



Georg Spengler

[E-Mail: georg.spengler@arlanis.com] arbeitet als Consultant für die arlanis Software AG und ist seit mehreren Jahren spezialisiert auf die Beratung zu und Umsetzung von Datenintegrationen im „as-a-Service“-Umfeld mit besonderem Fokus auf Aspekten der Datensicherheit.



Andreas Holubek

[E-Mail: andreas.holubek@arlanis.com] arbeitet als VP Engineering für die arlanis Software AG und blickt auf eine langjährige Arbeit in Konzeption, Design und Programmierung von Software zurück. Daneben beschäftigt er sich mit der Einführung und Umsetzung von modernen Technologien in der Praxis.



Gernot Saborowski

[E-Mail: gernot.saborowski@brainlab.com] ist als Lead Project Manager bei der Brainlab AG tätig. Er beschäftigt sich seit den Anfängen des WWW mit Webtechnologien und hat u. a. im Rahmen seiner langjährigen Tätigkeit für die Brainlab AG Erfahrungen sowohl im Bereich Medizinproduktentwicklung wie auch mit Datenbanken und Netzwerken sammeln können.

Mastering Cloud-Anwendungen – Architektur und Technologie anhand einer Community für Ärzte

Fast jede neue oder zu erweiternde Business-Anwendung wird heute auf die Realisierbarkeit mittels Plattform-as-a-Service, respektive Cloud-Technologien, überprüft. So bleibt dieses Feld nicht mehr nur klassischen Domänen, wie zum Beispiel CMS, vorbehalten. In diesem Artikel werden wir das Projekt „Community für Ärzte“ der Firma Brainlab AG vorstellen [QUE]. Hier wird anhand eines realen Beispiels ein Einblick in die aktuelle Welt der Plattform-Anwendungen gegeben.

Die Everything-as-a-Service Welt

Seit einiger Zeit lässt sich ein Wandel in der IT-Dienstleistungswelt beobachten. Software wird immer mehr in Richtung der Cloud verschoben, spezieller gesagt in die von den Herstellern bereit gestellten Plattformen. Darunter können wir uns Azure von Microsoft [AZU], AWS von Amazon [AWS] oder auch Force.com von salesforce.com [FOR] vorstellen, um nur einige ausgewählte Beispiele zu nennen.

Der hinter diesem Konzept stehende Gedanke scheint auf den ersten Blick nur Vorteile aufzuweisen. Anwender müssen keine Software mehr installieren und auch die Wartung entfällt. Entwickler müssen sich nicht mehr um Infrastruktur, benötigte Software und Verteilung der Ergebnisse kümmern (siehe [Abbildung 1](#)). Eigentlich die perfekte Welt für alle Beteiligten. Aus Sicht der Entwicklung sind jedoch neue Aufgaben zu lösen und neue Konzepte zur Bereitstellung, Wartung und den Betrieb zu entdecken.

Stück für Stück wurden in den letzten Jahren die Dienste in das Internet gelegt, nicht unbemerkt, aber doch eher ohne den

sonst allgegenwärtigen „Hype“ (man denke nur einmal an SOA...). Heute können wir fast alles als Dienst beziehen, sei es Festplattenspeicher, Rechenleistung oder eben der Platz für ganze Anwendungen. An

dieser Stelle ist es schier unmöglich, alle Dienste – in unserem Internet der Dienste – aufzuzählen, weshalb wir uns hier auf zwei grundlegende Konzepte Software- und Plattform-as-a-Service konzentrieren.

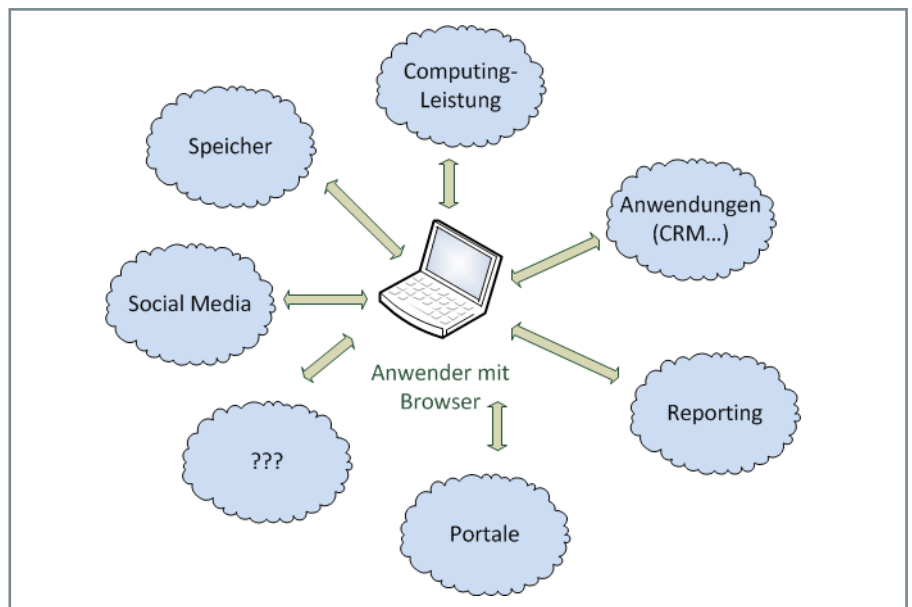


Abb. 1: Jedes Detail einer Software ist als Dienst im Internet vorhanden (Internet der Dienste).

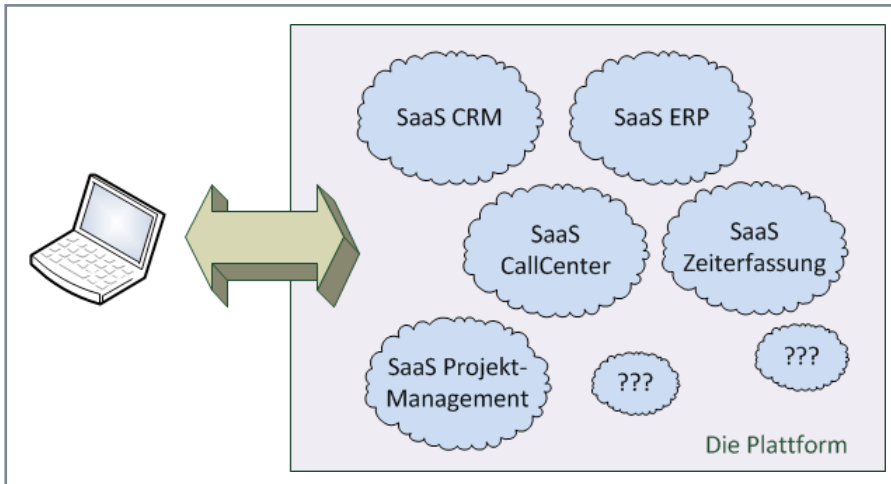


Abb. 2: Die Plattform bildet die vereinheitlichte Grundlage der SaaS-Anwendungen.

Platform-as-a-Service

Stellen wir uns einfach einmal vor, es sind eine Menge, vom Architekturstandpunkt ähnlicher Systeme zu entwickeln (CRM, beliebige Verwaltungen...). Alle diese Anwendungen haben im Prinzip einen ähnlich aufgebauten Kern, wiederum aus der Softwarearchitektur betrachtet. Anstatt diesen Kern jedes Mal neu zu entwerfen, zu durchdenken und zu implementieren, wurde dieser in eine generische Form gebracht und steht nun allen Software-as-a-Service-Anwendungen gleichermaßen zur Verfügung. Wohlgermerkt, bezogen auf eine konkrete Plattform. **Abbildung 2** verdeutlicht das Prinzip.

Architektonische Hintergründe

Bevor wir jetzt in die eigentliche Technik einsteigen, möchten wir an dieser Stelle einen Überblick über die Architektur des vollständigen Systems geben. Aufgrund der verschiedensten Anforderungen an die einzelnen Bereiche des Community-Portals mussten verschiedene Lösungen und deren Hersteller für eine vollständige Lösung gefunden werden. In **Abbildung 3** sind die beteiligten technischen Komponenten und ihre jeweilige Aufgabe zu sehen.

Die jeweiligen Teilbereiche des Portals haben dabei sehr spezielle Aufgaben zu meistern und werden durch entsprechende technische Komponenten hinterlegt:

■ *Verwaltung der Benutzer und Benutzergruppen*

Dieses Teilsystem wurde durch die Force.com-Plattform realisiert, welche auch als Klammer für alle anderen technischen Komponenten dient.

■ *Viewer und Uploader für medizinische Bilder*

Aufgrund der hohen Anforderungen an Ausführungsgeschwindigkeit, Sicherheit und Speicherbedarf ist dieses Teilsystem in Microsoft Silverlight implementiert. Die Daten liegen verschlüsselt auf der Amazon Simple Storage Server-Plattform (Amazon S3) sowie auf der Amazon RDS. Rechenleistung zur Kalkulation der dargestellten Bilder wird durch je nach Last automatisch gestartete Amazon Elastic Cloud Computing (EC2) Instanzen zur Verfügung gestellt.

■ *Austausch von Nachrichten, Diskussionen und Kontaktdaten*
Der Austausch von Nachrichten, die

Diskussionsboards sowie die Verwaltung der Kontakte ist auf der Force.com-Plattform implementiert. Zur Bereitstellung der Oberfläche kam das Dojo Framework zum Einsatz.

Die Kommunikation zwischen den technischen Komponenten geschieht mithilfe von Web Services. Erst durch die Standardisierung der Beschreibung von Schnittstellen lassen sich Elemente aus verschiedenen Plattformen leicht miteinander verbinden.

Das Entwicklungsmodell am Beispiel

Die Entwicklung von Software im Cloud-Umfeld unterscheidet sich vom klassischen Entwicklungsansatz auf lokalen Systemen. Das diesem Artikel zugrunde liegende Beispiel wurde auf Basis der force.com-Plattform realisiert, angereichert mit zusätzlichen Komponenten der Amazon Web Services, insbesondere Rechnerressourcen in der „Elastic Compute Cloud“ (EC2) und Speicherplatz als „Simple Storage Service“ (S3). Die in diesem Zusammenhang dargestellten Entwicklungstechniken geben sicherlich kein vollständiges Bild der as-a-Service-Entwicklung auf den unterschiedlichen Plattformen, jedoch gestatten sie einen guten Einblick in wesentliche Unterschiede zur lokalen Entwicklung und Bereitstellung.

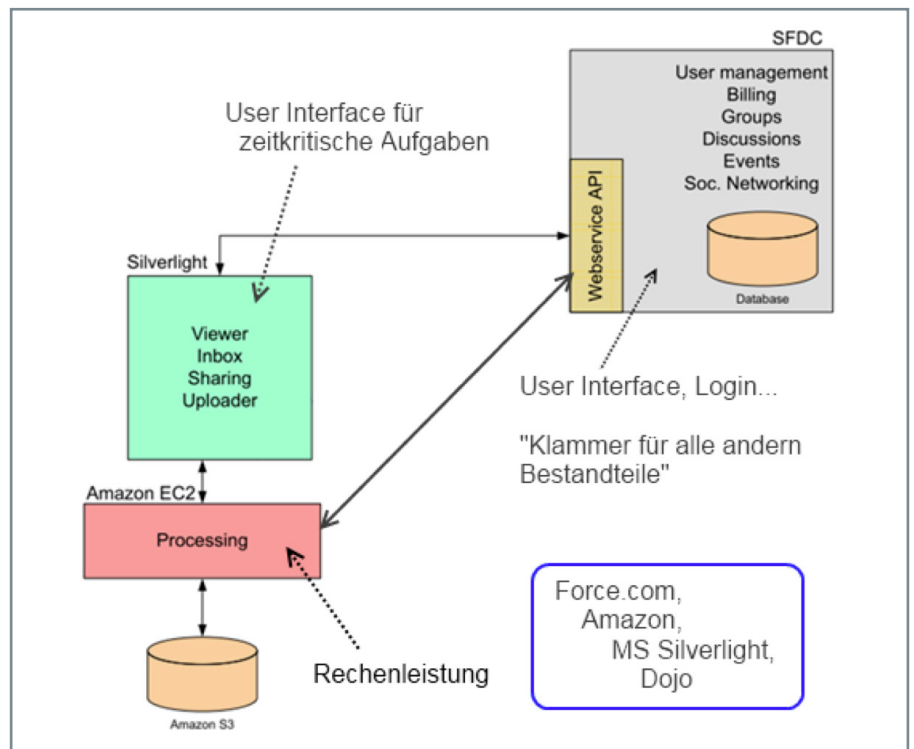


Abb. 3: Überblick über die Architektur des gesamten Projektes.

The screenshot shows the Salesforce configuration interface for a Custom Object named 'User Group Member'. The left sidebar contains navigation menus for 'Personal Setup', 'App Setup', and 'Administration Setup'. The main content area is titled 'Custom Object Definition Detail' and includes a table of 'Standard Fields' and a table of 'Custom Fields & Relationships'.

Field Label	Field Name	Data Type	Controlling Field	Track History
Created By	CreatedBy	Lookup(User)		<input type="checkbox"/>
Last Modified By	LastModifiedBy	Lookup(User)		<input type="checkbox"/>
User Group Member Number	Name	Auto Number		<input type="checkbox"/>

Action	Field Label	API Name	Data Type	Controlling Field	Modified By	Track History
Edit Del	Is Protected	Is_Protected__c	Checkbox		Gernot Saborowski, 19.01.2011 14:59	<input type="checkbox"/>
Edit Del	Member	Member__c	Lookup(UserC)		Arlanis, 26.10.2010 12:14	<input type="checkbox"/>
Edit Del	Member Approved	MemberApproved__c	Checkbox		Gernot Saborowski, 16.09.2010 15:40	<input type="checkbox"/>
Edit Del Replace	Membership Status	MembershipStatus__c	Picklist		Gernot Saborowski, 02.02.2011 13:50	<input type="checkbox"/>
Edit Del	User Group	User_Group__c	Master-Detail(User Group)		Gernot Saborowski, 16.09.2010 15:40	<input type="checkbox"/>

Abb. 4: Konfigurationsseite für ein Objekt des Datenmodells.

Metadaten als bestimmende Größe

Einer der Vorteile der as-a-Service-Plattformen ist die für alle Benutzer einheitlich bereitgestellte Infrastruktur und Plattform, die durch Modifikationen an die speziellen Bedürfnisse angepasst werden kann. Dem entsprechend besteht ein wesentlicher Teil der Softwareentwicklung aus der Definition der benötigten Komponenten in den Plattform-Metadaten.

Um eine Community auf der force.com-Plattform zu realisieren, wurde bei-

spielsweise zunächst eine sogenannte „Site“ mit angebundenem „Customer Portal“ konfiguriert. Die Site ermöglicht das Bereitstellen einer öffentlich zugänglichen Internetpräsenz, während das Customer Portal die Verknüpfung zur Benutzer- und Rechteverwaltung der Plattform herstellt.

Da die force.com-Plattform stets eine Datenbank enthält, kann anhand von Benutzerprofilen genau konfiguriert werden, auf welche Seiten und Datenbanktabellen

die Benutzer zugreifen können und auf welche nicht. Dem weiteren Design der Anwendung liegt die Untergliederung nach Model, View und Controller zugrunde. Auch zur Definition des Datenmodells muss keine einzige Zeile Quelltext geschrieben werden (siehe **Abbildung 4**), da alle zulässigen Strukturen bequem über eine Web-Applikation konfiguriert werden können.

Auf diesen Metadaten aufbauend wird die visuelle Ebene auf der force.com-Platt-

The screenshot displays two windows. The left window shows the source code for a Visualforce page, including Apex controller logic and HTML markup for a table and navigation links. The right window shows the rendered page in a browser, featuring a navigation bar with 'Home', 'Messages', 'Groups', 'Events', 'Contacts', and 'Profile' links, and three main action buttons: 'Upload & Share', 'View & Consult', and 'Invite & Connect'.

Abb. 5: Quelltexte und Aussehen einer Visualforce Page.



Abb. 6: Mittels Silverlight-Komponente werden medizinische Daten von der Amazon-Plattform im force.com-Kontext angezeigt.

form durch sogenannte „Visualforce Pages“ bereitgestellt, die sich syntaktisch an Java Server Faces orientieren. Die einzig verbleibenden Bestandteile, die tatsächlich programmiert werden müssen, sind die Controller, die den Seiten Interaktion mit dem Datenmodell und den Daten ermöglichen. Die force.com-Plattform stellt für Controller und Hilfsklassen die „Apex“ Programmiersprache zur Verfügung. Diese Sprache orientiert sich von der Grammatik und Syntax stark an Java und C# (siehe Abbildung 5).

Unter anderem wegen der großen Datenmengen und der erforderlichen Rechenleistung können die medizinischen Daten der Community nicht auf der force.com-Plattform gehostet werden – hierfür stehen entsprechende EC2-, RDS- und S3-Instanzen der Amazon Web Services zur Verfügung. Die Integration in die force.com-Plattform erfolgt dabei einerseits über die Web Serviceschnittstelle der Plattform, etwa um Nachrichten in die Community einstellen zu können, sowie durch eine Silverlight-Komponente zur Darstellung medizinischer Informationen in die Seiten der force.com-Plattform (siehe Abbildung 6).

Lokale vs. entfernte Entwicklung

Auch die Entwicklung von Visualforce Pages und Apex Controllern kann über die force.com- Web-Applikation vorgenommen werden. Dieser Ansatz ist bei Entwicklern jedoch nicht sonderlich beliebt,

da hier bis auf die Syntaxhervorhebung praktisch alle unterstützenden Elemente, an die man sich bei modernen IDEs gewöhnt hat, fehlen. Aus diesem Grund gibt es auch zusätzlich eine auf der Eclipse-Plattform basierende force.com IDE, die versucht, diesen Mangel zu adressieren – das Programm wird lokal auf dem Entwicklungsrechner installiert und synchronisiert die Projekte aus dem Workspace mit der force.com-Plattform.

Dennoch bleiben trotz Einsatz der force.com IDE noch wesentliche Unterschiede bestehen. Die Arbeit von mehreren Entwicklern an demselben Projekt gestaltet sich relativ schwierig, denn aufgrund der Synchronisation mit der Plattform ist es nicht möglich, mit Check-in bzw. Check-out von Quelltexten zu arbeiten. An dieser Stelle bedarf es also detaillierter Absprachen, insbesondere im Fall einer Einbeziehung von externen Dienstleistern.

Im Gegensatz zur klassischen Vorgehensweise beim Deployment von Web-Anwendungen ist auf der force.com-Plattform dieser Prozess stark vereinfacht worden: Statt den Quelltext lokal zu kompilieren und den dadurch entstandenen Binärcode auf einem Application Server zu deployen, wird dies zu einem einzigen Schritt zusammengefasst. Die Quelltexte werden auf einem gesonderten Bereich der force.com-Datenbank gespeichert und dabei validiert.

Das bedeutet, dass die Speicherung nur dann erfolgreich ist, wenn der Quelltext erfolgreich kompiliert wurde und auch keine weiteren Fehler – etwa bei der Ausführung der obligatorischen Unit Tests – festgestellt wurden. Dieses Vorgehen impliziert zwei Dinge, die Entwickler als hinderlich empfinden können: Erstens entsteht durch Kompilation und Unit Tests eine merkliche Wartezeit bei jedem Speichern, zweitens sind Änderungen nach erfolgreichem Speichern stets sofort aktiviert.

Sicherheitsanforderungen und Verfügbarkeit

Während bei lokaler Datenhaltung Quelltexte, fertige Applikationen und Server das eigene Netzwerk in der Regel nicht verlassen, befinden sich all diese Daten beim Cloud-Ansatz auf einem Fremdsystem, das von vielen Kunden des as-a-Service-Anbieters gemeinsam verwendet wird. Dadurch sind die Daten im Internet wesentlich vielfältigeren Gefahren ausgesetzt. Andererseits erreicht man in der Cloud durch Clustering von Systemen eine wesentlich höhere Verfügbarkeit. Die hohen Kosten hierfür verteilen sich auf viele Schultern, sodass der Einzelne nur noch einen kleinen Teil beisteuern muss.

Aufgrund der Gefahren, denen die Daten im Netz ausgesetzt sind, ist es besonders wichtig, dass der Dienstleister die Anforderungen des Datenschutzes sowie alle weiteren rechtlichen Rahmenbedingungen erfüllt. Hierbei ist speziell im Umgang mit medizinischen Daten auch zu prüfen, in welchen Ländern der as-a-Service-Provider seine Rechenzentren betreibt und wie die Daten darüber verteilt werden. Ferner müssen Anbieter und Kunden gemeinsam an einem bestmöglichen Schutz des Datenbestands vor unberechtigtem Zugriff und auch Angriffen arbeiten.

Sind die Sicherheitsanforderungen erfüllt, erhält man ein hochverfügbares System, mit dem sich gut arbeiten lässt. Doch was nutzt eine Verfügbarkeit von 99,9 % im Rechenzentrum, wenn die Verfügbarkeit der Anbindung des Kunden an das Internet nur bei 97,5 % liegt und außerdem die Bandbreite nicht ausreicht, um die anfallenden Datenmengen zu bewältigen? Auch für diesen Aspekt des Cloud Computings ist also der ganzheitliche Ansatz äußerst wichtig, da nur dann eine gute Arbeitserfahrung zu erzielen ist, wenn alle zusammenspielenden Komponenten optimal ineinander greifen – angefangen beim

lokalen Arbeitsplatz, über die Anbindung des Arbeitsplatzes ans Internet bis hin zur redundant ausgelegten Infrastruktur des as-a-Service-Anbieters.

Fazit

Seit über einer Dekade hat sich das Konzept von Software-as-a-Service, Platform-as-a-Service und Cloud Computing von den ersten Anfängen bis hin zu den heutigen Lösungen entwickelt. Wie Sie im Rahmen des Artikels gesehen haben, sind professionelle Anwendungen auf Basis dieser Technologie machbar. Das dies möglich ist, hat nicht zuletzt seinen Grund darin, dass viele vorgefertigte Konzepte und auch

Funktionalitäten aus der Plattform genutzt werden können. Mittlerweile würde es weit den Rahmen eines Artikels sprengen, alle Einzelheiten zu beschreiben oder auch nur vorstellen zu wollen.

Danke

Leider können im Rahmen eines solchen Artikels nur wenige Personen als Autoren stehen. Deshalb möchten wir an dieser Stelle einen besonderen Dank an die Entwicklungsteams der Brainlab AG und arlanis Software AG für die kritischen und hilfreichen Kommentare während der gesamten Zeit, welche nicht zuletzt auch in diesem Artikel mündeten, senden. ■

Literatur & Links

[AWS] Amazon Web Services (AWS):

<http://aws.amazon.com>

[AZU] Microsoft Azure Plattform:

<http://www.azure.com>

[FOR] Salesforce.com Force.com Plattform:

<http://www.force.com>

[HEU] O. Heuser, A. Holubek: Java Web Services in der Praxis: dpunkt.verlag 2010; ISBN 978-3-89864-596-6

[QUE] Qentry: Online Portal für Mediziner: <http://www.qentry.com>