E-Book

Red Hat

Agile Integration:

Agile Integration: Ein Blueprint für die Unternehmensarchitektur

Inhaltsverzeichnis

facebook.com/redhatinc
@RedHatDACH
linkedin.com/company/red-hat

> *„Die Modernisierung der IT ist ein enorm wichtiger und oft missverstandener Bestandteil der digitalen Transformation, weil sie Unternehmen ermöglicht, Innovationen zu beschleunigen und die Leistung zu steigern. Zu den weiteren wichtigen Transformationsaufgaben gehört die Schaffung einer digitalen Kultur."*[1]
>
> **Michael Bender und Paul Wilmot**
> Digital McKinsey

Die Rolle der Integration in der Innovation

Geschäftlicher Erfolg hängt zunehmend davon ab, ob und wie Unternehmen auf Änderungen reagieren können. Die Zahl der auf den Markt drängenden disruptiven Player sowie neuer Technologien, die Verbrauchererwartungen auf den Kopf stellen, steigt ständig an. Organisationen müssen in der Lage sein, sich flexibel an solche Änderungen anpassen zu können – und das in immer kürzeren Zeitabständen. Moderne Softwarearchitekturen und -prozesse können dafür die so dringend benötigte Effizienz bieten und den Erfolg von Unternehmen auf dem Markt sichern.

Technologie transformiert ganze Branchen. Heute haben die meisten Unternehmen E-Commerce-Services im Angebot und interagieren mit Verbraucher regelmäßig über digitale Medien. Disruptive Veränderungen zwingen Organisationen dazu, ihre IT-Umgebungen radikal zu transformieren, damit sie die vom Kunden geforderten neuen digitalen Services – besser und schneller als die Konkurrenz – bereitstellen können.

Eine neue Art und Weise, Anwendungen und Services im gesamten Unternehmen zu verbinden und in jeder Hinsicht eine fundamentale Lösung ist die agile Integration. Sie vereint drei leistungsstarke Architekturfunktionen – verteilte Integration, Application Programming Interfaces (APIs) und Container – zur Förderung von Agilität und Implementierung neuer Prozesse, all das mit dem Ziel der Bereitstellung von Wettbewerbsvorteilen.

Branchen wie Reisen und Gastgewerbe wurden durch neue Geschäftsstrategien komplett transformiert. So werden unter anderem neue Services angeboten, mit denen Verbraucher auf unterschiedliche Art und Weise interagieren. Dieser Trend hin zur kontinuierlichen Disruption überträgt sich auch auf andere wichtige Geschäftszweige, von Finanzdienstleistungen bis hin zu Behörden. Angetrieben werden sie von neuen Technologien und Ansätzen in Bezug auf die Interaktion zwischen Unternehmen und Verbrauchern. Die durch die Bereitstellung dieser neuen Services entstehenden Herausforderungen sind der Grund dafür, dass Unternehmen ihre IT-Technologien radikal transformieren müssen.

Integration ist für das digitale Geschäft von größter Wichtigkeit

Die Bereitstellung herausragender Kundenerfahrungen ist für Unternehmen nicht mehr länger nur eine Frage der Differenzierung, sondern des wirtschaftlichen Überlebens. Individuelle Erfahrungen und Interaktionen sind Bausteine der Kundenbindung, ein holistisches Kundenerlebnis der Zement, der diese Bausteine zusammenhält.

Bei Hotels zum Beispiel ist der Aufenthalt selbst ebenso wichtig wie die Buchung, sich über WiFi-Probleme unterhaltende Gäste, das Treueprogramm usw. Kunden hatten nie zuvor höhere Erwartungen an die digitale Wirtschaft sowie an personalisierte und kontextrelevante Interaktionen. Letztendlich ist die Beziehung zu diesen Kunden nur so stark wie das schwächste Glied in der Kette der Kundenerfahrungen.

Um Verbrauchererwartungen von heute erfüllen zu können, müssen Unternehmen eine nahtlose gemeinsame Nutzung von Daten für alle Apps gewährleisten. Dies wiederum erfordert die Integration zahlreicher unterschiedlicher Anwendungen, mit denen der Kunden während seiner digitalen Erfahrungen interagiert. Mit einer erfolgreichen Integrationsstrategie können Unternehmen multidimensionale Kundeneinblicke generieren, Kundenerwartungen antizipieren und die Abwanderungsrate gering halten.

Das perfekte Beispiel hierfür ist Uber. Dieses Unternehmen wird oft als Beispiel für digitale Disruption genannt, der Einfluss der Integration auf seinen Erfolg aber zumeist ignoriert. Taxiunternehmen hatten in den vergangenen 20 Jahren Zugriff auf die gleichen Daten, waren bis dato aber nicht in der Lage, Anwendungen zu verbinden und Informationen auf eine Weise zu nutzen, mit der sich Prognosen erstellen und Kundenerwartungen ändern lassen. Agile Integration schließt diese Lücke, indem sie unbegrenzte Möglichkeiten für Innovationen in der modernen digitalen Welt eröffnet und die Art und Weise des Wettbewerbs zwischen Unternehmen transformiert.

---

1 *Michael Bender und Paul Wilmott, Digital McKinsey, „Digital reinvention: Unlocking the ‚how'", Januar 2018,* https://www.mckinsey.com/~/media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20Insights/Digital%20Reinvention%20Unlocking%20the%20how/Digital-Reinvention_Unlocking-the-how.ashx

„Wenn Sie Ihren Wettbewerbern nicht kontinuierlich in Sachen Innovationen, Markteinführung und Agilität voraus sind, haben Sie keine Chance. Features präsentieren immer ein gewisses Risiko. Wenn Sie Glück haben, bringen zehn Prozent von ihnen den gewünschten Erfolg. Aus diesem Grund müssen Sie sie so schnell wie möglich auf den Markt bringen und testen. Übrigens amortisiert sich dadurch auch das investierte Kapital schneller und die Gewinne stellen sich zügiger ein."[2]

Gene Kim
The Phoenix Project

Die aktuell progressivsten Marktführer verbinden einfach alles mit allem, denn sie wissen, dass Integration für ihre digitale Transformation und damit auch ihren Erfolg unerlässlich ist.

Agile Integration für mehr Flexibilität und Geschwindigkeit

Geschwindigkeit ist eines der wichtigsten Kriterien der digitalen Welt. Um auf dem Markt relevant zu bleiben, müssen Unternehmen Änderungen an Softwaresystemen zeitnah planen und ausführen können. Deshalb sind diejenigen Organisationen, die über Nacht Preise ändern und neue Produkte anbieten können, denen weit voraus, die geplante Rollouts über drei Monate mit manuellen Verifizierungsschritten einsetzen.

Um aber Software in der von der modernen Geschäftswelt geforderten Geschwindigkeit bereitstellen zu können, wird eine agile Infrastrukturgrundlage benötigt. In diesem Fall bezieht sich „agil" aber nicht auf den in „agile Softwareentwicklung" verwendeten Ausdruck, sondern auf seine traditionelle englische Bedeutung, also flexibel, anpassbar oder flink.[3]

Bis dato haben agile Methodologien auf der Softwareentwicklung und der Verbesserung und Rationalisierung der Entwicklung von Anwendungen fokussiert. Mit DevOps[4]-Praktiken soll diese Methodologie jetzt auch in deren Bereitstellung integriert werden. DevOps ist allerdings nicht allumfassend, sondern hauptsächlich auf von den Organisationen selbst entwickelte neue Softwareanwendungen ausgelegt.

Infrastrukturagilität geht da viel weiter, weil sie die Erstellung einer Umgebung ermöglicht, die alle IT-Systeme vereint, darunter auch Legacy-Software. Mit einer agilen Infrastruktur soll die Komplexität bestehender Systeme, unterschiedlicher Datentypen und -ströme sowie von Kundenerwartungen in einer Methodologie vereint werden.

Red Hat nennt diesen Prozess „agile Integration". Der Begriff Integration meint hier keine Untergruppe von Infrastrukturen, sondern einen konzeptionellen Ansatz, der Daten und Anwendungen mit Hardware und Plattformen integriert. Durch die Ausrichtung von Integrationstechnologien mit agilen und DevOps-Technologien können Organisationen Plattformen schaffen, mithilfe derer Entwicklungs-Teams flexibel auf Marktänderungen reagieren können.

Die Planung hat ausgedient: Organisationen und Agilität

Jim Whitehurst, Red Hat CEO, sagte einmal: „Die Planung, wie wir sie kennen, gibt es nicht mehr ... Denn in einer praktisch unbekannten Umgebung kann sie keine Effizienz bieten."[5] Mit der zunehmenden Beschleunigung von Geschäftsumgebungen sind Pläne schnell überholt und eine Festlegung auf bestimmte Strategien kann extrem teuer werden.

Das bedeutet, je weniger Informationen ein Unternehmen besitzt oder je instabiler seine Umgebung ist, desto geringer die Wertigkeit seiner Pläne.

Sie wissen nicht, was Sie nicht wissen

Die Infrastrukturplanung ist üblicherweise eine langfristige Angelegenheit und nimmt zuweilen Jahre in Anspruch. Ein Mehrjahresplan wiederum kann die Fähigkeit zur Innovation oder ein flexibles Reagieren auf Marktänderungen stark beeinträchtigen. Agilität ist die Fähigkeit, Pläne schneller erstellen und ausführen zu können. In einer solchen Umgebung haben Planungen eine kürzere Lebenserwartung, weil kontinuierlich neue Konzepte kultiviert werden.

---

2 *Gene Kim, Kevin Behr und George Spafford, „The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win", Portland, Oregon: IT Revolution Press, 2013.*

3 *Oxford English Dictionary*

4 *„Understanding DevOps",* https://www.redhat.com/de/topics/devops

5 *Rede von Jim Whitehurst auf dem Red Hat Summit 2017.* https://www.youtube.com/watch?v=8MCbJmZQM9c

Solche kurzfristigen Änderungen sind speziell für diejenigen Teams eine Herausforderung, die an Entwicklungszyklen von 6 oder gar 24 Monaten gewöhnt sind. Die Problematik verschärft sich insbesondere für eher traditionell strukturierte Unternehmen, die sich mit Startups und ihren völlig neuen Marktstrategien auseinandersetzen müssen. Offensichtliche Beispiele hierfür wären Netflix und Blockbuster oder Uber und traditionelle Taxidienste. Der disruptive Effekt solcher Startups aber geht sogar bis zu den Anfängen des Informationszeitalters zurück, wie z. B. Amazon im Jahre 1993 oder Personal Computer in den 1980ern.

Tabelle 1: Disruptoren in allen Branchen

Branche	Traditioneller Service	Disruptor	Auswirkungen
Transportwesen	Taxis, öffentliche Verkehrsmittel	Uber, Lyft	Schaffung einer einheitlichen Kundenerfahrung, die kleine lokale Firmen unmöglich replizieren können
Wealth Management	Investmentfirmen	Automatische Fonds	Verschiebung der Unterscheidungsmerkmale für das Fondsmanagement vom Menschen zum Algorithmus
Einzelhandel	Physisches Einkaufen	Amazon	Änderung des Einkaufverhaltens vom Offline- zum Online-Shopping
Suchmaschinen	Google, browser-basierte Suche	Sprachsuche	Einfluss auf den primären Absatzkanal von Google sowie Aufkommen von Neueinsteigern

Der Vorteil von Startups und Disruptoren ist, dass sie Infrastruktur, Teams, Anwendungen, Architektur und selbst Bereitstellungsprozesse auf neuartige Weise flexibel strukturieren können. Es geht hier um mehr als nur innovative Ideen. Die wahre Stärke dieser Player ist die Fähigkeit zu deren Umsetzung, denn sie werden nicht von Legacy-Infrastrukturen oder, wie Rachel Laycock sie scherzend genannt hat, „Legacy-Menschen", ausgebremst.[6] Sie sind ganz einfach nur agil.

Diese Organisationen sind in der Lage, neben neuen Anwendungen auch Systeme zu entwickeln, die auf die flexible Anpassung an Änderungen ausgelegt sind. Ein wichtiges Unterscheidungsmerkmal ist dabei die Softwareinfrastruktur, denn praktisch alle Komponenten des Systems können je nach Marktbedürfnissen ausgewechselt, aktualisiert oder entfernt werden. Mit zunehmendem Alter mancher Startups leidet auch ihre Fähigkeit zur Anpassung. Die besten von ihnen aber stellen sicher, dass genau diese Fähigkeit um jeden Preis geschützt wird.

6 *Rachel Laycock, „Continuous Delivery", Nachmittagssitzung, Red Hat Summit - DevNation 2016. 1. Juli 2016, San Francisco, Kalifornien. https://youtube.com/watch?v=y87SUSOfgTY*

Schaffung einer Basis für Agilität

Um in sich schnell verändernden Umgebungen erfolgreich zu sein, muss die gesamte IT-Infrastruktur agil funktionieren können.

Die notwendigen Anpassungen erfolgen auf zwei Ebenen:

1. Organisatorischer und kultureller Support agiler Prozesse, vom Architekturdesign bis hin zur Teamkommunikation
2. Technische Infrastruktur mit der Möglichkeit für Nutzer, Funktionen umgehend aktualisieren, hinzufügen und entfernen zu können

Technische und kulturelle Änderungen alleine sorgen noch nicht für Agilität. Sie bilden lediglich die Basis dafür.

Marty Cagan, Produkt-Manager bei eBay, rechnet in sogenannten Projekt-„Steuern", also in Routineprojekte einkalkulierte Arbeitszeit und Ressourcen, die für neue Infrastrukturprojekte aufgewendet werden.[7] Dieser Ansatz räumt neuen Projekten und Innovationen eine hohe Priorität ein.

Die Infrastruktur der Agilität

Selbst eine Flut neuer Technologien führt in den seltensten Fällen zur Erstellung einer agilen Infrastruktur, da sich verschiedene Gruppen bei der Entwicklung von Verbesserungen in unterschiedliche Richtungen bewegen. Ohne einen kohärenten Satz an Top-Strategiezielen ist es praktisch unmöglich zu bestimmen, welche Features die Gesamtfunktion des Unternehmens am stärksten beeinflussen.

Die drei Grundpfeiler der agilen Integration

Der Ansatz der agilen Integration basiert auf drei grundlegenden Technologien:

1. **Verteilte Integration:** Einige Dutzend hochrangige Integrationsmuster spiegeln Arbeits- und Datenabläufe eines Unternehmens wider. In Containern lassen sich diese Muster für spezifische Anwendungen und Teams in beliebiger Skalierung und an beliebigen Orten implementieren. Dieser Ansatz repräsentiert keine traditionelle zentralisierte, sondern eine verteilte Integrationsarchitektur, die individuellen Teams die Definition und Implementierung der für eine optimale Agilität benötigten Integrationsmuster ermöglicht.
2. **APIs:** Stabile und effizient verwaltete APIs bieten enorme Vorteile für die Zusammenarbeit zwischen Teams, Entwicklung und Betrieb. APIs stellen wichtige Vermögenswerte als stabile, wiederverwendbare Schnittstellen bereit, die wiederum als Bausteine im gesamten Unternehmen sowie gemeinsam mit Partnern und Kunden genutzt werden können. Sie lassen sich zudem zusammen mit Containern in unterschiedlichen Umgebungen implementieren und ermöglichen Nutzern die Interaktion mit unterschiedlichen API-Sätzen.
3. **Container:** Sowohl für APIs als auch verteilte Integrationstechnologien dienen Container als zugrundeliegende Bereitstellungsplattform. Sie ermöglichen die Verfügbarmachung exakter Services in spezifischen Umgebungen, und das auf eine Art und Weise, die eine einfache Entwicklung, Prüfung und Wartung unterstützt. Und da sie die bevorzugte Plattform für DevOps-Umgebungen und Microservices sind, sorgt ihr Einsatz als Integrationsplattform für mehr Transparenz bei und eine bessere Zusammenarbeit zwischen Entwicklungs- und Infrastruktur-Teams.

---

7 *Cagan, Marty, „Inspired: How to Create Products Customers Love", Wiley Press, 2017.*

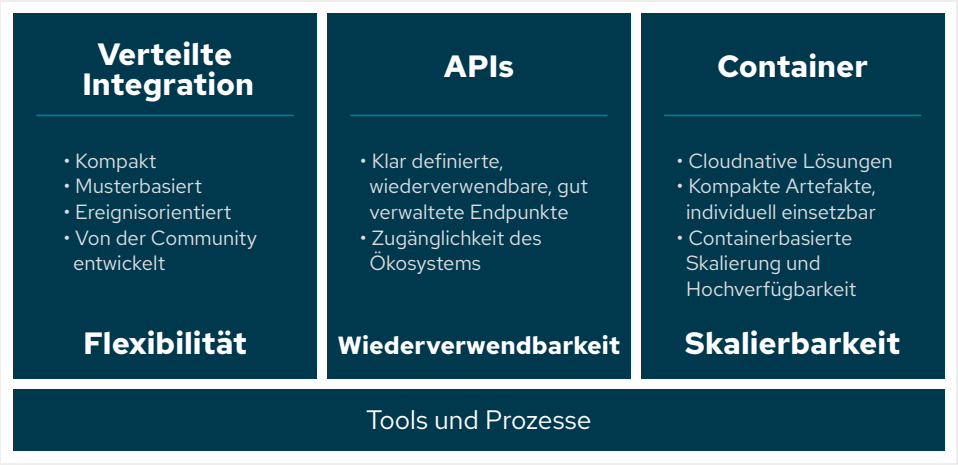


Abb. 1: Die drei Grundpfeiler der agilen Integration1

Die drei Grundpfeiler der agilen Integration machen die IT-Infrastruktur agiler, da sie das Abstraktionsniveau erhöhen, auf dem verschiedene Teams zusammenarbeiten können. Durch den Einsatz einer Containerplattform mit APIs und verteilten Integrationen lässt sich die Implementierung der Integration von der Integration selbst abstrahieren. Teams gewinnen an Agilität, weil APIs und verteilte Integrationsmuster spezifische Vermögenswerte auf einer Ebene paketieren, die von der Mehrheit der Beteiligten verstanden wird – und das, ohne dass genaue Informationen zur zugrundeliegenden Infrastruktur oder Änderungen daran benötigt werden.

Diese Technologien bieten individuell bereits eine enorme Agilität für spezifische Integrationsherausforderungen. Man stelle sich nur einmal vor, welches Potenzial sie als Kombination bereithalten. Die Unternehmenskultur unterstützt die Technologie, deren Vorteile wiederum durch die Kombination mit DevOps-Praktiken, speziell Automatisierungs- und Bereitstellungsprozesse, weiter verstärkt werden.

Grundpfeiler Nr. 1: Verteilte Integration

Eine der größten Herausforderungen traditioneller IT-Systeme ist die Notwendigkeit der Verknüpfung von Anwendungen im gesamten Unternehmen und darüber hinaus. In der Vergangenheit mündete genau dieser Umstand in immer komplexeren und zentralisierteren Integrations-Hubs. Diese oft als ESBs (Enterprise Service Buses) implementierten Hubs haben sich zu starren und komplexen Engpässen entwickelt, die für schnelle Änderungen einfach nicht geeignet sind.

Grund ist, dass sie die obligatorische Nutzung aller ESB-Tools über ihre gesamte Lebensdauer voraussetzen, und zwar zusätzlich zu den Tools für Entwicklungs- und Betriebsumgebungen. Und genau diese Einschränkung führt nicht selten zu umständlichen, ineffizienten und fehleranfälligen Betriebsprozessen.

Mit der verteilten Integration lassen sich die gleichen technischen Ziele früherer ESB-Generationen umsetzen, aber auf eine Weise, die mehr Flexibilität für die Teams einer Organisation bietet. Auch dieser Ansatz integriert Transformation, Routing, Parsing, Fehlerbehandlung und Alarmfunktionen.

„Wenn man Software mit einem Körperteil vergleicht, der schmerzt, dann ist die einzige Möglichkeit, diese Schmerzen zu lindern, Vorgänge öfter und nicht weniger oft auszuführen."[8]

David Farley
Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation

Der Unterschied liegt in der Architektur der Integration. Bei der verteilten Variante wird jeder Integrationspunkt als separate und individuelle Bereitstellung und nicht als Bestandteil einer großen zentralen Anwendung angesehen. Die Integration kann dann containerisiert und lokal für spezifische Projekte oder Teams bereitgestellt werden, ohne andere in der Organisation implementierte Integrationen zu beeinträchtigen. Hier wird die Integration praktisch als Microservice behandelt[9], wodurch eine Beschleunigung der Entwicklung sowie die Unterstützung von Rapid Release-Zyklen erzielt wird.

Dieser verteilte Ansatz bietet maximale Flexibilität. Dazu verwendet er die gleiche Toolchain wie die agilen oder DevOps-Teams. Das heißt, er nutzt die zugrundeliegende Containerplattform, um Teams in die Lage zu versetzen, ihre eigenen Integrationen mit eigenen Tools und Zeitplänen zu verwalten.

Die Ausrichtung von Entwickler-Tools und -prozessen ist unerlässlich. Die verteilte Integration ist keine zentrale Softwareinfrastruktur, die mithilfe einer speziellen Nutzergruppe einer Abteilung entwickelt und verwaltet und abseits des Softwareentwicklungsprozesses implementiert wird. Die Verteilung der Integrationsarchitektur mit einer gemeinsamen Plattform bzw. gemeinsamen Tools gewährt allen Entwicklern Zugriff auf Projektebene und unterstützt kompakte Bereitstellungen, und zwar egal wo oder wann Integration benötigt wird.

Tabelle 2. Ein Vergleich der Integrationstechnologien in jeder Phase des Software-Lifecycles

Lifecycle-Schritt	ESBs, die meisten iPaaS (Integration-Platform-as-a-Service)	Unterstützung verteilter Integrationstechnologien
Versionskontrolle	Proprietär	Github und andere
Build	Proprietär	Maven und andere
Bereitstellung	Proprietär	Container und andere DevOps-Tools
Verwaltung und Skalierung	Proprietär	Container und andere DevOps-Tools

Streaming über eine verteilte Backplane

Agile Integration bietet die Freiheit, je nach den Anforderungen der Anwendung entweder die synchrone oder asynchrone Integration zu verwenden. Bei der verteilten Integration wird häufig die synchrone Methode eingesetzt, und zwar per Bereitstellung von Containern zur gemeinsamen Nutzung von Daten unterschiedlicher Nutzer (wie oben erwähnt). Die zweite Verteiloption ist das Streaming, eine asynchrone Methode, die es agilen Entwicklern ermöglicht, bei Bedarf Events von anderen Quellen zu empfangen, neu zu lesen oder abzuspielen. Messaging wird zur Replizierung von Daten in einem Zwischenspeicher eingesetzt, damit die Informationen von mehreren Teams und ihren Anwendungen geteilt werden können. Dieser Zwischenspeicher ist vor allem für Microservice-Teams von Vorteil, weil sie so nicht gezwungen sind, Daten kontinuierlich synchron von anderen Quellen abzufragen.

8 *David Farley und Jez Humble, „Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment Automation", Addison-Wesley Professional, 2010.*

9 *Siehe dazu die äußerst hilfreiche Definition von Martin Fowler zum Thema Microservices:* https://martinfowler.com/articles/microservices.html

Mit einer verteilten Streaming-Plattform lassen sich Datenströme in Echtzeit veröffentlichen, abonnieren, speichern und verarbeiten. Sie ist darauf ausgelegt, Informationen aus mehreren Quellen zu handhaben und sie an verschiedene Kunden weiterzuleiten. Streaming-Plattformen können pro Sekunde Millionen von Datenpunkten handhaben. Dies wiederum erweist sich speziell für Use Cases wie IT-Ops und E-Commerce als nützlich, die Echtzeitverfügbarkeit benötigen. In der nachfolgenden Abbildung wird die organisatorische Struktur der meisten Use Cases für Streaming dargestellt. Alle Anwendungen produzieren im Rahmen ihrer Ausführung Änderungen und propagieren diese als Events an einen oder mehrere Streaming-Prozessoren. Diese Prozessoren wiederum führen eine Mustererkennung oder Transformation durch und gleichen dann die aktuelle Situation mit historischen Einblicken ab, wie sie oft von ihren Analysesystemen bereitgestellt werden.

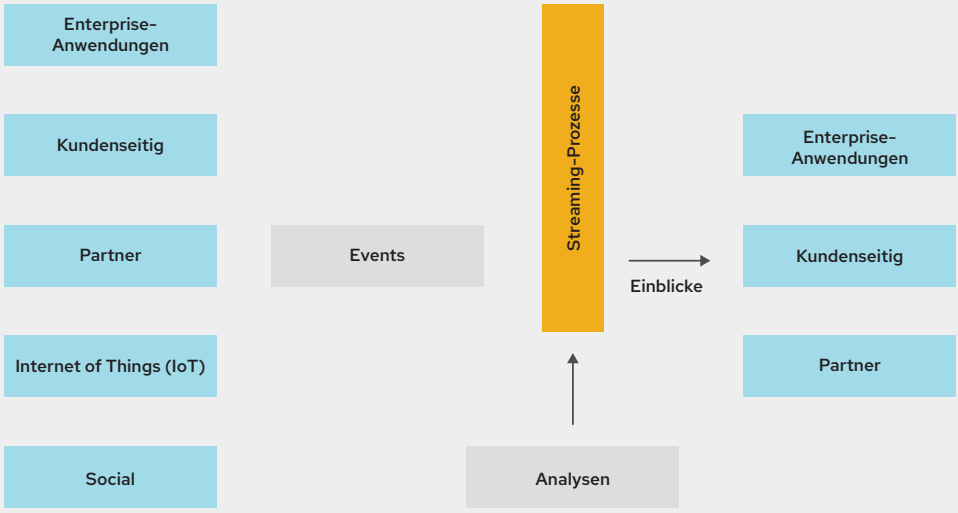


Abb. 2. Event Flow der Verarbeitungsplattform zur Generierung von Einblicken

Die asynchrone Integration bietet für manche Use Cases größere Vorteile als die synchrone Variante. Letztere erfordert für eine erfolgreiche Kommunikation einen einsatzbereiten Empfänger, während die messaging-basierte asynchrone Integration ein Versenden von Nachrichten auch bei nicht durchgängig verfügbaren Empfängern erlaubt. Bei der synchronen Kommunikation sind Timeouts gang und gäbe, weil die Ausführung der Anwendung von einer Antwort abhängig ist. Beim asynchronen Messaging gibt es solche Verzögerungen nicht. Die Verarbeitung wird unabhängig davon fortgesetzt, ob eine Antwort eingegangen ist oder nicht. Dank dieser Flexibilität eignet sich die asynchrone Integration, wie z. B. Streaming, ideal für große Datenvolumina. Dazu gewährleistet sie Hochverfügbarkeit und Zuverlässigkeit, weil die Wahrscheinlichkeit für Timeouts nur sehr gering ist.

Die Nutzung von Varianten der ereignisgesteuerten Integration, wie Streaming, zur Ergänzung der synchronen Integration und von APIs sorgt für eine wesentlich verbesserte agile Integration. Die Kombination aus Integration und Messaging steigert die allgemeine Performance der Integrationsarchitektur, und zwar durch ein effizienteres Routing, Support für mehrere Sprachen und Protokolle, hohen Durchsatz und eine herausragende Datenverwaltung.

Event Mesh

Das Event Mesh stellt eine neue Option der asynchronen verteilten Integration dar. Es handelt sich hier um eine konfigurierbare und dynamische Infrastrukturschicht zur Verteilung von Ereignissen auf entkoppelte Anwendungen, Geräte und Clouds.

Es besteht aus einem Netzwerk an miteinander verbundenen Event Brokern und dient zur Handhabung asynchroner, ereignisbasierter Interaktionen. Dazu ist es umgebungsagnostisch, das heißt, Ereignisse von einer Anwendung können ohne jegliche Konfiguration des Event Routings an eine andere weitergeleitet und von dieser empfangen werden – und zwar unabhängig von ihrem Standort – lokal, Private, Public oder Hybrid Cloud, Platform-as-a-Service (PaaS) oder gar das Internet of Things (IoT).

Dazu ist ein Event Mesh in der Lage, alles miteinander zu verbinden, von Microservices und cloudnativen Services bis hin zu Legacy-Anwendungen, Geräten und Datenbanken – mit einem Wort: beliebige Event-Produzenten mit beliebigen Event-Verbrauchern. Dazu ist es höchst effizient und bestimmt stets den schnellsten Weg vom Produzenten zum Verbraucher, um eine Echtzeit-Interaktion zu ermöglichen.

Angesichts des steigenden Verteilfaktors von IT-Umgebungen stellt das Event Mesh eine moderne Lösung dar, mit der sich Kommunikationen flexibel, zuverlässig, schnell und zudem sicherer handhaben lassen.

Asynchrone Interaktionen und ereignisgesteuerte Architekturmuster sind gestandene Technologien, das Event Mesh aber ein bahnbrechender neuer Ansatz, der im Bereich der Integration noch nicht allzu lange präsent ist. Und auch wenn es aktuell noch nicht überall verwendet wird, geht Red Hat davon aus, dass es in den kommenden Jahren deutlich an Beliebtheit gewinnen wird, weil Organisationen im Zuge ihrer digitalen Transformation vermehrt auf die Flexibilität der ereignisgesteuerten Integration setzen müssen.

Ein Event Mesh darf allerdings nicht mit einem Service Mesh verwechselt werden. Das eine unterstützt die asynchrone und das andere die synchrone Integration basierend auf APIs.

Aus der Perspektive der Architektur haben Integrationen in einer verteilten Integration den gleichen Stellenwert wie Microservices. Man kann sie containerisieren und einfach lokal bereitstellen. Dazu bieten sie kurze Release-Zyklen.

Integrationstechnologie muss in der Lage sein, eine kompakte microservice-basierte Architektur zu unterstützen. Mit Red Hat® Fuse lassen sich Integrationen als Code handhaben, der überall ausgeführt werden kann, auch in Containern.

Dazu stellt das im Bundle mit Red Hat AMQ bereitgestellte Produkt eine robuste Messaging-Struktur bereit, die wiederum ein effizientes Routing von Events und Daten zwischen Systemen gewährleistet. Messaging ist bei der Arbeit mit Microservices hilfreich, da es aufgrund seiner asynchronen Natur keine Dependenzen erfordert.

Red Hat AMQ stellt Apache Kafka bereit, eine verteilte Streaming-Plattform, und zwar via AMQ Streams über Red Hat OpenShift® Container Platform, die unternehmensfähige Kubernetes-Plattform, der fast die Hälfte aller Fortune 100 Unternehmen vertrauen.[10] Bei AMQ Streams handelt es sich um eine massiv skalierbare und verteilte hochleistungsfähige Daten-Streaming-Funktion, die auf dem Apache Kafka Projekt basiert. Diese Kombination ermöglicht den Datenaustausch zwischen Microservices und anderen Anwendungen, und zwar mit hohem Durchsatz und niedriger Latenz.

Red Hat bietet dazu ein Event Mesh über sein globales Peer-to-Peer Event-Bereitstellungssystem AMQ interconnect an. AMQ interconnect gewährleistet als Manager für verteilten Datenverkehr eine Weiterleitung vorbei an Engpässen und Fehlerzonen und damit eine einfache und zuverlässige Übertragung an den Verbraucher sowie Robustheit gegen Knoten- und Cloud-Probleme.

Grundpfeiler Nr. 2: APIs (Application Programming Interfaces)

Die meisten Informationsinfrastrukturen enthalten Hunderte oder gar Tausende von Systemen, Anwendungen und Vermögenswerten, weshalb sich die Interaktion dieser Komponenten als schwierig erweisen kann. Es kann sogar vorkommen, dass IT-Administratoren nicht wissen, welche Systeme tatsächlich verfügbar sind. Mit APIs lässt sich diese Herausforderung bewältigen, weil sie als Schnittstellen für alle Vermögenswerte dienen, die mithilfe der Integrationstechnologie verbunden werden können.

Da Organisationen immer mehr von der zentralisierten Integration hin zu einem verteilten Ansatz tendieren, kommt der Self-Service-Funktion eine entscheidende Bedeutung zu. Agile Teams müssen über die Befugnis und Freiheit verfügen, inner- und außerhalb des Unternehmens entwickelte Services suchen, testen und nutzen zu können. Dies lässt sich mit einer robusten API-Funktion bewerkstelligen. So bekommen die Teams die Integrationsflexibilität, die sie benötigen, während die Organisation gewährleisten kann, dass Sicherheits-, Berechtigungs- und Nutzungsrichtlinien verwaltet und durchgesetzt werden.

Mit APIs lässt sich ein Satz an Definitionen oder Regeln bereitstellen, mit dem die Art und Weise der Kommunikation von Anwendungen untereinander festgelegt wird. So erhalten Entwickler eine gemeinsame Sprache für die Integration und gleichzeitig eine Referenz für deren Design.

10 *Red Hat Pressemitteilung, „More than 1,000 enterprises across the globe adopt Red Hat OpenShift Container Platform to power business applications", 8. Mai 2019,* https://www.redhat.com/de/about/press-releases/more-1000-enterprises-across-globe-adopt-red-hat-openshift-container-platform-power-business-applications?extIdCarryOver=true&sc_cid=701f2000001OH74AAG

Entwickler nutzen APIs als Bausteine für ihre Projekte. Allerdings sind APIs auch für eine gemeinsame Verwendung gedacht. Unterschiedliche APIs oder API-Subsets lassen sich verschiedenen Zielgruppen nach Bedarf zur Verfügung stellen, auch wenn diese komplett unterschiedliche Anforderungen haben, wie zum Beispiel Anbieter auf der einen und interne Entwickler-Teams oder Community-Entwickler auf der anderen Seite.

Um APIs erfolgreich nutzen zu können, muss die betreffende Organisation über Funktionen für deren Management verfügen. Dazu gehören die Konzipierung der API für die Anwendung bzw. die Nutzergruppe sowie die Verwaltung des API-Lifecycles. APIs werden zunehmend als Produkte verwaltet, für die unterschiedliche Teams verantwortlich sind. Allerdings müssen auch Einheitlichkeit und Benutzerfreundlichkeit all dieser Ressourcen gewährleistet werden.

Service Mesh

Ein Service Mesh geht bei der API-Integration noch einen Schritt weiter, indem es eine konfigurierbare Infrastrukturschicht für Microservices bereitstellt und die Kommunikation so flexibler, zuverlässiger, schneller und verwaltbarer macht.

Bei der ersten Nutzung von Microservices wird die Kommunikation zwischen Services von der integrierten Governance-Funktion mit geringstmöglicher Beeinträchtigung der Operationen gehandhabt. Da allerdings kontinuierlich neue Anwendungen, Services und Features in Form von Microservices hinzugefügt werden, wird die zunehmende Komplexität der Architektur irgendwann unweigerlich zum Problem. Zum einen bringt jeder neue Service ein gewisses Fehlerpotenzial mit sich und zum anderen kann es öfter zu einer Überlastung von Services mit Anfragen kommen. So können Hunderte oder Tausende Microservices, die kontinuierlich versuchen, sich miteinander zu verbinden, Kommunikationslatenzen und Anwendungsausfälle verursachen. Dazu wird es in einer solch komplexen Microservice-Architektur immer schwieriger, die Ursache von Problemen zu finden. Und genau dafür wurde das Service Mesh entwickelt.

Es handelt sich hier um eine in Apps integrierte dedizierte Infrastrukturschicht, die die Logik zur Verwaltung der Kommunikation zwischen Services aus den einzelnen Services entfernt und sie in einer Infrastrukturschicht abstrahiert. „Neben" Microservices angeordnete Sidecar Proxies, die zusammen ein Mesh-Netzwerk bilden, leiten Anfragen an andere Proxies weiter. So gelangen mithilfe des Service Mesh Anfragen von einem Service zum nächsten, wodurch das Zusammenspiel aller Microservices weiter optimiert wird.

Zudem erlaubt es Nutzern, Funktionen zu Microservices hinzuzufügen, darunter Routing, Fehlertoleranz und Sicherheit sowie Transparenz, Überwachung und Prüfung, ohne dass dabei irgendwelche Änderungen an deren Komponenten vorgenommen werden. Diese Funktion erfüllen die Sidecar Proxies, die Informationen und Funktionalität in den jeweiligen Microservice injizieren.

Das Service Mesh ermöglicht eine einfache Behebung von Kommunikationsproblemen, weil deren Ursache nicht im Microservice versteckt ist, sondern in einer Infrastrukturschicht „neben" den Services klar ersichtlich ist.

Dazu verleiht das Service Mesh Anwendungen mehr Robustheit in Bezug auf Ausfallzeiten, weil es in der Lage ist, Anfragen von ausgefallenen Services umzuleiten.

Des Weiteren protokolliert es Leistungsmetriken, um in solchen Situationen bessere Diagnosefunktionen bereitzustellen und Ausfallzeiten zu minimieren. Die gleichen Leistungsdaten können auch von Entwicklern zur Optimierung von Designs für zukünftige Releases verwendet werden.

Beim Service Mesh müssen Microservices nicht mit einer Logik für die Interservice-Kommunikation programmiert werden, was den Entwickler nur Zeit kostet und seinen Fokus beeinträchtigt. Auf diese Weise wird der Entwicklungsprozess weiter optimiert. Red Hat geht davon aus, dass sich die Beliebtheit dieser Technologie in den nächsten Jahren deutlich steigern wird.

Die wahre Leistungsfähigkeit von APIs besteht darin, dass sie von Dritten verwendet werden können, und zwar sowohl internen Entwicklern als auch externen Nutzern. Red Hat 3scale API Management bietet nützliche Tools für alle. Die Lösung dient als Entwicklerportal zur Förderung der Zusammenarbeit in Sachen API-Erstellung sowie als Admin-Portal zur Veröffentlichung von APIs. Dazu sorgt sie für die externe Marktfähigkeit von APIs, indem sie Authentifizierung, Integration mit wichtigen Cloud-Anbietern sowie die Ausführung in Containern ermöglicht.

API-Strategien vereinigen API-Designs mit Methodologien zur API-Veröffentlichung. 3scale API Management, speziell das auf einer Container-Plattform aufsetzende 3scale, stellt die Mittel zu ihrer Umsetzung bereit.

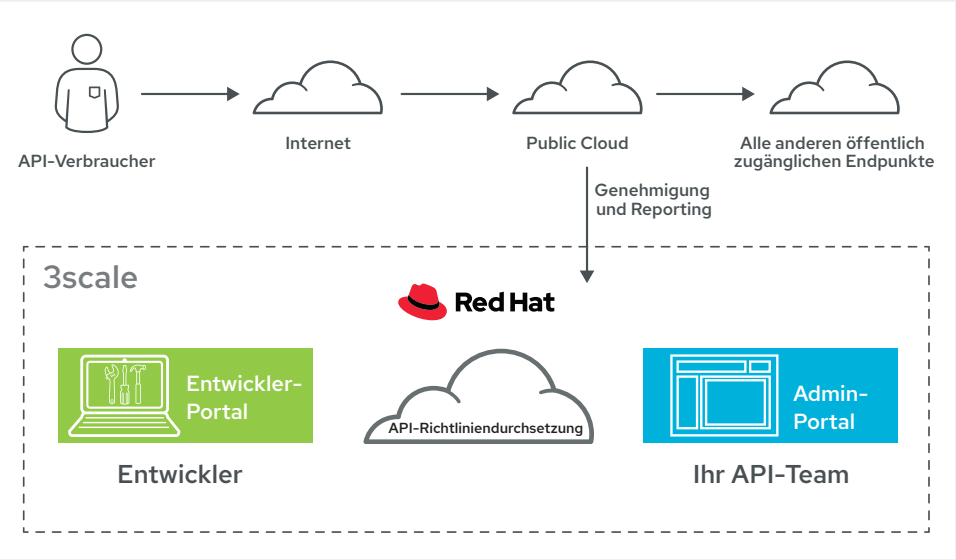


Abb. 3. Eine Übersicht über API-Management, Endpunkte und die Public Cloud

Red Hat bietet dazu OpenShift Service Mesh an, das das beliebte Istio Service Mesh mit anderen wichtigen Projekten kombiniert, darunter Jaeger (für Tracing) und Kiali (für Visualisierung), um eine bessere Verwaltbarkeit und Überwachung für Microservice-Bereitstellungen zu gewährleisten.

In 3scale enthalten ist der 3scale Istio Adapter, mit dem vollständige API-Managementfunktionen für das Istio Service Mesh in Kubernetes verfügbar gemacht werden.

„Um eine optimale Multi-Cloud-Bereitstellung zu erzielen, muss man sich viele Gedanken um Prozesse, Mitarbeiterqualifikationen und Organisationsstrukturen machen. Allerdings stimmt die Branche überein, dass Container und Kubernetes die Tech-Plattformen sind, die solche Multi-Clouds erst möglich machen."[11]

Suresh Vasudevan
Forbes

Grundpfeiler Nr. 3: Container

Virtualisierungs-, Cloud- und Container-Technologien verfolgen das gleiche Ziel: Sie abstrahieren die Software-Betriebsumgebung weg von der physischen Hardware, um dort mehrere Instanzen ausführen sowie Nutzung, Skalierung und Bereitstellung effizienter verwalten zu können. Allerdings setzen sie dazu unterschiedliche Vorgehensweisen ein. Bei der Virtualisierung wird die Betriebssystemschicht abstrahiert. Bei der Cloud wird das Konzept der permanenten dedizierten Serverinstanzen ad acta gelegt. Und mit Containern wird ein rudimentäres Betriebssystem nebst Bibliotheken definiert, das auf die Ausführung einer einzelnen Anwendung ausgelegt ist.

Der gezieltere, kompakte Ansatz von Containern ist genau das, was sie für moderne Softwareumgebungen so ideal macht. Jede Instanz verwendet eine unabänderliche Definition, und zwar vom Betriebssystem bis hin zur exakten Version der integrierten Bibliotheken. Das sich daraus ergebende Konstrukt verleiht der Umgebung in jeder Instanz eine hohe Wiederholbarkeit und Einheitlichkeit, weswegen sie sich bestens für CI/CD Pipelines (Continuous Integration und Continuous Delivery) eignen. Ihre Kompaktheit und Wiederholbarkeit machen Container zur idealen Technologieplattform für die agile Integration.

Und da Container-Images lediglich die benötigten Komponenten für einzelne Funktionseinheiten spezifizieren, erfüllen Container die Vision von Microservices und die Containerorchestrierung erleichtert die Implementierung und Verwaltung großer Microservice-Infrastrukturen.

Traditionelle Integrationsansätze besaßen eine stark zentralisierte Struktur, bei der ESBs an den Schlüsselpositionen der Infrastruktur zu finden waren. Die verteilte Integration sowie das API-Management verfügen über eine dezentrale Architektur, mit der Standorten oder Teams lediglich die benötigten Funktionen bereitgestellt werden. Container dienen bei beiden Ansätzen als zugrundeliegende Plattform, weil sie aufgrund ihrer unabänderlichen Natur für die Einheitlichkeit von Images und Bereitstellungen in allen Umgebungen sorgen und umgehend und ohne undurchsichtige Dependenzen oder Konflikte bereitgestellt oder ersetzt werden können.

Der Schlüssel zu einer verteilten Architektur ist, egal ob mit Integrationen oder APIs, dass sie Möglichkeiten zur Konzipierung und Bereitstellung neuer Services ohne komplexe Genehmigungsprozesse bietet.

Mithilfe von Containern lassen sich verteilte Integrationen und APIs als Microservices behandeln. Sie bieten einen gemeinsamen Tool-Satz sowohl für Dev- als auch Ops-Teams sowie die Nutzung beschleunigter Entwicklungsprozesse mit verwalteten Release-Workflows.

Jeder Container steht für individuelle Services oder Anwendungen so wie ein Microservice eine einzelne diskrete Funktion repräsentiert. In einer Microservice-Architektur können Dutzende oder gar Hunderte von separaten Services enthalten sein, die dann über Entwicklungs-, Test- und Produktionsumgebungen hinweg dupliziert werden. Für diese Anzahl von Instanzen ist die Möglichkeit zur Orchestrierung sowie die Ausführung fortschrittlicher Administrationsaufgaben für eine hohe Effizienz der Container-Umgebung unerlässlich.

11 *Vasudevan, Suresh, „Containers And Kubernetes Are Powering The Second Cloud Adoption Cycle", Forbes, 10. Juli 2019,* https://www.forbes.com/sites/forbestechcouncil/2019/07/10/containers-and-kubernetes-are-powering-the-second-cloud-adoption-cycle/#171ce5006929

Red Hat OpenShift vereinigt Linux®-Container mit dem Kubernetes-Orchestrierungsprojekt von Google und integriert eine zentrale Verwaltung, darunter Instanzmanagement, Überwachung, Protokollierung, Verkehrsmanagement und Automatisierung, was in einer ausschließlich aus Containern bestehenden Umgebung unmöglich wäre.

Red Hat OpenShift bietet dazu entwicklerfreundliche Tools wie Self-Service-Kataloge, Instanz-Clustering, Anwendungspersistenz sowie Isolation auf Projektebene.

Mit dieser Kombination lässt sich eine Balance zwischen den Anforderungen von Betrieb, speziell in Bezug auf Stabilität und Prüfung, sowie von Entwicklern für eine einfache Nutzung und schnelle Bereitstellung herstellen.

Implementierung der agilen Integration

Team-Praktiken

Die drei Grundpfeiler der agilen Integration sind am effizientesten, wenn sie Teams als wiederverwendbare Kapazitäten bereitgestellt werden. Mit Kapazitäten meint Red Hat, dass autorisierte Gruppen Technologien in Form eines Self-Service nutzen, organisatorische Richtlinien auf einfache Weise erfüllen können und Zugriff auf Best Practice-Informationen erhalten.

Informationsarchitekten oder IT-Administratoren müssen klare Prozesse für individuelle Teams definieren, und zwar:

- Bereitstellung allgemein verfügbarer Nutzungsrichtlinien
- Durchsetzung von Nutzungs- und Best Practice-Regeln (wo notwendig) bei gleichzeitiger Gewährleistung des Experimentierens über diese Regeln hinaus
- Wohl durchdachte Prozesse für den gesamten Lifecycle, von Prototypen und Tests über das Go-live und Updates bis hin zur Einstellung
- Gemeinsame Nutzung von Informationen für neue Bereitstellungen und Entwicklungen
- Einsatz von Infrastruktur-Teams als Enabler und Anbieter von Self-Service-Funktionen anstatt ihre zwangsweise Beteiligung an jedem Prozess

So sollte es für ein Software-Team z. B. möglich sein, neue APIs ausschließlich für die Lancierung via Self-Service zu entwickeln, zu testen und für die Markteinführung vorzubereiten. Dabei sollten Prozesse zur Aktualisierung anderer Gruppen und Dokumente zur Verfügung stehen.

Möglicherweise müssen vor der Veröffentlichung oder Produktionsphase manuelle Prozesse und Gegenkontrollen mit anderen Teams durchgeführt werden, allerdings sollte der Infrastrukturprozess trotzdem soweit wie möglich automatisiert werden.

Infrastrukturarchitektur

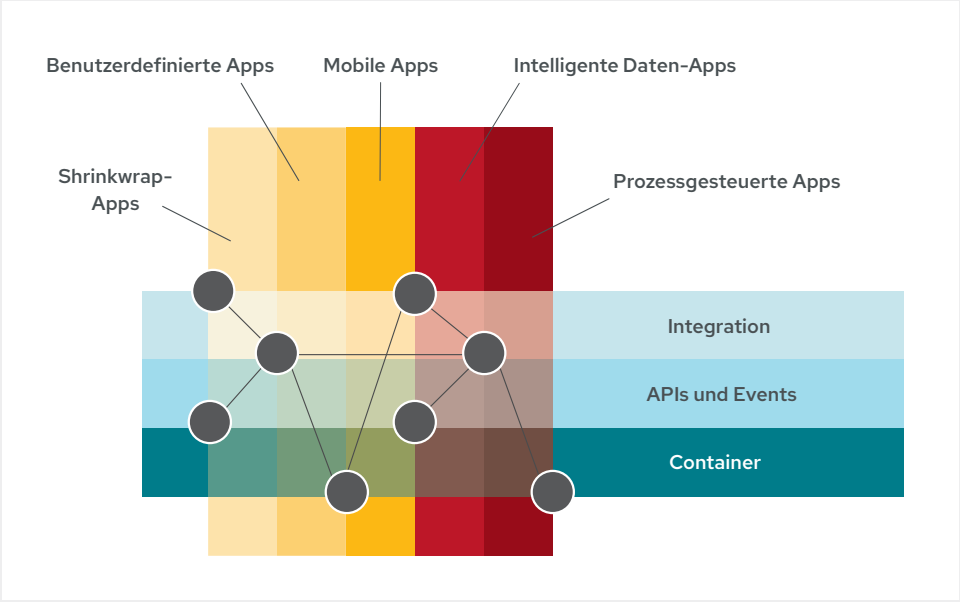


Abb. 4. So funktionieren agile Integrationstechnologien über Anwendungs-Stacks hinweg

Die internen Software-Ökosysteme einer Organisation – und in vielen Fällen auch die Zugangspunkte für externe Integrationen – bestehen aus einer synchronen und einer asynchronen Schicht. Die synchrone Schicht beinhaltet Container, APIs und Integration. Bei der asynchronen Schicht ist es ähnlich, nur dass APIs durch Events ersetzt werden. Das Event Processing besteht aus einem Event Mesh und verteilten Streaming-Plattformen.

Unterschiedliche Typen von Systemen ermöglichen eine Vielfalt an wiederverwendbaren Endpunkten, von denen jeder als wiederverwendbare API sichtbar ist. Dabei werden viele von ihnen zwecks Steigerung der Skalierbarkeit und einfacher Bereitstellung in Containern ausgeführt. Integrationen bieten bei Bedarf Transformation, Komposition oder Inline-Geschäftslogik im gesamten System, und zwar durch die Integration einer Gruppe an individuellen Services oder die Erfassung von Ergebnissen aus unterschiedlichen Teilen der Organisation.

Integrierte Anwendungen können vor der Bereitstellung von Endbenutzeranwendungen weiter aggregiert werden.

Es wird nicht davon ausgegangen, dass alle Systeme in immer kleinere Komponenten aufgeschlüsselt werden oder mehrere Schichten der API-Abstraktion durchlaufen. Solche Prozesse können die Effizienz verringern, die Latenz steigern oder unnötige Komplexität schaffen. In manchen Bereichen kann es angebracht sein, die aktuelle ESB-Legacy-Funktionalität und mit ihr die Verbindung zwischen spezifischen Anwendungen beizubehalten. Die Dependenzen zwischen verteilten Systemen müssen dazu mit geeigneten Tools überwacht und verwaltet werden.

Allerdings bedeutet für das System als Ganzes die Neudefinition der Architektur in Bezug auf Container, APIs und Integration, dass für alle Services, Integrationspunkte und Kundeninteraktionen die richtigen Entscheidungen getroffen werden können. So kann z. B. ein hohes Volumen an eingehenden Anfragen sicherheitsgeprüft und direkt an die entsprechenden Backend-Services weitergeleitet werden, ohne einen einzigen ESB-Engpass passieren zu müssen.

Bei verteilten Hybrid Cloud-Umgebungen können sich die betreffenden Backend-Systeme an unterschiedlichen physischen Standorten befinden. Die Integration von sich nahe beieinander befindenden Systemen zur Erfüllung lokaler Bedürfnisse bietet mehr Effizienz und Sicherheit als das umfassende Routing über ein einzelnes zentrales Integrationssystem mit zentraler Geschäftslogik.

Agile Organisationen und Unternehmenskultur

Der Lifecycle der Infrastruktur unterscheidet sich deutlich von dem von Softwareentwicklung und -betrieb. Die Softwareentwicklung besteht aus der Vervollständigung eines und dem Beginn eines neuen Projekts. Effizienz definiert, wie schnell ein Produkt veröffentlicht oder wie viele Features in einem bestimmten Zeitraum entwickelt werden können. Selbst für den Betriebsbereich, bei dem es hauptsächlich um Wartung und Stabilität geht, ist es von Vorteil, die Anwendung von Sicherheits-Patches und Updates, Bereitstellung neuer Services oder Rücknahme von Änderungen schneller und effizienter durchzuführen.

Für die Infrastruktur gilt allerdings ein ganz anderer Ansatz. So involviert dieser einen längeren Zeitrahmen sowie disparate, hochspezialisierte Gruppen, also nicht vergleichbar mit funktionsübergreifenden Teams, die an spezifischen Software-Engineering-Projekten arbeiten.

Infrastrukturprojekte sind typischerweise viel umfangreicher als Softwareprojekte, d. h., dass in den kurzen Release-Zyklen keine wirklichen Ergebnisse geliefert oder höchstens unfertige Projekte hervorgebracht werden. Wie Andrew Froehlich, Experte für Unternehmens-IT, in der Online-Publikation InformationWeek schrieb, ist Infrastruktur durch einen sogenannten Point-of-no-Return eingeschränkt, speziell in Bezug auf Hardware und Rechenzentren. Aber selbst bei der Public Cloud ist irgendwann ein Punkt erreicht, an dem man das Projekt nicht einfach verwerfen und neu beginnen kann.[12] Infrastruktur ist permanent. Allerdings ist es möglich, Methodologien mit der Leistung der Infrastruktur in Einklang zu bringen.

Die Vorteile reaktiver iterativer Prozesse wie Agilität und DevOps sind für Entwicklungs- und Betriebs-Teams klar ersichtlich, nicht aber für Infrastruktur-Teams. Es ist allerdings überaus wichtig, Infrastruktur- mit DevOps-Teams auszurichten, um eine maximale Effizienz zu erzielen. Das internationale Consulting-Unternehmen McKinsey hat es so formuliert: „Die Nutzung der agilen Transformation zur Modernisierung der Organisation einer IT-Infrastruktur ist keine einfache Aufgabe, lohnt sich aber in jedem Fall. Unseren Erkenntnissen nach sind IT-Infrastrukturgruppen in der Lage, ihre Produktivität mithilfe des agilen Ansatzes je nach Unternehmensgröße binnen sechs bis achtzehn Monaten um 25 bis 30 % zu steigern."[13]

Agile Integrationstechnologien sind die Grundlage für eine agilere Infrastruktur. APIs, Container-Images und verteilte Integrationen sind das neue Vokabular in der Diskussion über Software-Infrastruktur.

---

12 *Froehlich, Andrew, „Should IT go agile? The pros and cons", 6. Oktober 2015.*http://www.informationweek.com/infrastructure/pc-and-servers/should-it-go-agile-the-pros-and-cons/d/d-id/1322448

13 *Cormella-Dorda, Santiago, et al. „Transforming IT infrastructure organizations using agile", McKinsey Digital, Oktober 2018,* https://www.mckinsey.com/business-functions/mckinsey-digital/our-insights/transforming-it-infrastructure-organizations-using-agile

Das Agile Manifest definiert vier Kernprinzipien der Softwareentwicklung.[14] In einer agilen, integrationsbasierten Infrastruktur können diese Prinzipien auf die Integrationsstrategie angewandt werden.

1	**Personen und Interaktionen haben Vorrang vor Prozessen und Tools.** Bei der Infrastruktur konzentriert sich die Diskussion in erster Linie auf die Interaktionen zwischen Teams. Dazu gehören die direkte Kommunikation, die von APIs, Messaging und Verkehrsmustern bestimmt wird, Interdependenzen auf Systemebene sowie Test- und Release-Prozesse wie CI/CD-Pipelines.
2	**Funktionsfähige Software hat Vorrang vor umfassender Dokumentation.** Infrastruktur muss schon ihrer Natur nach rund um die Uhr funktionieren. Das Einpflegen von Änderungen sollte dabei eher Schritt für Schritt als mit wenigen großen Paketen erfolgen. Aus diesem Grund ist eine funktionierende Infrastruktur stets eine selbstverständliche Anforderung. Im Sinne der Infrastrukturstrategie bedeutet der Begriff „funktionierend" hier, dass die Infrastrukturkomponente das erwartete Endbenutzerverhalten gemäß Leistungsprofil bietet.
3	**Zusammenarbeit mit dem Kunden hat Vorrang vor Vertragsverhandlungen.** Bei Infrastruktursystemen wird mithilfe von Verträgen geregelt, wie Infrastruktur-Teams mit Systemdependenzen wie Sicherheitsrichtlinien, SLAs und sogar veröffentlichten APIs umgehen. Mit Kunden sind sowohl interne als auch externe Nutzer dieser Systeme gemeint. Agilität verleiht diesen Nutzern ein Mitspracherecht bei potenziellen Änderungen der mit den Systemen verknüpften Richtlinien und Schnittstellen und sorgt für deren schnelle Ausführung. Mithilfe von verteilten Integrationen wird diese Kooperation ausgeweitet, und zwar indem Teams die direkte Kontrolle über die Entwicklung und Implementierung von Integrationen eingeräumt wird.
4	**Das Eingehen auf Änderungen hat Vorrang vor strikter Planverfolgung.** Bei diesem Prinzip werden Prozesse mithilfe von Technologie unterstützt. Was Infrastrukturen betrifft, sollten diese Systeme stets stabil bleiben. Allerdings stellen neuere Technologien wie Container eine elastische Plattform bereit. Es ist möglich, Instanzen je nach Bedarf hinzuzufügen oder zu entfernen, Bereitstellungen und Updates zu automatisieren und Änderungen über mehrere Instanzen hinweg zu orchestrieren. Veröffentlichte API-Definitionen bieten wiederverwendbare Tools für eine einheitlichere Entwicklung. Durch diesen Ansatz wird eine Plattform möglich, die auf die Anpassung an Änderungen ausgerichtet ist.

Abb. 5. Kernprinzipien der Softwareentwicklung aus dem Agilen Manifest

Agile Integration nutzt Technologie zur Unterstützung kultureller Änderungen bei Infrastruktur-Teams und dient als Basis für die Infrastrukturstrategie. Mit ihr können Infrastrukturtechnologien und Teams effektiver auf Entwicklungs- und Geschäftsstrategien ausgerichtet werden.

Die agile Methodologie dient zur Identifizierung einiger wichtiger Bestandteile von Softwareprojekten wie Personen, Builds und Dependenzen. Außerdem können damit Beziehungen zwischen diesen Elementen definiert werden. Wenn man die Integrationsinfrastruktur als agiles Projekt erachtet, findet man ähnliche Elemente und Beziehungen bzw. Parallelen zur agilen Methodologie, darunter Teams, Container-Images, APIs und Integrationspunkte. In Tabelle 3 werden einige dieser Parallelen beschrieben.

14 *„Manifesto for Agile Software Development"*, http://agilemanifesto.org/

Tabelle 3. Vergleich der Elemente der agilen Software mit Infrastrukturagilität

Projekt	Organisation	Detail
Personen	Teams	Teams sind für bestimmte Bereiche der Infrastruktur verantwortlich. So werden Informationen rund um Team-Zuständigkeiten definiert, wie die vom Team verwalteten Systeme oder APIs, Teamleiter sowie die Ziele des Teams.
Module	APIs	Gut definierte Schnittstellen (APIs) sind langfristig stabil, haben ihre eigenen Roadmaps, werden von spezifischen Teams ausgeführt und stellen eine bestimmte, für die Organisation wichtige Funktion bereit.
Builds	Container Images	Releases basieren auf einsatzfähigen Einheiten, die getestet und getaggt wurden und von jedem Team mit Zugriff zuverlässig bereitgestellt werden können. Sie ersetzen monolithischen versionierten Code.
Kompilierung von Dependenzen	Integrationen	Mit diesem Element werden die Integrationen und Mappings zwischen verschiedenen Komponenten dieser verteilten Systeme bestimmt. Diese Integrationspunkte können wie jeder andere Bestandteil des Systems auch verwaltet, aktiviert oder deaktiviert, versioniert und getestet werden.
Build-Test	Infrastruktur-Automatisierung	Es handelt sich hier um ein vollständiges Lifecycle-Management, von der Prüfung von Software-Builds, Performance und Benutzeranforderungen bis hin zur Ausführung und Überwachung mehrerer Systeme.

Die Anwendung agiler Prinzipien auf die Infrastrukturplanung

Die meisten Änderungsmanagementansätze erfordern eine umfassende Dokumentation aller Subsysteme. Damit müssen im Detail alle Aspekte des Systems erfasst werden, von der Überwachungsmethode über die Leistungsparameter bis hin zu den verantwortlichen Teams. Agile Prinzipien erfordern Zusammenarbeit und Anpassungsfähigkeit, ein Faktum, das dem dokumentlastigen Änderungsmanagement entgegensteht.

Statt einer präskriptiven Definition aller Beteiligten, Änderungen und Systemkomponenten sollten Sie einen Satz Richtlinien und Standards zur Bewertung des Änderungsanfrage- und -planungsprozesses definieren. Ziehen Sie dabei folgende Fragen in Betracht:

- Welche durchgehende Nutzererfahrung soll erreicht werden?
- Welchen Beitrag leisten die Beteiligten, also Teams, APIs und Systeme, zur Verbesserung dieser Erfahrung im Laufe der Zeit?
- Wie werden Überwachung und Alarmierung für die Aufrechterhaltung der Service Levels definiert? Nach welchen Vorgaben?
- Welche Form der Automatisierung wird zur Verifizierung des erwarteten Verhaltens eingesetzt?
- Welche Release-Pipeline verwenden die Teams für den Test/die Bereitstellung neuer Versionen der eigenen Subsysteme ohne Unterbrechung der Nutzererfahrung?
- Auf welche Art und Weise beeinflusst der Ausfall eines Komponenten-Service die Service Levels des gesamten Systems?

Das Änderungsmanagement in einer agilen Infrastruktur sollte nicht nur eine Vertragsbeziehung, sondern eine fortlaufende Zusammenarbeit sein.

Wie stehen Ihre Erfolgschancen?

Wie hoch ist die Wahrscheinlichkeit für einen Erfolg Ihres Projekts? Dies hängt in erster Linie von Ihren Erfolgskriterien ab, also ob es um die Erfüllung von Spezifikationen, die Steigerung der Kundenakzeptanz oder um die reine Veröffentlichung geht. In einer Studie von PMI (Project Management Institute) hat sich herausgestellt, dass aktuell mehr Projekte ihre Zielvorgaben erfüllen als noch in den vergangenen fünf Jahren. Als Grund für diese Verbesserung wurde die bessere Ausrichtung zwischen IT- und Geschäfts-Teams ausgemacht, die nützlichere Daten in Bezug auf Strategie und Kundenanforderungen ermöglicht hat.[15]

Eine der Ursachen für diese strategische Ausrichtung ist die Implementierung agiler Teams. Dadurch werden Zusammenarbeit und Feedback, eine ganzheitliche Ansicht aller Probleme und Systeme sowie kreative Ansätze gefördert.

Dank eines gemeinsamen Technologie-Stacks verschiebt sich die Diskussion von unabhängigem Code hin zu Systemen und ihren Interdependenzen. Die Analyse erfolgt so auf Systemebene, d. h. die gesamte Sammlung an Softwareinfrastruktur, darunter intern entwickelte Programme, Anbietersysteme und ihre Verbindungen untereinander, werden als ein einziges System betrachtet. APIs und Messaging-Systeme können sich durch die gesamte Infrastruktur ziehen und die Vereinheitlichung der Softwaresysteme unterstützen.

Weil APIs und verteilte Integrationen von individuellen Dev- und Ops-Teams entwickelt und verstanden werden können, zeigt sich ein viel klareres Bild der Verantwortlichkeiten der Teams in Sachen Integrationen. Diese wiederum werden besser verständlich, da die Interdependenzen zwischen Systemen und Anwendungen von den für Entwicklung und Bereitstellung zuständigen Teams erkannt werden.

Der Einsatz von Integration als Grundlage für die Infrastruktur und die Verteilung der dazugehörigen Verantwortung auf verschiedene Teams schafft eine Infrastrukturumgebung, in der agile Ansätze mehr Relevanz erhalten.

15 *Florentine, Sharon, „IT project success rates finally improving", 27. Februar 2017.* https://www.cio.com/article/3174516/project-management/it-project-success-rates-finally-improving.html

Fazit: Die Bereitstellung agiler Integration

Agilität ist ein Prozess und kein Projekt.

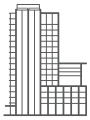
Nie zuvor war es für Organisationen wichtiger, schnell auf Marktänderungen reagieren zu können. Diese Fähigkeit muss über IT-Systeme bereitgestellt werden, um neue oder bestehende Services umgehend veröffentlichen oder aktualisieren zu können. So ist also eine Umstrukturierung der IT-Infrastruktur unerlässlich, da sie die Basis für digitale Services darstellt.

Infrastruktur-Teams werden schon seit jeher mit langwierigen, modulierten Prozessen assoziiert, die Risikominderung und Stabilität gewährleisten. Heute aber ist es möglich, diesen Infrastrukturansatz von der Fokussierung auf Hardware oder Plattformen hin zur Integration zu verschieben. Integration ist kein Subset der Infrastruktur. Es handelt sich hier um einen konzeptionellen Ansatz, der Daten und Anwendungen mit Hardware und Plattformen integriert.

Red Hat nennt dieses Konzept agile Integration, eine Methode zur Nutzung ihrer drei Grundpfeiler – verteilte Integration, APIs und Container – für die Schaffung einer agileren und anpassungsfähigeren Infrastruktur. Zur Unterstützung kultureller Veränderungen wird Technologe benötigt. Das heißt mehr Agilität, nicht nur für die Software, sondern auch für Infrastruktur-Teams. Während sich Infrastruktur-Teams auf agile Prinzipien ausrichten, kann die Unterstützung für diese Änderungen durch die schrittweise Integration von Technologie realisiert werden. Es gibt kein einzelnes Projekt, mit dem sich die Architektur eines gesamten Unternehmens auf 100 % Agilität umgestalten lässt. Es ist daher wohl effizienter, eine agile Integrationstechnologie zu implementieren oder einen Geschäftsbereich umzuwandeln und dann diese Änderungen inkrementell auf die gesamte Organisation auszuweiten.

Die Verbesserung der Reaktionsfähigkeit einer IT-Infrastruktur in Bezug auf Änderungen ist ein langfristiges strategisches Ziel. Um Fortschritte zu realisieren, muss man nicht gleich grundlegende unternehmensweite Änderungen implementieren. Vielleicht ist es noch nicht einmal notwendig, Änderungen isoliert zu entwickeln und dann im Rahmen eines Rollouts zu implementieren.

Die agile Integration bietet einen technischen und organisatorischen Rahmen zur Umstrukturierung der IT-Infrastruktur.



ÜBER RED HAT

Red Hat, weltweit führender Anbieter von Open Source Software-Lösungen für Unternehmen, folgt einem community-basierten Ansatz, um verlässliche und leistungsstarke Technologien in den Bereichen Linux, Hybrid Cloud, Container und Kubernetes bereitzustellen. Wir unterstützen Kunden bei der Integration neuer und bestehender IT-Anwendungen, der Entwicklung cloudnativer Anwendungen, der Standardisierung auf unserem branchenführenden Betriebssystem sowie der Automatisierung, Sicherung und Verwaltung komplexer Umgebungen. Dank unserer vielfach ausgezeichneten Support-, Training- und Consulting-Services ist Red Hat ein bewährter Partner der Fortune 500 Unternehmen. Als strategischer Partner für Cloud-Anbieter, Systemintegratoren, Anwendungsanbieter, Kunden und Open Source Communities hilft Red Hat Organisationen auf ihrem Weg in die digitale Zukunft.

facebook.com/redhatinc
@RedHatDACH
linkedin.com/company/red-hat

EUROPA, NAHOST, UND AFRIKA (EMEA)
00800 7334 2835
de.redhat.com
europe@redhat.com

TÜRKEI
00800 448820640

ISRAEL
1 809 449548

VAE
8000-4449549

de.redhat.com
#F20123_1119