AWSWeißbuch

# Grundlagen der SaaS-Architektur

# Grundlagen der SaaS-Architektur: AWSWeißbuch

# Table of Contents

# Grundlagen der SaaS-Architektur

Datum der Veröffentlichung: 3. August 2022 (Dokumentversionen)

Umfang, Ziele und Art der Unternehmensführung in einem Software-as-a-Service (SaaS) -Modell können schwierig zu definieren sein. Die Terminologie und Muster, die zur Charakterisierung von SaaS verwendet werden, variieren je nach Herkunft. Ziel dieses Dokuments ist es, die grundlegenden Elemente von SaaS besser zu definieren und ein klareres Bild von Mustern, Begriffen und Wertesystemen zu vermitteln, die bei der Entwicklung und Bereitstellung eines SaaS-Systems angewendet werdenAWS. Das umfassendere Ziel besteht darin, eine Sammlung grundlegender Erkenntnisse bereitzustellen, die den Kunden einen klareren Überblick über die Optionen geben, die sie bei der Einführung eines SaaS-Bereitstellungsmodells in Betracht ziehen sollten.

Dieses Whitepaper richtet sich an SaaS-Entwickler und -Architekten, die am Anfang ihrer SaaS-Reise stehen, sowie an erfahrenere Entwickler, die ihr Verständnis der SaaS-Kernkonzepte verfeinern möchten. Einige dieser Informationen können auch für Besitzer und Strategen von SaaS-Produkten nützlich sein, die sich mit der SaaS-Landschaft vertraut machen möchten.

## Einführung

Der Begriff Software as a Service (SaaS) wird verwendet, um ein Geschäfts- und Bereitstellungsmodell zu beschreiben. Die Herausforderung besteht jedoch darin, dass das, was es bedeutet, SaaS zu sein, nicht allgemein verstanden wird.

Über einige der Kernpfeiler von SaaS herrscht zwar Einigkeit, aber es herrscht nach wie vor einige Verwirrung darüber, was es bedeutet, SaaS zu sein. Es ist natürlich, dass Teams SaaS unterschiedlich betrachten. Gleichzeitig kann die mangelnde Klarheit in Bezug auf SaaS-Konzepte und -Begriffe bei denjenigen, die sich mit einem SaaS-Bereitstellungsmodell befassen, zu einiger Verwirrung führen.

Dieses Dokument konzentriert sich auf die Erläuterung der Terminologie, die zur Beschreibung der SaaS-Kernkonzepte verwendet wird. Eine gemeinsame Denkweise zu diesen Konzepten schafft ein klares Bild von den grundlegenden Elementen einer SaaS-Architektur und bietet Ihnen ein gemeinsames Vokabular für die Beschreibung von SaaS-Architekturkonstrukten. Dies ist besonders nützlich, wenn Sie sich mit zusätzlichen Inhalten befassen, die auf diesen Themen aufbauen.

Dieses Whitepaper geht von den Architekturdetails von Multi-Tenancy zurück und untersucht, wie wir die Grundlagen dessen definiert haben, was es bedeutet, SaaS zu sein. Im Idealfall bietet dies auch

eine klarere Terminologie, die es Unternehmen ermöglicht, sich schneller auf die Art und Weise ihrer SaaS-Lösungen einzustellen.

## Sind Sie gut durchdacht?

Das AWSWell-Architected Framework hilft Ihnen dabei, die Vor- und Nachteile der Entscheidungen zu verstehen, die Sie beim Aufbau von Systemen in der Cloud treffen. Die sechs Säulen des Frameworks ermöglichen es Ihnen, bewährte architektonische Verfahren für den Entwurf und Betrieb zuverlässiger, sicherer, effizienter, kostengünstiger und nachhaltiger Systeme zu erlernen. Mithilfe der AWS Well-Architected Tool, die kostenlos in der verfügbar ist AWS Management Console, können Sie Ihre Workloads anhand dieser Best Practices überprüfen, indem Sie für jede Säule eine Reihe von Fragen beantworten.

In der SaaS-Linse konzentrieren wir uns auf bewährte Methoden für die Architektur Ihrer Software-as-a-Service (SaaS) -Workloads. AWS

Weitere Ratschläge von Experten und Best Practices für Ihre Cloud-Architektur — Referenzarchitekturbereitstellungen, Diagramme und Whitepapers — finden Sie im Architecture Center. AWS

# SaaS ist ein Geschäftsmodell

Um zu definieren, was es bedeutet, SaaS zu sein, müssen Sie sich auf ein wichtiges Prinzip einigen: SaaS ist ein Geschäftsmodell. Dies bedeutet, dass — vor allem — die Einführung eines SaaS-Bereitstellungsmodells direkt von einer Reihe von Geschäftszielen abhängt. Ja, Technologie wird eingesetzt werden, um einige dieser Ziele zu erreichen, aber bei SaaS geht es darum, eine Denkweise und ein Modell einzuführen, die auf bestimmte Geschäftsziele abzielen.

Schauen wir uns einige der wichtigsten Geschäftsziele genauer an, die mit der Einführung eines SaaS-Bereitstellungsmodells verbunden sind.

- Agilität — Dieser Begriff fasst das umfassendere Ziel von SaaS zusammen. Erfolgreiche SaaS-Unternehmen basieren auf der Idee, dass sie bereit sein müssen, sich kontinuierlich an die Markt-, Kunden- und Wettbewerbsdynamik anzupassen. Große SaaS-Unternehmen sind so strukturiert, dass sie ständig neue Preismodelle, neue Marktsegmente und neue Kundenbedürfnisse berücksichtigen.
- Betriebliche Effizienz — SaaS-Unternehmen verlassen sich auf betriebliche Effizienz, um Skalierbarkeit und Agilität zu fördern. Dies erfordert die Einführung einer Kultur und Tools, die darauf ausgerichtet sind, eine betriebliche Präsenz zu schaffen, die die häufige und schnelle Veröffentlichung neuer Funktionen fördert. Es bedeutet auch, dass Sie über ein einziges, einheitliches Erlebnis verfügen, das es Ihnen ermöglicht, alle Kundenumgebungen gemeinsam zu verwalten, zu betreiben und bereitzustellen. Die Idee, einmalige Versionen und Anpassungen zu unterstützen, ist vorbei. SaaS-Unternehmen legen großen Wert auf betriebliche Effizienz als zentrale Säule ihrer Fähigkeit, das Geschäft erfolgreich auszubauen und zu skalieren.
- Reibungsloses Onboarding — Um agiler zu sein und Wachstum zu begrüßen, müssen Sie auch Wert darauf legen, jegliche Reibungsverluste im Onboarding-Prozess für Mandantenkunden zu verringern. Dies gilt allgemein für Business-to-Business-Kunden (B2B) und business-to-customer (B2C). Unabhängig davon, welches Kundensegment oder welchen Kundentyp Sie betreuen, müssen Sie sich immer auf die Amortisierungszeit für Ihre Kunden konzentrieren. Die Umstellung auf ein serviceorientiertes Modell erfordert, dass sich das SaaS-Geschäft auf alle Aspekte des Kundenerlebnisses konzentriert, wobei der Wiederholbarkeit und Effizienz des gesamten Onboarding-Lebenszyklus besondere Bedeutung beigemessen wird.
- Innovation — Bei der Umstellung auf SaaS geht es nicht nur darum, die Bedürfnisse der aktuellen Kunden zu erfüllen, sondern auch darum, die grundlegenden Elemente zu schaffen, die Ihnen Innovationen ermöglichen. Sie möchten in Ihrem SaaS-Modell auf Kundenbedürfnisse reagieren und darauf eingehen. Sie möchten diese Flexibilität jedoch auch nutzen, um future Innovationen

voranzutreiben, die es Ihnen ermöglichen, neue Märkte, Chancen und Effizienzsteigerungen für Ihre Kunden zu erschließen.

- Reaktion des Marktes — SaaS entfernt sich von der traditionellen Vorstellung vierteljährlicher Veröffentlichungen und Zweijahresplänen. Es ist auf seine Agilität angewiesen, um dem Unternehmen die Möglichkeit zu geben, nahezu in Echtzeit auf Marktdynamiken zu reagieren und zu reagieren. Die Investition in die organisatorischen, technischen und kulturellen Elemente von SaaS bietet die Möglichkeit, die Geschäftsstrategie auf der Grundlage der sich abzeichnenden Kunden- und Marktdynamik anzupassen.
- Wachstum — SaaS ist eine wachstumsorientierte Geschäftsstrategie. Durch die Ausrichtung aller beweglichen Teile des Unternehmens auf Agilität und Effizienz können SaaS-Organisationen ein Wachstumsmodell anstreben. Dies bedeutet, dass Sie die Mechanismen einrichten müssen, die die schnelle Einführung Ihres SaaS-Angebots unterstützen und begrüßen.

Sie werden feststellen, dass jeder dieser Punkte auf ein Geschäftsergebnis ausgerichtet ist. Es gibt eine Vielzahl von technischen Strategien und Mustern, die zum Aufbau eines SaaS-Systems verwendet werden können. Nichts an diesen technischen Strategien ändert jedoch die allgemeine Geschäftsgeschichte.

Wenn wir uns mit Unternehmen zusammensetzen und sie fragen, was sie im Rahmen ihrer Einführung von SaaS erreichen wollen, beginnen wir immer mit dieser geschäftsorientierten Diskussion. Die technologischen Entscheidungen sind wichtig, müssen jedoch im Zusammenhang mit diesen Geschäftszielen realisiert werden. Multi-Tenant zu sein, ohne beispielsweise Agilität, betriebliche Effizienz oder reibungsloses Onboarding zu erreichen, würde den Erfolg Ihres SaaS-Geschäfts untergraben.

Vor diesem Hintergrund wollen wir versuchen, dies zu einer präziseren Definition von SaaS zu formalisieren, die den zuvor skizzierten Prinzipien entspricht:

SaaS ist ein Geschäfts- und Softwarebereitstellungsmodell, das Unternehmen die Möglichkeit gibt, ihre Lösungen in einem reibungslosen, serviceorientierten Modell anzubieten, das den Wert für Kunden und Anbieter maximiert. Es stützt sich auf Agilität und betriebliche Effizienz als Eckpfeiler einer Geschäftsstrategie, die Wachstum, Reichweite und Innovation fördert.

Sie sollten sehen, wie die Geschäftsziele aufeinander abgestimmt sind und wie sie davon abhängen, dass alle Kunden ein gemeinsames Erlebnis haben. Ein großer Teil der Umstellung auf SaaS bedeutet, sich von den einmaligen Anpassungen zu entfernen, die Teil eines traditionellen Softwaremodells sein könnten. Jeder Versuch, unseren Kunden Spezialisierung anzubieten, lenkt uns im Allgemeinen von den Kernwerten ab, die wir mit SaaS erreichen wollen.

# Sie sind eine Dienstleistung, kein Produkt

Die Einführung eines „Service“ -Modells ist mehr als nur Marketing oder Terminologie. In einer Dienstleistungsmentalität werden Sie sich von Aspekten des traditionellen produktbasierten Entwicklungsansatzes entfernen. Features und Funktionen sind sicherlich für jedes Produkt wichtig, aber SaaS legt mehr Wert auf die Erfahrung, die Kunden mit Ihrem Service haben werden.

Was bedeutet das? In einem serviceorientierten Modell denken Sie mehr darüber nach, wie Kunden in Ihren Service aufgenommen werden, wie schnell sie einen Mehrwert erzielen und wie schnell Sie Funktionen einführen können, die den Kundenbedürfnissen entsprechen. Details, die sich darauf beziehen, wie Ihr Service aufgebaut, betrieben und verwaltet wird, entziehen sich der Sicht Ihres Kunden.

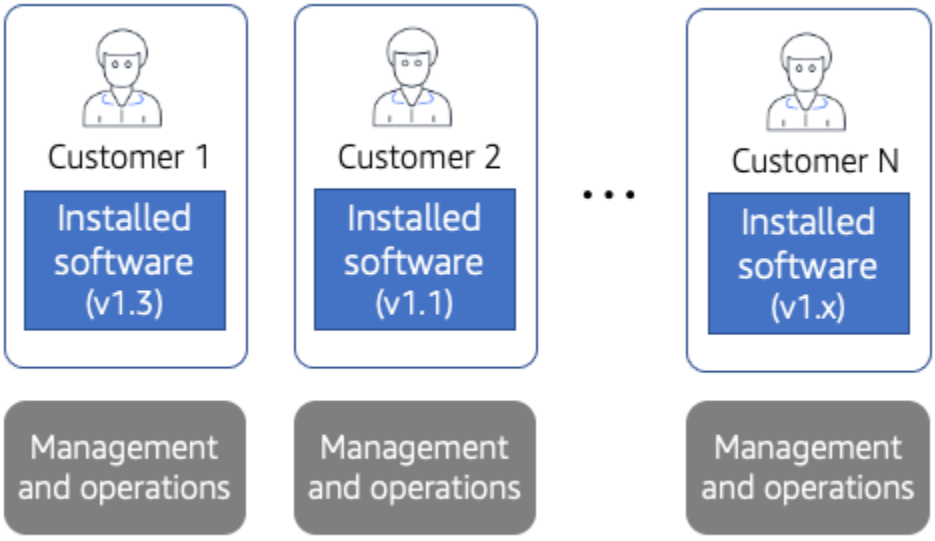
In diesem Modus denken wir über diesen SaaS-Dienst wie über jeden anderen Dienst nach, den wir nutzen könnten. Wenn wir in einem Restaurant sind, ist uns das Essen sicherlich wichtig, aber wir kümmern uns auch um den Service. Wie schnell Ihr Server an Ihren Tisch kommt, wie oft er Ihr Wasser nachfüllt, wie schnell das Essen kommt — all dies sind Messwerte für die Serviceerfahrung. Dies ist dieselbe Denkweise und dasselbe Wertesystem, die unsere Denkweise über den Aufbau eines SaaS-Dienstes prägen sollten.

Dieses as-a-serviceModell sollte einen starken Einfluss darauf haben, wie Sie Ihre Teams und Ihren Service zusammenstellen. Ihr Arbeitsstau wird diese Erfahrungsattribute nun auf die gleiche oder höhere Ebene stellen als Merkmale und Funktionen. Das Unternehmen wird diese auch als Grundlage für das langfristige Wachstum und den Erfolg Ihres SaaS-Angebots betrachten.

# Die Die Die Die Die Die Die

Um SaaS zu verstehen, beginnen wir mit einer relativ einfachen Vorstellung davon, was wir mit der Gründung eines SaaS-Unternehmens erreichen wollen. Der beste Ausgangspunkt ist, sich anzusehen, wie herkömmliche Software (ohne SaaS) erstellt, verwaltet und betrieben wurde.

Das folgende Diagramm bietet einen konzeptionellen Überblick darüber, wie verschiedene Anbieter ihre Lösungen verpackt und bereitgestellt haben.




Das klassische Modell für die Paketierung und Bereitstellung von Softwarelösungen

In diesem Diagramm haben wir eine Sammlung von Kundenumgebungen beschrieben. Diese Kunden repräsentieren die verschiedenen Unternehmen oder Organisationen, die die Software eines Anbieters gekauft haben. Jeder dieser Kunden arbeitet im Wesentlichen in einer eigenständigen Umgebung, in der er das Produkt eines Softwareanbieters installiert hat.

In diesem Modus wird die Installation jedes Kunden als eigenständige Umgebung behandelt, die diesem Kunden gewidmet ist. Das bedeutet, dass sich die Kunden selbst als Eigentümer dieser Umgebungen betrachten und möglicherweise einmalige Anpassungen oder individuelle Konfigurationen wünschen, die ihren Bedürfnissen entsprechen.

Ein häufiger Nebeneffekt dieser Denkweise ist, dass Kunden kontrollieren, welche Version eines Produkts sie verwenden. Das Problem kann aus einer Vielzahl von Gründen auftreten. Kunden sind

möglicherweise besorgt über neue Funktionen oder machen sich Sorgen über Unterbrechungen im Zusammenhang mit der Einführung einer neuen Version.

Sie können sich vorstellen, wie sich diese Dynamik auf den betrieblichen Fußabdruck des Softwareanbieters auswirken kann. Je mehr Sie Ihren Kunden individuelle Umgebungen ermöglichen, desto schwieriger wird es, die unterschiedlichen Konfigurationen der einzelnen Kunden zu verwalten, zu aktualisieren und zu unterstützen.

Dieser Bedarf an einmaligen Umgebungen erfordert oft, dass Unternehmen eigene Teams zusammenstellen, die für jeden Kunden separate Management- und Betriebserfahrungen bieten. Einige dieser Ressourcen können zwar von Kunden gemeinsam genutzt werden, aber bei diesem Modell fallen in der Regel zusätzliche Kosten für jeden neuen Kunden an, der aufgenommen wird.

Wenn jeder Kunde seine Lösungen in seiner eigenen Umgebung (in der Cloud oder vor Ort) ausführt, wirkt sich dies ebenfalls auf die Kosten aus. Sie können zwar versuchen, diese Umgebungen zu skalieren, die Skalierung ist jedoch auf die Aktivität eines einzelnen Kunden beschränkt. Im Wesentlichen beschränkt sich Ihre Kostenoptimierung auf das, was Sie im Rahmen einer individuellen Kundenumgebung erreichen können. Dies bedeutet auch, dass Sie möglicherweise separate Skalierungsstrategien benötigen, um den Aktivitätsschwankungen zwischen den Kunden Rechnung zu tragen.

Zunächst werden einige Softwareunternehmen dieses Modell als mächtiges Konstrukt betrachten. Sie nutzen die Möglichkeit, einmalige Anpassungen als Verkaufstool anzubieten, sodass neue Kunden Anforderungen stellen können, die für ihre Umgebung einzigartig sind. Die Kundenzahl und das Wachstum des Unternehmens bleiben zwar bescheiden, aber dieses Modell scheint absolut nachhaltig zu sein.

In dem Maße, wie Unternehmen zunehmend erfolgreich sind, stellen die Beschränkungen dieses Modells jedoch echte Herausforderungen dar. Stellen Sie sich zum Beispiel ein Szenario vor, in dem Ihr Unternehmen einen deutlichen Wachstumsschub erreicht, sodass Sie in schnellem Tempo viele neue Kunden gewinnen. Dieses Wachstum wird den operativen Aufwand, die Verwaltungskomplexität, die Kosten und eine Vielzahl anderer Probleme erhöhen.

Letztlich können der kollektive Aufwand und die Auswirkungen dieses Modells den Erfolg eines Softwareunternehmens grundlegend untergraben. Der erste Schwachpunkt könnte die betriebliche Effizienz sein. Der zusätzliche Personalaufwand und die Kosten, die mit der Gewinnung von Kunden verbunden sind, beginnen, die Margen des Unternehmens zu schmälern.

Betriebliche Probleme sind jedoch nur ein Teil der Herausforderung. Das eigentliche Problem ist, dass sich dieses Modell bei seiner Skalierung direkt auf die Fähigkeit des Unternehmens auswirkt,

neue Funktionen zu veröffentlichen und mit dem Markt Schritt zu halten. Wenn jeder Kunde über seine eigene Umgebung verfügt, müssen Anbieter eine Vielzahl von Update-, Migrations- und Kundenanforderungen abwägen, wenn sie versuchen, neue Funktionen in ihr System zu integrieren.

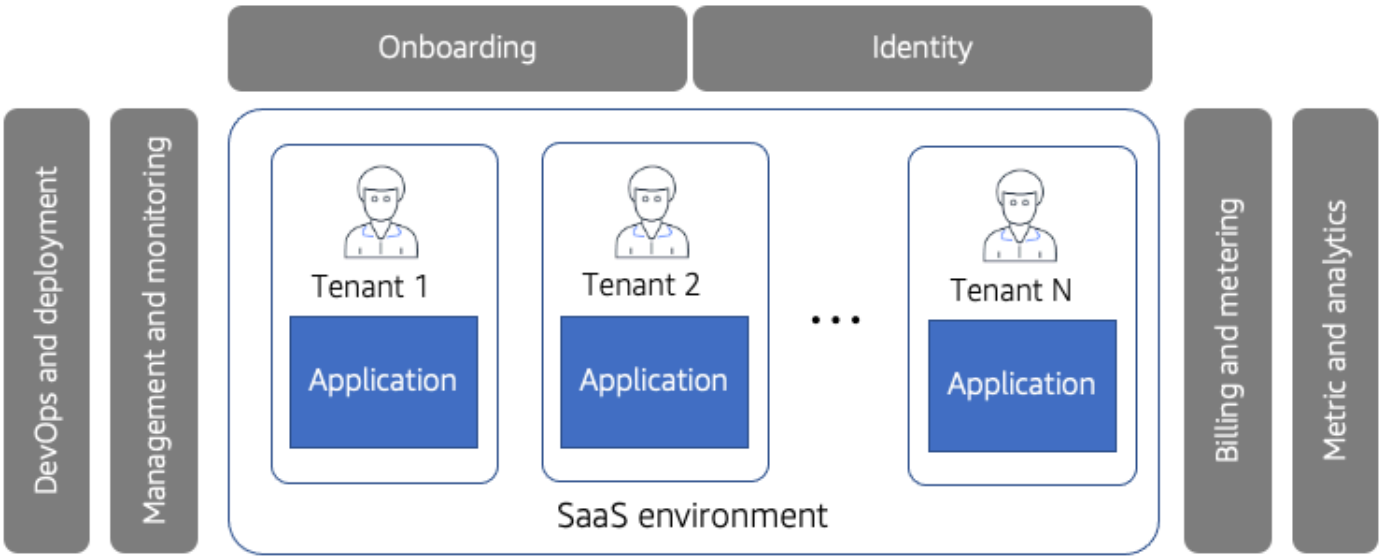
Dies führt in der Regel zu längeren und komplexeren Veröffentlichungszyklen, wodurch die Anzahl der Veröffentlichungen, die jedes Jahr veröffentlicht werden, tendenziell reduziert wird. Noch wichtiger ist, dass die Teams aufgrund dieser Komplexität mehr Zeit darauf verwenden, jede neue Funktion zu analysieren, lange bevor sie für einen Kunden veröffentlicht wird. Die Teams konzentrieren sich zunehmend auf die Validierung neuer Funktionen und weniger auf die Geschwindigkeit der Bereitstellung. Der Aufwand für die Veröffentlichung neuer Funktionen wird so groß, dass sich die Teams mehr auf die Testmechanik konzentrieren und weniger auf die neuen Funktionen, die die Innovation ihres Angebots vorantreiben.

In diesem langsameren, vorsichtigeren Modus haben Teams in der Regel lange Zykluszeiten, wodurch die Lücken zwischen der Entstehung einer Idee und dem Zeitpunkt, an dem sie in den Händen eines Kunden landet, immer größer werden. Insgesamt kann dies Ihre Fähigkeit beeinträchtigen, auf Marktdynamik und Wettbewerbsdruck zu reagieren.

# Umstellung auf ein einheitliches Erlebnis

Um den Anforderungen dieses klassischen Softwaredilemmas gerecht zu werden, wenden sich Unternehmen einem Modell zu, das es ihnen ermöglicht, ein einziges, einheitliches Erlebnis zu schaffen, das es ihnen ermöglicht, Kunden gemeinsam zu verwalten und zu bedienen.

Das folgende Diagramm bietet eine konzeptionelle Ansicht einer Umgebung, in der alle Kunden über ein gemeinsames Modell verwaltet, aufgenommen, abgerechnet und betrieben werden.




Eine konzeptionelle Ansicht einer Umgebung, in der alle Kunden über ein gemeinsames Modell verwaltet, aufgenommen, abgerechnet und betrieben werden

Auf den ersten Blick scheint sich dies nicht allzu sehr vom Vorgängermodell zu unterscheiden. Wenn wir jedoch etwas näher darauf eingehen, werden Sie feststellen, dass es zwischen diesen beiden Ansätzen grundlegende, signifikante Unterschiede gibt.

Zunächst werden Sie feststellen, dass die Kundenumgebungen in Mandanten umbenannt wurden. Diese Vorstellung von einem Mandanten ist grundlegend für SaaS. Die Grundidee ist, dass Sie über eine einzige SaaS-Umgebung verfügen und jeder Ihrer Kunden als Mieter dieser Umgebung betrachtet wird und die Ressourcen verbraucht, die er benötigt. Ein Mandant kann ein Unternehmen mit vielen Benutzern sein, oder er könnte direkt mit einem einzelnen Benutzer korrelieren.

Um die Idee eines Mieters besser zu verstehen, sollten Sie die Idee von Wohn- oder Geschäftshäusern in Betracht ziehen. Die Fläche in jedem dieser Gebäude wird an einzelne Mieter

vermietet. Die Mieter verlassen sich auf einige der gemeinsam genutzten Ressourcen des Gebäudes (Wasser, Strom usw.) und zahlen für das, was sie verbrauchen.

SaaS-Mandanten folgen einem ähnlichen Muster. Sie haben die Infrastruktur Ihrer SaaS-Umgebung und Mandanten, die die Infrastruktur dieser Umgebung nutzen. Die Menge der von jedem Mandanten verbrauchten Ressourcen kann variieren. Diese Mieter werden ebenfalls gemeinsam verwaltet, abgerechnet und betrieben.

Wenn Sie zum Diagramm zurückkehren, werden Sie sehen, wie der Begriff des Mietverhältnisses zum Leben erweckt wird. Hier haben Mieter kein eigenes Umfeld mehr. Stattdessen werden alle Mieter innerhalb einer kollektiven SaaS-Umgebung untergebracht und verwaltet.

Das Diagramm enthält auch eine Reihe von gemeinsam genutzten Diensten, die Ihre SaaS-Umgebung umgeben. Diese Dienste stehen allen Mandanten Ihrer SaaS-Umgebung global zur Verfügung. Das bedeutet, dass beispielsweise Onboarding und Identität von allen Mietern dieser Umgebung gemeinsam genutzt werden. Das Gleiche gilt für Verwaltung, Betrieb, Bereitstellung, Abrechnung und Kennzahlen.

Diese Idee eines einheitlichen Satzes von Diensten, die universell auf alle Ihre Mandanten angewendet werden, ist grundlegend für SaaS. Indem Sie diese Konzepte teilen, können Sie eine Reihe der Herausforderungen angehen, die mit dem oben beschriebenen klassischen Modell verbunden waren.

Ein weiteres wichtiges, etwas subtiles Element dieses Diagramms ist, dass alle Mandanten in dieser Umgebung dieselbe Version Ihrer Anwendung ausführen. Die Idee, für jeden Kunden separate, einmalige Versionen laufen zu lassen, ist vorbei. Dass alle Mandanten dieselbe Version ausführen, ist eines der grundlegenden Unterscheidungsmerkmale einer SaaS-Umgebung.

Wenn alle Kunden dieselbe Version Ihres Produkts ausführen, stehen Sie nicht mehr vor vielen Herausforderungen, die ein klassisches installiertes Softwaremodell mit sich bringt. Im einheitlichen Modell können neue Funktionen über einen einzigen, gemeinsamen Prozess für alle Mandanten bereitgestellt werden.

Dieser Ansatz bietet Ihnen die Möglichkeit, eine zentrale zentrale Oberfläche zu verwenden, über die alle Mieter verwaltet und verwaltet werden können. Auf diese Weise können Sie Ihre Mieter anhand einer gemeinsamen Betriebserfahrung verwalten und überwachen, sodass neue Mieter hinzugefügt werden können, ohne zusätzlichen betrieblichen Aufwand zu verursachen. Dies ist ein zentraler Bestandteil des SaaS-Wertversprechens, das Teams die Möglichkeit gibt, die Betriebskosten zu senken und die allgemeine organisatorische Agilität zu verbessern.

Stellen Sie sich vor, was es bedeuten würde, in diesem Modell 100 oder 1.000 neue Kunden hinzuzufügen. Anstatt sich Gedanken darüber zu machen, wie diese neuen Kunden die Margen schmälern und die Komplexität erhöhen könnten, können Sie dieses Wachstum als die Chance betrachten, die es darstellt.

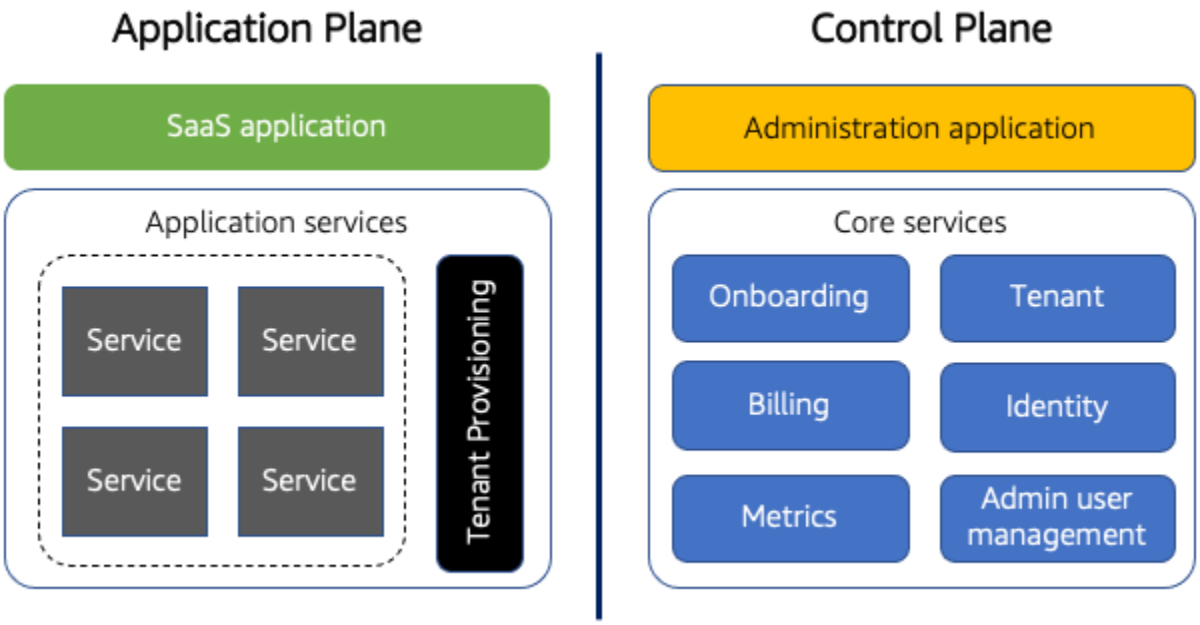
Im Allgemeinen liegt der Schwerpunkt von SaaS darauf, wie die Anwendung in der Mitte dieses Modells implementiert wird. Unternehmen möchten sich darauf konzentrieren, wie Daten gespeichert werden, wie Ressourcen gemeinsam genutzt werden usw. Die Realität ist jedoch, dass diese Details zwar auf jeden Fall wichtig sind, es jedoch viele Möglichkeiten gibt, Ihre Anwendung zu erstellen und sich Ihren Kunden dennoch als SaaS-Lösung zu präsentieren.

Entscheidend ist das umfassendere Ziel einer einzigen, einheitlichen Erfahrung, die Ihre Tenant-Umgebungen umgibt. Diese gemeinsame Erfahrung ermöglicht es Ihnen, das Wachstum, die Agilität und die betriebliche Effizienz voranzutreiben, die mit den Gesamtzielen eines SaaS-Unternehmens verbunden sind.

# Steuerungsebene im Vergleich zur Anwendungsebene

Das vorherige Diagramm bietet eine konzeptionelle Ansicht der Kernkonzepte der SaaS-Architektur. Lassen Sie uns nun näher darauf eingehen und besser definieren, wie sich Ihre SaaS-Umgebung in verschiedene Ebenen aufgeteilt hat. Dieses klarere Bild der Grenzen zwischen SaaS-Konzepten wird es einfacher machen, die beweglichen Teile einer SaaS-Lösung zu beschreiben.

Das folgende Diagramm unterteilt Ihre SaaS-Umgebung in zwei verschiedene Ebenen. Auf der rechten Seite befindet sich die Steuerungsebene. Diese Seite des Diagramms enthält alle Funktionen und Dienste, die für das Onboarding, die Authentifizierung, die Verwaltung, den Betrieb und die Analyse einer Multi-Tenant-Umgebung verwendet werden.




Steuerungsebene im Vergleich zur Anwendungsebene

Diese Steuerungsebene ist die Grundlage für jedes mehrinstanzenfähige SaaS-Modell. Jede SaaS-Lösung — unabhängig von der Anwendungsbereitstellung und dem Isolationsschema — muss die Services beinhalten, mit denen Sie Ihre Mandanten über ein einziges, einheitliches Erlebnis verwalten und betreiben können.

Innerhalb der Kontrollebene haben wir dies weiter in zwei verschiedene Elemente unterteilt. Die Kerndienste hier stellen die Sammlung von Diensten dar, die zur Orchestrierung Ihres Multi-Tenant-Erlebnisses verwendet werden. Wir haben einige der gängigen Beispiele für Dienste aufgeführt, die

normalerweise Teil des Kerns sind, wobei wir berücksichtigen, dass die Kerndienste für jede SaaS-Lösung etwas variieren können.

Sie werden auch feststellen, dass wir eine separate Administrationsanwendung zeigen. Dies stellt die Anwendung (eine Webanwendung, eine Befehlszeilenschnittstelle oder eine API) dar, die von einem SaaS-Anbieter zur Verwaltung seiner Multi-Tenant-Umgebung verwendet werden könnte.

Ein wichtiger Vorbehalt ist, dass es sich bei der Steuerungsebene und ihren Diensten nicht um mehrere Mandanten handelt. Die Funktionalität stellt nicht die tatsächlichen funktionalen Attribute Ihrer SaaS-Anwendung bereit (die mehrinstanzenfähig sein muss). Wenn Sie sich beispielsweise einen der Kerndienste ansehen, werden Sie keine Tenant Isolation und die anderen Konstrukte finden, die Teil Ihrer Multi-Tenant-Anwendungsfunktionalität sind. Diese Dienstleistungen stehen allen Mietern weltweit zur Verfügung.

Die linke Seite des Diagramms bezieht sich auf die Anwendungsebene einer SaaS-Umgebung. Hier befindet sich die Multi-Tenant-Funktionalität Ihrer Anwendung. Was im Diagramm erscheint, muss etwas vage bleiben, da jede Lösung je nach den Anforderungen Ihrer Domain, dem Platzbedarf Ihrer Technologie usw. unterschiedlich eingesetzt und aufgeteilt werden kann.

Die Anwendungsdomäne ist in zwei Elemente unterteilt. Es gibt die SaaS-Anwendung, die das Mandantenerlebnis/die Anwendung für Ihre Lösung darstellt. Dies ist die Oberfläche, die Mieter berühren, um mit Ihrer SaaS-Anwendung zu interagieren. Dann gibt es die Backend-Services, die die Geschäftslogik und die Funktionselemente einer SaaS-Lösung darstellen. Dies können Microservices oder andere Pakete Ihrer Anwendungsdienste sein.

Sie werden auch feststellen, dass wir die Bereitstellung aufgeteilt haben. Dies wird getan, um die Tatsache hervorzuheben, dass jede Bereitstellung von Ressourcen für Mieter während des Onboardings Teil dieser Anwendungsdomäne wäre. Einige könnten argumentieren, dass dies in die Steuerungsebene gehört. Wir haben es jedoch in die Anwendungsdomäne aufgenommen, da die Ressourcen, die es bereitstellen und konfigurieren muss, direkter mit Diensten verbunden sind, die auf der Anwendungsebene erstellt und konfiguriert werden.

Wenn Sie dies in verschiedene Ebenen aufteilen, ist es einfacher, über die Gesamtlandschaft einer SaaS-Architektur nachzudenken. Noch wichtiger ist, dass es die Notwendigkeit einer Reihe von Diensten unterstreicht, die völlig außerhalb des Funktionsumfangs Ihrer Anwendung liegen.

# Kernwerkfächer

Die zuvor erwähnte Steuerungsebene erwähnt eine Reihe von Kerndiensten, die die typischen Dienste darstellen, die für die Integration, Verwaltung und den Betrieb einer SaaS-Umgebung verwendet werden. Es kann hilfreich sein, die Rolle einiger dieser Dienste weiter hervorzuheben, um ihren Umfang und Zweck in einer SaaS-Umgebung hervorzuheben. Im Folgenden finden Sie eine kurze Zusammenfassung der einzelnen Dienste:

- Onboarding — Jede SaaS-Lösung muss einen reibungslosen Mechanismus für die Einführung neuer Mandanten in Ihre SaaS-Umgebung bieten. Dies kann eine Self-Service-Anmeldeseite oder ein intern verwaltetes Erlebnis sein. In jedem Fall sollte eine SaaS-Lösung alles tun, um interne und externe Probleme bei dieser Erfahrung zu beseitigen und Stabilität, Effizienz und Wiederholbarkeit für diesen Prozess zu gewährleisten. Es spielt eine wichtige Rolle bei der Unterstützung des Wachstums und der Skalierung eines SaaS-Unternehmens. Im Allgemeinen orchestriert dieser Dienst andere Dienste, um Benutzer-, Mandanten-, Isolationsrichtlinien, Bereitstellungs- und Mandantenressourcen zu erstellen.
- Mandant — Der Mandantenservice bietet eine Möglichkeit, die Richtlinien, Attribute und den Status von Mandanten zu zentralisieren. Der Schlüssel ist, dass Mieter keine einzelnen Benutzer sind. Tatsächlich ist ein Mandant wahrscheinlich mit vielen Benutzern verbunden.
- Identität — SaaS-Systeme benötigen eine klare Methode, um Benutzer mit Mandanten zu verbinden, die den Mandantenkontext in die Authentifizierungs- und Autorisierungserfahrung ihrer Lösungen einbringt. Dies wirkt sich sowohl auf das Onboarding-Erlebnis als auch auf die allgemeine Verwaltung von Benutzerprofilen aus.
- Abrechnung — Im Rahmen der Einführung von SaaS nutzen Unternehmen häufig neue Abrechnungsmodelle. Sie können auch die Integration mit externen Abrechnungsanbietern prüfen. Dieser Kerndienst konzentriert sich hauptsächlich auf die Unterstützung der Aufnahme neuer Mieter und die Erfassung von Verbrauchs- und Aktivitätsdaten, die zur Erstellung von Rechnungen für Mieter verwendet werden.
- Kennzahlen — SaaS-Teams verlassen sich in hohem Maße auf ihre Fähigkeit, umfangreiche Metrikdaten zu erfassen und zu analysieren, die mehr Transparenz darüber bieten, wie Mieter ihr System nutzen, wie sie Ressourcen verbrauchen und wie ihre Mieter ihre Systeme nutzen. Diese Daten werden zur Gestaltung von Betriebs-, Produkt- und Geschäftsstrategien verwendet.
- Admin-Benutzerverwaltung — SaaS-Systeme müssen sowohl Tenant-Benutzer als auch Admin-Benutzer unterstützen. Die Admin-Benutzer repräsentieren die Administratoren eines SaaS-

Anbieters. Sie werden sich in Ihre Betriebserfahrung einloggen, um Ihre SaaS-Umgebung zu überwachen und zu verwalten.

# Mehrmandantenfähigkeit neu definiert

Die Begriffe Multi-Tenancy und SaaS sind oft eng miteinander verknüpft. In einigen Fällen beschreiben Unternehmen SaaS und Multi-Tenancy als dasselbe. Dies mag zwar natürlich erscheinen, aber die Gleichsetzung von SaaS und Multi-Tenancy führt dazu, dass Teams SaaS eher aus technischer Sicht betrachten, obwohl SaaS in Wirklichkeit eher ein Geschäftsmodell als eine Architekturstrategie ist.

Um dieses Konzept besser zu verstehen, beginnen wir mit der klassischen Ansicht von Multi-Tenancy. In dieser rein infrastrukturorientierten Sichtweise wird Multi-Tenancy verwendet, um zu beschreiben, wie Ressourcen von Mietern gemeinsam genutzt werden, um Agilität und Kosteneffizienz zu fördern.

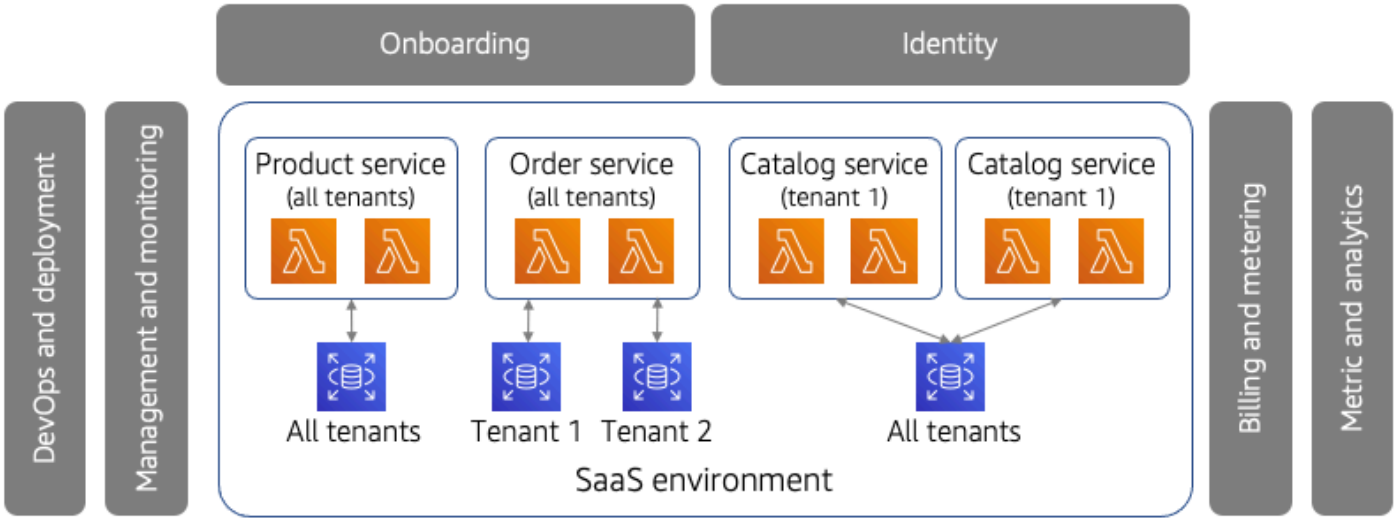
Nehmen wir zum Beispiel an, Sie haben einen Microservice oder eine Amazon Elastic Compute Cloud (Amazon EC2) -Instance, die von mehreren Mandanten Ihres SaaS-Systems genutzt wird. Es würde davon ausgegangen werden, dass dieser Service in einem Mehrmandantenmodell ausgeführt wird, da die Mandanten die Infrastruktur, auf der dieser Service ausgeführt wird, gemeinsam nutzen.

Die Herausforderung dieser Definition besteht darin, dass sie den technischen Begriff der Mehrmandantenfähigkeit zu direkt mit SaaS verknüpft. Es wird davon ausgegangen, dass das entscheidende Merkmal von SaaS darin besteht, dass es über eine gemeinsam genutzte, mandantenfähige Infrastruktur verfügen muss. Diese Auffassung von SaaS gerät allmählich auseinander, wenn wir uns die verschiedenen Arten ansehen, wie SaaS in verschiedenen Umgebungen realisiert wird.

Das folgende Diagramm bietet einen Überblick über ein SaaS-System, das einige der Herausforderungen aufzeigt, mit denen wir bei der Definition von Mehrmandantenfähigkeit konfrontiert sind.

Hier sehen Sie das zuvor beschriebene klassische SaaS-Modell mit einer Reihe von Anwendungsdiensten, die von Shared Services umgeben sind und es Ihnen ermöglichen, Ihre Mandanten gemeinsam zu verwalten und zu betreiben.

Neu sind die Microservices, die wir aufgenommen haben. Das Diagramm enthält drei Beispiele für Microservices: Produkt, Bestellung und Katalog. Wenn Sie sich das Mietmodell der einzelnen Dienste genau ansehen, werden Sie feststellen, dass sie alle leicht unterschiedliche Mietverhältnisse verwenden.




SaaS und Mehrmandantenfähigkeit

Der Produktservice teilt alle seine Ressourcen (Rechenleistung und Speicher) mit allen Mandanten. Dies entspricht der klassischen Definition von Mehrmandantenfähigkeit. Wenn Sie sich jedoch den Bestellservice ansehen, werden Sie feststellen, dass er über gemeinsam genutzte Rechenleistung verfügt, aber über separaten Speicher für jeden Mandanten verfügt.

Der Katalogdienst fügt eine weitere Variante hinzu, bei der die Rechenleistung für jeden Mandanten separat ist (eine separate Microservice-Bereitstellung für jeden Mandanten), der Speicher jedoch für alle Mandanten gemeinsam genutzt wird.

Variationen dieser Art sind in SaaS-Umgebungen üblich. Lauter Nachbar, Tiering-Modelle, Isolationsanforderungen — dies sind nur einige der Gründe, warum Sie Teile Ihrer SaaS-Lösung selektiv teilen oder isolieren könnten.

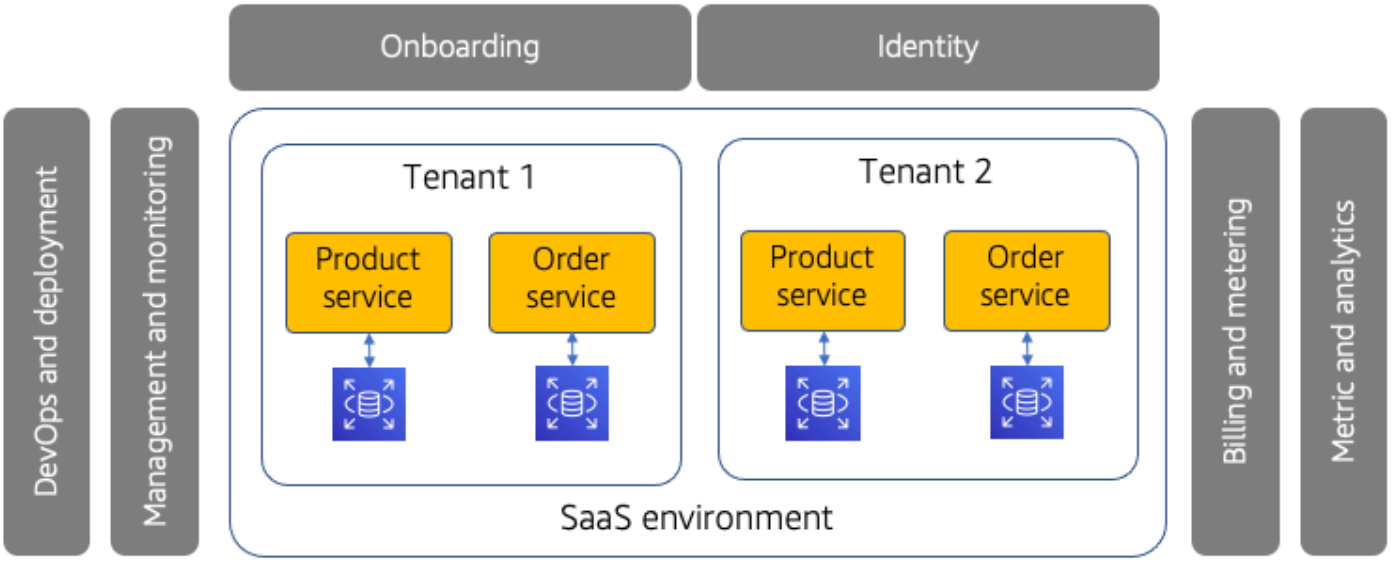
Angesichts dieser Variationen — und vieler anderer Möglichkeiten — wird es schwieriger, herauszufinden, wie Sie den Begriff Multi-Tenant verwenden sollten, um diese Umgebung zu charakterisieren. Insgesamt handelt es sich, soweit es den Kunden betrifft, um eine Umgebung mit mehreren Mandanten. Wenn wir jedoch die technischste Definition verwenden, sind einige Teile dieser Umgebung mandantenfähig und andere nicht.

Aus diesem Grund wird es notwendig, den Begriff Multi-Tenant nicht mehr zur Charakterisierung von SaaS-Umgebungen zu verwenden. Stattdessen können wir darüber sprechen, wie Mehrmandantenfähigkeit in Ihrer Anwendung implementiert ist, aber vermeiden, es zu verwenden, um eine Lösung als SaaS zu charakterisieren. Wenn der Begriff Multi-Tenant verwendet werden

soll, ist es sinnvoller, damit die gesamte SaaS-Umgebung als mandantenfähig zu beschreiben, da man weiß, dass einige Teile der Architektur gemeinsam genutzt werden können und andere nicht. Insgesamt betreiben und verwalten Sie diese Umgebung immer noch in einem Mehrmandantenmodell.

## Der Extremfall

Um diesen Begriff des Mietverhältnisses besser hervorzuheben, schauen wir uns ein SaaS-Modell an, bei dem sich Mandanten keine Ressourcen teilen. Das folgende Diagramm zeigt ein Beispiel für eine SaaS-Umgebung, die von einigen SaaS-Anbietern verwendet wird.




Stapel pro Mandant

In diesem Diagramm sehen Sie, dass wir immer noch unsere gemeinsame Umgebung rund um diese Mandanten haben. Jeder Mandant wird jedoch mit einer speziellen Sammlung von Ressourcen bereitgestellt. In diesem Modell wird nichts von den Mandanten gemeinsam genutzt.

In diesem Beispiel wird hinterfragt, was es bedeutet, mehrere Mandanten zu haben. Handelt es sich um eine Umgebung mit mehreren Mandanten, obwohl keine der Ressourcen gemeinsam genutzt wird? Die Mandanten, die dieses System nutzen, haben dieselben Erwartungen, die Sie von einer SaaS-Umgebung mit gemeinsam genutzten Ressourcen haben würden. Tatsächlich sind sie sich möglicherweise nicht bewusst, wie ihre Ressourcen unter der Haube der SaaS-Umgebung eingesetzt werden.

Obwohl diese Mandanten in einer isolierten Infrastruktur betrieben werden, werden sie dennoch gemeinsam verwaltet und betrieben. Sie verfügen über ein einheitliches Onboarding-, Identitäts-, Metriken-, Abrechnungs- und Betriebserlebnis. Wenn eine neue Version veröffentlicht wird, wird sie außerdem für alle Mandanten bereitgestellt. Damit dies funktioniert, können Sie keine einmaligen Anpassungen für einzelne Mandanten zulassen.

Dieses extreme Beispiel bietet ein gutes Modell, um das Konzept von Multi-Tenant-SaaS zu testen. Auch wenn sie möglicherweise nicht alle Effizienzen einer gemeinsam genutzten Infrastruktur nutzt, handelt es sich um eine absolut gültige mehrinstanzenfähige SaaS-Umgebung. Für einige Kunden kann es aufgrund ihrer Domain erforderlich sein, dass einige oder alle Kunden dieses Modell verwenden. Das heißt nicht, dass sie keine SaaS sind. Wenn sie diese gemeinsamen Dienste verwenden und alle Mandanten dieselbe Version ausführen, entspricht dies immer noch den Grundprinzipien von SaaS.

Angesichts dieser Parameter und dieser umfassenderen Definition von SaaS können Sie die Notwendigkeit erkennen, die Verwendung des Begriffs Multi-Tenant weiterzuentwickeln. Es ist sinnvoller, jedes SaaS-System, das gemeinsam verwaltet und betrieben wird, als mandantenfähig zu bezeichnen. Anschließend können Sie auf eine detailliertere Terminologie zurückgreifen, um zu beschreiben, wie Ressourcen bei der Implementierung einer SaaS-Lösung gemeinsam genutzt oder zugewiesen werden.

# Streichung des Begriffs „Einzelmandant“

Im Rahmen der Verwendung des Begriffs Multi-Tenant ist es nur natürlich, dass Benutzer den Begriff Single-Tenant verwenden möchten, um SaaS-Umgebungen zu beschreiben. Vor dem zuvor skizzierten Hintergrund sorgt der Begriff Single-Tenant jedoch für Verwirrung.

Handelt es sich bei dem obigen Diagramm um eine Single-Tenant-Umgebung? Technisch gesehen hat zwar jeder Mandant seinen eigenen Stack, aber diese Mandanten werden immer noch in einem Multi-Tenant-Modell betrieben und verwaltet. Aus diesem Grund wird der Begriff Single-Tenant generell vermieden. Stattdessen werden alle Umgebungen als Multi-Tenant-Umgebungen bezeichnet, da sie lediglich eine Variante der Tenancy implementieren, bei der einige oder alle Ressourcen gemeinsam genutzt oder dediziert werden.

# Wir stellen vor: Silo und Pool

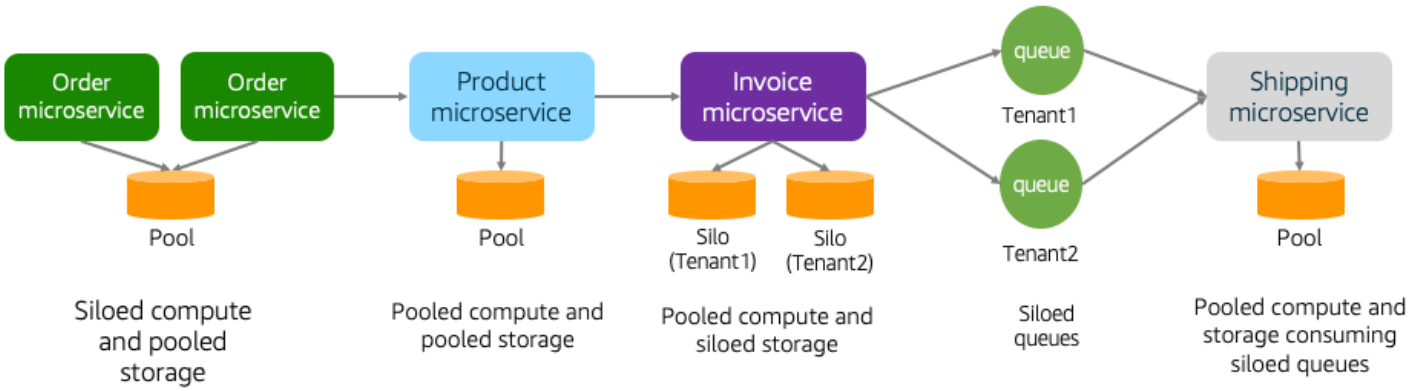
Angesichts all dieser Modellvarianten und angesichts der Herausforderungen rund um den Begriff Multi-Tenancy haben wir einige Begriffe eingeführt, mit denen wir die verschiedenen Modelle, die beim Aufbau eines SaaS verwendet werden, genauer erfassen und beschreiben können.

Zwei Begriffe, die wir verwenden, um die Nutzung von Ressourcen in einer SaaS-Umgebung zu charakterisieren, sind Silo und Pool. Diese Begriffe ermöglichen es uns, die Natur von SaaS-Umgebungen zu kennzeichnen, indem wir Multi-Tenant als übergreifende Beschreibung verwenden, die auf eine beliebige Anzahl von zugrunde liegenden Modellen angewendet werden kann.

Auf der einfachsten Ebene soll der Begriff Silo Szenarien beschreiben, in denen eine Ressource einem bestimmten Mandanten zugewiesen wird. Umgekehrt wird das Poolmodell verwendet, um Szenarien zu beschreiben, in denen eine Ressource von Mietern gemeinsam genutzt wird.

Wenn wir uns ansehen, wie die Begriffe Silo und Pool verwendet werden, ist es wichtig, sich darüber im Klaren zu sein, dass Silo und Pool keine all-or-nothing Konzepte sind. Silo und Pool könnten für den Ressourcenstapel eines gesamten Mandanten gelten, oder sie könnten selektiv auf Teile Ihrer gesamten SaaS-Umgebung angewendet werden. Wenn wir also sagen, dass eine Ressource ein Silomodell verwendet, bedeutet das nicht, dass alle Ressourcen in dieser Umgebung isoliert sind. Das Gleiche gilt für die Verwendung des Begriffs gepoolt.

Das folgende Diagramm zeigt ein Beispiel dafür, wie isolierte und gepoolte Modelle in einer SaaS-Umgebung detaillierter verwendet werden:




Silo- und Poolmodelle

Dieses Diagramm enthält eine Reihe von Beispielen, die den gezielteren Charakter der Silo- und Poolmodelle veranschaulichen sollen. Wenn Sie dies von links nach rechts verfolgen, werden Sie feststellen, dass wir mit einem Bestell-Microservice beginnen. Dieser Microservice verfügt über isolierte Rechenleistung und gepoolten Speicher. Es interagiert mit einem Produktservice, der über gepoolte Rechenleistung und gepoolten Speicher verfügt.

Der Produktservice interagiert dann mit einem Rechnungs-Microservice, der über gepoolte Rechenleistung und isolierten Speicher verfügt. Dieser Dienst sendet Nachrichten über Warteschlangen an den Versandservice. Die Warteschlangen werden in einem Silomodell bereitgestellt.

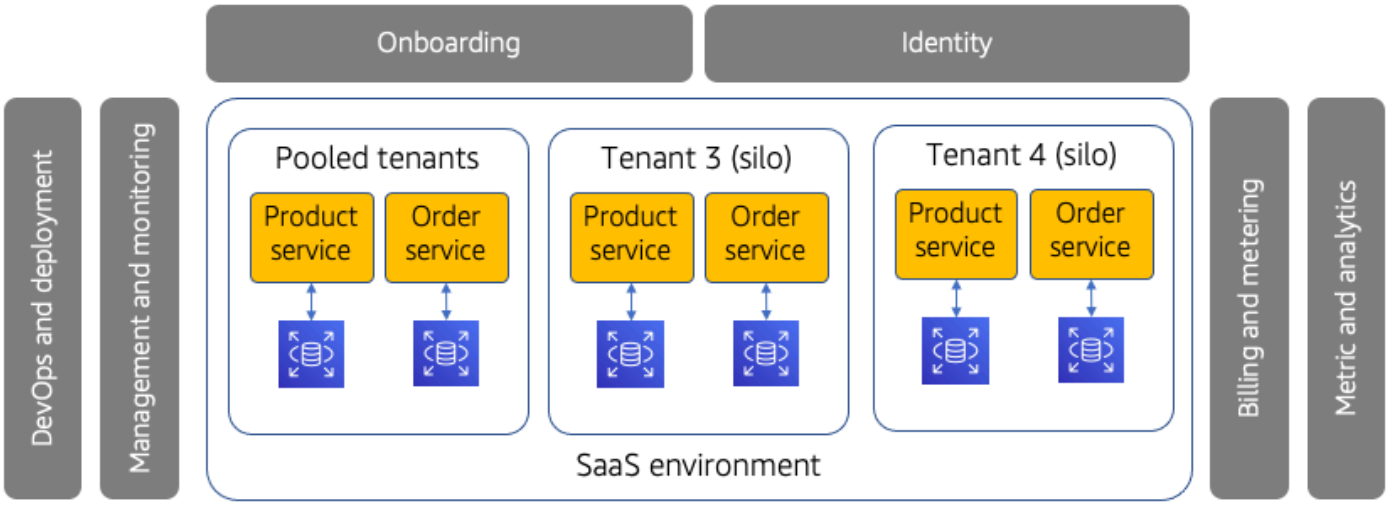
Schließlich erfasst der Versand-Microservice Nachrichten aus den isolierten Warteschlangen. Es verwendet gepoolte Rechen- und Speicherressourcen.

Dies mag zwar etwas kompliziert erscheinen, aber das Ziel besteht darin, den granularen Charakter der Silo- und Poolkonzepte hervorzuheben. Beim Entwerfen und Erstellen Ihrer SaaS-Lösung wird erwartet, dass Sie diese Silo- und Poolentscheidungen auf der Grundlage der Bedürfnisse Ihrer Domain und Ihrer Kunden treffen.

Lauter Nachbar, Isolation, Tiering und eine Vielzahl anderer Gründe können beeinflussen, wie und wann Sie das Silo- oder Poolmodell anwenden.

# Silo und Pool für den kompletten Stapel

Silo und Pool können auch verwendet werden, um einen gesamten SaaS-Stack zu beschreiben. Bei diesem Ansatz werden alle Ressourcen für einen Mandanten entweder dediziert oder gemeinsam genutzt. Das folgende Diagramm zeigt ein Beispiel dafür, wie dies in einer SaaS-Umgebung landen könnte.




Silo- und Poolmodelle für den kompletten Stack

In diesem Diagramm sehen Sie, dass es drei verschiedene Modelle für Ihre Full-Stack-Tenant-Bereitstellungen gibt. Zunächst werden Sie feststellen, dass es eine Full-Stack-Pool-Umgebung gibt. Die Mandanten in diesem Pool teilen sich alle Ressourcen (Rechenleistung, Speicher usw.).

Die anderen beiden abgebildeten Stacks sollen Silo-Tenant-Umgebungen mit vollem Stack darstellen. In diesem Fall werden Mandant 3 und Tenant 4 so dargestellt, dass sie jeweils ihre eigenen dedizierten Stacks haben, bei denen keine der Ressourcen mit anderen Mandanten gemeinsam genutzt wird.

Diese Mischung aus Silo- und Poolmodellen in derselben SaaS-Umgebung ist gar nicht so untypisch. Stellen Sie sich zum Beispiel vor, Sie haben eine Reihe von Mandanten der Basisstufe, die einen moderaten Preis für die Nutzung Ihres Systems zahlen. Diese Mieter werden in der gepoolten Umgebung untergebracht.

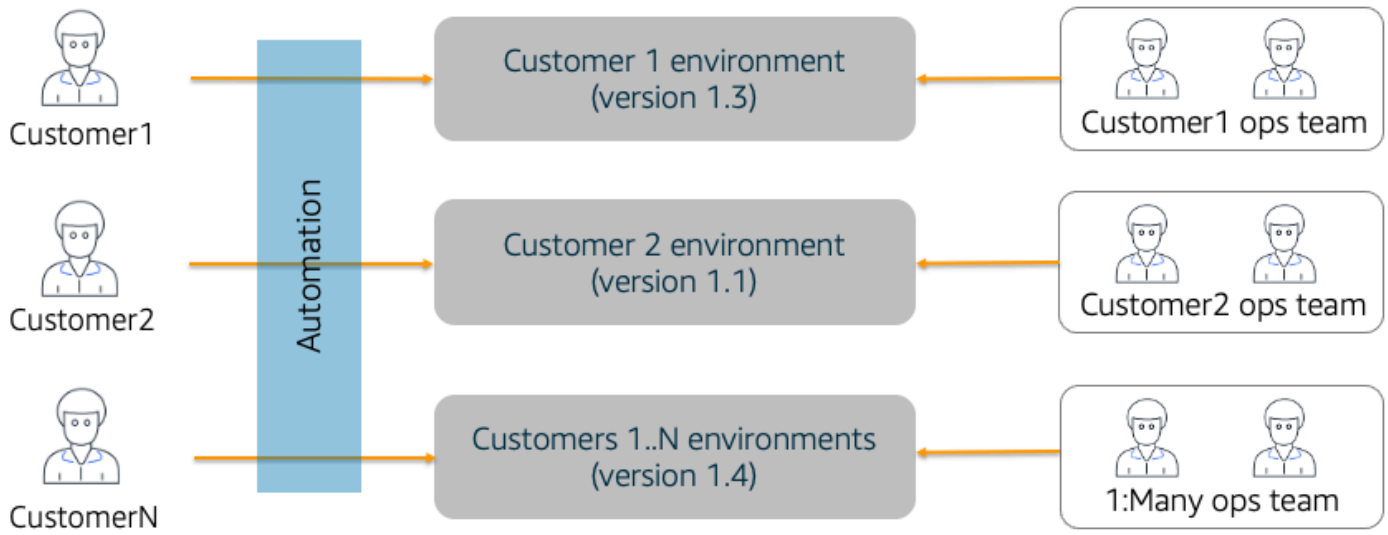
In der Zwischenzeit haben Sie möglicherweise auch Premium-Mieter, die bereit sind, mehr für das Privileg zu zahlen, in einem Silo zu arbeiten. Diese Kunden werden mit separaten Stacks bereitgestellt (wie in der Abbildung dargestellt).

Selbst in diesem Modell, bei dem Sie Mandanten möglicherweise erlaubt haben, in ihrem eigenen Full-Stack-Silo zu arbeiten, wäre es wichtig, dass diese Silos keine einmaligen Variationen oder Anpassungen für diese Mandanten zulassen. In jeder Hinsicht sollte auf jedem dieser Stacks dieselbe Konfiguration des Stacks mit derselben Version der Software ausgeführt werden. Wenn eine neue Version veröffentlicht wird, wird sie in der gepoolten Mandantenumgebung und in jeder der isolierten Umgebungen bereitgestellt.

# SaaS im Vergleich zu Managed Service Provider (MSP)

Es gibt auch einige Verwirrung, was die Grenzen zwischen SaaS- und Managed Service Provider (MSP) -Modellen angeht. Wenn Sie sich ein MSP-Modell ansehen, kann es den Anschein haben, dass es ähnliche Ziele verfolgt wie das SaaS-Modell.

Wenn Sie sich jedoch etwas mehr mit MSP befassen, werden Sie feststellen, dass MSP und SaaS tatsächlich unterschiedlich sind. Das folgende Diagramm bietet eine konzeptionelle Ansicht einer MSP-Umgebung.




Modell eines Managed Service Providers (MSP)

Dieses Diagramm stellt einen Ansatz für das MSP-Modell dar. Auf der linken Seite sehen Sie Kunden, die das MSP-Modell verwenden. Im Allgemeinen würde hier der Ansatz darin bestehen, jede verfügbare Automatisierung zu verwenden, um jede Kundenumgebung bereitzustellen und die Software für diesen Kunden zu installieren.

Auf der rechten Seite finden Sie eine ungefähre Darstellung des betrieblichen Fußabdrucks, den der MSP für die Unterstützung dieser Kundenumgebungen bereitstellen würde.

Es ist wichtig zu beachten, dass der MSP häufig eine Version des Produkts installiert und verwaltet, das ein bestimmter Kunde ausführen möchte. Möglicherweise verwenden alle Kunden dieselbe Version, dies ist jedoch in einem MSP-Modell normalerweise nicht erforderlich.

Die allgemeine Strategie besteht darin, das Leben eines Softwareanbieters zu vereinfachen, indem die Installation und Verwaltung dieser Umgebungen selbst in die Hand genommen werden. Dies erleichtert dem Anbieter zwar das Leben, entspricht aber nicht direkt den Werten und der Denkweise, die für ein SaaS-Angebot unerlässlich sind.

Der Schwerpunkt liegt auf der Verlagerung der Managementverantwortung. Dieser Schritt ist nicht gleichbedeutend damit, dass alle Kunden dieselbe Version mit einer einzigen, einheitlichen Verwaltungs- und Betriebserfahrung verwenden. Stattdessen lässt MSP oft separate Versionen zu und behandelt jede dieser Umgebungen oft als betrieblich separate Umgebungen.

Es gibt sicherlich Bereiche, in denen sich MSP mit SaaS zu überschneiden beginnen könnte. Wenn der MSP im Wesentlichen verlangen würde, dass alle Kunden dieselbe Version ausführen und der MSP in der Lage wäre, alle Mandanten über ein einziges Erlebnis zentral zu integrieren, zu verwalten, zu betreiben und zu fakturieren, könnte das eher SaaS als MSP sein.

Das umfassendere Thema ist, dass die Automatisierung der Installation von Umgebungen nicht gleichbedeutend mit einer SaaS-Umgebung ist. Erst wenn Sie alle anderen zuvor besprochenen Vorbehalte hinzufügen, würde dies eher ein echtes SaaS-Modell darstellen.

Wenn wir uns von den technologischen und betrieblichen Aspekten dieser Geschichte entfernen, wird die Grenze zwischen MSP und SaaS noch deutlicher. Im Allgemeinen hängt der Erfolg Ihres Angebots als SaaS-Unternehmen von Ihrer Fähigkeit ab, tief in alle wichtigen Teile des Erlebnisses eingebunden zu sein.

Dies bedeutet in der Regel, dass Sie bei der Onboarding-Erfahrung am Puls der Zeit sind, verstehen, wie sich betriebliche Ereignisse auf Mieter auswirken, wichtige Kennzahlen und Analysen verfolgen und Ihren Kunden nahe sein. In einem MSP-Modell, bei dem dies an eine andere Person übergeben wird, kann es sein, dass Sie sich auf einer Ebene befinden, die von den wichtigsten Details entfernt ist, die für den Betrieb eines SaaS-Unternehmens von zentraler Bedeutung sind.

# Migration über SaaS

Viele der Anbieter, die SaaS einführen, migrieren von einem traditionellen installierten Softwaremodell (zuvor beschrieben) zu SaaS. Für diese Anbieter ist es besonders wichtig, die Kernprinzipien von SaaS gut aufeinander abzustimmen.

Auch hier kann es zu einiger Verwirrung darüber kommen, was es bedeutet, zu einem SaaS-Modell zu migrieren. Manche betrachten beispielsweise die Umstellung auf die Cloud als Migration zu SaaS. Andere betrachten das Hinzufügen von Automatisierung zu ihrem Installations- und Bereitstellungsprozess als Erfolg der Migration.

Man kann mit Fug und Recht sagen, dass jedes Unternehmen an einem anderen Ort anfängt, unterschiedliche Überlegungen zur Altlasten anstellt und wahrscheinlich einem unterschiedlichen Markt- und Wettbewerbsdruck ausgesetzt ist. Das bedeutet, dass jede Migration anders aussehen wird.

Obwohl jeder Weg anders ist, gibt es einige Bereiche, in denen es Unterschiede in Bezug auf die Kernprinzipien gibt, die die Migrationsstrategien prägen. Eine gute Abstimmung der Konzepte und Prinzipien kann erhebliche Auswirkungen auf den Gesamterfolg Ihrer SaaS-Migration haben.

Auf der Grundlage der zuvor skizzierten Konzepte sollte klar sein, dass die Umstellung auf SaaS mit der Geschäftsstrategie und den Unternehmenszielen beginnt. Dieser Punkt kann in einer Migrationsumgebung verloren gehen, in der der Druck besteht, so schnell wie möglich zu SaaS zu gelangen.

In diesem Modus betrachten Unternehmen die Migration oft in erster Linie als technische Maßnahme. Die Realität ist, dass jede SaaS-Migration mit einem klaren Überblick über die Zielkunden, das Serviceerlebnis, die betrieblichen Ziele usw. beginnen sollte. Ein klarerer Fokus darauf, wie Ihr SaaS-Geschäft aussehen muss, hat tiefgreifende Auswirkungen auf die Form, die Prioritäten und den Weg, den Sie bei der Migration Ihrer Lösung auf SaaS einschlagen.

Wenn Sie von Beginn Ihrer Migration an diese klare Vision haben, legen Sie die Grundlage dafür, wie Sie im Rahmen Ihrer Umstellung auf SaaS sowohl die Technologie als auch das Geschäft migrieren. Wenn Sie sich auf diesen Weg begeben, konzentrieren Sie sich auf die Fragen, die Ihnen am meisten darüber sagen können, wohin Sie wollen.

Die folgende Tabelle gibt einen Überblick über den unterschiedlichen Charakter der Denkweisen zur technischen Migration und zur Unternehmensmigration.

Tabelle 1 — Migration an erster Stelle in technischer Hinsicht im Vergleich zur unternehmenseigenen Migration

| Die Denkweise, bei der Technik an erster Stelle steht | Die betriebswirtschaftliche Denkweise |
|---|---|
| Wie isolieren wir Mandantendaten? | Wie kann SaaS uns helfen, unser Geschäft auszubauen? |
| Wie verbinden wir Benutzer mit Mietern? | Auf welche Segmente zielen wir ab? |
| Wie vermeiden wir laute Nachbarschaftsbedingungen? | Was ist die Größe und das Profil dieser Segmente? |
| Wie führen wir A/B-Tests durch? | Welche Stufen müssen wir unterstützen? |
| Wie skalieren wir basierend auf der Auslastung der Mieter? | Welches Serviceerlebnis streben wir an? |
| Welchen Abrechnungsanbieter sollten wir verwenden? | Was ist unsere Preis- und Verpackungsstrategie? |

Im Folgenden finden Sie ein Beispiel dafür, wie eine Migration über Technologie aussehen könnte. Das Entwicklungsteam konzentriert sich sehr darauf, die klassischen Multi-Tenant-Themen zu verfolgen, die für jede SaaS-Architektur sicherlich wichtig sind.

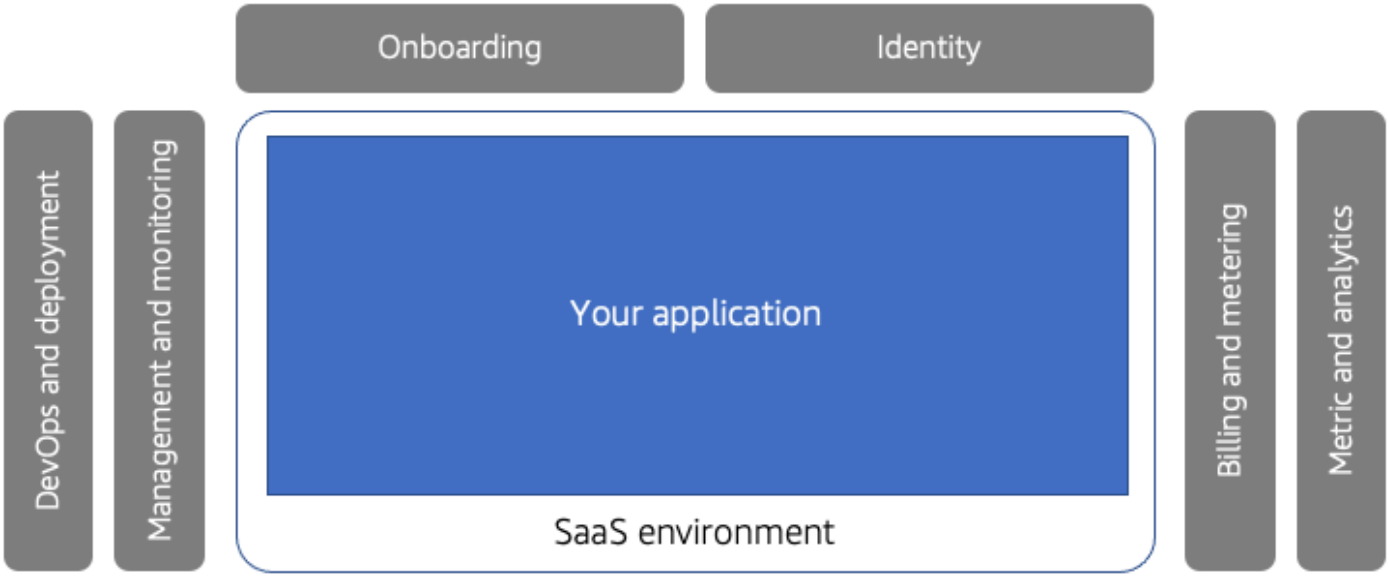
Das Problem ist, dass die Antworten auf viele der Fragen auf der linken Seite oft direkt von den Antworten auf die Frage auf der rechten Seite beeinflusst werden. Es ist unwahrscheinlich, dass dieser Punkt für jemanden neu ist, der sich mit Migration befasst. Die Realität ist jedoch, dass viele Unternehmen zunächst der Betriebs- und Kosteneffizienz als ersten Schritt nachjagen, vorausgesetzt, dass sich die geschäftlichen Aspekte von selbst regeln.

Im Rahmen dieser Migrationsstrategie kann es auch zu Unklarheiten darüber kommen, wie Ihre Legacy-Umgebung weiterentwickelt werden könnte, um in ein SaaS-Modell zu passen. Auch in diesem Bereich gibt es eine Vielzahl von Optionen für die Migration zu SaaS. Es gibt jedoch ein gemeinsames Wertesystem, das wir generell für jede Migration befürworten.

In unserer vorherigen Diskussion der SaaS-Prinzipien haben wir verschiedene Muster und Terminologien skizziert, die zur Beschreibung von SaaS-Umgebungen verwendet werden. Ein

gemeinsames Thema, das all diese Lösungen durchzieht, ist die Idee, gemeinsam genutzte Dienste zu haben, die Ihre Anwendung umgeben. Identität, Onboarding, Metriken, Abrechnung — all dies werden als gemeinsame Elemente bezeichnet, die für jede SaaS-Umgebung von zentraler Bedeutung sind.

Wenn wir uns nun die Migration ansehen, werden Sie feststellen, dass dieselben Shared Services in jeder Migrationsgeschichte eine wichtige Rolle spielen. Das folgende Diagramm bietet eine konzeptionelle Ansicht der Migrationslandschaft.




Migration zu SaaS

Dieses Diagramm stellt das Zielerlebnis für jeden Migrationspfad dar. Es umfasst dieselben gemeinsamen Dienste, die zuvor beschrieben wurden. In der Mitte befindet sich ein Platzhalter für Ihre Bewerbung.

Die Grundidee ist, dass Sie inmitten dieser Umgebung beliebig viele Anwendungsmodelle platzieren können. In Ihrem ersten Schritt der Migration wird möglicherweise jeder Tenant in seinem eigenen Silo ausgeführt. Oder Sie haben vielleicht eine Hybridarchitektur, in der Elemente isoliert sind und andere Funktionalitäten durch eine Sammlung modernisierter Microservices abgedeckt werden.

Wie stark Sie Ihre Anwendung zunächst modernisieren, hängt von der Art Ihrer Legacy-Umgebung, den Marktanforderungen, Kostenüberlegungen usw. ab. Was sich nicht ändert, ist die Einführung dieser Shared Services.

Jede SaaS-Migration muss diese grundlegenden Shared Services unterstützen, damit Ihr Unternehmen in einem SaaS-Modell arbeiten kann. Alle Variationen der Anwendungsarchitektur benötigen beispielsweise eine SaaS-Identität. Für die Verwaltung und Überwachung Ihrer SaaS-Lösung benötigen Sie Prozesse, bei denen die Mandanten berücksichtigt werden.

Wenn Sie diese Shared Services an der Spitze Ihrer Migration einsetzen, können Sie Ihren Kunden ein SaaS-Erlebnis bieten — auch wenn die zugrunde liegende Anwendung immer noch in einem Full-Stack-Silo für jeden Mandanten ausgeführt wird.

Das allgemeine Ziel besteht darin, Ihre Anwendung in einem SaaS-Modell zum Laufen zu bringen. Anschließend können Sie sich der weiteren Modernisierung und Verfeinerung Ihrer Anwendung zuwenden. Dieser Ansatz ermöglicht es Ihnen auch, die anderen Bereiche Ihres Unternehmens (Marketing, Vertrieb, Support usw.) schneller voranzubringen. Noch wichtiger ist, dass Sie auf diese Weise beginnen können, Kundenfeedback einzuholen und zu sammeln, das zur Gestaltung der laufenden Modernisierung Ihrer Umgebung verwendet werden kann.

Es ist wichtig zu beachten, dass die von Ihnen bereitgestellten Shared Services möglicherweise nicht alle Funktionen oder Mechanismen beinhalten, die Sie letztendlich benötigen. Das Hauptziel besteht darin, die gemeinsamen Mechanismen zu schaffen, die zu Beginn Ihrer Migration erforderlich sind. Auf diese Weise können Sie sich auf die Elemente des Systems konzentrieren, die für die Entwicklung Ihrer Anwendungsarchitektur und Ihre betriebliche Entwicklung von entscheidender Bedeutung sind.

# SaaS-Identität

SaaS-Lösungen fügen eine -Datei zum Identitätsmodell der Anwendung hinzu. Da jeder Benutzer authentifiziert ist, muss er mit einem bestimmten Mandantenkontext verbunden sein. Dieser Mandantenkontext enthält wichtige Informationen über Ihren Mandanten, die in Ihrer SaaS-Umgebung verwendet werden.

Diese Bindung von Mandanten an Benutzer wird oft als SaaS-Identität Ihrer Anwendung bezeichnet. Wenn sich jeder Benutzer authentifiziert, gibt Ihr Identitätsanbieter in der Regel ein Token aus, das sowohl die Benutzeridentität als auch die Mandantenidentität enthält.

Die Verbindung von Mandanten mit Benutzern ist ein grundlegender Aspekt Ihrer SaaS-Architektur, der viele nachgeordnete Auswirkungen hat. Das Token aus diesem Identitätsprozess fließt in die Microservices Ihrer Anwendung und wird verwendet, um mandantenorientierte Protokolle zu erstellen, Kennzahlen aufzuzeichnen, die Abrechnung zu messen, die Mandantenisolierung durchzusetzen usw.

Es ist wichtig, dass Sie Szenarien vermeiden, die auf separaten, eigenständigen Mechanismen beruhen, die Benutzer Mandanten zuordnen. Dies kann die Sicherheit Ihres Systems untergraben und führt häufig zu Engpässen in Ihrer Architektur.

# Mandantenisolierung

Je mehr Kunden auf ein Mehrmandantenmodell umsteigen, desto mehr werden sie sich Gedanken darüber machen, ob ein Mandant möglicherweise auf die Ressourcen eines anderen Mandanten zugreifen kann. SaaS-Systeme beinhalten explizite Mechanismen, die sicherstellen, dass die Ressourcen jedes Mandanten — auch wenn sie auf einer gemeinsam genutzten Infrastruktur laufen — isoliert sind.

Dies bezeichnen wir als Mandantenisolierung. Die Idee hinter der Mandantenisolierung ist, dass Ihre SaaS-Architektur Konstrukte einführt, die den Zugriff auf Ressourcen streng kontrollieren und jeden Versuch blockieren, auf Ressourcen eines anderen Mandanten zuzugreifen.

Beachten Sie, dass die Mandantenisolierung von den allgemeinen Sicherheitsmechanismen getrennt ist. Ihr System unterstützt Authentifizierung und Autorisierung. Die Tatsache, dass ein Mandantenbenutzer authentifiziert ist, bedeutet jedoch nicht, dass Ihr System isoliert ist. Die Isolierung wird getrennt von der grundlegenden Authentifizierung und Autorisierung angewendet, die Teil Ihrer Anwendung sein können.

Um dies besser zu verstehen, stellen Sie sich vor, Sie haben einen Identitätsanbieter verwendet, um den Zugriff auf Ihr SaaS-System zu authentifizieren. Das Token aus dieser Authentifizierungserfahrung kann auch Informationen über die Rolle eines Benutzers enthalten, mit denen der Zugriff dieses Benutzers auf eine bestimmte Anwendungsfunktion gesteuert werden kann. Diese Konstrukte bieten Sicherheit, aber keine Isolation. Tatsächlich könnte ein Benutzer authentifiziert und autorisiert werden und trotzdem auf die Ressourcen eines anderen Mandanten zugreifen. Nichts an Authentifizierung und Autorisierung wird diesen Zugriff unbedingt blockieren.

Die Mandantenisolierung konzentriert sich ausschließlich auf die Verwendung des Mandantenkontextes, um den Zugriff auf Ressourcen einzuschränken. Es wertet den Kontext des aktuellen Mandanten aus und bestimmt anhand dieses Kontextes, welche Ressourcen für diesen Mandanten zugänglich sind. Diese Isolierung wird für alle Benutzer innerhalb dieses Mandanten angewendet.

Dies wird schwieriger, wenn wir uns ansehen, wie die Mandantenisolierung in all den verschiedenen SaaS-Architekturmustern realisiert wird. In einigen Fällen kann eine Isolierung erreicht werden, indem ganze Stapel von Ressourcen einem Mandanten zugewiesen werden, wobei Netzwerkrichtlinien (oder grobkörnere) den mandantenübergreifenden Zugriff verhindern. In anderen Szenarien haben Sie möglicherweise Ressourcen (Elemente in einer Amazon DynamoDB-Tabelle) gebündelt, für die genauere Richtlinien zur Steuerung des Zugriffs auf die Ressourcen erforderlich sind.

Jeder Versuch, auf eine Mandantenressource zuzugreifen, sollte nur auf die Ressourcen beschränkt werden, die zu diesem Mandanten gehören. Es ist die Aufgabe der SaaS-Entwickler und -Architekten, zu bestimmen, welche Kombination von Tools und Technologien die Isolationsanforderungen Ihrer spezifischen Anwendung unterstützt.

# Datenpartitionierung

Datenpartitionierung wird verwendet, um verschiedene Strategien zu beschreiben, die zur Darstellung von Daten in einer Umgebung mit mehreren Mandanten verwendet werden. Dieser Begriff wird allgemein verwendet, um eine Reihe verschiedener Ansätze und Modelle abzudecken, mit denen verschiedene Datenkonstrukte einzelnen Mandanten zugeordnet werden können.

Beachten Sie, dass oft die Versuchung besteht, Datenpartitionierung und Mandantenisolierung als austauschbar anzusehen. Diese beiden Konzepte sollen nicht gleichwertig sein. Wenn wir über Datenpartitionierung sprechen, sprechen wir darüber, wie Mandantendaten für einzelne Mieter gespeichert werden. Die Partitionierung von Daten stellt nicht sicher, dass die Daten isoliert sind. Die Isolierung muss dennoch separat angewendet werden, um sicherzustellen, dass ein Mandant nicht auf die Ressourcen eines anderen Mandanten zugreifen kann.

Jede AWS Speichertechnologie bringt ihre eigenen Überlegungen zur Strategie der Datenpartitionierung mit ein. Beispielsweise sieht das Isolieren von Daten in Amazon DynamoDB ganz anders aus als das Isolieren von Daten mit Amazon Relational Database Service (Amazon RDS).

Wenn Sie über Datenpartitionierung nachdenken, denken Sie im Allgemeinen zunächst darüber nach, ob die Daten isoliert oder gepoolt werden sollen. In einem Silomodell haben Sie für jeden Mandanten ein eigenes Speicherkonstrukt, ohne dass Daten miteinander vermischt werden. Bei der gepoolten Partitionierung werden die Daten gemischt und auf der Grundlage einer Mandanten-ID partitioniert, die bestimmt, welche Daten jedem Mandanten zugeordnet sind.

Beispielsweise verwendet ein Silomodell mit Amazon DynamoDB eine separate Tabelle für jeden Mandanten. Das Zusammenlegen von Daten in Amazon DynamoDB wird erreicht, indem die Mandanten-ID im Partitionsschlüssel jeder Amazon DynamoDB-Tabelle gespeichert wird, die Daten für alle Mandanten verwaltet.

Sie können sich vorstellen, dass dies je nach Serviceangebot unterschiedlich sein kann. Jeder dieser AWS Dienste führt seine eigenen Konstrukte ein, sodass bei jedem Service möglicherweise ein anderer Ansatz zur Realisierung von Silo- und Pool-Speichermodellen erforderlich ist.

Datenpartitionierung und Mandantenisolierung sind zwar separate Themen, aber die von Ihnen gewählten Datenpartitionierungsstrategien werden wahrscheinlich vom Isolationsmodell Ihrer Daten beeinflusst. Sie könnten beispielsweise einen Teil des Speichers isolieren, weil dieser Ansatz am besten zu den Anforderungen Ihrer Domain oder Ihrer Kunden passt. Oder Sie entscheiden sich für

Silo, weil das Pool-Modell es Ihnen möglicherweise nicht ermöglicht, die Isolierung mit dem für Ihre Lösung erforderlichen Granularitätsgrad zu erzwingen.

Ein lauter Nachbar kann sich auch auf Ihren Umgang mit Isolation auswirken. Einige Workloads oder Anwendungsfälle in Ihrer Anwendung müssen möglicherweise getrennt werden, um die Auswirkungen anderer Mandanten zu begrenzen oder Service Level Agreements (SLAs) einzuhalten.

# Messung, Kennzahlen und Abrechnung

Diskussionen über SaaS beinhalten in der Regel auch die Begriffe Messung, Kennzahlen und Abrechnung. Diese Konzepte werden oft zu einem Konzept zusammengefasst. Es ist jedoch wichtig, die verschiedenen Rollen zu unterscheiden, die Messung, Kennzahlen und Abrechnung in einer SaaS-Umgebung spielen.

Die Herausforderung dieser Konzepte besteht darin, dass sie sich oft überschneiden, dasselbe Wort verwenden. Wir können zum Beispiel über Zähler sprechen, die zur Erstellung Ihrer Rechnung verwendet werden. Gleichzeitig können wir auch über Zählerkennungen sprechen, mit denen der interne Verbrauch von Ressourcen verfolgt wird, der nicht mit der Abrechnung verbunden ist. Wir sprechen auch über Metriken und SaaS in vielen Kontexten, die in diese Diskussion einfließen können.

Um das zu klären, lassen Sie uns jedem dieser Begriffe einige spezifische Konzepte zuordnen (wobei wir wissen, dass es hier keine Absolutwerte gibt).

- Metering — Dieses Konzept hat zwar viele Definitionen, passt aber am besten in den Bereich der SaaS-Abrechnung. Die Idee ist, dass Sie die Aktivität oder den Ressourcenverbrauch der Mieter messen, um die Daten zu sammeln, die für die Erstellung einer Rechnung erforderlich sind.
- Kennzahlen — Kennzahlen stellen alle Daten dar, die Sie erfassen, um Trends in Ihren Geschäfts-, Betriebs- und Technologiebereichen zu analysieren. Diese Daten werden in vielen Kontexten und Rollen innerhalb des SaaS-Teams verwendet.

Diese Unterscheidung ist nicht entscheidend, hilft uns aber dabei, unsere Vorstellung von der Rolle von Metering und Metriken in einer SaaS-Umgebung zu vereinfachen.

Wenn wir nun diese beiden Konzepte mit Beispielen verbinden, können Sie sich vorstellen, Ihre Anwendung mit bestimmten Messvorgängen zu instrumentieren, anhand derer die für die Erstellung einer Rechnung erforderlichen Daten angezeigt werden. Dabei kann es sich um die Anzahl der Anfragen, die Anzahl der aktiven Benutzer oder um eine Zuordnung zu einem Gesamtverbrauch (Anfragen, CPU, Arbeitsspeicher) handeln, der mit einer Einheit korreliert, die für Ihre Kunden sinnvoll ist.

In Ihrer SaaS-Umgebung veröffentlichen Sie diese Abrechnungsereignisse von Ihrer Anwendung aus und sie werden vom Abrechnungskonstrukt Ihres SaaS-Systems aufgenommen und angewendet. Dies kann ein Abrechnungssystem eines Drittanbieters oder etwas Benutzerdefiniertes sein.

Im Gegensatz dazu besteht die Denkweise hinter Kennzahlen darin, die Aktionen, Aktivitäten, Konsummuster usw. zu erfassen, die für die Bewertung der Gesundheit und des betrieblichen Fußabdrucks, den verschiedene Mieter Ihrem System auferlegen, unerlässlich sind. Die Kennzahlen, die Sie hier veröffentlichen und aggregieren, hängen eher von den Bedürfnissen verschiedener Personas ab (Betriebsteams, Product Owner, Architekten usw.). Hier werden diese Metrikdaten veröffentlicht und in einigen Analysetools zusammengefasst, die es diesen verschiedenen Benutzern ermöglichen, Ansichten über die Systemaktivität zu erstellen, die die Aspekte des Systems analysieren, die am besten zu ihrer Persona passen. Ein Product Owner möchte vielleicht verstehen, wie verschiedene Mandanten Funktionen nutzen. Ein Architekt benötigt möglicherweise Ansichten, anhand derer er verstehen kann, wie Mieter Infrastrukturressourcen verbrauchen usw.

# B2B- und B2C-SaaS

SaaS-Angebote wurden sowohl für den B2B- als auch für den B2C-Markt entwickelt. Obwohl die Märkte und Kunden sicherlich unterschiedliche Dynamiken aufweisen, ändern die allgemeinen Prinzipien von SaaS nicht irgendwie jeden dieser Märkte.

Wenn Sie sich beispielsweise das Onboarding ansehen, können B2B- und B2C-Kunden unterschiedliche Onboarding-Erfahrungen haben. Es stimmt, dass sich B2C-Systeme eher auf einen Self-Service-Onboarding-Flow konzentrieren (obwohl das B2B-System dies möglicherweise auch unterstützt).

Es mag zwar Unterschiede in der Art und Weise geben, wie das Onboarding den Kunden angeboten wird, aber die grundlegenden Werte für das Onboarding sind größtenteils dieselben. Auch wenn Ihre B2B-Lösung auf einem internen Onboarding-Prozess basiert, würden wir dennoch erwarten, dass dieser Prozess so reibungslos und automatisiert wie möglich abläuft. B2B zu sein bedeutet nicht, dass wir unsere Erwartungen im Laufe der Zeit ändern, um unseren Kunden einen Mehrwert zu bieten.

# Schlussfolgerung

Ziel dieses Dokuments ist es, grundlegende Konzepte der SaaS-Architektur zu skizzieren und einige Erläuterungen zu den Modellen und der Terminologie zu geben, die zur Charakterisierung von SaaS-Mustern und -Strategien verwendet werden. Es besteht die Hoffnung, dass Unternehmen dadurch einen klareren Überblick über die gesamte SaaS-Landschaft erhalten.

Vieles, was hier behandelt wird, konzentrierte sich darauf, was es bedeutet, SaaS zu sein, wobei der Schwerpunkt auf der Schaffung einer Umgebung lag, in der Sie alle Ihre SaaS-Mandanten über ein einheitliches Erlebnis verwalten und betreiben können. Dies steht im Zusammenhang mit der Kernidee, dass SaaS in erster Linie ein Geschäftsmodell ist. Die von Ihnen erstellte SaaS-Architektur soll diese grundlegenden Geschäftsziele fördern.

# Weitere Informationen

Es gibt eine Reihe von Ressourcen, die detaillierter auf SaaS-Architekturmuster eingehen, die den hier beschriebenen Mustern entsprechen.

Weitere Informationen finden Sie unter:

- Strategien zur Isolierung von SaaS-Mandanten (AWSWhitepaper)
- SaaS-Speicherstrategien (AWSWhitepaper)
- Gut konzipiertes SaaS-Objektiv (Whitepaper) AWS

# Beitragende Faktoren

Folgende Personen und Organisationen haben zu diesem Dokument beigetragen:

- Tod Golding, Hauptpartner, Lösungsarchitekt, AWS SaaS Factory

# Dokumentversionen

Wenn Sie über Aktualisierungen dieses Whitepapers möchten, abonnieren Sie den RSS-Feed.

| Änderung | Beschreibung | Datum |
| --- | --- | --- |
| Erstveröffentlichung | In. | 03. August 2022 |

# Hinweise

Die Kunden sind dafür verantwortlich, die Informationen in diesem Dokument selbst unabhängig zu bewerten. Dieses Dokument: (a) dient nur zu Informationszwecken, (b) stellt aktuelle AWS Produktangebote und Praktiken dar, die ohne vorherige Ankündigung geändert werden können, und (c) enthält keine Verpflichtungen oder Zusicherungen von AWS und seinen verbundenen Unternehmen, Lieferanten oder Lizenzgebern. AWSProdukte oder Dienstleistungen werden „wie sie sind“ ohne ausdrückliche oder stillschweigende Garantien, Zusicherungen oder Bedingungen jeglicher Art bereitgestellt. Die Verantwortlichkeiten und Verbindlichkeiten AWS gegenüber seinen Kunden werden durch AWS Vereinbarungen geregelt, und dieses Dokument ist nicht Teil einer Vereinbarung zwischen seinen Kunden AWS und seinen Kunden und ändert diese auch nicht.

# AWS-Glossar

Die neueste AWS-Terminologie finden Sie im AWS-Glossar in der AWS-Glossar-Referenz.

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