und Abfragedurchsatz) und Partitionen (für Indexgröße) dimensionieren; Latenz- und Throughput-Anforderungen definieren und mit realistischen Lasttests validieren; Speicherbedarf für Vektoren und Metadaten einkalkulieren.
- **Kostenfaktoren überwachen**: Einbettungsberechnungen (Token-basiert bei OpenAI/Azure OpenAI), Suchabfragen, Vektorspeicher und Datenbewegungen kontinuierlich beobachten; Caching für häufige Queries, Semantic Caching und Ergebniswiederverwendung einsetzen; Azure Cost Management für Transparenz nutzen.
- **Monitoring und Alerting**: Query-Latenzen, Fehlerraten, Index-Refresh-Zeiten und Ressourcenauslastung überwachen; proaktive Alerts bei SLA-Verletzungen oder Anomalien konfigurieren.
- **Änderungsmanagement**: Wechsel des Einbettungsmodells oder Schemaanpassungen als kontrollierte Re-Indexierung mit Staging-Index planen; Blue-Green-Deployments oder Aliasing nutzen; Abwärtskompatibilität der Anwendung durch Versionierung sicherstellen.
- **Ausfallsicherheit und Disaster Recovery**: Backup/Restore-Strategien für Indexe und Konfigurationen vorhalten; Multi-Region-Deployments für kritische Workloads evaluieren; Notfallprozeduren dokumentieren und regelmäßig testen.

Zusammenführung zum ganzheitlichen Ansatz

Azure AI Foundry bündelt den gesamten technischen Pfad von der Datenaufnahme über Vektorisierung und Hybridsuche bis zur agentenbasierten Nutzung in einer konsistenten Plattform. Diese Integration ist mehr als eine Zusammenstellung von Einzeldiensten – sie schafft einen durchgängigen Workflow, in dem jeder Schritt nahtlos ineinandergreift.

Die Datenaufnahme erfolgt über standardisierte Konnektoren (Azure AI Search Indexer für Blob Storage, Cosmos DB, SQL) und Data Pipelines. Die Transformation in durchsuchbare Repräsentationen geschieht automatisiert durch Skillsets mit integrierten Embedding-Modellen. Die Nutzung durch Agenten wird durch native SDKs, Prompt Flow-Orchestrierung (bei Hub-Basierten Projekten) und vortrainierte Evaluatoren vereinfacht.

Das Resultat sind nachvollziehbare, berechtigungskonforme Antworten mit klarer Trennung zwischen Anwendungslogik, Retrieval-Mechanismen und Governance-Strukturen:

- **Nachvollziehbarkeit**: Quellenangaben (Citations) mit Dokumentreferenzen, Seitenzahlen und Textausschnitten werden zu jedem abgerufenen Segment mitgeliefert, durchgängiges Tracing über Azure Monitor und Application Insights macht den Datenpfad transparent.
- **Berechtigungskonformität**: Security Trimming filtert Suchergebnisse basierend auf Entra ID-Identitäten und ACLs, Zugriffskontrolle erfolgt auf Dokument- und Segmentebene, Zero-Trust-Architektur mit Managed Identities.
- **Architektonische Modularität**: Klare Trennung ermöglicht unabhängige Entwicklung, Testen und Optimierung von Index-Design, Retrieval-Logik und Generation-Prompts, Versionierung von Index, Prompts und Modellen ohne Breaking Changes.

Diese Modularität ist entscheidend für die langfristige Wartbarkeit und iterative Weiterentwicklung komplexer RAG-Systeme in Unternehmensumgebungen. Die Plattform ermöglicht schnelle Experimente und kontrollierten Rollout in produktive Deployments mit vollständiger Lineage-Dokumentation.

6. Sicherheit, Identität und Netzwerk

Sicherheit ist kein nachträgliches Beiwerk, sondern ein fundamentaler Grundpfeiler jeder unternehmenskritischen KI-Plattform. Azure AI Foundry verankert Identität, Autorisierung, Netzwerkisolation und Inhaltskontrollen direkt im Ausführungsweg von Modellen und Agenten, sodass diese Mechanismen nicht umgangen werden können. Das übergeordnete Ziel ist eine minimale Angriffsfläche bei vollständiger Nachverfolgbarkeit aller Zugriffe und Operationen. In regulierten Umgebungen, etwa im Finanz- oder Versicherungswesen, ist diese tiefe Integration von Sicherheitsmechanismen nicht nur wünschenswert, sondern häufig eine Voraussetzung für den produktiven Einsatz von KI-Systemen.

Identität und Autorisierung

Die Grundlage jeder Sicherheitsarchitektur bildet ein robustes Identitäts- und Autorisierungssystem, das zuverlässig kontrolliert, wer auf welche Ressourcen zugreifen darf. Azure AI Foundry bietet hierfür mehrere Ansätze, die unterschiedliche Anforderungen und Migrationsszenarien bedienen.

Schlüssellose Anmeldung mit Microsoft Entra ID

Der empfohlene Standard für Unternehmensumgebungen ist die tokenbasierte Anmeldung über Microsoft Entra ID, die früher als Azure Active Directory bekannt war. Dieser Ansatz vermeidet die

unkontrollierte Verbreitung von API-Schlüsseln. API-Schlüssel werden oft in Konfigurationsdateien abgelegt, in Code-Repositories eingecheckt oder in Skripten hartcodiert, was zu unüberschaubaren Sicherheitslücken führt.

Die tokenbasierte Authentifizierung hingegen nutzt zeitlich begrenzte Token, die automatisch rotiert werden und deren Gültigkeit an Bedingungen geknüpft werden kann. Entwickler verwenden die Azure-Identity-Bibliotheken, insbesondere die `DefaultAzureCredential`-Klasse, die eine intelligente Kaskade von Authentifizierungsmethoden durchläuft. Lokal während der Entwicklung authentifiziert sich der Entwickler mit seinen persönlichen Anmeldedaten, während in produktiven Umgebungen verwaltete Identitäten zum Einsatz kommen. Diese Flexibilität ermöglicht es, denselben Code ohne Änderungen in verschiedenen Umgebungen zu betreiben.

Der wesentliche Vorteil liegt in der zentralen Richtliniendurchsetzung. Wenn beispielsweise ein Mitarbeiter das Unternehmen verlässt, werden durch die Deaktivierung seiner Entra-Identität automatisch alle seine Zugriffe widerrufen, ohne dass API-Schlüssel mühsam in verschiedenen Systemen gesucht und entfernt werden müssen.

API-Schlüssel als Übergangslösung

Viele Endpunkte in Azure AI Foundry akzeptieren zusätzlich zur tokenbasierten Authentifizierung auch API-Schlüssel. Diese Kompatibilität ist wichtig für Migrationsszenarien und bestimmte isolierte Anwendungsfälle. Unternehmen sollten API-Schlüssel jedoch nur übergangsweise oder für klar abgegrenzte Szenarien nutzen und mittelfristig auf die Entra-basierte Anmeldung umstellen. Der Grund liegt nicht nur in den oben beschriebenen Sicherheitsrisiken, sondern auch in den eingeschränkten Möglichkeiten zur feingranularen Zugriffskontrolle und Auditierung, die API-Schlüssel bieten.

Rollenbasierte Zugriffssteuerung

Die rollenbasierte Zugriffssteuerung in Azure AI Foundry folgt dem bewährten Prinzip der minimalen Berechtigung. Berechtigungen werden über vordefinierte Azure-Rollen vergeben, die präzise auf bestimmte Aufgaben zugeschnitten sind. Die Rolle „Cognitive Services OpenAI User" gewährt die Möglichkeit, Inferenzanfragen zu stellen und Bereitstellungen sowie Modelle anzuzeigen, ohne jedoch Änderungen vornehmen zu können. Die Rolle „Cognitive Services OpenAI Contributor" erweitert diese Berechtigungen und ermöglicht zusätzlich das Erstellen und Verwalten von Bereitstellungen, das Durchführen von Fine-Tuning sowie das Hinzufügen von Datenquellen. Administrative Aufgaben wie die Verwaltung von Kontingenten oder das Einsehen von API-Schlüsseln erfordern weitergehende Berechtigungen auf Abonnement- oder Ressourcengruppenebene.

Diese granulare Trennung ermöglicht es, Verantwortlichkeiten sauber nach Aufgabe und Geltungsbereich zu separieren. Ein Entwicklungsteam kann Inferenzrechte für Entwicklungs- und

Testumgebungen erhalten, während nur das Betriebsteam Bereitstellungen in der Produktion verwalten darf. Ein Sicherheitsteam kann Audit-Rechte erhalten, ohne selbst Inferenz durchführen zu können. Diese Trennung reduziert das Risiko versehentlicher oder böswilliger Fehlkonfigurationen erheblich.

Bedingter Zugriff für erweiterte Kontrolle

Ein besonders mächtiges Feature ist der bedingte Zugriff über „Entra Conditional Access", der sich auf menschliche Benutzer und Gruppen anwenden lässt und standort- oder risikobasierte Bedingungen erzwingt. Dies ermöglicht beispielsweise Richtlinien, die den Zugriff nur aus bestimmten vertrauenswürdigen Netzwerksegmenten erlauben oder bei erkannten Anomalien im Zugriffsverhalten zusätzliche Validierungen verlangen.

Für die Absicherung von nicht-interaktiven Identitäten wie Service Principals steht mit der Microsoft Entra Workload ID Premium Lizenz eine erweiterte Funktionalität zur Verfügung. Diese ermöglicht es, Conditional Access Policies auch auf Single-Tenant Service Principals anzuwenden, die im eigenen Tenant registriert sind. Allerdings ist der Funktionsumfang hier derzeit eingeschränkter als bei Benutzerkonten und beschränkt sich primär auf standortbasierte Filter und Risikobewertungen. Verwaltete Identitäten (Managed Identities) werden von Conditional Access Policies nicht abgedeckt und sollten stattdessen über andere Mechanismen wie Netzwerksegmentierung und Access Reviews geschützt werden.

Strategische Empfehlungen

Die Strategie sollte auf Entra-Token als Authentifizierungsstandard basieren, mit klar definierten Rollen nach dem Prinzip der minimalen Berechtigung. Breit wirkende Platzhalter in benutzerdefinierten Rollen sollten vermieden werden, da sie die Präzision der Zugriffskontrolle untergraben und im Falle einer Kompromittierung größeren Schaden anrichten können. Wo immer möglich, sollten verwaltete Identitäten gegenüber Service Principals bevorzugt werden, da sie die Credential-Verwaltung vollständig eliminieren und die Sicherheit durch automatische Rotation erhöhen.

Netzwerkisolation

Die Netzwerkisolation bildet eine zusätzliche Sicherheitsebene, die den Datenverkehr auf vertrauenswürdige Pfade beschränkt und unautorisierten Zugriff selbst bei kompromittierten Zugangsdaten erschwert. Azure AI Foundry bietet hierfür mehrere komplementäre Mechanismen.

Private Endpoints und Microsoft-Backbone

Foundry-Ressourcen und alle abhängigen Dienste wie Azure AI Services, Azure AI Search, Storage, Key Vault oder Cosmos DB lassen sich über Private Link in Ihr virtuelles Netzwerk einbinden. Dies bedeutet

konkret, dass diese Ressourcen über private IP-Adressen aus Ihrem eigenen Adressraum erreichbar werden, statt über öffentliche Internet-Endpunkte. Der Datenpfad verbleibt dabei vollständig im Microsoft-Backbone-Netzwerk, verlässt also nie das globale Microsoft-Netzwerk und ist damit vor Internet-basierten Angriffen geschützt.

Der praktische Nutzen liegt auf der Hand: Selbst, wenn ein Angreifer Zugang zum öffentlichen Internet hat, kann er ohne Zugang zu Ihrem virtuellen Netzwerk nicht auf die Ressourcen zugreifen. Dies ist besonders wichtig für Szenarien, in denen sensible Daten verarbeitet werden oder strenge Compliance-Anforderungen zu erfüllen sind.

Verwaltete virtuelle Netzwerke auf Hub-Ebene

Azure AI Foundry-Hubs unterstützen ein verwaltetes virtuelles Netzwerk (Managed Virtual Network) mit konfigurierbaren Regelwerken für ausgehenden Datenverkehr. Es ist wichtig zu verstehen, dass diese Managed Virtual Networks auf Hub-Ebene konfiguriert werden und von dort auf alle untergeordneten Projekte vererbt werden. Azure AI Foundry unterstützt derzeit kein „Bring Your Own Virtual Network" für die Compute-Isolation.

Für den Betriebsmodus stehen zwei Varianten zur Verfügung: „Allow Internet Outbound" erlaubt allen ausgehenden Internet-Datenverkehr, während „Allow Only Approved Outbound" (Nur genehmigte ausgehende Verbindungen) den ausgehenden Datenverkehr auf explizit freigegebene Ziele begrenzt. Im letztgenannten Modus wird automatisch ein Datenexfiltrationsschutz aktiviert, der verhindert, dass Daten unkontrolliert das Netzwerk verlassen können. Dies ermöglicht es, zentrale Inspection-Points durch Azure Firewall zu erzwingen, sodass jeglicher Datenverkehr, der den Hub verlässt, überwacht und gefiltert werden kann.

Dieser Ansatz ist besonders wertvoll in Szenarien, in denen Datenexfiltration verhindert werden muss. Ein kompromittierter Agent oder eine fehlerhafte Konfiguration könnte theoretisch versuchen, Daten an externe Systeme zu senden. Durch die Begrenzung des ausgehenden Datenverkehrs auf ausschließlich genehmigte Ziele wird dies technisch unterbunden. Für die Namensauflösung empfiehlt sich die Integration von Private DNS-Zone, damit Private Endpoints ordnungsgemäß aufgelöst werden.

Deaktivierung öffentlicher Zugriffe

Für Azure AI Services und Azure AI Search kann der öffentliche Zugriff auf die Datenebene vollständig deaktiviert werden, sodass ausschließlich der Zugriff über Private Endpoints möglich ist. Diese Konfiguration lässt sich über Azure Policy erzwingen, sodass auch bei versehentlichen Fehlkonfigurationen oder bei der Bereitstellung neuer Ressourcen die Sicherheitsrichtlinien automatisch durchgesetzt werden.

Die Governance durch Azure Policy ist hier entscheidend, denn sie verhindert, dass einzelne Teams oder Projekte aus Bequemlichkeit oder Unwissenheit die Netzwerkisolation aufweichen. Die Richtlinien

werden zentral definiert und automatisch auf alle relevanten Ressourcen angewendet, unabhängig davon, wie oder von wem diese erstellt werden.

Konsistente regionale Platzierung

In Agent-Szenarien ist eine konsistente regionale Platzierung aller Komponenten nicht nur empfohlen, sondern technisch erforderlich. Der Foundry-Hub, alle Projekte, OpenAI/AI-Services, Storage, Search, Cosmos DB und die verwendeten Identitäten müssen in derselben Region wie das Virtual Network bereitgestellt werden. Dies vereinfacht nicht nur die Konfiguration von Private DNS, sondern reduziert auch die Latenz erheblich und verhindert komplexe Routing-Probleme, die bei regionsübergreifender Kommunikation auftreten können.

Anbindung von On-Premises-Ressourcen

Für Hub-Workspaces existiert ein Pfad, um die verwalteten Netzwerke privat an On-Premises-Ressourcen anzubinden. Dies erfolgt über Azure Application Gateway als vermittelnde Komponente, die als sicherer Proxy fungiert. Das Application Gateway wird in Ihrem eigenen Azure Virtual Network eingerichtet und über ExpressRoute oder VPN-Gateways mit der On-Premises-Umgebung verbunden. Vom Managed Virtual Network des Hubs wird dann ein Private Endpoint zum Application Gateway erstellt, das wiederum den Zugriff auf lokale HTTP(S)-Endpunkte ermöglicht.

Diese Architektur ist relevant für hybride Szenarien, in denen KI-Anwendungen auf Datenquellen oder Dienste in lokalen Rechenzentren zugreifen müssen, ohne Daten über das öffentliche Internet übertragen zu müssen. Dabei ist zu beachten, dass Application Gateway nur HTTP(S)-Protokolle unterstützt und für andere Protokolle alternative Lösungen erforderlich sind.

Datenresidenz und regionale Verfügbarkeit

Die Frage, wo Daten physisch gespeichert und verarbeitet werden, ist für viele Organisationen von höchster Bedeutung, insbesondere in regulierten Branchen oder bei der Einhaltung von Datenschutzgesetzen wie der DSGVO.

Azure AI Foundry-Funktionen und Modelle sind an spezifische Azure-Regionen gebunden, wobei die Verfügbarkeit je nach Feature und Modell variiert. Bei der Architekturplanung ist es entscheidend, die Regionsmatrix sorgfältig zu prüfen und funktionsübergreifende Regions-Mischungen in streng regulierten Umgebungen zu vermeiden. Wenn beispielsweise ein bestimmtes Modell nur in der Region „East US" verfügbar ist, Ihre Compliance-Anforderungen aber verlangen, dass alle Daten in der EU verarbeitet werden, entsteht ein Konflikt, der bereits in der Planungsphase erkannt und adressiert werden muss.

Deployment-Typen und Datenverarbeitung

Für Modelle, die direkt von Azure bereitgestellt werden, einschließlich Azure OpenAI, ist die Wahl des Deployment-Typs entscheidend für die Datenresidenz. Azure OpenAI bietet verschiedene Deployment-Optionen mit unterschiedlichen Garantien bezüglich der Datenverarbeitung:

- **Standard (Regional) Deployments** verarbeiten und speichern Daten ausschließlich in der gewählten Azure-Region. Dies bietet die strengste Datenresidenz-Garantie, kann jedoch bei hohem Durchsatz zu Kapazitätsengpässen führen.
- **Global Standard Deployments** nutzen die globale Azure-Infrastruktur und können Datenverkehr dynamisch zu jedem Rechenzentrum weltweit routen. Während Daten im Ruhezustand in der designierten Azure-Region verbleiben, kann die Inferenzverarbeitung in jeder verfügbaren Azure OpenAI Location weltweit erfolgen. Dies maximiert Performance und Verfügbarkeit, entspricht jedoch nicht strengen EU-Datenresidenz-Anforderungen.
- **Data Zone Standard Deployments** stellen einen Mittelweg dar: Sie nutzen die globale Infrastruktur, beschränken die Datenverarbeitung aber auf eine von Microsoft definierte Data Zone. Für EU-basierte Ressourcen bedeutet dies, dass die Inferenzverarbeitung innerhalb der EU-Mitgliedsstaaten erfolgt, während Daten im Ruhezustand in der ursprünglichen Region verbleiben.

EU Data Boundary und Compliance

Microsoft hat im Februar 2025 die EU Data Boundary für Microsoft Cloud Services vollständig implementiert. Diese Initiative umfasst Microsoft 365, Dynamics 365, Power Platform und die meisten Azure Services, einschließlich Azure OpenAI. Die EU Data Boundary gewährleistet, dass Kundendaten und pseudonymisierte personenbezogene Daten innerhalb der EU und EFTA-Regionen gespeichert und verarbeitet werden.

Für Azure OpenAI ist jedoch zu beachten, dass die tatsächliche Datenresidenz vom gewählten Deployment-Typ abhängt. Organisationen mit strikten DSGVO-Anforderungen, die sicherstellen müssen, dass alle Datenverarbeitungen ausschließlich in der EU stattfinden, sollten explizit Data Zone Standard (EUR) Deployments wählen. Global Standard Deployments erfüllen diese Anforderung nicht, da die Inferenzverarbeitung weltweit erfolgen kann.

Die relevanten Datenschutzhinweise, Product Terms und die Data Protection Addendum sollten vor der Implementierung sorgfältig geprüft werden, um sicherzustellen, dass der gewählte Deployment-Typ den spezifischen Compliance-Anforderungen Ihrer Organisation entspricht. Für europäische Organisationen bietet die Kombination aus EU-Region, Data Zone Standard Deployment und den Microsoft EU Data Boundary-Commitments die höchste Sicherheit bezüglich der Datenresidenz.

Inhalts- und Qualitätsschutz

Neben der Kontrolle darüber, wer auf Systeme zugreifen darf und wie der Datenverkehr fließt, ist auch die Kontrolle darüber wichtig, welche Inhalte verarbeitet und generiert werden. Azure AI Foundry bietet hierfür mehrere Schutzmechanismen.

Content Filtering für sichere Interaktionen

Sowohl eingehende Prompts als auch generierte Antworten werden über Klassifizierer geprüft, die verschiedenen Kategorien problematischer Inhalte erkennen. Die Konfiguration erlaubt es, Schwellenwerte pro Kategorie festzulegen, sodass Teams den Kompromiss zwischen Sicherheit und Flexibilität entsprechend ihren Anforderungen justieren können. Für Modelle aus der Kategorie „Models sold directly by Azure" ist Content Filtering standardmäßig aktiviert, was ein grundlegendes Sicherheitsniveau ohne zusätzliche Konfiguration gewährleistet.

Prompt Shields gegen Angriffsvektoren

Prompt Shields bieten spezialisierten Schutz gegen direkte und indirekte Prompt-Angriffe. Direkte Angriffe, oft als „Jailbreaks" bezeichnet, versuchen, das Modell durch geschickt formulierte Prompts dazu zu bringen, seine Sicherheitsrichtlinien zu umgehen. Indirekte Angriffe (auch „Indirect Prompt Injection Attacks" oder „Cross-Domain Prompt Injection Attacks" genannt) verstecken bösartige Anweisungen in Dokumenten, die das Modell verarbeitet, etwa in einer PDF-Datei, die über ein RAG-System eingebunden wird.

Die Integration von Prompt Shields erfolgt direkt in Foundry-Workflows, sodass der Schutz automatisch greift, ohne dass Anwendungscode geändert werden muss. Die Signale aus dieser Schutzschicht lassen sich in zentrale Sicherheitsüberwachung einbinden und mit weiteren Sicherheitsereignissen korrelieren.

Groundedness-Erkennung für faktenbasierte Antworten

In RAG-Szenarien, wo Modelle auf externe Wissensquellen zugreifen, ist es kritisch, dass generierte Antworten tatsächlich durch die abgerufenen Quellen gestützt werden. Die Groundedness-Erkennung mittels Azure AI Content Safety prüft, ob Antworten durch die zur Verfügung gestellten Quellen gedeckt sind. Optional können ungestützte Aussagen automatisch korrigiert werden, indem sie entfernt oder durch qualifizierte Formulierungen ersetzt werden.

Diese Funktionalität ist mehr als nur ein Qualitätsmerkmal – in vielen Anwendungsfällen, etwa im rechtlichen oder medizinischen Bereich, kann die Generierung ungestützter Behauptungen ernsthafte Konsequenzen haben. Die Ergebnisse der Groundedness-Prüfung fließen in Evaluierungen und Metriken ein, sodass die Qualität kontinuierlich überwacht werden kann.

Schlüsselverwaltung und Verschlüsselung

Die Verwaltung kryptographischer Schlüssel ist ein fundamentaler Aspekt der Datensicherheit, der besondere Aufmerksamkeit verdient. Azure AI Foundry bietet hier mehrere Optionen, die unterschiedliche Kontroll- und Compliance-Anforderungen bedienen.

Projekte verwenden standardmäßig einen Microsoft-verwalteten Key Vault für die Speicherung von Geheimnissen wie Verbindungszeichenfolgen oder API-Schlüsseln. Dieser ist nicht in Ihrem Abonnement sichtbar, funktioniert ohne zusätzliche Konfiguration und ist für viele Szenarien ausreichend. Organisationen mit strengeren Anforderungen können jedoch einen eigenen Key Vault anbinden, um die vollständige Kontrolle über den Lebenszyklus ihrer Geheimnisse zu behalten. Wichtig ist, dass pro Foundry-Ressource (Hub) nur eine Key Vault-Verbindung gleichzeitig möglich ist. Diese Verbindung verwaltet dann alle Secrets auf Ressourcen- und Projektebene. Die Planung von Verbindungen und insbesondere der Löschreihenfolge muss sorgfältig durchdacht werden: Eine Key Vault-Verbindung kann nur gelöscht werden, wenn keine anderen Verbindungen mehr existieren. Da Foundry keine Secret-Migration unterstützt, müssen bei einem Wechsel alle Verbindungen neu erstellt werden.

Für ruhende Daten kann die Foundry-Ressource konfiguriert werden, um kundenseitig verwaltete Schlüssel (Customer-Managed Keys, CMK) zu verwenden. Dies gibt Organisationen die Möglichkeit, ihre Verschlüsselungsschlüssel selbst zu rotieren oder im Ernstfall den Zugriff auf verschlüsselte Daten zu widerrufen, ohne auf Microsoft angewiesen zu sein. Diese Funktion ist besonders in regulierten Umgebungen relevant, wo explizite Anforderungen an die Kontrolle über Verschlüsselungsschlüssel bestehen.

Betriebs- und Sicherheitsstatus-Überwachung

Sicherheit ist kein einmaliger Konfigurationsvorgang, sondern ein kontinuierlicher Prozess, der laufende Überwachung und Anpassung erfordert. Microsoft Defender for Cloud bietet hierfür spezialisierte Funktionen für KI-Workloads.

Die Erkennung umfasst die automatische Identifikation von KI-Ressourcen in Ihrer Azure-Umgebung. Die Bewertung des Sicherheitsstatus analysiert Konfigurationen gegen Best Practices und identifiziert Abweichungen. Die Laufzeiterkennung überwacht den tatsächlichen Betrieb generativer Anwendungen und erkennt anomales Verhalten in Echtzeit.

Die generierten Warnungen umfassen verschiedene Bedrohungskategorien. Datenabfluss-Warnungen erkennen ungewöhnliche Muster beim Zugriff auf oder der Übertragung von Daten. Prompt-Angriffs-Warnungen identifizieren Versuche, Systeme durch bösartige Eingaben zu kompromittieren. Credential-Theft-Warnungen erkennen, wenn Anmeldeinformationen in Modellantworten enthalten

sind. Data-Poisoning-Warnungen identifizieren Versuche, Trainingsdaten zu korrumpieren. Anomalie-Warnungen erfassen statistisch ungewöhnliches Verhalten, das auf eine Kompromittierung hindeuten könnte. Die Integration von Prompt Shields bedeutet, dass erkannte Angriffsversuche nicht nur blockiert, sondern auch als Sicherheitsereignisse in Defender XDR dokumentiert werden, was forensische Untersuchungen und die Verbesserung der Sicherheitsstrategie ermöglicht.

Ganzheitliche Sicherheitsstrategie

Die verschiedenen Sicherheitsmechanismen in Azure AI Foundry bilden keine isolierten Features, sondern fügen sich zu einer ganzheitlichen Sicherheitsstrategie zusammen. Die Identitäts- und Autorisierungsmechanismen kontrollieren den Zugang auf Personenebene. Die Netzwerkisolation schützt den Datenverkehr auf Infrastrukturebene. Content Filtering und Prompt Shields sichern die Interaktionen auf Inhaltsebene. Die Verschlüsselung schützt ruhende und übertragene Daten. Die kontinuierliche Überwachung durch Defender erkennt und meldet Anomalien im laufenden Betrieb.

Für eine robuste Implementierung empfiehlt sich die bevorzugte Nutzung von Entra ID-Authentifizierung statt API-Schlüsseln, kombiniert mit der Durchsetzung von bedingtem Zugriff auch für Service Principals und Managed Identities. Der öffentliche Zugriff sollte deaktiviert und stattdessen Private Endpoints für alle Foundry-Ressourcen und Abhängigkeiten genutzt werden. Verwaltete virtuelle Netzwerke mit explizit genehmigten Zielen für ausgehenden Datenverkehr bieten zusätzlichen Schutz, wobei DNS- und Routing-Pfade sorgfältig dokumentiert werden sollten. Content Filtering, Prompt Shields und Groundedness-Evaluierung sollten vor der Produktivsetzung aktiviert und getestet werden. Die Einbindung von Defender for Cloud für die Überwachung des Sicherheitsstatus und Laufzeit-Signale rundet die Sicherheitsstrategie ab.

Diese mehrschichtige Verteidigung, auch als „Defense in Depth" bekannt, stellt sicher, dass selbst wenn eine Sicherheitsebene überwunden wird, weitere Mechanismen den Schaden begrenzen oder die Kompromittierung erkennen können. Dies ist besonders wichtig in der dynamischen Welt der KI-Sicherheit, wo neue Angriffsvektoren kontinuierlich entdeckt werden und Systeme sich entsprechend anpassen müssen.

7. Qualität, Safety und Red Teaming

Produktionsreife KI-Systeme erfordern messbare Qualität und nachweisbare Sicherheit. Dies ist keine optionale Ergänzung, sondern eine fundamentale Anforderung für den verantwortungsvollen Einsatz generativer KI in unternehmenskritischen Szenarien. Azure AI Foundry liefert für diese Herausforderung umfassende Evaluationsfunktionen, die sowohl im Portal als auch programmatisch per SDK genutzt werden können. Ergänzt wird dies durch einen AI Red Teaming Agent, welcher aktuell

ein Preview bereit steht, der automatisierte, adversariale Überprüfungen ermöglicht. Diese Werkzeuge erlauben es, Evaluierungen lokal während der Entwicklung, in der Cloud unter produktionsnahen Bedingungen und kontinuierlich im laufenden Betrieb auszuführen. Die Ergebnisse werden in Projekten zentral erfasst und mit Observability-Signalen verknüpft, sodass eine durchgängige Qualitäts- und Sicherheitsüberwachung über den gesamten Lebenszyklus hinweg möglich wird.

Evaluationen und Metriken im KI-Lebenszyklus

Evaluierungen sind systematische, wiederholbare Tests, die gegen Modelle, Agenten oder ganze Anwendungen ausgeführt werden. Sie bilden das objektive Fundament für Qualitätsentscheidungen und ersetzen das subjektive „es fühlt sich gut an" durch messbare Kennzahlen. Die Ausführung kann auf zwei Wegen erfolgen: entweder grafisch gesteuert im Portal ohne Code-Anforderungen oder programmatisch mit dem Azure AI Evaluation SDK, das einen „Evaluation as Code"-Ansatz ermöglicht. Der zweite Weg ist besonders wertvoll für die Integration in automatisierte Pipelines und ermöglicht es, Evaluierungen als versionierten Code zu verwalten.

Das Spektrum der Metrik-Familien

Die verfügbaren Metriken decken verschiedene Dimensionen der KI-Qualität und -Sicherheit ab, die zusammen ein umfassendes Bild der Systemleistung zeichnen. Im Bereich der Qualitätsmetriken steht Groundedness an vorderster Stelle, besonders für RAG-Szenarien, wo sie misst, ob Antworten tatsächlich durch die bereitgestellten Quellen gestützt werden. Relevanz bewertet, wie gut eine Antwort zur gestellten Frage passt, während Kohärenz die innere Konsistenz und logische Struktur der Antwort prüft. Flüssigkeit misst die sprachliche Natürlichkeit und Lesbarkeit, und Ähnlichkeit quantifiziert, wie nah eine generierte Antwort an einer Referenzantwort liegt.

Ergänzend dazu stehen klassische NLP-Metriken aus der Forschung zur Verfügung. BLEU, ursprünglich für maschinelle Übersetzung entwickelt, misst n-Gramm-Überlappungen zwischen generiertem und Referenztext. ROUGE fokussiert auf Recall-orientierte Metriken und wird häufig für Zusammenfassungen verwendet. METEOR berücksichtigt Synonyme und Wortstammformen für eine semantisch sensitivere Bewertung. GLEU ist eine Variante von BLEU mit verbesserter Strafberechnung, und F1-Scores kombinieren Präzision und Recall für eine ausgewogene Bewertung.

Für agentenbasierte KI-Systeme existieren spezialisierte Metriken, die deren einzigartige Eigenschaften adressieren. Task Adherence bewertet, ob ein Agent die ihm zugewiesene Aufgabe tatsächlich erfüllt, ohne davon abzuweichen oder zusätzliche, nicht angeforderte Aktionen durchzuführen. Die Tool-Call-Genauigkeit misst, ob Agenten die richtigen Werkzeuge zur richtigen Zeit mit korrekten Parametern aufrufen. Die Intentauflösung prüft, ob Agenten die Absicht hinter Benutzeranfragen korrekt identifizieren und entsprechend handeln.

Die Safety-Metriken adressieren verschiedene Kategorien problematischer Inhalte. Dazu gehören Hate und Unfairness, die hasserfüllte oder diskriminierende Inhalte erkennen, sowie Gewalt, Sexual und Selbstverletzung als weitere kritische Kategorien. Indirekte Angriffe erfassen Versuche, Sicherheitsmechanismen durch in Dokumente eingebettete Anweisungen zu umgehen. Schutzrechte prüfen, ob generierte Inhalte urheberrechtlich geschütztes Material reproduzieren.

Schließlich gibt es betriebliche Metriken, die die technische Leistung quantifizieren. Latenz misst die Antwortzeit, Durchsatz die Anzahl verarbeiteter Anfragen pro Zeiteinheit, und Kostenabschätzungen helfen bei der Budgetplanung und -kontrolle. Diese Metriken sind entscheidend, um zu verstehen, ob ein System nicht nur qualitativ hochwertig, sondern auch wirtschaftlich tragfähig und performant genug für produktive Workloads ist.

Der Evaluationsprozess in der Praxis

Der typische Ablauf einer Evaluation beginnt mit der Definition von Testsätzen. Teams stellen Sammlungen von Eingaben zusammen, ergänzt um Referenzantworten, die das gewünschte Verhalten repräsentieren, und optional um Kontextquellen, die das System zur Beantwortung nutzen sollte. Diese Testsätze sollten die reale Nutzung widerspiegeln und sowohl typische als auch Randfälle abdecken. Foundry berechnet dann die konfigurierten Metriken und präsentiert die Ergebnisse in verschiedenen Perspektiven: Verteilungen zeigen, wie sich Scores über den Testsatz verteilen, Schwellenverletzungen identifizieren Fälle, die definierte Mindestanforderungen unterschreiten, und Trends über mehrere Evaluationsläufe hinweg machen Verbesserungen oder Verschlechterungen sichtbar.

Die Flexibilität bei der Ausführung ist bemerkenswert. Evaluierungen können lokal auf dem Entwicklerrechner ausgeführt werden, was schnelles Feedback während der Entwicklung ermöglicht, und die Ergebnisse werden dann ins Projekt hochgeladen für die zentrale Erfassung. Alternativ können sie als Remote-Runs in der Cloud skaliert werden, was besonders bei großen Testsätzen oder rechenintensiven Evaluatoren sinnvoll ist, die mehr Ressourcen benötigen als auf einem einzelnen Rechner verfügbar sind.

Groundedness und automatisierte Korrektur

Für RAG-Szenarien ist die Groundedness-Erkennung von zentraler Bedeutung. Sie prüft systematisch, ob jede Aussage in einer generierten Antwort durch die bereitgestellten Quellen faktisch gestützt wird. Dies geht über eine einfache Ähnlichkeitsprüfung hinaus und bewertet, ob die semantische Bedeutung einer Aussage tatsächlich in den Quellen nachweisbar ist. Eine besonders interessante Funktion, die sich derzeit in Preview befindet, ist die automatische Korrektur. Sie kann ungestützte Passagen erkennen und automatisch an die Quellen angleichen. Diese Funktion sollte jedoch mit Bedacht eingesetzt werden, da automatische Änderungen die Intention der ursprünglichen Antwort verändern können. Eine sorgfältige Validierung ist unerlässlich, bevor diese Funktion in produktiven Systemen aktiviert wird.

Integration in CI/CD-Pipelines

Evaluierungen entfalten ihre volle Wirkung, wenn sie als Quality Gates in Continuous-Integration- und Continuous-Deployment-Pipelines integriert werden. Microsoft stellt dafür GitHub Actions und einen cloudbasierten Evaluationspfad bereit. Das Konzept ist: Bevor eine neue Version eines Agenten oder Modells in die Produktion übertragen wird, muss sie definierte Qualitätsschwellen überschreiten. Wenn beispielsweise die Groundedness unter 0,8 fällt oder die Relevanz unter einen festgelegten Wert sinkt, wird das Deployment automatisch blockiert. Dies verhindert, dass Regressionen in die Produktion gelangen und stellt sicher, dass jede Version mindestens die Qualität der vorherigen Version erreicht.

Kontinuierliche Evaluation und Observability

Die Evaluation endet nicht mit dem Deployment. Foundry unterstützt die kontinuierliche Evaluierung von Agenten im laufenden Betrieb, was einen fundamentalen Perspektivwechsel darstellt. Während Vor-Deployment-Tests auf kuratierten Testsätzen basieren, bewertet kontinuierliche Evaluation echte Produktionsinteraktionen. Eine konfigurierbare Stichprobe laufender Interaktionen wird fortlaufend auf Qualität, Safety und Performance bewertet, sodass Teams verstehen, wie ihr System sich unter realen Bedingungen mit echten Nutzern verhält.

Die Ergebnisse dieser kontinuierlichen Bewertung erscheinen im Observability-Dashboard und sind mit den detaillierten Traces einzelner Interaktionen verknüpft. Wenn beispielsweise ein Qualitätsscore auffällig niedrig ist, können Entwickler direkt zur zugehörigen Interaktion springen und den kompletten Ablauf nachvollziehen – welche Prompts verwendet wurden, welche Werkzeuge aufgerufen wurden, welche Quellen abgerufen wurden. Diese Verbindung zwischen aggregierten Metriken und individuellen Traces ist entscheidend für das Debugging und die gezielte Verbesserung.

Die Integration mit Azure Monitor und Application Insights erweitert diese Funktionalität erheblich. Teams können benutzerdefinierte Abfragen formulieren, um spezifische Muster zu identifizieren, Dashboards erstellen, die verschiedene Metriken in Beziehung setzen, und Warnungen konfigurieren, die bei kritischen Verschlechterungen automatisch Benachrichtigungen auslösen. Ein praktisches Beispiel wäre eine Warnung, die ausgelöst wird, wenn die durchschnittliche Groundedness über einen Zeitraum von 30 Minuten unter einen kritischen Schwellenwert fällt, was auf ein Problem mit den Datenquellen oder der Retrieval-Logik hindeuten könnte.

Es ist wichtig zu beachten, dass kontinuierliche Evaluation ein Foundry-Projekt und eine verbundene Application-Insights-Ressource voraussetzt. Zudem ist die Funktion nicht in allen Azure-Regionen verfügbar, sodass die Dokumentation bezüglich unterstützter Regionen konsultiert werden sollte, bevor diese Funktion in die Architektur eingeplant wird.

AI Red Teaming Agent für adversariale Prüfungen

Der AI Red Teaming Agent als Preview-Feature automatisiert einen Prozess, der traditionell manuell und zeitaufwändig ist: die adversariale Prüfung von KI-Systemen durch simulierte Angriffe. Der Agent nutzt PyRIT, das Python Risk Identification Toolkit for generative KI, in Kombination mit Foundry-Evaluatoren, um systematisch Angriffe zu generieren, deren Erfolg zu bewerten und die Ergebnisse zu dokumentieren. Die zentrale Kennzahl ist die Attack Success Rate, kurz ASR, die quantifiziert, wie viele Angriffsversuche erfolgreich die Sicherheitsmechanismen umgehen konnten.

Abgedeckte Risikokategorien und Angriffstechniken

Der Agent prüft verschiedene Risikokategorien, die besonders für Text-Szenarien relevant sind. Dazu gehören Hateful und Unfair Content, Sexual Content, Violence und Self-Harm. Für jede Kategorie generiert der Agent Angriffe, die darauf abzielen, das System dazu zu bringen, problematische Inhalte zu erzeugen, die normalerweise durch Content-Filter blockiert werden sollten.

Die verwendeten Angriffstechniken sind vielfältig und spiegeln reale Umgehungsversuche wider, die in der Praxis beobachtet wurden. Unicode-Verfremdung ersetzt Buchstaben durch ähnlich aussehende Unicode-Zeichen, um Textfilter zu umgehen. Base64- oder ROT13-Kodierung verschlüsselt bösartige Anweisungen in der Hoffnung, dass das Modell sie dekodiert und ausführt, während Filter nur die kodierte Form sehen. Zeichen-Flips vertauschen oder spiegeln Buchstaben. Leetspeak ersetzt Buchstaben durch Zahlen oder Sonderzeichen in einer Weise, die für Menschen lesbar bleibt, aber Textfilter verwirren kann.

Jailbreak-Suffixe sind besonders raffinierte Techniken, bei denen an eine scheinbar harmlose Anfrage ein speziell konstruiertes Suffix angehängt wird, das das Modell dazu bringt, seine Sicherheitsrichtlinien zu ignorieren. Indirekte Angriffe verstecken bösartige Anweisungen in Dokumenten, die das System über RAG einbindet, sodass die eigentliche Benutzeranfrage harmlos aussieht, während die kompromittierenden Anweisungen aus einem verarbeiteten Dokument stammen.

Diese Varianten testen systematisch verschiedene Dimensionen der Systemrobustheit: die Fähigkeit, Umgehungen von Schutzmechanismen zu erkennen, die Stabilität bei ungewöhnlich langen Eingaben, die möglicherweise das Context Window erschöpfen oder Performance-Probleme auslösen könnten, und das Verhalten bei mehrdeutigen oder absichtlich verwirrenden Eingaben.

Integration in den Entwicklungs- und Sicherheitsprozess

Der AI Red Teaming Agent sollte als Ergänzung zu manuellen Red-Team-Übungen verstanden werden, nicht als deren Ersatz. Menschliche Red Teams bringen Kreativität und kontextbezogenes Verständnis mit, die automatisierte Systeme nicht vollständig replizieren können. Die Kombination beider Ansätze ist am wirksamsten: Der automatisierte Agent führt systematische, umfassende Tests durch, die verschiedene Angriffstechniken und Risikokategorien standardisiert abdecken, während menschliche

Experten sich auf komplexe, kontextspezifische Szenarien und neuartige Angriffsvektoren konzentrieren können.

Die Ergebnisse werden als detaillierter Evaluationsbericht dokumentiert, der nach Kategorien und spezifischen Techniken aufgeschlüsselt ist. Diese Dokumentation kann über Zeit nachverfolgt werden, um zu zeigen, wie sich die Sicherheitsposition des Systems entwickelt. Wenn die ASR über mehrere Testzyklen hinweg sinkt, ist dies ein objektiver Beweis für Verbesserungen in den Sicherheitsmechanismen.

Der Agent sollte mit Content-Safety-Filtern und projektspezifischen Guardrails kombiniert werden. Content-Safety-Filter bilden die erste Verteidigungslinie gegen problematische Inhalte, während Guardrails anwendungsspezifische Regeln durchsetzen, etwa dass ein medizinischer Chatbot keine Finanzberatung geben sollte. Das Ziel all dieser Maßnahmen ist das frühzeitige Aufdecken von Schwachstellen, lange bevor das System produktiv geht und echten Nutzern ausgesetzt ist.

Strategien für robuste Qualitäts- und Sicherheitsprozesse

Die erfolgreiche Implementierung von Qualitäts- und Sicherheitsmechanismen erfordert einen systematischen, mehrschichtigen Ansatz, der verschiedene Aspekte des KI-Lebenszyklus adressiert.

Das Fundament bildet die Messbarkeit. Teams sollten repräsentative Testsätze anlegen, die sowohl typische als auch herausfordernde Szenarien abdecken. Die Auswahl relevanter Metriken sollte sich an den spezifischen Anforderungen der Anwendung orientieren – ein Kundenservice-Chatbot benötigt andere Metriken (z.B. Relevanz, Kohärenz, Tonalität) als ein technisches Dokumentationssystem (z.B. Faktentreue, Vollständigkeit, Präzision). Schwellenwerte müssen definiert werden, die klare Qualitätsanforderungen repräsentieren, aber auch realistisch erreichbar sind. Diese Evaluationsläufe sollten vor jedem Release automatisch ausgeführt und als Pipeline-Gates integriert werden, sodass keine Version in die Produktion gelangen kann, die die definierten Standards nicht erfüllt.

Für RAG-Systeme ist besondere Sorgfalt erforderlich. Die Groundedness-Prüfung sollte aktiviert und konsequent genutzt werden, um sicherzustellen, dass alle Antworten faktisch gestützt sind. Die Retrieval-Abdeckung muss getestet werden – erfasst das System die relevanten Dokumente? Die Antwortstützung muss validiert werden – nutzt das System die abgerufenen Dokumente tatsächlich? LLM-basierte Nachbearbeitung von Antworten (z.B. zur Verbesserung der Lesbarkeit oder Strukturierung) sollte nur nach bewusster Entscheidung und umfassender Validierung eingesetzt werden, da sie subtile Bedeutungsänderungen oder Informationsverluste einführen kann.

Die Beobachtung im laufenden Betrieb schließt die Lücke zwischen Vor-Deployment-Tests und realer Nutzung. Kontinuierliche Evaluation sollte aktiviert werden, um ein kontinuierliches Bild der Systemqualität unter Produktionsbedingungen zu erhalten. Die Verknüpfung von Traces und Metriken

mit Application Insights ermöglicht tiefgehende Analysen und die Korrelation von Qualitätssignalen mit anderen Betriebsmetriken wie Latenz oder Fehlerrate. Alarme sollten definiert werden, die bei signifikanten Abweichungen von erwarteten Qualitätsniveaus automatisch Benachrichtigungen auslösen.

Die Robustheit gegenüber Adversarial Attacks muss regelmäßig getestet werden. AI-Red-Teaming-Scans sollten in regelmäßigen Abständen durchgeführt werden, nicht nur einmalig vor der ersten Produktivsetzung. Die Attack Success Rate (ASR) sollte bewertet und über Zeit verfolgt werden, um Trends zu erkennen. Ein Nicht-Bestehen dieser Tests sollte als harter Blocker für die Produktivsetzung behandelt werden, unabhängig von anderen Qualitätsmetriken. Sicherheit ist keine verhandelbare Anforderung, und ein System, das anfällig für bekannte Angriffsmuster ist, sollte nicht produktiv gehen, bis die Schwachstellen adressiert sind.

Zusammenführung zur ganzheitlichen Qualitätsstrategie

Die verschiedenen Komponenten – Evaluationen mit diversen Metrik-Familien, kontinuierliche Überwachung im Betrieb und adversariale Tests durch Red Teaming – fügen sich zu einer umfassenden Qualitäts- und Sicherheitsstrategie zusammen. Vor-Deployment-Evaluationen stellen sicher, dass neue Versionen Mindeststandards erfüllen. Kontinuierliche Evaluation überwacht die tatsächliche Leistung unter realen Bedingungen. Red Teaming deckt Sicherheitslücken auf, die durch normale Tests möglicherweise nicht erfasst werden.

Diese mehrschichtige Strategie ist notwendig, weil keine einzelne Testmethode alle Aspekte der KI-Qualität und -Sicherheit erfassen kann. Kuratierte Testsätze können nie alle realen Nutzungsszenarien abdecken. Produktionsüberwachung ist reaktiv und kann Probleme erst nach ihrem Auftreten identifizieren, nicht präventiv verhindern. Red Teaming fokussiert auf adversariale Szenarien, aber nicht auf typische Qualitätsprobleme. Nur die Kombination aller drei Ansätze bietet das Sicherheitsnetz, das für den verantwortungsvollen Einsatz generativer KI in geschäftskritischen Anwendungen erforderlich ist.

Die Integration dieser Praktiken in den Entwicklungsprozess etabliert datengetriebene Qualitätsstandards, bei denen Entscheidungen auf objektiven Metriken basieren. Teams können die Entwicklung von ASR, Groundedness-Scores, Relevanz und anderen Kennzahlen über Modellversionen hinweg nachverfolgen und so kontinuierliche Verbesserungen systematisch validieren und dokumentieren.

8. Observability und Monitoring

Beobachtbarkeit ist für produktive KI-Systeme keine optionale Zusatzfunktion, sondern eine fundamentale Anforderung. Ohne tiefe Einblicke in das Laufzeitverhalten können Teams weder die Qualität ihrer Anwendungen sicherstellen noch Probleme effektiv diagnostizieren oder die Nutzung und entstehenden Kosten verstehen. Azure AI Foundry begegnet dieser Herausforderung durch die umfassende Integration von Traces, Metriken und Logs in Azure Monitor und Application Insights. Diese Integration liefert detaillierte Einblicke in Verhalten, Qualität, Sicherheit, Nutzung und Kosten von Modellen und Agenten – sowohl über grafische Portal-Ansichten als auch durch mächtige KQL-Abfragen für komplexe Analysen. Ein spezialisiertes Observability-Dashboard in Foundry ergänzt die allgemeinen Azure-Monitor-Sichten und ist spezifisch auf die Anforderungen agentischer Workloads zugeschnitten.

End-to-End-Ablaufverfolgung mit OpenTelemetry

Das Fundament der Beobachtbarkeit in Azure AI Foundry bildet die Ablaufverfolgung auf Basis von OpenTelemetry und dem W3C Trace Context Standard. Diese Wahl offener Standards ist strategisch bedeutsam, da sie Vendor-Lock-in vermeidet und die Integration mit verschiedenen Monitoring-Plattformen ermöglicht.

Die Ablaufverfolgung erfasst jeden Aufruf als strukturierten Trace mit hierarchischem Span, die den Weg einer Anfrage durch das System dokumentieren. Die Hierarchie beginnt typischerweise auf Anwendungsebene, führt über den Foundry-Endpunkt oder Agent Service, verzweigt sich in Werkzeugaufrufe wie Abfragen an Azure AI Search oder Datenbankzugriffe und endet schließlich im Antwortpfad zurück zur Anwendung.

Jeder Span trägt detaillierte Attribute, die für die Analyse kritisch sind:

- Modellkennung: Welches spezifische Modell oder welche Modellversion wurde verwendet
- Latenzwerte: Für jeden Schritt benötigte Zeit
- Statuscodes: Erfolg oder Fehler
- Token-Zähler: Anzahl verarbeiteter Input- und Output-Token (relevant für Kostenanalysen und Prompt-Optimierung)

Diese Traces sind sowohl im Foundry-Portal in einer für KI-Workloads optimierten Darstellung sichtbar als auch vollständig in Application Insights verfügbar, wo sie mit KQL analysiert werden können.

Automatische Instrumentierung und Aktivierung

Ein besonders wertvolles Feature ist die automatische Instrumentierung für Agenten. Diese erzeugen Traces ohne zusätzlichen Instrumentierungscode von Entwicklern. Die Telemetrie wird transparent

erfasst, sobald die entsprechenden SDK-Einstellungen aktiviert sind. Die SDK-Dokumentation bietet ausführliche Beispiele, die zeigen, wie die OpenTelemetry-Telemetrie aktiviert und einem Application-Insights-Arbeitsbereich zugeordnet wird.

Für Prompt-Flow-Bereitstellungen im hub-basierten Modell können Traces und System Metriken erfasst werden, wenn die entsprechenden Diagnoseoptionen aktiviert oder die Eigenschaft `app_insights_enabled: true` in der Deployment-YAML-Datei gesetzt wird. Diese Konfiguration ermöglicht auch für ablaufbasierte Orchestrierungen vollständige Observability.

Sampling-Strategien für Kosteneffizienz

Bei der Erfassung von Telemetriedaten entsteht ein fundamentales Spannungsfeld zwischen Vollständigkeit und Kosten. Die vollständige Erfassung aller Traces liefert maximale Einsicht, verursacht aber bei hohen Durchsatzraten erhebliche Ingestions- und Speicherkosten.

Application Insights Custom Sampler

Application Insights bietet einen spezialisierten Sampler in den Azure-Monitor-OpenTelemetry-Distributionen. Dieser trifft Sampling-Entscheidungen am Ende der Span-Generierung – nachdem ein Span vollständig ist, aber bevor er exportiert wird. Obwohl dies in mancher Hinsicht tail-based Sampling ähnelt, wartet der Sampler nicht darauf, mehrere Spans desselben Trace zu sammeln. Stattdessen nutzt er einen Hash der Trace-ID, um Trace-Vollständigkeit sicherzustellen. Dieser Ansatz balanciert Trace-Vollständigkeit und Effizienz und vermeidet die höheren Kosten eines vollständigen tail-based Samplings.

Der Nachteil: Die Sampling-Entscheidung wird getroffen, bevor bekannt ist, ob die Anfrage interessant sein wird. Eine Anfrage, die in einem kritischen Fehler endet, wird mit derselben Wahrscheinlichkeit gesampelt wie eine erfolgreiche Routine-Anfrage.

Vollständiges Tail-based Sampling

Tail-based Sampling adressiert diese Limitierung, indem es die Entscheidung erst trifft, nachdem eine Anfrage vollständig verarbeitet wurde. Dies ermöglicht intelligentere Strategien – etwa alle fehlgeschlagenen Anfragen zu behalten oder solche mit ungewöhnlich hoher Latenz. Allerdings erfordert echtes tail-based Sampling den Einsatz des OpenTelemetry Collectors oder eines Drittanbieter-Exporters, da die Traces zunächst vollständig erfasst und zwischengespeichert werden müssen. Diese zusätzliche Infrastruktur bedeutet mehr Komplexität, kann aber in produktiven Umgebungen mit hohem Durchsatz die Observability-Qualität erheblich verbessern.

Notbremse: Tageslimit

Als zusätzliche Schutzmaßnahme kann ein Tageslimit in Application Insights konfiguriert werden, dass die Ingestion deckelt. Dies fungiert als Notbremse für den Fall unerwarteter Telemetrie-Explosionen, etwa durch fehlerhafte Konfigurationen oder Angriffe, die zu einer Flut von Anfragen führen.

Integration in Azure Monitor und Application Insights

Die erfassten Telemetriedaten fließen in Log Analytics und Application Insights, wo sie zum Gegenstand umfassender Analysen werden. Beide Dienste bieten komplementäre Perspektiven auf die gleichen Daten und nutzen die Kusto Query Language als mächtige Abfragesprache. KQL ermöglicht es, komplexe Fragen zu stellen, die mehrere Dimensionen kombinieren, Zeitreihen analysieren und Anomalien identifizieren.

Workbooks und Dashboards bieten vorkonfigurierte oder benutzerdefinierte Visualisierungen, die Schlüssel Metriken auf einen Blick erfassbar machen. Live Metriken liefern Echtzeit-Einblicke in das laufende System mit minimaler Latenz zwischen Ereignis und Darstellung. Alert-Regeln ermöglichen es, proaktiv auf Schwellenverletzungen oder Anomalien zu reagieren, indem sie Benachrichtigungen an definierte Empfänger senden oder automatisierte Reaktionen auslösen.

Die möglichen Abfragen und Visualisierungen decken ein breites Spektrum ab. Fehlerquoten zeigen, wie viele Anfragen scheitern, idealerweise aufgeschlüsselt nach Fehlertyp und betroffener Komponente. Latenz-Perzentile sind aussagekräftiger als Durchschnittswerte, da sie zeigen, wie die langsamsten Anfragen abschneiden – das 95. oder 99. Perzentil offenbart Probleme, die im Durchschnitt unsichtbar bleiben. Abhängigkeitsketten visualisieren, welche externen Dienste aufgerufen werden und wo Engpässe entstehen. RAG-Toolpfade zeigen, welche Retrieval-Strategien wie häufig zum Einsatz kommen. Nutzungsprofile identifizieren Muster in der Systemnutzung, etwa Spitzenzeiten oder ungewöhnliches Verhalten.

Smart Detection für automatisierte Anomalieerkennung

Eine besonders wertvolle Funktion ist Smart Detection, die auf maschinellem Lernen basiert und automatisiert Ausreißer im Systemverhalten erkennt. „Failure Anomalies" identifizieren plötzliche Anstiege in Fehlerraten, die auf neu eingeführte Bugs oder Infrastrukturprobleme hindeuten könnten. Performance Degradation erkennt schleichende Verschlechterungen der Antwortzeiten, die möglicherweise auf Ressourcenengpässe oder ineffiziente Abfragen zurückzuführen sind. Diese automatisierten Erkennungen generieren Alerts, die Teams proaktiv auf Probleme aufmerksam machen, bevor diese von Endnutzern bemerkt werden oder eskalieren.

Vendor-neutrale Nutzung durch offene Standards

Die Verwendung von OpenTelemetry als Basis der Instrumentierung bietet einen strategischen Vorteil über die Azure-Integration hinaus. Organisationen können die gleichen Traces auch in Drittplattformen auswerten, etwa Elastic, Grafana oder Datadog. Für viele dieser Plattformen existieren fertige Dashboards und Integrationen, die speziell für OpenTelemetry-Daten optimiert sind. Dies gibt Teams die Flexibilität, ihre bevorzugten Monitoring-Werkzeuge zu nutzen oder eine Multi-Vendor-Strategie zu verfolgen, ohne die Anwendungen neu instrumentieren zu müssen.

Einheitliches Observability-Dashboard in Foundry

Ergänzend zu den umfassenden Möglichkeiten von Azure Monitor bietet Foundry ein spezialisiertes Dashboard, das zentral Kennzahlen zu Gesundheit, Leistung, Nutzung, Qualität und Safety für Agenten und Anwendungen präsentiert. Dieses Dashboard ist spezifisch auf agentenbasierte Workloads zugeschnitten und hebt die Aspekte hervor, die für KI-Systeme besonders relevant sind.

Im Bereich der Nutzung zeigt das Dashboard die Gesamtzahl der Anfragen, aufgeschlüsselt nach Endpunkt oder Modell. Diese Aufschlüsselung hilft zu verstehen, welche Teile des Systems am stärksten belastet sind. Token-Ein- und Ausgabe werden separat dargestellt, was sowohl für Kostenprognosen als auch für die Optimierung von Prompt-Designs wertvoll ist. Lange Input-Prompts verursachen höhere Kosten und potenziell längere Latenzen, während übermäßig lange Ausgaben auf ineffiziente Antwortgenerierung hindeuten können.

Die Leistungs- und Zuverlässigkeits-Metriken umfassen Latenzverteilungen, die zeigen, wie schnell das System typischerweise antwortet und wo Ausreißer auftreten. Durchsatzmessungen quantifizieren, wie viele Anfragen pro Zeiteinheit verarbeitet werden. Fehlerarten werden kategorisiert, um zu verstehen, ob Probleme von Timeouts, Kapazitätsgrenzen, ungültigen Eingaben oder anderen Ursachen herrühren. Abhängigkeitsketten visualisieren die Komplexität des Systems und identifizieren kritische Pfade, deren Ausfall oder Verlangsamung das Gesamtsystem beeinträchtigen würde.

Besonders wertvoll für KI-spezifische Observability sind die kontinuierlichen Qualitäts- und Safety-Signale. Diese stammen aus den Foundry-Evaluierungen, die wir im vorherigen Kapitel diskutiert haben, sowie aus Content-Safety-Checks. Das Dashboard macht diese Signale im Zeitverlauf sichtbar, sodass Teams erkennen können, ob die Qualität stabil bleibt, sich verbessert oder verschlechtert. Ein plötzlicher Rückgang in der Groundedness könnte beispielsweise auf Probleme mit den Datenquellen hinweisen, während eine Zunahme von Safety-Verletzungen auf Änderungen im Nutzerverhalten oder Angriffsmuster hindeuten könnte.

Kosten- und Nutzungsanalyse

Die Transparenz über Kosten ist entscheidend für den wirtschaftlich nachhaltigen Betrieb von KI-Systemen. Für Azure-OpenAI-Ressourcen stehen token-basierte Nutzungs-Metriken bereit, die präzise quantifizieren, wie viele Token in einem bestimmten Zeitraum verarbeitet wurden. Diese Metriken können nach verschiedenen Dimensionen aufgeschlüsselt werden, um zu verstehen, welche Modelle, Bereitstellungen oder Anwendungsteile die höchsten Kosten verursachen.

Die eigentliche Kostenanalyse erfolgt in Cost Analysis, dem Azure-weiten Werkzeug für Kostentransparenz. Dort können Kosten auf Ressourcen- oder Ressourcengruppen-Ebene verfolgt werden. Modelle aus der Kategorie §Models sold directly by Azure" erscheinen als Metriken, die der

jeweiligen Foundry-Ressource zugeordnet sind, was eine direkte Zuordnung ermöglicht. Partner- und Marketplace-Modelle werden unter dem Ressourcengruppen-Scope erfasst, was eine breitere Zuordnung darstellt, aber dennoch Transparenz über die Gesamtkosten dieser Modellkategorie bietet.

Die Kombination von Token-Metriken und Kostenanalyse erlaubt es, Kosteneffizienz zu quantifizieren. Teams können beispielsweise die Kosten pro tausend Token für verschiedene Modelle vergleichen oder analysieren, wie Änderungen in der Prompt-Struktur die Token-Nutzung und damit die Kosten beeinflussen. Diese Einblicke sind fundamental für Optimierungsentscheidungen, etwa ob ein größeres, genaueres Modell trotz höherer Kosten pro Token insgesamt wirtschaftlicher ist, weil es häufigere Nachfragen vermeidet.

Datenschutz, Maskierung und Aufbewahrung

Die Protokollierung von Telemetriedaten muss stets datenschutzkonform erfolgen, was in Zeiten strenger Datenschutzgesetze wie der DSGVO nicht optional ist. Eine Herausforderung bei KI-Systemen besteht darin, dass Prompts und Completions häufig personenbezogene oder vertrauliche Informationen enthalten. Die unreflektierte Protokollierung dieser Daten könnte Datenschutzverletzungen darstellen.

Als Grundprinzip sollten Prompts und Completions nur in dem Umfang protokolliert werden, der für Debugging und Qualitätssicherung tatsächlich notwendig ist. Häufig genügt es, Metadaten wie Token-Zähler, Latenzen und Fehlertypen zu erfassen, ohne den vollständigen Prompt- oder Antworttext zu speichern. Wenn vollständige Inhalte protokolliert werden müssen, sollten sensible Informationen vor dem Logging systematisch redigiert werden. Dies kann personenbezogene Daten wie Namen, Adressen oder Identifikationsnummern umfassen, aber auch geschäftskritische Informationen, deren Protokollierung Sicherheitsrisiken birgt.

Die Zugriffskontrolle auf Protokolldaten muss streng geregelt sein. Nicht jedes Teammitglied benötigt Zugang zu vollständigen Trace-Daten. Rollenbasierte Zugriffskontrollen sollten sicherstellen, dass nur autorisierte Personen, etwa Sicherheitsteams oder Senior-Entwickler, auf potenziell sensible Logs zugreifen können.

Log Analytics erlaubt die Konfiguration von Aufbewahrungsfristen, die definieren, wie lange Daten gespeichert bleiben. Kürzere Aufbewahrungsfristen reduzieren sowohl Kosten als auch Datenschutzrisiken. Purge- und Delete-Operationen ermöglichen es, spezifische Datensätze vorzeitig zu entfernen, etwa wenn versehentlich sensible Daten protokolliert wurden oder wenn ein Betroffenenantrag auf Löschung vorliegt.

Für die automatisierte Erkennung und Redaktion von personenbezogenen Informationen kann Azure AI Language PII-Detection eingesetzt werden. Dieser Dienst identifiziert verschiedene Kategorien von PII

in Text und kann vor der Protokollierung eingebettet werden, um automatisch sensible Informationen zu maskieren oder zu entfernen.

Ganzheitliche Monitoring-Strategie

Eine robuste Monitoring-Strategie für Azure AI Foundry beginnt mit der Aktivierung von OpenTelemetry und der korrekten Bindung des Geltungsbereichs. Das Foundry-Projekt sollte fest mit einer Application-Insights-Ressource gekoppelt werden, und Tracing muss im SDK explizit eingeschaltet werden. Diese initiale Konfiguration ist das Fundament, ohne das keine der fortgeschrittenen Funktionen verfügbar ist.

Die Sampling-Konfiguration erfordert sorgfältige Überlegung. Head-based Sampling im OpenTelemetry-SDK bietet einen guten Ausgangspunkt für die meisten Szenarien und ist einfach zu konfigurieren. Für anspruchsvollere Anforderungen, etwa die bevorzugte Erfassung von Fehler-Traces, sollte Tail-based Sampling über einen Collector in Betracht gezogen werden. Das Tageslimit fungiert als Notbremse und sollte so konfiguriert werden, dass es normale Schwankungen toleriert, aber vor extremen Kostenexplosionen schützt.

Dashboards und Workbooks sollten aktiv genutzt werden, nicht nur als passive Anzeigen. KQL-Berichte für Latenz sollten Perzentile statt Durchschnittswerte verwenden. Fehleranalysen sollten nach Fehlertyp, betroffenem Modell und zeitlichem Auftreten gruppieren. Tool-Pfad-Anteile zeigen, welche RAG-Strategien dominant sind. Token-Verbrauch sollte nach Modell und Zeitperiode aufgeschlüsselt werden. Alerts und Smart Detection sollten nicht nur konfiguriert, sondern auch regelmäßig auf ihre Relevanz überprüft und angepasst werden.

Die Kopplung von Token-Metriken und Kostenanalyse liefert die wirtschaftliche Perspektive. Regelmäßige Reviews sollten analysieren, wie sich Kosten nach Ressource und Meter-Typ verteilen, welche Optimierungspotenziale existieren und ob die tatsächlichen Kosten den Erwartungen entsprechen. Diese finanzielle Observability ist genauso wichtig wie die technische, denn selbst das beste System ist nicht nachhaltig, wenn es die verfügbaren Budgets überschreitet.

Die Minimierung personenbezogener Informationen in Logs sollte von Anfang an in die Architektur eingebaut werden. Redaktion sollte vor der Protokollierung erfolgen, nicht nachträglich. Aufbewahrungsfristen und Zugriffspolitiken sollten klar definiert und per Azure Policy durchgesetzt werden. Diese proaktive Herangehensweise an Datenschutz verhindert Compliance-Probleme, bevor sie entstehen.

Klarstellungen technischer Missverständnisse

Einige Aspekte der Observability in Azure AI Foundry werden häufig missverstanden und verdienen explizite Klarstellung. Die Frage, warum ein Agent ein bestimmtes Werkzeug gewählt hat, wird nicht automatisch protokolliert. Die Traces erfassen objektive Fakten – welche Werkzeuge aufgerufen wurden, mit welchen Parametern, wie lange die Ausführung dauerte und welcher Status zurückgegeben wurde. Die Begründung für diese Entscheidung ist eine Funktion der Modelllogik und wird nur dann Teil der Telemetrie, wenn die Anwendung sie explizit als Span-Attribut mitschreibt. Dies erfordert typischerweise, dass das Modell aufgefordert wird, seine Reasoning-Schritte offenzulegen, und dass diese dann strukturiert erfasst werden.

Tail-based Sampling ist kein einfacher Schalter in Foundry-Einstellungen. Es erfordert Collector-basierte Verarbeitung, was bedeutet, dass ein OpenTelemetry Collector eingerichtet und konfiguriert werden muss, der Traces zunächst puffert, bevor er Sampling-Entscheidungen trifft. Standardmäßig arbeitet Application Insights mit SDK-basiertem Head-Sampling, was für viele Szenarien ausreichend ist, aber die genannten Limitierungen hat.

Das Monitoring von Prompt Flow setzt Hub-basierte Projekte voraus und erfordert, dass die entsprechenden Diagnoseoptionen explizit aktiviert werden. In modernen Foundry-Projekten ohne Hub-Struktur ist Prompt Flow nicht verfügbar, und Teams sollten stattdessen direkt mit dem Agent Service und dessen integrierter Observability arbeiten.

Zusammenfassung zur umfassenden Observability

Die verschiedenen Komponenten – OpenTelemetry-basierte Traces, Azure-Monitor-Integration, das Foundry-Dashboard, Kostenanalyse und Datenschutzmechanismen – bilden zusammen ein umfassendes Observability-System. Traces liefern die detaillierte, zeitlich geordnete Perspektive auf einzelne Anfragen. Azure Monitor aggregiert und analysiert diese Daten über viele Anfragen hinweg, um Muster und Trends zu identifizieren. Das Foundry-Dashboard präsentiert die für KI-Workloads relevantesten Metriken in einer zugänglichen Form. Die Kostenanalyse verbindet technische Metriken mit wirtschaftlichen Realitäten. Datenschutzmechanismen stellen sicher, dass Observability nicht auf Kosten der Compliance geht.

Diese mehrschichtige Observability ist entscheidend, weil KI-Systeme auf Weisen versagen können, die sich von traditioneller Software unterscheiden. Ein Agent kann technisch funktionieren, aber qualitativ schlechte Antworten liefern. Ein Modell kann korrekte Resultate produzieren, aber wirtschaftlich untragbare Kosten verursachen. Eine Konfigurationsänderung kann die durchschnittliche Latenz verbessern, aber die Latenz des 99. Perzentils verschlechtern. Nur durch umfassende Observability

über alle diese Dimensionen hinweg können Teams die Komplexität produktiver KI-Systeme beherrschen und kontinuierlich verbessern.

9. CI/CD und GenAIOps

Mit wachsender Reife von KI-Anwendungen müssen diese in belastbare Software-Lebenszyklen eingebettet werden. Die Herausforderung liegt darin, dass generative KI-Systeme grundlegend anders sind als traditionelle Software. Während ein klassisches Programm deterministisch arbeitet und durch Unit-Tests validiert werden kann, erzeugen KI-Modelle probabilistische Ausgaben, deren Qualität sich nur statistisch bewerten lässt. Klassische CI/CD-Praktiken bleiben zwar gültig und wertvoll, brauchen aber KI-spezifische Ergänzungen: reproduzierbare Artefakte trotz nicht-deterministischer Komponenten, automatisierte Evaluierungen als Freigabekriterien statt binärer Tests und kontrollierte Rollouts mit kontinuierlicher Qualitätsüberwachung. Azure AI Foundry unterstützt diese erweiterten Anforderungen über Evaluation-SDKs, fertige Pipeline-Integrationen und umfassende Unterstützung für Infrastructure as Code. Diese Kombination ermöglicht es, die Prinzipien moderner DevOps-Praktiken auf die spezifischen Anforderungen generativer KI zu übertragen.

Integration von Evaluierungen als Freigabekriterien

Azure AI Foundry stellt umfassende Evaluierungsfunktionen für Modelle, RAG-Abläufe und Agenten bereit, die nahtlos in CI/CD-Pipelines integriert werden können. Diese Evaluierungen können lokal während der Entwicklung über das Azure AI Evaluation SDK ausgeführt werden, was schnelles Feedback ermöglicht. Für automatisierte Pipelines existieren spezialisierte Integrationen in Form einer GitHub Action und einer Azure-DevOps-Erweiterung, die den Evaluierungsprozess vollständig in den Deployment-Workflow einbetten.

Die Evaluierungsergebnisse liefern detaillierte Metriken zu Qualität und Sicherheit, die wir in früheren Kapiteln ausführlich diskutiert haben. Dazu gehören Qualitätsmetriken wie Relevanz, Kohärenz und Groundedness sowie Sicherheits-Metriken für problematische Inhalte wie Hate, Violence und Self-harm. Besonders wertvoll für Pipeline-Entscheidungen ist, dass diese Metriken optional mit Konfidenzintervallen und Signifikanztests angereichert werden können. Dies ermöglicht es, nicht nur zu wissen, dass ein Modell einen bestimmten Score erreicht, sondern auch, wie zuverlässig diese Messung ist und ob Unterschiede zwischen Versionen statistisch signifikant sind.

Der typische Pipeline-Ablauf

Ein robuster CI/CD-Prozess für KI-Anwendungen folgt einem strukturierten Ablauf, der Qualitätssicherung in mehreren Stufen implementiert. Zunächst werden Änderungen – sei es am

Modell, am Prompt-Design, an der RAG-Logik oder an den Agenten-Definitionen – in eine produktionsnahe Staging-Umgebung bereitgestellt. Diese Staging-Umgebung sollte die Produktionsumgebung so genau wie möglich spiegeln, um aussagekräftige Testergebnisse zu gewährleisten. Nach der Bereitstellung startet automatisch ein Offline-Evaluation-Run gegen kuratierte Testsätze, die repräsentative und herausfordernde Szenarien abdecken.

Die Pipeline trifft dann eine kritische Entscheidung: Falls vordefinierte Schwellenwerte verfehlt werden, bricht der Deployment-Prozess vor der Produktion ab. Dies verhindert, dass Regressionen oder qualitativ minderwertige Versionen Endnutzer erreichen. Die GitHub Action und die Azure-DevOps-Erweiterung sind so gestaltet, dass sie die notwendigen Parameter entgegennehmen – den Projekt-Endpunkt oder die Verbindungszeichenfolge, die Modell-Bereitstellung, die zu testende Anwendung und die Referenz-Datensätze. Die Resultate werden sowohl als Workflow-Ausgabe für die unmittelbare Pipeline-Logik als auch als Projekt-Artefakte für die langfristige Nachverfolgbarkeit geschrieben.

Praktische Gestaltung von Qualitätsschwellen

Die Definition sinnvoller Schwellenwerte erfordert sorgfältige Überlegung und sollte sich an domänenspezifischen Anforderungen orientieren. Eine Beispielkonfiguration könnte verlangen, dass die Relevanz mindestens einen Score von 4 auf einer 5-Punkte-Skala erreicht, die Groundedness mindestens 0,85 beträgt und keine Safety-Verletzungen auftreten. Diese Schwellen sollten nicht willkürlich gewählt werden, sondern auf empirischen Daten basieren – etwa dem Durchschnitt der letzten stabilen Versionen plus einer Sicherheitsmarge oder auf Nutzerfeedback, das Qualitätserwartungen quantifiziert.

Entscheidend ist, dass Evaluierungen selbst versioniert werden. Die Testsätze, Evaluierungslogik und Schwellenwerte sollten gemeinsam mit dem Anwendungscode im Repository liegen. Dies ermöglicht es, nachzuvollziehen, warum eine bestimmte Version akzeptiert oder abgelehnt wurde, und stellt sicher, dass sich die Bewertungskriterien nicht unbemerkt ändern. Evaluierungen sollten für jeden Pull Request und für jeden Release-Kandidaten automatisiert ausgeführt werden, sodass Qualitätsprobleme so früh wie möglich erkannt werden.

Rollout-Strategien: Gestaffelte Einführung statt Vollausrollung

Eine wichtige Erkenntnis für den produktiven Betrieb von KI-Systemen ist, dass selbst nach erfolgreichen Tests in Staging-Umgebungen unvorhergesehene Probleme in der Produktion auftreten können. Die Interaktion mit echten Nutzern bringt oft Randfälle und Nutzungsmuster zutage, die in Tests nicht antizipiert wurden. Aus diesem Grund sind gestaffelte Rollouts, auch als progressive Rollouts bezeichnet, eine kritische Risikominderungsstrategie.

Azure AI Foundry selbst implementiert keine prozentuale Verkehrsumlenkung auf Plattformebene. Stattdessen werden progressive Rollouts in der Zugriffs-Schicht realisiert, was mehrere Vorteile bietet: Es ermöglicht die Verwendung etablierter, erprobter Werkzeuge und hält die Foundry-Architektur fokussiert auf ihre Kernfunktionalität. Verschiedene Azure-Dienste bieten spezialisierte Funktionen für Traffic-Management.

Azure Front Door kann gewichtete Verteilung zwischen verschiedenen Backend-Pools implementieren, sodass beispielsweise 95 Prozent des Traffics zur stabilen Version und 5 Prozent zur neuen Version geleitet werden. App Service Deployment Slots ermöglichen das parallele Betreiben mehrerer Versionen mit konfigurierbaren Traffic-Prozentsätzen, die über das Portal oder API gesteuert werden können. API-Management bietet Policy-basiertes Routing, das auf Request-Eigenschaften reagieren kann, um sophisticated Rollout-Strategien zu implementieren. Container Apps unterstützen Revisions-basiertes Routing mit feingranularer Traffic-Kontrolle.

Der gestaffelte Rollout-Prozess

Ein typischer gestaffelter Rollout beginnt konservativ mit 5 Prozent des Produktions-Traffics, der zur neuen Version geleitet wird. In dieser Phase wird intensiv überwacht, ob die neue Version sich wie erwartet verhält. Wenn keine Anomalien auftreten, wird der Anteil schrittweise erhöht – etwa auf 25 Prozent, dann 50 Prozent und schließlich 100 Prozent. Diese Progression kann über Stunden oder Tage erfolgen, abhängig vom Risikoappetit und der Kritikalität der Anwendung.

Entscheidend ist die Automatisierung von Rollback-Entscheidungen. Die Metriken aus Foundry und Application Insights sollten kontinuierlich analysiert werden, und das Rollout sollte automatisch gestoppt oder zurückgesetzt werden, wenn kritische Schwellenwerte überschritten werden. Dies könnte eine plötzliche Zunahme von Fehlerraten sein, eine Verschlechterung der Latenz-Perzentile – besonders des 95. oder 99. Perzentils – oder eine Zunahme von Safety-Signalen, die auf problematische Ausgaben hindeuten. Die Automatisierung dieses Prozesses ist entscheidend, denn menschliche Reaktionszeiten könnten zu langsam sein, um Schaden zu begrenzen, insbesondere bei hochfrequentierten Anwendungen.

Infrastructure as Code für konsistente Umgebungen

Die deklarative Bereitstellung von Foundry-Ressourcen, Projekten und abhängigen Diensten über Infrastructure as Code ist fundamental für Reproduzierbarkeit und Konsistenz. Bicep und ARM-Templates sind die primären Pfade für Azure-native Bereitstellung und werden von Microsoft direkt unterstützt. Sie bieten typsichere Ressourcendefinitionen und native Integration mit Azure-Diensten. Terraform hat sich als weitverbreitete Alternative etabliert, besonders in Multi-Cloud-Umgebungen oder wenn bestehende Terraform-Expertise vorhanden ist.

Microsoft stellt Referenz-Module und Beispiele bereit, die „secure by default"-Konfigurationen demonstrieren. Diese Templates implementieren Best Practices wie die Verwendung von Private Endpoints statt öffentlicher Zugriffe, die Aktivierung von Verschlüsselung für Daten im Ruhezustand und in Übertragung sowie die korrekte Konfiguration von Identitäts- und Zugriffsmechanismen. Teams sollten diese Referenzen als Ausgangspunkt nutzen und an ihre spezifischen Anforderungen anpassen, statt von Grund auf neu zu beginnen.

Kritische Ressourcentypen

Die Azure AI Foundry-Ressource wird technisch als „`Microsoft.CognitiveServices accounts`" mit der Art-Kennzeichnung „`AIServices`" bereitgestellt. Diese Ressource fungiert als Anker für Foundry-Projekte und definiert den Geltungsbereich für Identität, Netzwerk und Abrechnung. Die Konfiguration dieser Ressource ist entscheidend, da sie die Sicherheits- und Compliance-Eigenschaften aller zugehörigen Projekte bestimmt.

Für ausgewählte Anwendungsfälle, insbesondere wenn erweiterte Machine-Learning-Funktionen oder spezifische Hub-Features benötigt werden, können Azure AI Hubs und Hub-Projekte auf Basis von „`Microsoft.MachineLearningServices workspaces`" bereitgestellt werden. Diese Ressourcentypen bringen zusätzliche Abhängigkeiten mit sich, bieten aber auch erweiterte Funktionen für bestimmte Szenarien. Die Wahl zwischen Foundry-Ressourcen und Hub-basierten Strukturen sollte basierend auf den spezifischen Anforderungen getroffen werden, wobei die modernen Foundry-Ressourcen für neue Implementierungen generell empfohlen werden.

Governance als Code

Ein besonders mächtiger Aspekt von Infrastructure as Code ist die Möglichkeit, Governance-Anforderungen direkt in Pipelines zu implementieren. Azure Policy ermöglicht es, Richtlinien als Code zu definieren und automatisch durchzusetzen. Deny-Policies verhindern die Erstellung von Ressourcen, die nicht den Vorgaben entsprechen, etwa Foundry-Ressourcen ohne Private Endpoints. `DeployIfNotExists`-Policies korrigieren automatisch fehlende Konfigurationen, etwa durch das Hinzufügen von Diagnoseeinstellungen oder Tags. Initiativen bündeln mehrere verwandte Policies zu kohärenten Regelwerken. Remediation-Tasks können bestehende Ressourcen nachträglich in Compliance bringen.

Praktische Anwendungsfälle umfassen die Durchsetzung einer Private-Link-Pflicht für alle Foundry-Ressourcen, Regionsvorgaben, die sicherstellen, dass Ressourcen nur in erlaubten Azure-Regionen erstellt werden können, Tagging-Standards für Kostenzuordnung und Lifecycle-Management sowie Modell-Nutzungsvorgaben, die etwa nur bestimmte Modellkategorien oder -versionen erlauben. Diese Governance-as-Code-Ansätze stellen sicher, dass Compliance-Anforderungen nicht durch manuelle Prozesse durchgesetzt werden müssen, die fehleranfällig und inkonsistent sind, sondern technisch garantiert werden.

Versionierung und Reproduzierbarkeit

Die Reproduzierbarkeit von KI-Systemen ist eine besondere Herausforderung, da viele Komponenten inhärent nicht-deterministisch sind. Dennoch können und müssen die kontrollierbaren Aspekte rigoros versioniert werden, um Nachvollziehbarkeit und die Möglichkeit zu Rollbacks zu gewährleisten.

Modellversionen und Update-Strategien

Azure AI Foundry unterscheidet zwischen Modellversionen, die das eigentliche Modellgewicht repräsentieren, und API-Versionen, die die Schnittstelle definieren. Bereitstellungen unterstützen verschiedene Update-Richtlinien, die unterschiedliche Szenarien adressieren. Die automatische Aktualisierung auf die Standard-Version bedeutet, dass das Deployment automatisch auf neue vom Anbieter empfohlene Versionen umgestellt wird. Dies hält das System aktuell, birgt aber das Risiko unerwarteter Verhaltensänderungen. Die Aktualisierung bei Ablauf bedeutet, dass ein Upgrade nur erfolgt, wenn die aktuell verwendete Version veraltet und nicht mehr verfügbar ist. Dies gibt mehr Kontrolle, erfordert aber proaktives Management. Die Option ohne automatische Aktualisierung gibt vollständige Kontrolle, verlangt aber manuelle Upgrades.

Für Produktionsumgebungen empfiehlt sich das Pinning auf datierte Modell-Snapshots mit einem kontrollierten Upgradepfad. Dies bedeutet, dass Deployments explizit auf eine spezifische Modellversion verweisen, etwa „gpt-5-2025-11-03" statt nur „gpt-5". Upgrades werden dann bewusst initiiert, durchlaufen den vollständigen Test- und Evaluierungsprozess und werden gestaffelt ausgerollt. Dieser Ansatz maximiert Stabilität und Vorhersagbarkeit auf Kosten von etwas mehr manuellem Aufwand.

Evaluierungen als versionierte Artefakte

Evaluierungsläufe werden projektseitig gespeichert und bleiben dauerhaft nachvollziehbar. Die Foundry-Oberfläche ermöglicht Vergleiche auf mehreren Ebenen. Auf Aggregat-Ebene können durchschnittliche Scores verschiedener Versionen gegenübergestellt werden. Auf Fall-Ebene können individuelle Testfälle inspiziert werden, um zu verstehen, wo sich Versionen unterscheiden. Der Run-Vergleich zeigt systematisch, welche Metriken sich verbessert oder verschlechtert haben.

Diese Vergleichsfunktionen machen Regression sichtbar und dokumentierbar. Wenn eine neue Version in einer wichtigen Metrik schlechter abschneidet als die vorherige, ist dies sofort ersichtlich und kann untersucht werden, bevor die Version produktiv geht. Die dauerhafte Speicherung ermöglicht auch retrospektive Analysen, etwa wenn Monate später ein Problem auftritt und verstanden werden muss, wann und warum eine bestimmte Verhaltensänderung eingeführt wurde.

Agenten und deren Definitionen

Der Entwicklungsfluss über die Visual-Studio-Code-Erweiterung für Azure AI Foundry erlaubt das Bearbeiten von Agenten in YAML-Format. Diese YAML-Definitionen können gemeinsam mit

begleitenden Dateien wie Prompt-Templates oder Konfigurationsdateien im Code-Repository versioniert werden. Dies bedeutet, dass jede Änderung an einem Agenten durch den gleichen Review-Prozess geht wie Code-Änderungen, mit Pull Requests, Code Reviews und Approval-Gates.

Die Projects-Clients in den Foundry-SDKs für verschiedene Programmiersprachen binden Projekt-Endpunkte konsistent an, sodass Agenten programmatisch erstellt, aktualisiert und bereitgestellt werden können. Dies ermöglicht vollständig automatisierte Deployment-Pipelines, in denen Agenten-Definitionen aus dem Repository gelesen, validiert, getestet und dann in verschiedenen Umgebungen bereitgestellt werden, ohne manuelle Schritte über das Portal.

RAG-Daten und Index-Versionen

Eine oft übersehene Quelle von Nicht-Reproduzierbarkeit sind sich ändernde Datenbestände. Wenn ein RAG-System auf einen Index zugreift, der kontinuierlich aktualisiert wird, können Evaluierungsergebnisse zwischen Läufen variieren, selbst wenn Code und Modell identisch bleiben. Für wahre Reproduzierbarkeit sollten Datenbestände und Index-Versionen explizit getaggt oder als Data Assets geführt werden.

Dies kann technisch über verschiedene Mechanismen realisiert werden. Storage-Versionierung in Azure Blob Storage ermöglicht es, auf spezifische Versionen von Dateien zu verweisen. Index-Snapshots in Azure AI Search können gesichert und bei Bedarf wiederhergestellt werden. Data Assets in Azure ML können Datensätze als versionierte, unveränderliche Artefakte behandeln. Die konkrete Wahl hängt von der Architektur ab, aber das Prinzip bleibt gleich: Für jeden Release sollte nachvollziehbar sein, nicht nur welcher Code und welches Modell verwendet wurden, sondern auch welche Daten zur Verfügung standen.

Automatisierte Aktualisierung und Wartungsroutinen

Viele KI-Systeme benötigen regelmäßige Aktualisierungen, etwa wenn neue Daten hinzukommen oder wenn Indizes neu aufgebaut werden müssen, um Optimierungen zu nutzen. Diese Wartungsroutinen lassen sich elegant als zeitgesteuerte Pipelines modellieren, die den gleichen Qualitätssicherungsprozess durchlaufen wie manuelle Releases.

Ein typischer Refresh-Prozess beginnt mit der Datenaufnahme aus Quellsystemen. Dies könnten neue Dokumente in SharePoint, aktualisierte Produktkataloge oder neue FAQ-Einträge sein. Im nächsten Schritt werden Indizes neu aufgebaut, um diese Daten durchsuchbar zu machen. Anschließend werden automatisierte Evaluierungen gegen den aktualisierten Index ausgeführt, um sicherzustellen, dass die Retrieval-Qualität nicht gelitten hat und dass neue Inhalte korrekt indexiert wurden.

Nur wenn diese Evaluierungen die definierten Freigabekriterien erfüllen, wird die Änderung in Staging und dann in Produktion befördert. Dies verhindert, dass fehlerhafte Datenaktualisierungen oder Index-Probleme die Produktionsqualität beeinträchtigen. Cloud-gestützte Evaluierungen unterstützen große Testmengen, die lokal nicht praktikabel wären, etwa Tausende von Testfällen gegen verschiedene Datenquellen.

Umgebungen und kontrollierte Beförderung

Die saubere Trennung von Entwicklung, Staging und Produktion ist ein fundamentales Prinzip für belastbare Systeme und gilt für KI-Anwendungen gleichermaßen. Jede Umgebung sollte isoliert sein, um Seiteneffekte zu vermeiden. Eine Fehlkonfiguration in der Entwicklungsumgebung sollte nicht die Produktion beeinflussen können. Ein fehlgeschlagenes Experiment in Staging sollte keine Auswirkungen auf Endnutzer haben.

Foundry-Projekte oder Hub-Strukturen sollten für jede Umgebung separat und konsistent über Infrastructure as Code erstellt werden. Dies stellt sicher, dass die Umgebungen strukturell identisch sind, abgesehen von umgebungsspezifischen Werten wie Skalierungsparametern oder Kostenlimits. Die Verwendung von IaC-Parameterdateien oder Template-Variablen ermöglicht es, die gleichen Definitionen für alle Umgebungen zu nutzen, während umgebungsspezifische Unterschiede explizit und nachvollziehbar bleiben.

Promotion-Workflows übernehmen die kontrollierte Beförderung geprüfter Artefakte von einer Umgebung zur nächsten. Ein typischer Workflow könnte verlangen, dass Änderungen zunächst in Entwicklung getestet werden, dann automatisierte Tests in Staging bestehen müssen, bevor sie für die Produktion freigegeben werden. Manuelle Approval-Gates können für kritische Übergänge eingefügt werden, etwa vor der Produktion, um eine finale menschliche Überprüfung zu ermöglichen.

Besonders wichtig ist die konsistente Planung von Observability- und kostenrelevanten Ressourcen. Application Insights, Log Analytics und andere Monitoring-Ressourcen sollten pro Umgebung identisch konfiguriert werden, sodass das Verhalten in verschiedenen Umgebungen vergleichbar überwacht werden kann. Dies erleichtert die Diagnose von Problemen erheblich, da Teams die gleichen Dashboards und Abfragen in allen Umgebungen nutzen können.

Zusammenführung zu robustem GenAIOps

Die verschiedenen Komponenten – Evaluation-Integrationen, Infrastructure as Code, rigoroses Versionieren und kontrollierte Rollouts – fügen sich zu einem ganzheitlichen GenAIOps-Ansatz

zusammen, der die spezifischen Herausforderungen generativer KI adressiert, während er auf bewährten DevOps-Prinzipien aufbaut.

Evaluation-Integrationen machen Freigabequalität objektiv messbar, statt auf subjektive Einschätzungen angewiesen zu sein. Teams können mit Zuversicht behaupten, dass eine neue Version bestimmte Qualitätsstandards erfüllt, weil sie quantitative Beweise haben. Infrastructure as Code macht Umgebungen reproduzierbar und konsistent, eliminiert Konfigurationsdrift und ermöglicht es, Infrastruktur mit denselben Review- und Approval-Prozessen zu behandeln wie Anwendungscode.

Strenges Versionieren macht Rollbacks sicher und günstig. Wenn eine neue Version Probleme verursacht, kann schnell zur vorherigen stabilen Version zurückgekehrt werden, weil alle Komponenten – Code, Modellversionen, Agenten-Definitionen und Datenversionen – eindeutig identifiziert sind. Kontrollierte Rollouts minimieren die Auswirkungen unvorhergesehener Probleme, indem sie die Exposition schrittweise erhöhen und automatisierte Rollbacks bei Anomalien ermöglichen.

Compliance-Vorgaben werden als Code durchsetzbar, statt auf manuelle Prozesse und Checklisten angewiesen zu sein. Azure Policies können technisch garantieren, dass Sicherheits- und Governance-Anforderungen erfüllt sind, unabhängig davon, wer eine Ressource erstellt oder ändert. Dies reduziert nicht nur Compliance-Risiken, sondern auch den operativen Aufwand für kontinuierliche Überprüfungen.

Zusammengenommen ermöglichen diese Praktiken es, generative KI-Systeme mit derselben Zuverlässigkeit, Vorhersagbarkeit und Geschwindigkeit zu betreiben, die moderne DevOps-Organisationen für traditionelle Software erreicht haben. Die KI-spezifischen Ergänzungen – statistische Evaluierungen statt binärer Tests, Versionierung nicht-deterministischer Komponenten und kontinuierliche Qualitätsüberwachung im Betrieb – adressieren die einzigartigen Herausforderungen dieser Technologie, ohne die fundamentalen Prinzipien guter Software-Engineering-Praxis aufzugeben.

10. Kosten- und Kapazitätsmanagement

Das wirtschaftliche Management von KI-Systemen ist eine fundamentale Herausforderung, die über traditionelle IT-Kostenmodelle hinausgeht. Während klassische Anwendungen oft vorhersagbare Ressourcenverbräuche haben, können KI-Workloads erheblich variieren – abhängig von der Prompt-Länge, der gewünschten Antwortqualität und der Komplexität der Aufgaben. Azure AI Foundry begegnet dieser Komplexität mit einem transparenten, nutzungsbasierten Preismodell ohne Plattform-Grundgebühr. Die Abrechnung erfolgt ausschließlich für die tatsächlich verwendeten Dienste und Modelle, was sowohl Kosteneffizienz als auch Flexibilität ermöglicht.

Preisstruktur und Abrechnungsmodell

Die Preisbildung in Azure AI Foundry erfolgt überwiegend tokenbasiert, wobei zwischen Eingabe-Tokens und Ausgabe-Tokens unterschieden wird. Diese Unterscheidung ist nicht willkürlich, sondern reflektiert unterschiedliche Verarbeitungskosten. Eingabe-Tokens müssen vom Modell verarbeitet und verstanden werden, während Ausgabe-Tokens generiert werden müssen, ein Prozess, der typischerweise rechenintensiver ist. Daher sind die Sätze für Ausgabe-Tokens in der Regel höher als für Eingabe-Tokens. Die konkreten Preise variieren erheblich nach Modell, Anbieter und Azure-Region, sodass eine Bezugnahme auf die offiziellen Azure-Preislisten unerlässlich ist für präzise Budgetierungen.

Die Kostenübersicht und Abrechnung erfolgen über die etablierten Azure-Mechanismen. Modelle aus der Kategorie „sold directly by Azure" erscheinen als Zähler direkt unter der zugehörigen Foundry-Ressource in der Kostenanalyse. Diese direkte Zuordnung vereinfacht das Kostentracking erheblich und ermöglicht es, die Ausgaben spezifischer Projekte oder Teams präzise zu erfassen. Partner- und Marketplace-Modelle dagegen erscheinen unter dem Ressourcengruppen-Scope, was eine etwas breitere, aber dennoch nachvollziehbare Kostenerfassung darstellt.

Zusatzkosten durch angebundene Dienste

Neben den direkten Modellkosten entstehen typischerweise Zusatzkosten durch die notwendigen unterstützenden Dienste, die ein vollständiges KI-System ausmachen. Azure AI Search verursacht Kosten basierend auf der gewählten Tarifstufe, wobei fortgeschrittene Features wie der Semantic Ranker zusätzliche Gebühren nach sich ziehen können. Diese Features verbessern die Retrieval-Qualität erheblich, müssen aber in die Gesamtkostenkalkulation einbezogen werden.

Storage-Dienste für Dokumente, Indizes und Protokolle verursachen sowohl Speicher- als auch Transaktionskosten. Application Insights und Log Analytics, die für umfassende Observability unerlässlich sind, rechnen nach Datenvolumen ab. Besonders bei hochfrequentierten Systemen mit detailliertem Tracing kann das Ingestion-Volumen erheblich werden. Netzwerk-Egress, also ausgehender Datenverkehr der Azure-Regionen verlässt oder ins Internet übertragen wird, verursacht zusätzliche Gebühren. Diese können besonders bei regions- oder internetübergreifenden Datenpfaden ins Gewicht fallen und sollten durch intelligente Architekturentscheidungen minimiert werden.

Die Einrichtung von Budgets und Kostenwarnungen über Cost Management + Billing ist eine kritische Schutzmaßnahme. Budgets definieren Ausgabengrenzen für definierte Zeiträume, während Warnungen proaktiv benachrichtigen, wenn Schwellenwerte erreicht werden. Dies ermöglicht rechtzeitiges Eingreifen, bevor Kostenexplosionen auftreten.

Quoten und Rate-Limitierung

Rate-Limits sind Schutzmechanismen, die Kapazitäten vor Überlastung bewahren und eine faire Ressourcenverteilung zwischen verschiedenen Nutzern sicherstellen. Diese Limits werden pro Bereitstellung und Modell durchgesetzt und operieren auf mehreren Dimensionen. Tokens per Minute quantifiziert die maximale Token-Verarbeitungsrate, Requests per Minute begrenzt die Anzahl der Anfragen unabhängig von ihrer Größe, und die Zahl paralleler Anfragen kontrolliert die Gleichzeitigkeit.

Wenn eine dieser Grenzen erreicht wird, antwortet der Dienst mit HTTP-Statuscode 429, der signalisiert, dass der Client zu viele Anfragen gestellt hat und es später erneut versuchen sollte. Diese Antwort ist keine Fehlfunktion, sondern ein normaler Teil des Rate-Limiting-Mechanismus. Anwendungen müssen darauf vorbereitet sein, 429-Antworten zu handhaben und ihre Anfragen entsprechend anzupassen.

Quotenerhöhungen sind über das offizielle Quota-Increase-Verfahren möglich, unterliegen aber der Verfügbarkeit von Kapazität und werden basierend auf der Nutzungshistorie und geplanten Verwendung bewertet. Microsoft prüft solche Anfragen sorgfältig, um sicherzustellen, dass erhöhte Quoten tatsächlich benötigt werden und sinnvoll genutzt werden. Es ist wichtig zu verstehen, dass feingranulare Limits pro Nutzer oder Team kein Plattform-Feature von Foundry sind. Wenn solche Kontrollen benötigt werden, etwa um verschiedene Abteilungen unterschiedliche Budgets zuzuordnen, müssen diese in der eigenen API-Schicht implementiert werden, die vor den Foundry-Aufrufen sitzt.

Empfohlene Praktiken für Rate-Limiting

Die Behandlung von 429-Antworten sollte Exponential Backoff mit Jitter implementieren. Exponential Backoff bedeutet, dass nach jeder fehlgeschlagenen Anfrage die Wartezeit exponentiell erhöht wird – etwa 1 Sekunde, dann 2 Sekunden, dann 4 Sekunden und so weiter. Jitter fügt eine zufällige Komponente hinzu, um zu verhindern, dass viele Clients, die gleichzeitig gedrosselt wurden, synchron wieder anfragen und dadurch eine neue Überlastungswelle auslösen. Diese Kombination hat sich als robust und effizient erwiesen.

Das Monitoring sollte systematisch auf gedrosselte Anfragen achten. Die Metrik „Throttled Requests" in Application Insights quantifiziert, wie häufig Rate-Limits erreicht werden. Frühwarnungen sollten bereits bei etwa 80 Prozent Quota-Auslastung ausgelöst werden, nicht erst wenn das Limit tatsächlich erreicht ist. Dies gibt Teams Zeit, entweder Optimierungen vorzunehmen oder Quotenerhöhungen zu beantragen, bevor die Limits die Produktion beeinträchtigen.

Bereitstellungsmodelle und ihre wirtschaftlichen Implikationen

Die Wahl des Bereitstellungsmodells hat fundamentale Auswirkungen sowohl auf die Kostenstruktur als auch auf die Leistungscharakteristik eines Systems. Jedes Modell adressiert unterschiedliche Anforderungsprofile und bietet spezifische Vor- und Nachteile, die sorgfältig gegen die Anwendungsanforderungen abgewogen werden müssen.

Serverlose Bereitstellung für Flexibilität

Das serverlose Modell, häufig als Pay-as-you-go bezeichnet, eliminiert die Notwendigkeit für Kapazitätsplanung vollständig. Teams können sofort mit der Nutzung beginnen, ohne Ressourcen vorab bereitstellen zu müssen. Die Abrechnung erfolgt strikt nach tatsächlich genutzten Tokens, was bedeutet, dass keine Kosten anfallen, wenn das System nicht genutzt wird. Dies macht serverlose Bereitstellungen ideal für Entwicklungs- und Testumgebungen, für Proof-of-Concepts, bei denen die zukünftige Nutzung noch unklar ist, und für Workloads mit stark variierender oder unvorhersehbarer Last.

Der wesentliche Trade-off liegt in der Latenz. Unter hoher Last kann die Antwortzeit variieren, da die Ressourcen zwischen verschiedenen Nutzern geteilt werden. Für viele Anwendungen ist dies akzeptabel, aber Szenarien mit strengen Latenzanforderungen könnten ein dediziertes Modell benötigen. Die Quoten in serverloser Bereitstellung sind moderat und reflektieren die geteilte Natur der Ressourcen. Sie können jedoch nach Bedarf erhöht werden, was Wachstum ermöglicht, ohne das Bereitstellungsmodell wechseln zu müssen.

Provisioned Throughput für Vorhersagbarkeit

Provisioned Throughput Units, kurz PTU, repräsentieren ein grundlegend anderes Modell. Statt nach Nutzung zu bezahlen, reservieren Teams dedizierte Durchsatzkapazität gegen einen stündlichen PTU-Preis. Diese Reservation garantiert planbare Latenz und verhindert Drosselung innerhalb der bereitgestellten Kapazität. Dies ist besonders wertvoll für produktive Workloads mit vorhersehbarer, konstanter Last, bei denen Latenz-Vorhersagbarkeit kritisch ist.

PTUs sind regionsbezogene Quoten, was bedeutet, dass die Kapazität in einer spezifischen Azure-Region reserviert wird und nicht einfach zwischen Regionen verschoben werden kann. Der effektive Durchsatz, den eine PTU liefert, wird in Tokens pro Minute gemessen und variiert je nach Modell. Größere Modelle verbrauchen mehr Ressourcen pro Token, sodass eine PTU bei einem großen Modell weniger Tokens pro Minute liefert als bei einem kleineren. Diese Modellabhängigkeit muss bei der Kapazitätsplanung berücksichtigt werden.

Die wirtschaftliche Entscheidung zwischen serverlos und PTU hängt vom Nutzungsprofil ab. Bei konstanter, vorhersagbarer Last kann PTU trotz höherer Grundkosten insgesamt günstiger sein, da die

Kosten pro Token bei hoher Auslastung niedriger sind als im Pay-as-you-go-Modell. Die Latenz-Vorhersagbarkeit ist ein zusätzlicher Wert, der schwer zu quantifizieren ist, aber für bestimmte Anwendungen wie Echtzeit-Kundeninteraktionen entscheidend sein kann.

Verwaltete Rechenressourcen für maximale Kontrolle

Das Managed-Compute-Modell ermöglicht es, offene oder individuell entwickelte Modelle auf eigener Azure-Infrastruktur zu betreiben, typischerweise auf virtuellen Maschinen oder Azure Kubernetes Service. Dies bietet volle Kontrolle über die Ausführungsumgebung, erfordert aber auch eigenen Betriebsaufwand für Bereitstellung, Skalierung, Überwachung und Wartung. Die Abrechnung erfolgt über die genutzten Compute-Ressourcen – VMs, Storage, Netzwerk – statt pro Token.

Im Foundry-Kontext ist dieses Modell primär in Hub-basierten Projekten relevant und adressiert spezialisierte Szenarien. Dazu gehören proprietäre oder stark angepasste Modelle, die nicht als Standard-Dienst verfügbar sind, Anforderungen an spezielle Hardware wie bestimmte GPU-Typen oder regulatorische Vorgaben, die dedizierte Infrastruktur verlangen. Es ist wichtig zu verstehen, dass die Funktionsabdeckung zwischen den Bereitstellungsoptionen variiert. Nicht alle Foundry-Features sind in allen Modi gleichermaßen verfügbar, was bei der Architekturplanung berücksichtigt werden muss.

Batch-Verarbeitung für Volumenszenarien

Für große Offline-Mengen bietet die Batch-API einen spezialisierten Pfad mit separaten Kontingenten und reduzierten Kosten gegenüber Online-Anfragen. Die Batch-API ist für Szenarien konzipiert, bei denen keine Echtzeit-Antworten benötigt werden und eine Abarbeitung innerhalb von 24 Stunden akzeptabel ist. Dies macht sie ideal für Log-Replays bei Incident-Analysen, Backfills beim Aktualisieren von Datensätzen mit neuen Erkenntnissen und Massentests bei der Evaluierung neuer Modellversionen.

Die Kostenreduktion in der Batch-API reflektiert die Tatsache, dass der Dienst Anfragen über Zeit akkumulieren und optimiert verarbeiten kann, ohne die Latenzanforderungen von Echtzeit-Interaktionen erfüllen zu müssen. Diese Flexibilität ermöglicht effizientere Ressourcennutzung und gibt die Einsparungen an die Kunden weiter.

Kostensteuerung in der Praxis

Die praktische Kontrolle von Kosten in KI-Systemen erfordert ein mehrschichtiges Vorgehen, das technische Optimierungen mit organisatorischen Prozessen kombiniert. Die Wahl des passenden Modells für jede Aufgabe ist fundamental. Kleine, einfache Aufgaben sollten nicht mit großen, teuren Modellen überversorgt werden. Ein einfacher Klassifikationsauftrag benötigt möglicherweise nicht die Fähigkeiten eines Frontier-Modells. Umgekehrt sollten anspruchsvolle Aufgaben, bei denen Qualität entscheidend ist, nicht aus Kostengründen an ungeeignete Modelle delegiert werden. Die Preislisten

geben detaillierte Sätze pro Million Tokens und medienspezifische Besonderheiten an, die bei der Modellauswahl konsultiert werden sollten.

Die Begrenzung von Prompt- und Antwortlänge hat direkten Einfluss auf die Kosten. Präzise Instruktionen vermeiden unnötige Ausschweifungen und bringen das Modell schneller zum Punkt. Harte Output-Limits über die max_tokens-Parameter senken die Ausgabetoken-Kosten. Token-Schätzungen sollten vorab in die Budgetierung eingeplant werden. Tools, die Token-Counts schätzen, helfen dabei, die Kosten von Prompts zu verstehen, bevor sie in Produktion gehen.

Intelligente RAG-Implementierung kann erhebliche Kosteneinsparungen bringen. Index-Hygiene – also das Entfernen veralteter oder redundanter Dokumente – verbessert nicht nur die Trefferqualität, sondern reduziert auch die Indexgröße und damit Kosten. Durchdachte Segmentierung sorgt dafür, dass nur relevante Teile von Dokumenten abgerufen werden, nicht ganze Dokumente. Effektive Filter grenzen die Suche auf relevante Dokumente ein, bevor teure Vektorsuchen durchgeführt werden. All dies senkt die Kontextlängen, die an Modelle übergeben werden, und damit die Kosten pro Anfrage.

Die Nutzung von Batch-Verarbeitung, wo immer möglich, ist eine einfache Kostenoptimierung. Asynchrone Stapel sind nicht nur günstiger, sondern entkoppeln auch von Online-Quoten, was zusätzliche Flexibilität schafft. Aufgaben wie nächtliche Berichte, regelmäßige Datenaktualisierungen oder umfassende Analysen sind oft hervorragende Kandidaten für Batch-Verarbeitung.

Das kontinuierliche Monitoring von Quoten ist entscheidend. Die Metrik „Throttled Requests" sollte systematisch in Application Insights überwacht werden. Die Token-per-Minute-Auslastung zeigt, wie nah ein System an seinen Limits operiert. Erhöhte Fehlerraten können ein Indikator für Kapazitätsprobleme sein. Alerts sollten proaktiv konfiguriert werden, um Teams zu warnen, bevor Probleme eskalieren.

Budgets und Kostenwarnungen in Cost Management sollten für alle Foundry-Projekte aktiviert werden. Wichtig ist dabei, Foundry-Kosten immer im Kontext aller beteiligten Azure-Ressourcen zu betrachten. Die Storage-Kosten für RAG-Dokumente, die Application-Insights-Kosten für Monitoring und die Netzwerkkosten für Datenübertragungen sind Teil des Gesamtbilds und sollten nicht übersehen werden.

Die Vermeidung von Netzwerk-Egress ist eine oft übersehene Optimierung. Das Halten von Daten innerhalb einer einzelnen Azure-Region minimiert Transferkosten. Internet- oder regionsübergreifende Transfers sollten nur erfolgen, wenn sie tatsächlich notwendig sind. Bei der Architekturplanung sollte die Datenlokalität explizit berücksichtigt werden, um diese Kosten zu vermeiden.

Entscheidungsfindung zwischen Bereitstellungsmodellen

Die Wahl des optimalen Bereitstellungsmodells ist keine einmalige Entscheidung, sondern sollte kontinuierlich auf Basis von Nutzungsmustern und Geschäftsanforderungen überprüft werden. Serverlose Bereitstellung eignet sich hervorragend für Szenarien, in denen schnelle Wertschöpfung im Vordergrund steht und die Last schwankend oder unvorhersehbar ist. Die niedrige Einstiegshürde ohne Vorabinvestitionen macht sie ideal für Experimente, frühe Entwicklungsphasen und Anwendungen mit variablem Datenverkehr.

Provisioned Throughput wird relevant, wenn planbare Latenz und garantierte Kapazität für stetige Produktionslast kritisch werden. Anwendungen mit Service-Level-Agreements, die bestimmte Antwortzeiten garantieren müssen, profitieren von der Vorhersagbarkeit von PTU. Ebenso Systeme mit konstanter, hoher Last, bei denen die wirtschaftliche Analyse zeigt, dass PTU trotz höherer Grundkosten insgesamt günstiger ist als die variable Pay-as-you-go-Abrechnung bei hoher Auslastung.

Verwaltete Rechenressourcen bieten volle Kontrolle und sind die richtige Wahl für spezialisierte Anforderungen. Dies umfasst Szenarien mit proprietären Modellen, die nicht als Managed Service verfügbar sind, Anforderungen an spezifische Hardware-Konfigurationen oder regulatorische Vorgaben, die dedizierte Infrastruktur verlangen. Der Trade-off ist der deutlich höhere Betriebsaufwand für das Selbstmanagement der Infrastruktur.

Viele Organisationen werden feststellen, dass eine Mischstrategie optimal ist. Entwicklungs- und Testumgebungen laufen serverlos für Kosteneffizienz und Flexibilität. Produktionssysteme mit vorhersagbarer Last nutzen PTU für Leistungsstabilität. Spezialisierte Workloads mit einzigartigen Anforderungen werden auf verwalteten Rechenressourcen betrieben. Diese hybride Strategie maximiert die Vorteile jedes Modells für die jeweils passenden Workloads.

Zusammenführung zu wirtschaftlich nachhaltigem Betrieb

Das Kosten- und Kapazitätsmanagement in Azure AI Foundry ist keine isolierte technische Aufgabe, sondern ein kontinuierlicher Prozess, der technisches Verständnis, wirtschaftliche Analyse und organisatorische Disziplin vereint. Die Transparenz des nutzungsbasierten Preismodells ist ein Vorteil, erfordert aber auch proaktives Management, um Kosten vorhersagbar und kontrolliert zu halten.

Die Kombination von kluger Modellauswahl, optimierten Prompt-Designs, effizienten RAG-Implementierungen und der richtigen Wahl des Bereitstellungsmodells kann die Kosten erheblich beeinflussen, ohne die Qualität zu kompromittieren. Das kontinuierliche Monitoring von Nutzungsmustern, Quoten-Auslastung und tatsächlichen Kosten liefert die Daten, die für fundierte Optimierungsentscheidungen notwendig sind.

Die Einrichtung von Budgets, Warnungen und automatisierten Governance-Regeln schafft Leitplanken, die verhindern, dass Kosten außer Kontrolle geraten. Die regelmäßige Überprüfung der Kostenverteilung, der Identifikation von Kostenausreißern und der Evaluation von Optimierungsmöglichkeiten sollte Teil des operativen Rhythmus sein, nicht eine einmalige Übung.

Letztlich ist das Ziel nicht, Kosten, um jeden Preis zu minimieren, sondern den optimalen Punkt zu finden, an dem Qualität, Leistung und Wirtschaftlichkeit im Einklang stehen. KI-Systeme erzeugen Geschäftswert, und die Kosten sollten im Verhältnis zu diesem Wert betrachtet werden. Ein System, das etwas mehr kostet, aber deutlich höhere Qualität liefert und damit bessere Geschäftsergebnisse ermöglicht, kann insgesamt wirtschaftlich vorteilhafter sein als ein billigeres System minderer Qualität. Die Werkzeuge und Mechanismen in Azure AI Foundry ermöglichen es, diese Balance bewusst zu gestalten und kontinuierlich zu optimieren.

11. Feinabstimmung und Anpassung

Feinabstimmung, im Englischen als Fine-Tuning bezeichnet, ist eine fortgeschrittene Technik zur Modellanpassung, die in einem spezifischen Kontext sinnvoll wird. Das Verständnis, wann Feinabstimmung angebracht ist und wann nicht, ist entscheidend für den effizienten Einsatz von Ressourcen und Zeit. Feinabstimmung sollte in Betracht gezogen werden, wenn Prompting-Techniken und Retrieval-Augmented Generation die angestrebte Zielqualität nicht stabil erreichen können oder wenn ein Modell intrinsisch neues Verhalten dauerhaft erlernen soll. Dies umfasst Szenarien wie die Verankerung eines spezifischen Sprachstils, die Verwendung domänenspezifischer Terminologie, die Einhaltung strikter Ausgabeschemata oder das Erlernen spezialisierter Schlussfolgerungsregeln.

Die goldene Regel lautet: Beginnen Sie stets mit Prompting-Optimierung und RAG-Integration. Diese Ansätze sind schneller zu implementieren, kostengünstiger und flexibler. Erst wenn deren Grenzen messbar erreicht sind und sich der erhebliche Aufwand der Feinabstimmung über eine große Anzahl von Anfragen amortisiert, sollte der Wechsel zu Fine-Tuning erfolgen. Azure AI Foundry stellt für diesen Prozess einen durchgängigen Ende-zu-Ende-Workflow bereit, sowohl über das Portal als auch programmatisch per SDK.

Der Entscheidungspfad: Von einfachen zu komplexen Ansätzen

Die Wahl zwischen verschiedenen Anpassungstechniken folgt einer logischen Progression, die von einfachen, flexiblen Methoden zu aufwendigeren, aber potenziell leistungsfähigeren Ansätzen führt. Das Verständnis dieser Progression hilft, fundierte Architekturentscheidungen zu treffen.

Prompting und Few-Shot Learning als Ausgangspunkt

Prompting, insbesondere mit wenigen Beispielen im Prompt selbst, ist die schnellste und kostengünstigste Methode zur Modellanpassung. Es entstehen keine Trainingskosten, und Änderungen können in Sekunden implementiert und getestet werden. Dieser Ansatz eignet sich hervorragend für die Steuerung von Tonalität und Stil, für einfache Formatvorgaben und für Aufgaben, bei denen einige wenige Beispiele im Prompt ausreichen, um das gewünschte Verhalten zu demonstrieren. Die Flexibilität ist unübertroffen – wenn sich Anforderungen ändern, wird einfach der Prompt angepasst.

Die Grenzen von Prompting werden erreicht, wenn das Kontextfenster durch zu viele Beispiele überladen wird, wenn das Verhalten über viele verschiedene Szenarien hinweg inkonsistent bleibt oder wenn die Prompt-Länge zu erheblichen Kosten führt, da jeder Token im Prompt bei jeder Anfrage verarbeitet werden muss.

RAG für dynamisches Wissen

Retrieval-Augmented Generation adressiert eine andere Dimension der Anpassung: das Hinzufügen von aktuellem, unternehmensspezifischem Wissen zur Laufzeit. Dies ist ideal, wenn die Informationsbasis sich häufig ändert – etwa bei Produktkatalogen, Unternehmensrichtlinien oder aktuellen Ereignissen. RAG ermöglicht es, das Wissen zu aktualisieren, ohne das Modell neu zu trainieren. Einfach die Dokumente im Index aktualisieren, und das Modell hat Zugriff auf die neuen Informationen.

RAG ist jedoch keine Lösung für Verhaltensanpassungen. Es kann dem Modell Fakten bereitstellen, aber es bringt dem Modell nicht bei, in einem bestimmten Stil zu schreiben oder spezifische Argumentationsmuster zu folgen. Diese Art von intrinsischem Verhalten erfordert Feinabstimmung.

Feinabstimmung für dauerhafte Verhaltensänderungen

Feinabstimmung wird relevant, wenn Verhalten dauerhaft im Modell verankert sein muss und nicht bei jeder Anfrage über lange Prompts re-etabliert werden soll. Typische Szenarien umfassen die Einhaltung strikter JSON-Schemata, bei denen das Modell komplexe, verschachtelte Strukturen fehlerfrei produzieren muss. Konsistente Fachsprache ist ein weiteres Beispiel – etwa medizinische oder juristische Terminologie, die präzise und durchgängig verwendet werden muss. Spezifische Argumentationsmuster, wie sie in bestimmten Branchen oder Methodologien üblich sind, können durch Feinabstimmung internalisiert werden.

Ein weiterer wichtiger Anwendungsfall ist die Kostenoptimierung. Wenn ein bestimmtes Verhalten durch sehr lange System-Prompts erzielt werden muss, verursacht dies bei hohen Anfragevolumen erhebliche Kosten. Durch Feinabstimmung kann dieses Verhalten ins Modell integriert werden, sodass deutlich kürzere Prompts ausreichen, was die Betriebskosten über Millionen von Anfragen hinweg signifikant senkt.

Technische Grundlagen und unterstützte Verfahren

Azure setzt bei der Feinabstimmung von Azure-OpenAI-Modellen standardmäßig Low-Rank Adaptation, kurz LoRA, ein. Diese Technik ist eine Form von Parameter-Efficient Fine-Tuning, abgekürzt PEFT. Das Konzept dahinter ist elegant: Statt alle Millionen oder Milliarden Parameter eines großen Modells zu aktualisieren, werden nur kleine, zusätzliche Adapter-Schichten trainiert. Diese Adapter modifizieren das Verhalten des Basismodells, ohne dessen ursprüngliche Gewichte vollständig zu verändern.

Der Vorteil liegt im reduzierten Trainingsaufwand und niedrigeren Kosten. LoRA benötigt weniger Rechenressourcen und Zeit als vollständiges Fine-Tuning und verringert gleichzeitig das Risiko des „Catastrophic Forgetting", bei dem das Modell seine ursprünglichen Fähigkeiten verliert. Die Basisgewichte bleiben intakt, und nur die Adapter werden gelernt und gespeichert.

Modellunterstützung und regionale Verfügbarkeit

Die Unterstützung für Feinabstimmung variiert erheblich nach Modell und Azure-Region. Dies ist keine willkürliche Einschränkung, sondern reflektiert die schrittweise Ausrollung von Funktionen und die Verfügbarkeit von Rechenkapazität in verschiedenen Regionen. Als gesichert gilt, dass gpt-4o-mini für Supervised Fine-Tuning unterstützt wird, was durch offizielle Tutorials bestätigt ist. Kleinere Varianten der 4.1-Modellfamilie sind ebenfalls verfügbar, wobei die genaue Unterstützung regionsabhängig ist.

Vor jedem Projekt ist es unerlässlich, die aktuelle Modell-Übersicht zu konsultieren. Diese wird von Microsoft kontinuierlich aktualisiert und reflektiert den aktuellen Stand der Verfügbarkeit. Planungen auf Basis veralteter Informationen können zu erheblichen Verzögerungen führen, wenn sich herausstellt, dass ein gewähltes Modell in der benötigten Region nicht für Fine-Tuning verfügbar ist.

Bereitstellungsmodi für Feinabstimmung

Feinabstimmung steht in Azure AI Foundry in zwei Modi bereit, die unterschiedliche Anforderungsprofile adressieren. Der serverlose Modus ist die Standardoption und bietet die einfachste Nutzung. Teams können sofort mit dem Training beginnen, ohne Infrastruktur bereitstellen zu müssen. Die Plattform verwaltet alle Aspekte der Ressourcenallokation und Skalierung transparent.

Managed Compute adressiert anspruchsvollere Szenarien, primär für offene Modelle mit erweiterten Trainingsparametern. Dies gibt mehr Kontrolle über die Trainingsumgebung, erfordert aber auch mehr Konfigurationsaufwand und tieferes technisches Verständnis. Für die meisten Azure-OpenAI-Feinabstimmungs-Szenarien ist der serverlose Modus ausreichend und empfohlen.

Reinforcement Fine-Tuning als zukünftige Ergänzung

Microsoft hat Reinforcement Fine-Tuning für das o1-mini-Modell angekündigt, wobei der Status als Vorschau oder „coming soon" ausgewiesen ist. RFT ist eine fortgeschrittenere Technik, die

Reinforcement Learning nutzt, um Modellverhalten durch Feedback zu optimieren. Zusätzliche Roadmap-Funktionen wie Pause und Resume von Trainingsläufen sowie Cross-Region-Copy für Checkpoints sind veröffentlicht oder angekündigt.

Die praktische Empfehlung lautet, konservativ zu planen. Supervised Fine-Tuning mit LoRA sollte als Standard angenommen werden, während Reinforcement Fine-Tuning projektweise evaluiert wird, sobald es allgemein verfügbar ist. Architekturen sollten nicht auf Funktionen aufgebaut werden, die noch in der Vorschau sind, es sei denn, das Risiko potenzieller Änderungen kann akzeptiert werden.

Der Workflow von Daten bis Bereitstellung

Der Feinabstimmungsprozess in Azure AI Foundry ist als durchgängiger Workflow konzipiert, der verschiedene Einstiegspunkte bietet, aber zu konsistenten Ergebnissen führt.

Portal-Erlebnisse und ihre Zielgruppen

Das Portal bietet zwei komplementäre Ansichten. Die Hub- und Projekt-Sicht unterstützt Feinabstimmung über mehrere Anbieter hinweg und richtet sich an Teams, die mit verschiedenen Modellquellen arbeiten. Die Azure-OpenAI-zentrierte Sicht bietet spezialisierte Features und eine Vorschau-Integration mit Weights & Biases, einer Plattform für Experiment-Tracking und Modell-Management. Diese Integration ermöglicht detaillierte Analysen von Trainingsläufen, Hyperparameter-Vergleiche und Visualisierungen des Trainingsprozesses.

Datenformat und Qualitätsanforderungen

Die Trainings- und Validierungsdaten müssen im JSONL-Format vorliegen, wobei jede Zeile ein Chat-Beispiel im Format der Chat-Completions-API repräsentiert. Dieses Format besteht aus einer Sequenz von Messages mit Rollen – typischerweise system, user und assistant. Die Mindestgröße beträgt zehn Beispiele, was für initiale Experimente ausreichen kann, aber für produktive Feinabstimmung unzureichend ist.

In der Praxis zeigt sich, dass etwa fünfzig gut kuratierte und repräsentative Beispiele ein guter Ausgangspunkt sind, während Hunderte oder mehr Beispiele für robuste Verhaltensänderungen empfohlen werden. Die Qualität der Beispiele ist dabei wichtiger als die reine Quantität. Sorgfältig kuratierte, diverse und repräsentative Beispiele führen zu besseren Ergebnissen als eine große Menge minderwertiger Daten.

Die maximale Dateigröße beträgt 512 Megabyte pro Datei, was für die meisten Anwendungsfälle mehr als ausreichend ist. Ein kritischer Aspekt ist die Konsistenz der System-Nachricht. Die System-Message, die beim späteren Inferenzaufruf verwendet wird, muss der beim Training verwendeten entsprechen.

Inkonsistenzen hier können zu unerwartetem Verhalten führen, da das Modell auf eine bestimmte System-Message-Struktur trainiert wurde.

Training und Monitoring

Der Trainingsstart erfolgt entweder über das Portal durch einen geführten Wizard oder programmatisch über das SDK für automatisierte Workflows. Während des Trainings bietet Foundry Monitoring-Metriken, die Einblicke in den Fortschritt geben. Der Loss, also die Verlustfunktion, zeigt, wie gut das Modell die Trainingsdaten zu jedem Zeitpunkt modelliert. Ein sinkender Loss indiziert Lernen, während ein stagnierender oder steigender Loss auf Probleme hinweisen kann.

Token-Accuracy misst, wie oft das Modell das nächste Token korrekt vorhersagt. Dies ist eine direkte Messung der Modellleistung auf der Trainingsaufgabe. Das Portal bietet auch Hinweise zur Überanpassung, ein kritisches Phänomen, bei dem das Modell die Trainingsdaten auswendig lernt, aber nicht gut auf neue, ungesehene Daten generalisiert.

Bereitstellung und Kostenfaktoren

Nach erfolgreichem Training können feinabgestimmte Modelle wie Standard-Bereitstellungen genutzt werden. Sie erscheinen als verfügbare Modelle und können über die gleichen APIs aufgerufen werden. Ein wichtiger Kostenfaktor, der oft übersehen wird, ist die stündliche Hosting-Gebühr für bereitgestellte benutzerdefinierte Modelle. Diese Gebühr fällt kontinuierlich an, auch wenn kein Traffic das Modell erreicht. Im Unterschied zu Standard-Modellen, die von vielen Kunden geteilt werden, ist ein feinabgestimmtes Modell kundendediziert und reserviert daher Ressourcen.

Um unnötige Kosten zu vermeiden, werden inaktive Bereitstellungen nach fünfzehn Tagen automatisch entfernt. Dies löscht nicht das zugrunde liegende Modell, sondern nur die Bereitstellung. Das trainierte Modell bleibt erhalten und kann jederzeit neu bereitgestellt werden. Teams sollten dies bei ihrer Kapazitätsplanung berücksichtigen und ungenutzte Bereitstellungen proaktiv entfernen.

Regionale Strategien und Datenresidenz

Die regionale Dimension der Feinabstimmung hat sowohl technische als auch regulatorische Implikationen. Supervised Fine-Tuning unterstützt sowohl regionale als auch globale Trainings-SKUs. Regionale SKUs beschränken Training und Inferenz auf eine spezifische Azure-Region, was Datenresidenz-Anforderungen erfüllt und Latenz minimiert.

Globale SKUs ermöglichen Kapazitätsnutzung außerhalb der Heimatregion des Projekts. Dies kann günstiger sein und Zugang zu mehr verfügbarer Trainingskapazität bieten, wenn die primäre Region ausgelastet ist. Der Trade-off ist, dass Daten möglicherweise Regionen verlassen, was für regulierte

Industrien oder bei strengen Datenschutzvorgaben problematisch sein kann. Vor der Wahl einer globalen SKU müssen Residenzanforderungen sorgfältig geprüft werden.

Cross-Region-Kopien von Checkpoints sind per API verfügbar. Dies ermöglicht interessante Szenarien: Ein Modell kann in einer Region mit verfügbarer Kapazität trainiert, dann in eine andere Region kopiert und dort weiter trainiert oder bereitgestellt werden. Dies bietet Flexibilität bei Kapazitätsengpässen und ermöglicht globale Bereitstellungsstrategien.

Datenschutz und Datenverwendung

Ein kritischer Aspekt für viele Organisationen ist die Frage, wie Trainingsdaten verwendet werden. Microsoft hat explizit zugesichert, dass Trainingsdaten für Feinabstimmung nicht ohne Kundenzustimmung zur Verbesserung von Grundmodellen verwendet werden. Benutzerdefinierte Modelle sind kundendediziert und deren Gewichte werden nicht mit anderen Kunden geteilt oder für das Training öffentlicher Modelle verwendet.

Dennoch sollten die spezifischen Bedingungen der gewählten Trainings-SKU beachtet werden, da verschiedene SKUs unterschiedliche Zusicherungen bieten können. Die Dokumentation zu den Data-Privacy-Aspekten der jeweiligen SKU sollte konsultiert werden, besonders wenn hochsensible Daten für das Training verwendet werden.

Qualitätssicherung vor Produktivsetzung

Feinabgestimmte Modelle sollten niemals direkt nach dem Training live geschaltet werden, ohne rigorose Qualitätsprüfungen durchlaufen zu haben. Die Evaluierungsfunktionen in Azure AI Foundry sollten als verbindliche Quality-Gates eingesetzt werden. Dies umfasst Qualitäts-Scores über relevante Metriken wie Relevanz, Kohärenz und Flüssigkeit. Safety-Checks stellen sicher, dass das feinabgestimmte Modell keine problematischen Inhalte generiert, die durch die Trainingsdaten eingeführt worden sein könnten.

Groundedness-Prüfung ist besonders relevant, wenn das feinabgestimmte Modell in RAG-Szenarien eingesetzt wird. Die optionale Korrektur-Funktion, derzeit in der Vorschau, kann ungestützte Passagen automatisch angleichen. Für Feinabstimmung existieren zusätzliche, spezialisierte Safety-Evaluationen, die auf die spezifischen Risiken dieses Prozesses zugeschnitten sind.

Risiken und ihre Mitigation

Feinabstimmung ist ein mächtiges Werkzeug, birgt aber spezifische Risiken, die verstanden und aktiv adressiert werden müssen. Jedes dieser Risiken hat etablierte Gegenmaßnahmen, die in den Prozess integriert werden sollten.

Catastrophic Forgetting

Catastrophic Forgetting beschreibt das Phänomen, bei dem ein Modell während der Feinabstimmung seine ursprünglichen Basisfähigkeiten verliert. Das Modell wird hervorragend in der spezifischen Trainingsaufgabe, vergisst aber grundlegende Fähigkeiten, die es zuvor hatte. Ein Beispiel wäre ein feinabgestimmtes Modell für medizinische Terminologie, das danach nicht mehr gut in allgemeiner Konversation ist.

Die Gegenmaßnahmen sind mehrschichtig. Ein gemischtes Trainingsset, das neben spezialisierten Beispielen auch allgemeine Beispiele enthält, hilft dem Modell, seine breiten Fähigkeiten zu bewahren. Konservative Hyperparameter, insbesondere eine niedrigere Lernrate und weniger Trainings-Epochen, verringern das Risiko drastischer Gewichtsänderungen. Die Verwendung von PEFT-Techniken wie LoRA statt vollständigem Fine-Tuning minimiert das Risiko strukturell, da die Basisgewichte unverändert bleiben.

Unabhängige Benchmarks sollten regelmäßig gegen das feinabgestimmte Modell laufen. Standardisierte Evaluierungen auf allgemeinen Aufgaben zeigen, ob Basisfähigkeiten erhalten geblieben sind. Ein Rückgang in diesen Benchmarks ist ein Warnsignal, dass zu aggressiv feinabgestimmt wurde.

Überanpassung und „Memorisierung“

Überanpassung tritt auf, wenn ein Modell die spezifischen Trainingsdaten auswendig lernt, statt allgemeine Muster zu extrahieren. Memorisierung ist eine extreme Form, bei der das Modell Trainingsbeispiele wortwörtlich reproduzieren kann, was besonders problematisch ist, wenn diese sensible Informationen enthalten.

Sorgfältige Validierung mit einem Hold-out-Validierungsset, das nicht im Training verwendet wurde, ist essentiell. Wenn das Modell auf Trainingsdaten deutlich besser abschneidet als auf Validierungsdaten, ist dies ein klares Zeichen für Überanpassung. Frühes Stoppen, bei dem das Training beendet wird, sobald die Validierungsleistung nicht mehr verbessert oder sich verschlechtert, verhindert übermäßige Anpassung.

Scans auf sensible Inhalte in den Trainingsdaten sind eine präventive Maßnahme. Wenn bekannt ist, dass keine sensiblen Daten im Trainingsset sind, kann Memorisierung kein Datenschutzproblem verursachen. Nachgelagerte Extraction-Tests, bei denen versucht wird, Trainingsbeispiele aus dem Modell zu extrahieren, validieren, dass keine wortwörtliche Memorisierung stattgefunden hat.

Bias-Verstärkung

Feinabstimmung kann existierende Biases in den Trainingsdaten verstärken oder neue Biases einführen. Wenn das Trainingsset nicht ausgewogen ist – etwa mehr Beispiele aus bestimmten

demografischen Gruppen oder Perspektiven enthält – wird das Modell diese Unausgewogenheit reflektieren und möglicherweise verstärken.

Datensätze sollten bewusst balanciert werden, um verschiedene Perspektiven, demografische Gruppen und Szenarien repräsentativ abzudecken. Fairness-Evaluierungen sollten explizit in den Evaluierungsplan eingeplant werden. Diese testen, ob das Modell verschiedene Gruppen unterschiedlich behandelt und können subtile Biases aufdecken, die in allgemeinen Qualitätsmetriken nicht sichtbar werden.

Compliance und Governance

Die Feinabstimmung wirft spezifische Compliance-Fragen auf, die über die normale Modellnutzung hinausgehen. Die Datenherkunft muss dokumentiert sein – woher stammen die Trainingsdaten und welche Rechte hat die Organisation an ihnen? Aufbewahrungspolitiken müssen definieren, wie lange Trainingsdaten und trainierte Modelle gespeichert werden. Zugriffskontrolle muss regeln, wer Modelle trainieren, Trainingsdaten einsehen oder feinabgestimmte Modelle nutzen darf.

Die Residenz der gewählten Trainings-SKU muss mit regulatorischen Anforderungen abgeglichen werden. In regulierten Branchen wie Finanzen oder Gesundheitswesen können strenge Vorgaben existieren, wo Daten verarbeitet werden dürfen. All diese Aspekte sollten bereits in der Planungsphase adressiert werden, nicht erst wenn das Modell trainiert ist.

Betrieb und kontinuierliche Optimierung

Feinabgestimmte Modelle verhalten sich im Aufruf wie Basis-Modelle und können mit denselben Werkzeugen überwacht und verwaltet werden. Die Observability-Funktionen von Foundry sollten genutzt werden, um Latenz, Fehlerraten, Token-Verbrauch und Hosting-Kosten kontinuierlich zu überwachen. Cost Analysis zeigt die Gesamtkosten, die sich aus Inferenzkosten und den stündlichen Hosting-Gebühren zusammensetzen.

Bei Bedarf können gestaffelte Rollouts implementiert werden, allerdings nicht auf Modell-Endpunkt-Ebene, sondern in der Zugriffs-Schicht. Azure Front Door, API Management oder ähnliche Dienste können verwendet werden, um Traffic graduell von einem alten zu einem neuen feinabgestimmten Modell zu verlagern. Dies minimiert das Risiko, wenn ein neu feinabgestimmtes Modell eingeführt wird.

Zusammenfassung zum robusten Feinabstimmungsprozess

Die erfolgreiche Feinabstimmung in Azure AI Foundry erfordert ein systematisches Vorgehen, das technisches Verständnis mit rigorosen Qualitätssicherungsprozessen verbindet. Der empfohlene Pfad nutzt Supervised Fine-Tuning mit LoRA als Standardmethode, da dies die beste Balance zwischen

Effektivität und Effizienz bietet. Die Wahl zwischen serverloser Bereitstellung und verwalteten Rechenressourcen sollte basierend auf den Kontroll- und Parametrisierungsbedürfnissen getroffen werden, wobei serverlos für die meisten Szenarien ausreicht.

Harte Quality-Gates und Safety-Checks vor der Produktivsetzung sind nicht optional, sondern essentiell. Kein feinabgestimmtes Modell sollte direkt nach dem Training produktiv gehen, ohne umfassende Evaluierungen durchlaufen zu haben. Die Prüfung der Modell- und Regionsmatrix vor jedem Projektstart verhindert Überraschungen und Verzögerungen während der Implementierung.

Wenn diese Prinzipien befolgt werden, bleibt die Feinabstimmung reproduzierbar durch sorgfältige Versionierung von Daten und Modellen, sicher durch rigorose Safety-Checks und Risiko-Mitigation, und kostenkontrolliert durch transparentes Monitoring und proaktives Management ungenutzter Bereitstellungen. Die Feinabstimmung wird dann zu einem wertvollen Werkzeug im Portfolio der Modellanpassungsstrategien, das eingesetzt wird, wenn einfachere Ansätze ihre Grenzen erreichen, aber mit der notwendigen Sorgfalt und Vorsicht, die diese mächtige Technik verlangt.

12. Interoperabilität der SDKs und Frameworks

Azure AI Foundry verfolgt konsequent das Prinzip der Einbindung statt des Ersatzes bestehender Werkzeuge. Die Plattform ruht auf drei zentralen Säulen: einheitliche Inferenz-Schnittstellen über alle Modellanbieter hinweg, offene Standards für Beobachtbarkeit und durchdachte Integrationen in etablierte Frameworks. Diese Architektur ermöglicht es, vorhandene Toolketten weiterzuverwenden und schrittweise zu professionalisieren, ohne bestehende Investitionen zu entwerten.

SDK-Portfolio mit klarer Aufgabenteilung

Die Azure AI Foundry SDKs folgen dem Prinzip der Verantwortungstrennung. Jedes SDK adressiert einen spezifischen Aspekt der KI-Entwicklung, ohne unnötige Überschneidungen zu erzeugen.

- Inference-SDK (`azure-ai-inference`). Das Inference-SDK stellt eine einheitliche Laufzeit-API für Modellaufrufe über verschiedene Anbieter bereit. Die Kernoperationen umfassen `complete()` für Chat-Completions und `embed()` für Text- und Bild-Embeddings, jeweils mit optionaler Streaming-Unterstützung. Die Authentifizierung erfolgt wahlweise per API-Schlüssel oder Entra-Token. Besonders hervorzuheben ist die integrierte OpenTelemetry-Tracing-Unterstützung, die sich optional auch auf die Aufzeichnung von Eingabe- und Ausgabeinhalten erstrecken lässt. Diese Funktionalität wird über Umgebungsvariablen gesteuert.

- Projects-SDK (`azure-ai-projects`). Der Projekt-Client fungiert als zentrale Koordinationsstelle und bündelt Zugriffe auf Bereitstellungen, Verbindungen, Indizes und Agenten innerhalb eines Projekts. Eine besonders nützliche Funktion ist `get_openai_client()`, die einen vollständig authentifizierten OpenAI-kompatiblen Client am Projektendpunkt bereitstellt. Die Authentifizierung läuft über die bewährte `DefaultAzureCredential`, was eine nahtlose Integration in Azure-Umgebungen mit verwalteten Identitäten ermöglicht.
- Evaluation-SDK (`azure-ai-evaluation`). Dieses SDK ermöglicht programmatische Qualitäts- und Sicherheitsevaluierungen für Modelle, RAG-Abläufe und Agenten. Evaluierungen können lokal durchgeführt oder als entfernte Ausführungen im Projekt protokolliert werden. Diese Flexibilität erlaubt sowohl schnelle Entwicklungsiterationen als auch reproduzierbare Evaluierungen im Team.
- Beobachtbarkeit und Fehlerbehandlung. Azure AI Foundry und die zugehörigen SDKs bauen auf OpenTelemetry und W3C Trace Context als Fundament für die Beobachtbarkeit. Telemetriedaten lassen sich wahlweise in Azure Monitor und Application Insights oder in jedem anderen OpenTelemetry-fähigen Backend auswerten. Für die Fehlerbehandlung implementieren die Azure-SDKs Exponential-Backoff als Standard-Wiederholungsstrategie. Das Verhalten lässt sich über die RetryPolicy präzise an spezifische Anforderungen anpassen.

OpenAI-Kompatibilität und Migrationspfade

Für OpenAI-Modelle stellt Azure AI Foundry neben der nativen Responses-API auch OpenAI-kompatible v1-Endpunkte bereit (Pfadmuster: `/openai/v1/...`). Diese Kompatibilitätsschicht reduziert Migrationsaufwände erheblich. In den meisten Fällen beschränken sich notwendige Anpassungen auf drei Bereiche: die Basis-URL (Umstellung auf Projekt- oder OpenAI-Endpunkt), die Authentifizierungsmethode (Entra-Token oder API-Schlüssel) und die Modellreferenz (Verwendung von Bereitstellungsnamen statt OpenAI-Modellnamen). Der Projects-Client kann über `get_openai_client()` einen fertig authentifizierten `AzureOpenAI`-Client bereitstellen, was die Integration zusätzlich vereinfacht.

Integration in etablierte Frameworks

LangChain

Die Integration von Azure AI Foundry in LangChain erfolgt über mehrere spezialisierte Komponenten:

- **Azure OpenAI in LangChain:** Für Azure-OpenAI-Bereitstellungen steht die Klasse `AzureChatOpenAI` aus dem Paket `langchain-openai` zur Verfügung. Diese Klasse bietet

vollständigen Zugriff auf Chat-Modelle mit allen LangChain-spezifischen Features wie Callbacks, Streaming und Batch-Verarbeitung.

- **Foundry-weite Inferenz in LangChain:** Um auch Modelle jenseits von Azure OpenAI anzusprechen, nutzt man `AzureAIChatCompletionsModel`. Diese Komponente implementiert das Foundry-Inferenzschema und ermöglicht den Zugriff auf das gesamte Modellportfolio der Plattform, einschließlich Drittanbieter-Modelle und Open-Source-Modelle.
- **Retriever für RAG-Szenarien:** Für Retrieval-Augmented-Generation ist `AzureAISearchRetriever` der aktuelle Standard. Der ältere `AzureCognitiveSearchRetriever` ist als Deprecated markiert und sollte in neuen Implementierungen vermieden werden. Der moderne Retriever bietet verbesserte Performance und bessere Integration in die aktuelle Azure AI Search-Architektur.

Semantic Kernel

Semantic Kernel (SK) integriert Azure-OpenAI-Modelle und Azure-Dienste als Konnektoren und Plugins und fügt sich nahtlos in .NET-typische Architekturmuster ein. Die Integration nutzt Dependency Injection und das etablierte .NET-Logging-Framework. Microsoft dokumentiert die Verwendung von Foundry-Modellen mit Semantic Kernel in der offiziellen Dokumentation ausführlich, was den Einstieg erheblich erleichtert. Semantic Kernel eignet sich besonders für Entwicklungsteams, die bereits im .NET-Ökosystem verankert sind und KI-Funktionalität in bestehende Unternehmensanwendungen integrieren möchten.

AutoGen

AutoGen unterstützt AAD/Entra-Authentifizierung für Azure-OpenAI-Endpunkte vollständig. Dies ermöglicht den Aufbau von Multi-Agenten-Szenarien mit Foundry-Bereitstellungen unter Nutzung verwalteter Identitäten. Die Kombination aus AutoGens Orchestrierungsfähigkeiten und Foundrys einheitlichen Schnittstellen schafft eine leistungsfähige Grundlage für komplexe agentenbasierte Anwendungen.

Offene Standards als Architekturprinzip

Azure AI Foundry setzt konsequent auf offene, herstellerunabhängige Standards. OpenTelemetry als CNCF-Standard bildet das Rückgrat der Beobachtbarkeit. Foundry und die zugehörigen SDKs erzeugen Traces nach OpenTelemetry-Konventionen und propagieren Kontextinformationen über W3C Trace Context. Diese Architekturentscheidung stellt sicher, dass Telemetriedaten nicht an ein spezifisches Backend gebunden sind. Entwicklungsteams können frei zwischen Azure Monitor, Application Insights oder beliebigen OpenTelemetry-kompatiblen Systemen wählen, ohne Vendor-Lock-in befürchten zu müssen.

REST-APIs als universelle Grundlage

Für Szenarien, in denen kein passendes SDK zur Verfügung steht oder direkte REST-Aufrufe bevorzugt werden, stellt Azure AI Foundry vollständige REST-Spezifikationen bereit. Die Azure-AI-Model-Inference-API dient als einheitliches Schema für Inferenzaufrufe über verschiedene Modellanbieter hinweg. Parallel dazu existiert die Azure-OpenAI-Spezifikation für OpenAI-spezifische Funktionalität. Diese REST-APIs bilden die gemeinsame Basis, auf der alle höheren Abstraktionsebenen aufbauen.

Typische Migrationspfade

Die Architektur von Azure AI Foundry ermöglicht verschiedene Migrationspfade mit kalkulierbarem Aufwand:

- **OpenAI-Anwendung zu Foundry:** Bei der Migration einer OpenAI-basierten Anwendung konzentrieren sich die Änderungen auf drei Konfigurationspunkte: die Basis-URL muss auf einen Projekt- oder OpenAI-Endpunkt in Foundry umgestellt werden, die Authentifizierung wechselt zu Entra-Token oder Foundry-API-Schlüsseln, und Modellreferenzen müssen auf Bereitstellungsnamen angepasst werden. Die Anwendungslogik selbst bleibt unverändert, was eine schrittweise Migration ermöglicht.
- **LangChain-Prototyp zu Produktionssystem:** Der Übergang von einem LangChain-POC zur Produktionsreife erfordert mehrere Professionalisierungsschritte: Verwendung von `AzureChatOpenAI` oder `AzureAIChatCompletionsModel` für robuste Modellansprache, Migration des Retrieval-Systems zu Azure AI Search mit dem modernen `AzureAISearchRetriever`, Umstellung der Authentifizierung auf `DefaultAzureCredential` für sichere Identitätsverwaltung, und Aktivierung der OpenTelemetry-basierten Telemetrie für vollständige Beobachtbarkeit.
- **.NET-Bestandssysteme zu KI-erweiterten Anwendungen:** Für die Integration von KI-Funktionalität in bestehende .NET-Anwendungen bietet Semantic Kernel einen strukturierten Pfad: Nutzung der SK-Konnektoren für Chat-Funktionalität und Embeddings, Konfiguration über Dependency Injection für testbare und wartbare Architekturen, und Integration des Monitorings in Application Insights für einheitliche Beobachtbarkeit über die gesamte Anwendungslandschaft.

Zusammenfassung

Interoperabilität ist in Azure AI Foundry kein nachträglich hinzugefügtes Feature, sondern ein fundamentales Entwurfsprinzip. Die Plattform vereint einheitliche Inferenz-Schnittstellen, präzise abgegrenzte SDKs nach Zuständigkeitsbereichen, OpenTelemetry und W3C-Standards für

herstellerunabhängige Beobachtbarkeit sowie erstklassige Integrationen in LangChain, Semantic Kernel und AutoGen. Diese Architektur macht vorhandene Investitionen nutzbar, ermöglicht schrittweise Migrationen und schafft eine solide Grundlage für produktionsreife KI-Systeme. Die Kombination aus offenen Standards und pragmatischen Framework-Integrationen reduziert technische Schulden und erhöht die Flexibilität bei zukünftigen Technologieentscheidungen.

13. Betriebs- und Governance-Leitplanken

Produktionsreife KI-Systeme verlangen nach robusten Leitplanken, die über reine Funktionalität hinausgehen. Im Zentrum stehen vier fundamentale Anforderungen: präzise Identitäts- und Berechtigungskonzepte, wirksame Netzwerkisolation mit Kontrolle über Datenabflüsse, lückenlos prüfbare Protokollierung sowie die Einbindung in etablierte Compliance-Rahmen. Azure AI Foundry adressiert diese Anforderungen durch ein durchdachtes Zusammenspiel aus Rollenmodellen, Richtlinien, Netzwerkkontrollen, Beobachtbarkeit und Integrationen in Sicherheits- sowie Datengovernance-Dienste. Diese Mechanismen bilden das Fundament für verantwortungsvolle Betriebsführung von KI-Anwendungen in Unternehmensumgebungen.

Zugriffssteuerung mit Azure RBAC

Azure AI Foundry nutzt die rollenbasierte Zugriffskontrolle von Azure mit spezialisierten Rollen für die Konto- und Projektebene. Die drei zentralen integrierten Rollen sind Azure AI User, Azure AI Project Manager und Azure AI Account Owner. Diese Rollenhierarchie ermöglicht eine klare Trennung zwischen administrativer Verwaltung auf Konto- und Projektebene, aktiver Entwicklungsarbeit innerhalb von Projekten und minimalen Datenoperationen für spezifische Anwendungsfälle. Ergänzend können die allgemeinen Azure-Rollen Owner, Contributor und Reader erforderlich sein, insbesondere wenn übergreifende Ressourcenverwaltung notwendig ist.

Für komplexere Berechtigungsstrukturen, die über die integrierten Rollen hinausgehen, lassen sich benutzerdefinierte Rollen definieren. Die zentrale Empfehlung lautet, das Prinzip der minimalen Rechte konsequent durchzusetzen und Rollenzuweisungen präzise am jeweiligen Geltungsbereich auszurichten. Dies verhindert übermäßige Privilegien und reduziert das Risiko unbeabsichtigter oder missbräuchlicher Aktionen erheblich.

Ein praxisnahes Rollenlayout könnte folgendermaßen aussehen: IT-Administratoren erhalten die Rolle Azure AI Account Owner auf der Ebene des Foundry-Kontos und können so grundlegende Infrastrukturentscheidungen treffen. Teamleiter bekommen Azure AI Project Manager für ihre jeweiligen Projekte zugewiesen, was ihnen die Kontrolle über Projektressourcen gibt, ohne globale Administrationsrechte zu erfordern. Entwickler arbeiten mit der Rolle Azure AI User innerhalb ihrer

zugewiesenen Projekte. Diese Struktur ermöglicht es, Projektaufbau und Entwicklungsaktivitäten zu delegieren, ohne dass weitreichende globale Berechtigungen verteilt werden müssen. Die klare Abgrenzung der Zuständigkeiten schafft zudem natürliche Prüfpunkte im Entwicklungsprozess.

Identitäten, bedingter Zugriff und Privileged Identity Management

Die schlüssellose Authentifizierung über Microsoft Entra ID bildet den Standard für den Zugriff auf Azure AI Foundry. Diese Architektur eliminiert die Risiken, die mit der Verwaltung und Rotation von API-Schlüsseln einhergehen. Zugriffe lassen sich über Richtlinien für bedingten Zugriff weiter absichern, und zwar nicht nur für menschliche Benutzer, sondern auch für Workload-Identitäten in Form von Dienstprinzipalen. Bedingungen können an Faktoren wie Standort, Risikobewertung oder spezifische Authentifizierungskontexte geknüpft werden, was eine flexible und gleichzeitig strenge Zugriffskontrolle ermöglicht.

Hochprivilegierte Rollen sollten über Privileged Identity Management zeitlich befristet als berechtigungsfähig definiert und zustimmungspflichtig aktiviert werden. PIM stellt sicher, dass privilegierte Zugriffe nur dann und nur so lange gewährt werden, wie sie tatsächlich benötigt werden, und dass jede Aktivierung dokumentiert und genehmigt wird. Dies reduziert das Zeitfenster, in dem kompromittierte Konten Schaden anrichten könnten, erheblich.

Ein wichtiger technischer Hinweis betrifft verwaltete Identitäten: Diese fallen derzeit nicht unter Richtlinien für bedingten Zugriff. Bei verwalteten Identitäten greifen stattdessen die RBAC-Berechtigungen und Netzwerkkontrollen als primäre Sicherheitsmechanismen. Diese Unterscheidung ist in der Architektur von Produktionssystemen zu berücksichtigen, insbesondere wenn automatisierte Prozesse mit verwalteten Identitäten arbeiten.

Netzwerk-Governance und private Konnektivität

Die Netzwerksicherheit folgt dem Prinzip der standardmäßigen Verweigerung, das technisch konsequent umgesetzt werden muss. Der öffentliche Netzwerkzugriff sollte deaktiviert werden, stattdessen sind Private Endpoints zu erzwingen. Wo immer möglich, sollte die Foundry-Umgebung in ein verwaltetes virtuelles Netzwerk eingebunden werden, das im Modus für ausschließlich genehmigten ausgehenden Verkehr betrieben wird. Diese Konfiguration verhindert, dass Daten unkontrolliert die Umgebung verlassen können.

Die Durchsetzung dieser Netzwerkrichtlinien lässt sich über Azure Policy automatisieren. Richtlinien können so konfiguriert werden, dass sie die Erstellung von Ressourcen ohne Private Endpoints

blockieren oder bestehende Ressourcen auf Nichtkonformität prüfen. Für abhängige Dienste wie Azure AI Services, Azure AI Search, Azure Storage, Azure Key Vault oder Azure Cosmos DB sind korrespondierende Private-Link-Verbindungen und DNS-Konfigurationen erforderlich. Diese Dienste müssen ebenfalls über Private Endpoints erreichbar sein, um eine durchgängige Netzwerkisolation zu gewährleisten.

Die Kontrolle ausgehenden Verkehrs bleibt eine zentrale Anforderung. Ausgehende Verbindungen sollten ausschließlich zu explizit genehmigten Zielen möglich sein, identifiziert über vollqualifizierte Domänennamen oder Azure Service Tags. Das verwaltete virtuelle Netzwerk für Hubs unterstützt die Betriebsart, die ausschließlich genehmigten ausgehenden Verkehr zulässt. Bei Bedarf für zusätzlichen domänenbasierten Zugriff müssen ergänzende Regeln in Azure Firewall konfiguriert werden. Diese mehrschichtige Kontrolle stellt sicher, dass selbst im Fall einer Kompromittierung die Exfiltration von Daten oder die Kommunikation mit schädlicher Infrastruktur unterbunden wird.

Werkzeug- und Aktionskontrolle für Agenten

Agenten in Azure AI Foundry sollten ausschließlich explizit zugelassene Werkzeuge nutzen dürfen. Die Werkzeugliste wird pro Agent konfiguriert, wodurch eine präzise Kontrolle über die Fähigkeiten jedes Agenten entsteht. Verbindungen zu externen Diensten oder Datenquellen sind über Private Endpoints einzubinden, um sicherzustellen, dass auch die Werkzeugnutzung den Netzwerksicherheitsrichtlinien entspricht.

Für eigene Werkzeuge und Verbindungen gelten die identischen Netzwerk- und Identitätsvorgaben wie für integrierte Komponenten. Dieser einheitliche Ansatz verhindert, dass durch benutzerdefinierte Erweiterungen Sicherheitslücken entstehen. Das Resultat ist eine Positivliste auf Anwendungsebene, die den Aktionsradius eines Agenten klar definiert. Diese Architektur erweist sich als besonders wertvoll als Verteidigungslinie gegen Prompt-Injection-Angriffe. Selbst wenn ein Angreifer es schafft, einen Agenten über manipulierte Eingaben zu steuern, bleibt der mögliche Schaden auf die zugelassenen Werkzeuge und deren definierte Einsatzbereiche beschränkt.

Daten-Governance mit Microsoft Purview

Microsoft Purview liefert die zentrale Infrastruktur für Datenkatalogisierung, Klassifizierung, Richtliniendurchsetzung und Verhinderung von Datenverlust über Datenquellen hinweg. Scans und Datenherkunftsanalysen werden auf der Datenebene etabliert, typischerweise für Quellen wie Azure Storage, Azure SQL, Azure Synapse oder Azure Data Factory. Die erfassten Metadaten und Klassifizierungen werden in der Data Map dokumentiert und stehen für nachgelagerte Anwendungen zur Verfügung.

KI-Anwendungen können Purview-Kontrollen über SDK-Integrationen berücksichtigen. Ein praktisches Beispiel ist die Ergebnisfilterung in RAG-Szenarien: Azure AI Search implementiert Security-Trimming zur Abfragezeit, indem Benutzer- und Gruppenkennungen in Dokumentmetadaten mit den Berechtigungen des anfragenden Benutzers abgeglichen werden. Dies stellt sicher, dass Suchergebnisse automatisch auf die Dokumente gefiltert werden, auf die der Benutzer zugreifen darf.

Wichtig ist die realistische Einordnung der Purview-Integration: Purview ersetzt keine rollenbasierte Zugriffskontrolle innerhalb der einzelnen Dienste und prüft nicht automatisch jeden Ausgabetext eines Agenten. Funktionen wie Data Loss Prevention oder Datenmaskierung müssen bewusst in die Anwendungsarchitektur integriert werden. Dies erfordert explizite Prüfpunkte im Datenfluss, an denen Purview-Richtlinien abgefragt und durchgesetzt werden. Die Verantwortung für die korrekte Integration liegt beim Anwendungsentwickler.

Prüfprotokolle, Nachvollziehbarkeit und Langzeitablage

Vollständige Nachvollziehbarkeit setzt umfassende Protokollierung voraus. Konfigurations- und Verwaltungsaktionen auf der Steuerungsebene werden automatisch im Azure Activity Log erfasst. Dies umfasst Aktionen wie das Erstellen von Projekten, Änderungen an Berechtigungen oder die Konfiguration von Netzwerkregeln. Für Datenebenen- und Dienstprotokolle, die die eigentliche Nutzung der KI-Dienste dokumentieren, kommen die Diagnostic Settings zum Einsatz. Diese Einstellungen definieren, welche Protokolle erfasst werden und wohin sie geleitet werden.

Als Ziele für Protokolldaten stehen Log Analytics-Arbeitsbereiche für interaktive Abfragen und Dashboards, Event Hubs für die Weiterleitung an externe Systeme und Azure Storage für kostengünstige Langzeitarchivierung zur Verfügung. Für manipulationssichere Aufbewahrung bietet Azure mehrere Mechanismen. Blob-WORM-Richtlinien machen Speicherobjekte unveränderbar für einen definierten Zeitraum. Für höchste Anforderungen an Manipulationssicherheit steht Azure Confidential Ledger zur Verfügung, ein kryptografisch gesichertes Append-Only-Register.

Diese Architektur ermöglicht belastbare Beweisführung, konforme Aufbewahrung gemäß regulatorischer Vorgaben und forensische Analysen im Fall von Sicherheitsvorfällen. Die Kombination aus vollständiger Erfassung, manipulationssicherer Speicherung und leistungsfähigen Analysewerkzeugen schafft die Grundlage für Transparenz und Rechenschaftsfähigkeit.

Compliance-Rahmen systematisch adressieren

Die Bausteine von Azure AI Foundry unterstützen die Umsetzung etablierter Compliance-Rahmen. Das NIST AI Risk Management Framework mit seinen vier Funktionen – Govern, Map, Measure und Manage

– findet direkte Entsprechungen in den Governance- und Observability-Funktionen der Plattform. ISO/IEC 42001, der Standard für Managementsysteme künstlicher Intelligenz, wird durch die strukturierten Prozesse für Änderungsmanagement, Evaluierung und Risikokontrolle adressiert.

Besonders relevant für europäische Organisationen sind die Anforderungen des EU-AI-Act für Hochrisikosysteme. Microsoft dokumentiert Sicherheits-Baselines für die Azure-Plattform und veröffentlicht spezifische Anleitungen zur Nutzung von Purview-Kontrollen im Kontext von KI-Anwendungen. Der aktuelle Zertifizierungs- und Konformitätsstatus für die verschiedenen Azure-Dienste ist jederzeit im Azure Trust Center und der zugehörigen Dokumentation einsehbar. Diese Transparenz erleichtert die eigene Compliance-Bewertung und die Dokumentation für Auditoren erheblich.

Änderungssteuerung und Freigabeprozesse

Änderungen an kritischen KI-Komponenten – Prompts, Werkzeugkonfigurationen, Modellauswahlen, Indizes und Agentendefinitionen – sollten als Code verwaltet und durch formale Prozesse laufen. Verzweigungsrichtlinien im Versionskontrollsystem erzwingen Code-Reviews, bevor Änderungen in Hauptzweige übernommen werden. Evaluierungen fungieren als Qualitätstore und müssen erfolgreich durchlaufen werden, bevor Änderungen für die Bereitstellung freigegeben werden. Gestaffelte Rollouts über verschiedene Zugriffsschichten ermöglichen es, Änderungen zunächst in kontrollierten Umgebungen zu testen, bevor sie produktionswirksam werden.

Kritische Änderungen erfordern das Vier-Augen-Prinzip und einen dokumentierten Rückfallplan. Das Vier-Augen-Prinzip stellt sicher, dass keine einzelne Person kritische Änderungen ohne Überprüfung durch eine zweite Person implementieren kann. Der Rückfallplan definiert, wie eine Änderung rückgängig gemacht werden kann, falls sie unerwartete Probleme verursacht. Die RBAC-Rollen auf Projektebene unterstützen diese Prozesse, indem sie die Trennung von Entwicklungs- und Freigabeberechtigungen technisch durchsetzen. Entwickler können Änderungen vorschlagen und implementieren, benötigen aber separate Freigabeberechtigungen, um diese in die Produktion zu überführen.

Incident-Response für KI-Szenarien

Ein wirksamer Incident-Response-Plan für KI-Systeme ruht auf mehreren Säulen. Frühwarnsysteme aus den Bereichen Beobachtbarkeit und kontinuierliche Evaluierung erkennen Anomalien, bevor sie zu größeren Problemen eskalieren. Vollständige Traces über die gesamte Verarbeitungskette ermöglichen die Rekonstruktion problematischer Interaktionen. Reproduzierbare Versionen aller Systemkomponenten stellen sicher, dass vergangene Zustände wiederhergestellt werden können. Ein

klarer Rückführungspfad ermöglicht es, schnell auf eine bekannte gute Konfiguration zurückzukehren, wenn akute Probleme auftreten.

Protokolldaten werden zentral gesammelt, idealerweise in einem Log Analytics-Arbeitsbereich, der strukturierte Abfragen und Korrelationsanalysen ermöglicht. Für umfassende Sicherheitsüberwachung können die Protokolle zusätzlich an ein Security Information and Event Management-System weitergeleitet werden, das erweiterte Anomalieerkennung und automatisierte Reaktionen bietet. Bei erkanntem Fehlverhalten ermöglicht die Versionierung eine schnelle Rückführung auf eine stabile Version. Ursachenanalysen stützen sich auf die kombinierten Informationen aus Traces, Prüfprotokollen und Telemetriedaten, um nicht nur Symptome zu beheben, sondern auch die zugrundeliegenden Probleme zu verstehen und strukturell zu adressieren.

Zusammenfassung

Die Betriebsführung produktionsreifer KI-Systeme auf Azure AI Foundry ruht auf ineinandergreifenden Governance-Mechanismen. Eine strikte RBAC-basierte Berechtigungstrennung verhindert übermäßige Privilegien und erzwingt Zuständigkeitsgrenzen. Die Entra-gestützte Authentifizierung mit bedingtem Zugriff und Privileged Identity Management schafft robuste Identitätskontrolle. Private Konnektivität in Kombination mit präziser Kontrolle ausgehenden Verkehrs isoliert KI-Workloads wirksam von öffentlichen Netzwerken. Purview-basierte Datengovernance ermöglicht differenzierte Zugriffskontrolle auf Datenebene. Prüfbare und unveränderbare Protokolle schaffen die Grundlage für Nachvollziehbarkeit und Compliance.

Diese Mechanismen wirken nicht isoliert, sondern verstärken sich gegenseitig. Die Kombination aus Identitätskontrollen, Netzwerkisolation, Werkzeugbeschränkungen und Datengovernance schafft mehrere Verteidigungsschichten. Selbst wenn eine Ebene kompromittiert wird, begrenzen die verbleibenden Kontrollen den möglichen Schaden. Die vollständige Protokollierung ermöglicht es, Vorfälle zu erkennen, zu analysieren und daraus zu lernen. Formale Änderungs- und Freigabeprozesse reduzieren das Risiko unbeabsichtigter Fehlkonfigurationen. So bleiben KI-Systeme nachvollziehbar, kontrollierbar und konform – Voraussetzungen für verantwortungsvolle KI in Unternehmensumgebungen.

14. Ausblick: Die Reise durch das KI-Orchester

Diese Betrachtung hat Sie durch die vielfältigen Facetten einer modernen Enterprise-KI-Plattform geführt. Sie haben kennengelernt, wie Azure AI Foundry eine vereinheitlichte, governance-fähige Entwicklungsumgebung bereitstellt, wie die Architektur mit Foundry-Ressourcen und Projekten strukturiert ist, welche Möglichkeiten der Modellkatalog eröffnet, wie die Orchestrierung mit Agenten

und im Hub-Modell mit Prompt Flow funktioniert, und wie fundamentale Aspekte wie Netzwerkisolation, Identitätsmanagement, systematische Evaluierungen, CI/CD-Integration und Compliance-Anbindung ineinandergreifen. Das Ergebnis dieser Reise ist mehr als technisches Wissen – es ist die Fähigkeit zum professionellen Aufbau agentenbasierter Systeme, die weit über einfaches Prompting hinausgehen und den Anforderungen von Produktionsumgebungen gerecht werden.

Die Metapher des Orchesters – Koordination statt Isolation

Die Metapher des KI-Orchesters, die sich durch diese Betrachtung zieht, erfasst eine zentrale Erkenntnis: Es genügt nicht, ein einzelnes Instrument virtuos zu beherrschen. Ein Sprachmodell mag beeindruckende Fähigkeiten besitzen, aber seine wahre Wirkung entfaltet es erst im koordinierten Zusammenspiel mit anderen Komponenten. Wer KI verantwortungsvoll und skalierbar einsetzen will, muss lernen, viele Instrumente aufeinander abzustimmen.

Diese Instrumente sind vielfältig. Da sind zunächst die Modelle selbst, jedes mit eigenen Stärken und Schwächen – von spezialisierten Embedding-Modellen über effiziente Chat-Modelle bis zu leistungsstarken Reasoning-Modellen. Hinzu kommen Datenquellen und Retrieval-Mechanismen, die das kontextuelle Wissen bereitstellen. Sicherheits- und Compliance-Kontrollen bilden das Fundament verantwortungsvoller Systeme. Monitoring und Beobachtbarkeit schaffen die notwendige Transparenz im Betrieb. DevOps-Prozesse ermöglichen kontrollierte, reproduzierbare Änderungen. Human-in-the-Loop-Prozesse stellen sicher, dass kritische Entscheidungen menschliche Überprüfung erfahren. Jedes dieser Instrumente bringt eigene Komplexität und eigene Anforderungen mit sich.

Die Kunst liegt darin, aus diesen diversen Komponenten ein harmonisches Ganzes zu formen. Azure AI Foundry übernimmt dabei die Rolle des Konzerthauses, das Bühne, Akustik und Infrastruktur bereitstellt. Es liefert das Instrumentarium in Form einheitlicher SDKs, die trotz unterschiedlicher unterliegender Dienste konsistente Schnittstellen bieten. Es stellt die Notenschrift bereit durch etablierte Muster für Evaluierungen, automatisierte Telemetrie und schlüssellose Authentifizierung. Und es ermöglicht all dies, ohne dass Sie die Kontrolle aus der Hand geben müssen.

Die Trennung der Verantwortlichkeiten, die in der Architektur verankert ist, schafft einen entscheidenden Mehrwert. Die Managementebene mit ihren zentralen Richtlinien und Governance-Mechanismen stellt sicher, dass übergreifende Anforderungen an Sicherheit, Compliance und Kostenmanagement durchgesetzt werden. Gleichzeitig bewahrt die Projektebene die Flexibilität, die Entwicklungsteams für schnelle Iterationen und Experimente benötigen. Diese Balance zwischen zentraler Kontrolle und dezentraler Agilität ist keine triviale Errungenschaft – sie ist das Resultat durchdachter Architekturentscheidungen, die Governance nicht als Bremse, sondern als Ermöglicher verstehen.

Der Paradigmenwechsel für Entscheider

Für Organisationen und ihre Entscheidungsträger bedeutet der professionelle Einsatz von KI einen fundamentalen Paradigmenwechsel. KI verlässt die Experimentierecke des Innovation Labs, in der sie oft jahrelang residiert hat, und wird zum strategischen Produktionsfaktor. Dieser Übergang verändert die Anforderungen grundlegend. Was im Lab als faszinierend galt, muss nun in Produktion zuverlässig, nachvollziehbar und konform funktionieren.

Systeme werden nicht mehr ad hoc zusammengebaut, sondern kontrolliert bereitgestellt über definierte Prozesse mit Qualitätstoren. Sie werden nicht mehr nach dem Start vergessen, sondern kontinuierlich evaluiert anhand messbarer Metriken. Sie operieren nicht im Verborgenen, sondern werden systematisch überwacht mit vollständiger Beobachtbarkeit über alle Systemebenen hinweg. Und sie existieren nicht als isolierte Experimente, sondern sind fest eingebettet in eine Unternehmenskultur, die Sicherheit, Compliance und Vertrauen als nicht-verhandelbare Grundlagen versteht.

Der Weg führt vom Proof-of-Concept zur Skalierung – nicht durch bloße technische Replikation, sondern durch die Etablierung tragfähiger Strukturen. Diese Strukturen umfassen technische Aspekte wie Deployment-Pipelines und Monitoring-Infrastruktur ebenso wie organisatorische Aspekte wie Rollenmodelle, Verantwortlichkeiten und Freigabeprozesse. Der Aufbau dieser Strukturen erfordert Investitionen, aber diese Investitionen zahlen sich vielfach aus, wenn aus einem erfolgreichen Prototyp ein stabiles Produktionssystem wird, das echten geschäftlichen Wert generiert.

Der Blick nach vorn – Kommende Entwicklungen

Die Entwicklung von KI-Plattformen und insbesondere von Azure AI Foundry steht nicht still. Mehrere Entwicklungslinien zeichnen sich ab, die die kommenden Jahre prägen werden.

Die Modalitätsgrenzen verschwimmen zunehmend. Während frühe KI-Systeme strikt auf Text beschränkt waren, ermöglichen multimodale Modelle bereits heute die integrierte Verarbeitung von Text, Bildern, Audio und Video. Zukünftige Agenten werden diese Modalitäten nicht nur einzeln verarbeiten, sondern nahtlos kombinieren – ein Agent könnte beispielsweise eine gesprochene Anfrage entgegennehmen, relevante Dokumente und Bilder recherchieren, diese analysieren und eine audiovisuelle Antwort komponieren, ohne dass die verschiedenen Modalitäten explizit orchestriert werden müssen.

Die Orchestrierungsmuster werden ausgefeilter. Die aktuellen Ansätze mit fest definierten Agentenhierarchien und Werkzeuglisten werden durch dynamischere, selbstorganisierende Architekturen ergänzt. Agenten könnten zur Laufzeit entscheiden, welche anderen Agenten oder Werkzeuge sie für eine gegebene Aufgabe hinzuziehen, und temporäre Koalitionen bilden, die sich

nach Abschluss der Aufgabe wieder auflösen. Framework-Integrationen werden tiefer, sodass die Grenzen zwischen Plattform und Framework verschwimmen.

Der regulatorische Druck verstärkt sich erheblich. Der EU AI Act mit seiner Klassifizierung von KI-Systemen nach Risikostufen und den konkreten Nachweispflichten für Hochrisikosysteme setzt neue Standards. Organisationen müssen nicht nur technische Compliance demonstrieren, sondern auch organisatorische Prozesse etablieren für Risikomanagement, Qualitätssicherung, Dokumentation und Post-Market-Monitoring. Azure AI Foundry wird diese Anforderungen durch differenziertere Compliance-Funktionen adressieren müssen – von automatisierter Dokumentationsgenerierung über Audit-Trails bis zu standardisierten Reporting-Formaten.

Branchenspezifische Integrationen werden tiefer und umfassender. Im Gesundheitswesen werden KI-Systeme nicht nur mit FHIR-Schnittstellen interagieren, sondern medizinisches Domänenwissen und klinische Richtlinien direkt einbinden. Im Finanzwesen werden regulatorische Reporting-Anforderungen direkt in die KI-Pipelines integriert sein. In der Fertigung werden KI-Systeme eng mit IoT-Plattformen und Edge-Computing-Infrastruktur verzahnt sein, um Echtzeit-Optimierungen in Produktionsprozessen zu ermöglichen.

Besonders faszinierend ist die Entwicklung von KI-für-KI-Funktionen. KI-Systeme werden zunehmend in der Lage sein, sich selbst zu optimieren. Automatisierte Prompt-Optimierung experimentiert systematisch mit Variationen und wählt die effektivsten Formulierungen. AutoML-artige Modellauswahl evaluiert verschiedene Modelle für eine gegebene Aufgabe und wählt automatisch das beste aus. Selbstadaptive Agenten lernen aus ihren Interaktionen und passen ihr Verhalten kontinuierlich an. Diese Meta-KI-Funktionen versprechen eine weitere Produktivitätssteigerung, bringen aber auch neue Herausforderungen für Nachvollziehbarkeit und Kontrolle mit sich.

Die Meisterklasse im Dirigieren – Ihre erworbenen Fähigkeiten

Wenn Sie diese Betrachtung durchlaufen haben – Konzepte verstanden, Architekturmuster verinnerlicht, Implementierungsdetails nachvollzogen und vielleicht erste praktische Erfahrungen gesammelt – dann sind Sie vorbereitet für die Arbeit mit der nächsten Generation von KI-Systemen. Diese Vorbereitung besteht nicht in der Kenntnis spezifischer APIs oder aktueller Modellnamen, die sich schnell ändern werden. Sie besteht in einem tieferen Verständnis fundamentaler Prinzipien.

Sie verstehen nun, wie moderne KI-Plattformen Komplexität managen, ohne Flexibilität zu opfern. Sie erkennen, warum die Trennung von Governance und Entwicklung essentiell ist. Sie sehen, wie Sicherheit und Compliance nicht als nachträgliche Ergänzungen, sondern als integraler Bestandteil der Architektur funktionieren. Sie wissen, warum Evaluierung und Monitoring keine optionalen Add-ons

sind, sondern Voraussetzungen für produktionsreife Systeme. Und Sie haben ein Gespür dafür entwickelt, wo die Möglichkeiten enden, und die Grenzen beginnen.

Dieses Verständnis versetzt Sie in die Lage, neue Entwicklungen nüchtern zu bewerten – weder von euphorischer Überschätzung noch von ängstlicher Unterschätzung getrieben. Sie können die Chancen erkennen, die neue Modelle oder Techniken bieten, aber auch die Risiken einschätzen, die mit ihrem Einsatz verbunden sind. Sie können zwischen marketinggetriebenen Versprechungen und realen Fähigkeiten unterscheiden. Und Sie können fundierte Entscheidungen treffen über den Einsatz von KI in Ihrer Organisation.

Das Ziel dieser Betrachtung war nie die Vermittlung isolierter Fakten oder Anleitungen. Das Ziel war und ist ein ganzheitliches Verständnis davon, wie verantwortungsvolle, wertschöpfende KI-Systeme im Enterprise-Kontext konzipiert, gebaut und betrieben werden. Dieses Verständnis ist übertragbar auf andere Plattformen, anwendbar auf zukünftige Technologien und wertvoll unabhängig von spezifischen Produktentwicklungen.

Die fortgesetzte Reise – Ihr Weg vorwärts

Die Reise endet hier nicht – sie beginnt in gewissem Sinne erst richtig. Was Sie durchlaufen haben, ist die Vermittlung von Landkarten, Kompass und Rüstzeug. Aber den eigenen Weg müssen Sie selbst gehen. Dieser Weg wird individuell sein, geprägt von den spezifischen Anforderungen Ihrer Organisation, den besonderen Charakteristika Ihrer Anwendungsfälle und den einzigartigen Herausforderungen Ihrer Branche.

Für manche wird der Weg zu hochspezialisierten Anwendungen führen, die tiefes Domänenwissen mit KI-Fähigkeiten verbinden. Für andere wird er zu breit einsetzbaren Plattformen führen, die viele verschiedene Anwendungsfälle unterstützen. Manche werden zunächst interne Prozesse optimieren, andere direkt kundengerichtete Innovationen schaffen. Die konkreten Projekte werden variieren, aber die Prinzipien bleiben bestehen.

Was bleibt als Konstante, ist die Notwendigkeit der Nachvollziehbarkeit. Ihre Systeme müssen erklärbar sein – nicht unbedingt in jedem technischen Detail, aber in ihren Entscheidungsgrundlagen und Funktionsweisen. Was bleibt, ist die Anforderung der Konformität. Ihre Systeme müssen den regulatorischen Rahmen einhalten, in dem Sie operieren, und dies nachweisbar dokumentieren. Was bleibt, ist das Streben nach verantwortungsvoller KI, die menschliche Werte respektiert und menschliches Wohlergehen fördert statt gefährdet.

Azure AI Foundry bietet die technische Infrastruktur und die methodischen Grundlagen für diesen Weg. Die Plattform wird sich weiterentwickeln, neue Funktionen werden hinzukommen, bestehende werden verfeinert. Aber die fundamentalen Architekturprinzipien – die Trennung von Governance und

Entwicklung, die Betonung von Beobachtbarkeit und Evaluierung, die Integration von Sicherheit und Compliance – werden Bestand haben, weil sie auf dauerhaften Anforderungen beruhen, nicht auf flüchtigen Trends.

Ihre Aufgabe ist es nun, dieses Wissen in die Praxis zu überführen. Beginnen Sie mit einem überschaubaren, aber bedeutungsvollen Projekt. Wenden Sie die gelernten Prinzipien an. Etablieren Sie von Anfang an Strukturen für Evaluierung und Monitoring. Integrieren Sie Sicherheit und Governance nicht als nachträgliche Ergänzung, sondern als fundamentale Designprinzipien. Sammeln Sie Erfahrung, lernen Sie aus Erfolgen und Fehlschlägen, und bauen Sie kontinuierlich Ihre Expertise aus.

Die Landkarte haben Sie studiert. Der Kompass ist justiert. Das Rüstzeug liegt bereit. Jetzt liegt es an Ihnen, Ihren eigenen, nachvollziehbaren und konformen Weg in die Produktions-KI zu beschreiten – mit der Zuversicht, die aus fundiertem Verständnis erwächst, und mit der Umsicht, die aus Bewusstsein für Möglichkeiten und Grenzen resultiert. Die Reise geht weiter, und sie verspricht spannend zu werden.