

SaaS (Software as a Service) kennzeichnet ein Prinzip zur Verteilung und Nutzung von Software. Anwender mieten dabei Software, wobei unter der Miete nicht nur die tatsächliche Anwendung gemeint ist, sondern vielmehr auch die Dienstleistung rund um den Betrieb und die Wartung der Anwendung. Der Anwender erhält bei SaaS eine komplette Infrastruktur – genau genommen mietet er jedoch einen kleinen Teil aus einer größeren Infrastruktur, denn er teilt sich Dinge wie CPU, Rechenleistung oder Speicherplatz mit anderen Anwendern – ohne dass das negative Einfluss auf die Arbeit hätte. Das betrifft vor allem die Anwenderseite.

Um Platform as a Service (PaaS) zu verstehen, muss man nun auf die Seite der Entwicklung wechseln. Vorgestellt sei, dass vom Architekturstandpunkt eine Menge ähnlicher Systeme zu entwickeln ist (CRM, beliebige Verwaltungen...). Die Anwendungen haben im Prinzip einen ähnlich aufgebauten Kern. Statt ihn jedes Mal neu zu entwerfen und zu implementieren, wurde er in eine generische Form gebracht und steht nun allen SaaS-Anwendungen gleichermaßen zur Verfügung. Wohlgemerkt, bezogen auf eine konkrete Plattform. Es existieren schon einige Plattformen, alle mit ihren jeweiligen Vor- und Nachteilen. Amazon Web Services, Microsofts Azure und Force.com mögen als Beispiele hierfür genannt sein. Bei ihnen findet man unter den Anwendungen neben den Klassikern CRM und Vertriebsunterstützung auch Projektmanagement, Urlaubsplanung, News und vieles andere mehr.

Die SaaS-Anwendungen, die gerade entstanden und noch im Entstehen sind, bilden eine große Serviceinfrastruktur im Netz. Folgerichtig spricht man deshalb auch von Infrastructure as a Service (IaaS). Anstelle eigener Hard- oder Software im Keller zielt das Prinzip auf die Miete der benötigten Teile und die anschließende Verknüpfung derselben über definierte Schnittstellen. Hierbei haben standardisierte Webservices ebenfalls einen positiven Effekt auf die Integration unterschiedlicher Systeme.

Eine Betrachtung der Architektur

Das Kernstück jeder SaaS-Anwendung bilden die Datenbank, eine Engine zur Ausführung von Programmen (oder Teilen davon) und Metadaten zur Beschrei-



Softwareentwicklung im Zeitalter der „As a Service“-Plattformen

Unter der Lupe

Andreas Holubek, Georg Spengler

Mit der Einführung von „As a Service“-Projekten geht ein grundsätzlicher Wandel für IT-Dienstleistung und Entwicklung von Softwaresystemen einher. Anwender müssen keine Software mehr installieren, auch die Wartung entfällt. Entwickler müssen sich nicht mehr um Infrastruktur, Software und Verteilung der Ergebnisse kümmern. Der Artikel gibt einen Einblick in den Anwendungsentwicklungsprozess und die Architektur in einem „As a Service“-Universum.

bung von Funktionen. Im Gegensatz zu klassischen Anwendungen nutzen die Anwender große Teile des Systems gemeinsam, woraus sich auch der Fachbegriff Multi-Tenancy-Architektur ableitet. An diese sind die Teilkomponenten anzupassen.

Die Benutzerschnittstelle einer SaaS-Anwendung wird in den meisten Fällen durch eine Weboberfläche oder ein

mobiles Gerät repräsentiert. Beides ist nicht verwunderlich, liegt doch die gesamte Programmlogik in der SaaS-Anwendung. Die Plattform erweitert die Architektur der SaaS-Anwendung um das Heraustrennen von generischen Bestandteilen, die sich zwischen den Anwendungen gemeinsam nutzen lassen. Ein klassisches Beispiel ist sicherlich die Benutzerverwaltung.

Die Integration einer SaaS-Anwendung in eine Systemlandschaft geht einfach und schnell über Webservices. Nahezu jedes ernst zu nehmende System stellt heute eine Webservice-Schnittstelle auf Basis von SOAP oder REST bereit. Somit lassen sich Anwendungen beliebig verknüpfen, sei es nun SAP, Lotus Notes, Salesforce.com oder Google App Engine. Ein wichtiges Augenmerk muss man jedoch auf die Lebensdauer und die hohe Interaktion mit der umgebenden Welt einer solchen Anwendung legen. Die aktuellsten Erfahrungen zeigen, dass eine Laufzeit von mehreren Jahren in Betracht zu ziehen ist. Deswe-

gen ist ein hoher Verknüpfungsgrad mit der IT-Landschaft von großer Bedeutung. Daraus resultiert wiederum, dass sich einzelne Programmteile einfach abschalten oder inkompatibel ersetzen lassen. Aus den Erfahrungen heraus ist es notwendig, von Anfang an ein Konzept zur Versionierung von Artefakten einzuplanen. Abbildung 3 zeigt solch eine Erweiterung der Architektur im Überblick.

Die Plattform enthält die folgenden grundlegenden Elemente für die Entwicklung eigener SaaS-Anwendungen.
 – Datenbank: Sie ist das Herzstück der Plattform. In ihr finden sich alle Unternehmens- und Nutzerdaten sowie die

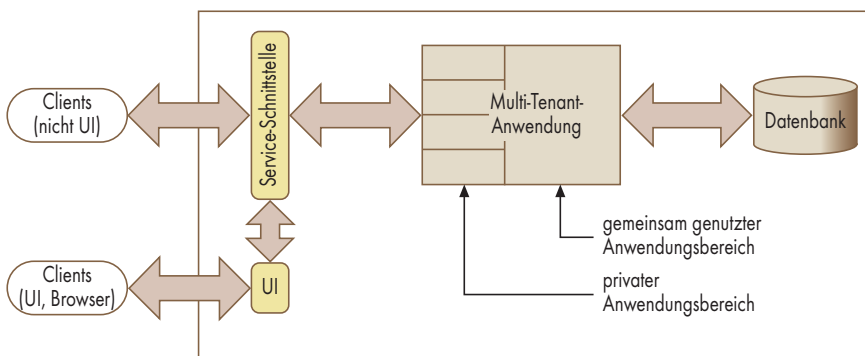
Metadaten. Die Datenbank bleibt meistens vor einer direkten Nutzung verborgen.

– Metadaten und metadaten-getriebene Entwicklung: Metadaten stellen ein wichtiges Artefakt in der Plattform dar und entscheiden nicht selten über die Qualität derselben. Mit Metadaten lassen sich sowohl Domainobjekte als auch die Anwendungen, Workflows und die Beziehungen zwischen den einzelnen Artefakten beschreiben, um nur einige Beispiele zu nennen. Während der Entwicklung von Artefakten für die Plattform werden die Metadaten angelegt, die die Laufzeitumgebung im Anschluss für die Präsentation der Anwendung nutzt.

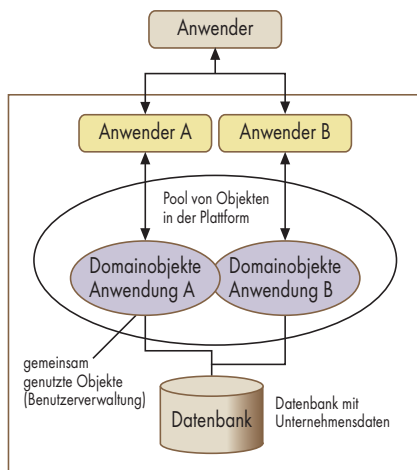
– grundlegende Dienste: Die Plattform stellt eine Menge von Diensten bereit, die sich direkt in den für die Plattform entwickelten Anwendungen nutzen lassen. Solche Dienste können zum Beispiel Benutzersteuerung und Authentifizierung, aber auch Verzeichnis- und Dateidienste sein.

– Sprachen, Entwicklungsumgebung: Letztere ist notwendig, um die Metadaten in der Plattform zu definieren und abzulegen. Spezielle Sprachen wie Apex Code von Salesforce.com können dabei eine besondere Unterstützung des SaaS-Programmiermodells als integralen Bestandteil mitbringen. Jedoch setzen sich aktuell die bekannten Programmiersprachen (Java, C# ...) für die Entwicklung auf der Plattform mehr und mehr durch – was wiederum einen größeren Entwicklerkreis zulässt.

– Multitenant-Architektur: Sie ist von der Eigenschaft gekennzeichnet, dass alle Benutzer die gleiche physische Instanz und Version einer Anwendung nutzen. Alle Updates oder Erweiterungen werden gleichzeitig und automatisch an alle Benutzer verteilt. Ein Vorteil ergibt sich daraus, dass jegliche administrative Aufgaben beim Betreiber der Plattform liegen. Die Nutzung der gleichen physischen Instanz bedeutet jedoch nicht, dass alle Benutzer Zugriff auf alle Daten der anderen Benutzer haben. Hier erfolgt eine sichere und geschützte Trennung in verschiedene Datenbereiche.

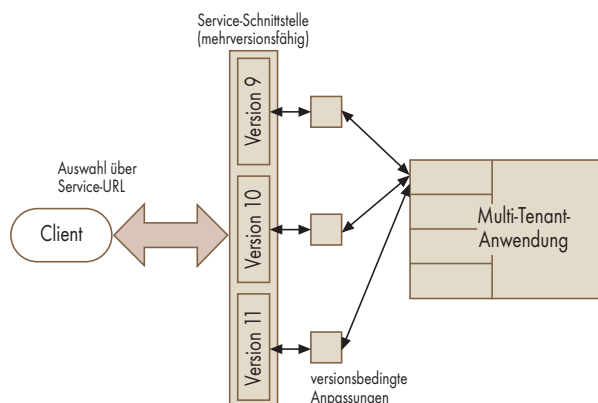


In einer SaaS-Architektur finden sich ein gemeinsam genutzter Bereich sowie spezielle Anpassungen und Datenspeicher für den einzelnen Benutzer. Für den Zugriff existieren Webservice-Schnittstellen und Weboberflächen (Abb. 1).



Bei PaaS teilen sich mehrere Anwendungen zahlreiche grundlegende Artefakte – beispielsweise Datenobjekte, Seiten und Sicherheit (Abb. 2).

Alle Artefakte in der Plattform haben eine definierte Version. Anwender nutzen immer die für sie beste; damit ist die Kompatibilität auch für die Zukunft gesichert (Abb. 3).



AaS-gerechtes Programmiermodell

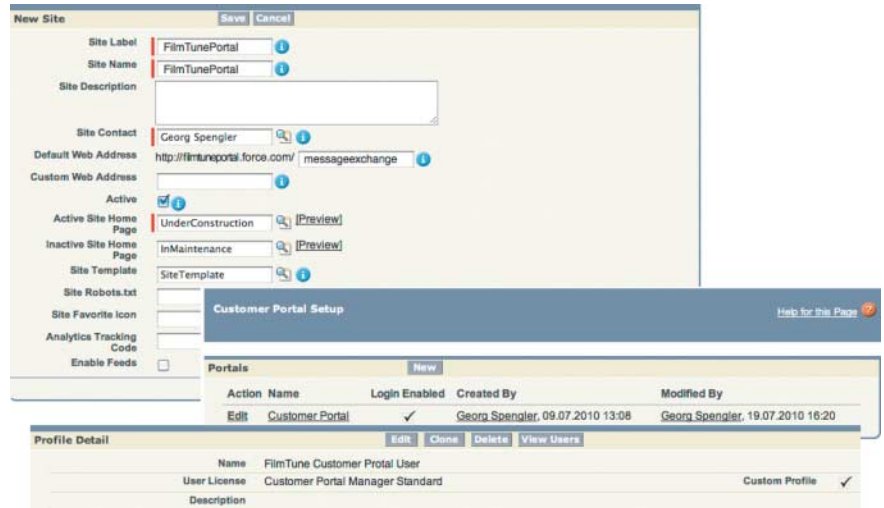
Beispielhaft sei nun die Entwicklung eines Portals auf der Force.com-Plattform mit Anbindung an die Amazon Web Services betrachtet, ohne hiermit eine Wertung oder Präferenz gegenüber

den beiden genannten Plattformen zum Ausdruck bringen zu wollen.

Man stelle sich die fiktive „Film Tune AG“ vor, die in Europa führend in der Produktion von Geräten zur Film- und Fernsehproduktion ist. Als CRM-System setzt die Firma an ihren Standorten Salesforce.com ein und möchte nun auf der Basis für ihre Kunden ein exklusives Portal zum fachlichen Erfahrungsaustausch schaffen, das ähnlich einem sozialen Netz funktioniert. Die Nutzung des Portals wird für Kunden zwar kostenpflichtig sein, soll dafür aber auch den Austausch ganzer Filmsequenzen im Rahmen fachlicher Diskussionen zulassen. Da Salesforce.com zwar gut geeignet für die Verwaltung der Benutzer und ihrer Rechte ist, bei der Verarbeitung der großen Filmsequenzen jedoch schwächelt, zieht die Film Tune AG Amazons Simple Storage Services (S3) für diesen Teilaspekt hinzu. Für die Integration der beiden Plattformen in demselben Projekt ist die Einbindung einer Flash-Komponente in die Force.com-Anwendungsplattform vorgesehen. Flash wiederum soll dann durch Austausch von SOAP-Nachrichten mit S3 kommunizieren und die Darstellung sowie Verarbeitung der Filmsequenzen übernehmen.

Auf der Salesforce.com-Instanz der Film Tune AG aufbauend bestehen die ersten Schritte in Richtung Portal darin, „force.com-Sites“ und „Customer-Portal“ zu aktivieren (siehe Abb. 4). Ersteres stellt eine öffentlich zugängliche Website dar, über die sich Anmeldung und Login durchführen lassen. Das mit den „force.com-Sites“ verknüpfte „Customer-Portal“ enthält die Inhalte, die nur angemeldeten Benutzern zur Verfügung stehen. Darauf wird für Plattformbenutzer ein eigenes Profil definiert, das die Zugriffsrechte der Benutzergruppe auf die Daten regelt. Die weitere Entwicklung auf der Force.com-Plattform folgt dem MVC-Pattern (Model View Controller).

Das Datenmodell wird entweder über die Set-up-Funktion der Salesforce.com-Weboberfläche oder unter Verwendung der Metadaten-API definiert. Da die Plattform der Film Tune AG es registrierten Benutzern erlauben soll, andere Benutzer im gegenseitigen Einvernehmen zu Kontakten zu erklären und auf der Basis einen Nachrichtenaustausch innerhalb des Portals bereitzustellen, sind auf der Datenbank einige spezielle Tabellen – „Custom Objects“ im Salesforce.com-Jargon – zu definieren, etwa *User_Contact__c* und *Message__c*. Da-



Nach Definition der Parameter der „force.com-Sites“ und des „Customer-Portal“ lässt sich ein Profil für Benutzer der Plattform anlegen (Abb. 4).

bei sollen *User_Contact__c*-Datensätze immer paarweise existieren, wobei der einzelne Satz die folgende Beziehung darstellt: „Der Datensatzeigentümer kennt *Target_User__c* als Kontakt und gewährt *Target_User__c* die spezifizierten Zugriffsrechte auf sein Benutzerprofil“ (vgl. Abb. 5). Ein *Message__c*-Datensatz speichert eine einzelne Nachricht zwischen zwei Portalbenutzern.

Zur Gestaltung der Benutzeroberfläche mit dem UI-Werkzeug Visualforce und der zugehörigen Controller in der Apex-Programmiersprache nutzt man die sich in Eclipse integrierende Force.com-IDE. Durch den geschickten Einsatz der Visualforce-Templates und -Komponenten und die Manipulation der Metadaten des Systems über die IDE wird die Implementierung auf Basis der Force.com-Plattform so flexibel gehalten, dass sowohl Änderungen am Datenmodell als auch an der Präsentationsschicht ohne allzu großen Entwicklungsaufwand möglich bleiben.

Der Ansatz erlaubt die schnelle Fertigung eines Prototypen, der dann durch Iterationen in das fertige Produkt für die Film Tune AG überführt wird. Die iterative Arbeitsweise ist eines der Merkmale, das die Entwicklung im „As a Service“-Universum von klassischer IT-Projektarbeit nicht selten unterscheidet. Ein weiteres Merkmal ist die verteilte Entwicklung.

Zwar wird die Force.com-IDE wie das klassische Eclipse lokal auf dem Entwicklungsrechner installiert, die IDE kann jedoch nicht (wie aus dem Java-Umfeld gewohnt) lokal auf einen Compiler oder Debugger zugreifen. Vielmehr erfordert jede noch so kleine Änderung am Code ein direktes Einspielen auf die Plattform, wo dann der Compiler seine Arbeit verrichtet. Ferner verlangt eine Force.com-Richtlinie, dass

der überwiegende Teil des Codes durch Unit-Tests abzudecken ist, die sich in Form von *testMethods* direkt in den Code einbinden lassen. Ein Deployment lässt sich nur erfolgreich abschließen, wenn der serverseitige Compiler den übermittelten Code als fehlerfrei befindet und zusätzlich auch die Ausführung aller Testmethoden erfolgreich ist. Für umfangreiche Projekte verursacht ein Deployment dementsprechend eine merkliche Wartezeit; zudem erweist sich die Eigenheit der Testmethoden, einerseits umfangreiche Abdeckung zu verlangen und sich andererseits immer auf den gesamten Code zu beziehen, als hinderlich – insbesondere im Zusammenhang mit iterativen Anpassungen des Datenmodells.

Die Kooperation mehrerer Entwickler an demselben Projekt auf der Force.com-Plattform lässt sich derzeit nur durch strikte Trennung der Aufgabenbereiche auf unterschiedliche Dateien bewerkstelligen, da keine zuverlässige Versionsverwaltung zur Verfügung steht. Ebenso muss der Entwickler noch auf den Debugger verzichten; zur Fehlersuche stehen ausschließlich textuelle Ausgaben im „Debug-Log“ zur Verfügung.

Sicherheitsanforderungen und Verfügbarkeit

Die lokale Datenhaltung – die Vorhaltung einer entsprechenden Redundanz bei Hardware und sonstiger Infrastruktur vorausgesetzt – bietet durch die Nähe zum Anwender die beste Verfügbarkeit und höchste Performance. Ferner ermöglicht sie optimal auf den lokalen Bedarf abgestimmte Geschäftsprozesse. Zugleich verursacht die Vorhaltung der redundanten Infrastruktur eines lokalen Datenzentrums jedoch auch Kosten für

qualifiziertes Personal und Material. In kleinen Unternehmen ist der Ansatz noch ohne allzu große Probleme umsetzbar. Mit wachsender Unternehmensgröße steht die lokale Datenhaltung jedoch vor der Herausforderung, für alle Mitarbeiter weltweit einen einheitlichen Datenbestand bei hoher Verfügbarkeit vorzuhalten. Eine Kostenexplosion entweder für eine schnelle hochgradig ausfallsichere Anbindung der einzelnen Standorte an das Firmennetz oder für die Synchronisierung der Datenbestände verschiedener unabhängiger Standorte sind die Folge.

Die entfernte Datenhaltung des „As a Service“-Universums versucht sich der Aufgabe anzunehmen. Durch die gemeinsame Nutzung der hochverfügbaren Datenzentren, die auch das technische Personal vorhalten, werden die Kosten für die Infrastruktur auf mehrere Unternehmen verteilt. Zusätzliche Vereinbarungen zur Servicequalität und -verfügbarkeit sowie zu Nutzungsrechten und -pflichten regeln zwischen den Vertragsparteien, dass die Dienste jederzeit in bester Qualität zur Verfügung stehen. Idealerweise sind alle Unternehmensstandorte weltweit gleichberechtigt beim Zugriff auf die Firmendaten, ohne dass hierfür enorme Kosten die Bilanz des Unternehmens belasten; kleinere Einschränkungen in Funktionsumfang und Anpassbarkeit gleichen die Synergieeffekte bei der globalen Zusammenarbeit mehr als aus.

Bei der Suche nach passenden Anbietern kommt verständlicherweise die Frage, wo die Daten zu finden sind. Kann der Diensteanbieter die Frage zum Beispiel, ohne zu zögern, mit den Worten

beantworten: „Unser Hochsicherheits-Datenzentrum liegt in der Heilmannstraße 30 in Pullach im schönen Isartal und wird dort 24/7 von unserer Abteilung für technische Unterstützung nach deutschen Qualitätsstandards betreut“, dann weicht die erste Freude über den vermeintlich sicheren Standort schon bald der Frage, wie es bei nur einem Standort um die Datensicherheit im Katastrophenfall bestellt sein mag.

Im Regelfall wird sich kein eindeutiger Ort für die Speicherung der Daten benennen lassen, vielmehr übernimmt immer ein Verbund von Rechenzentren die Aufgabe. Bestenfalls lässt sich eine Region benennen, die alle für den Servicenutzer relevanten Daten verarbeitet. Im Beispiel etwa lagern die auf der Force.com-Plattform verarbeiteten Daten in den Vereinigten Staaten (primär in Ashburn, Virginia, und zusätzlich auf einem Failover-System im kalifornischen San José, CA), die genutzten S3-Dienste der Amazon Web Services werden in der EU, genauer gesagt in Irland, gehostet.

Auf diese geografischen Angaben des Serviceanbieters vertraut man als Servicenutzer ebenso wie auf die Zusagen hinsichtlich der Sicherheit und Verfügbarkeit des gesamten Rechenzentrumsnetzes. Die Bindung an einen Partner in der Datenverarbeitung darf nicht nur aus einem bloßen Bauchgefühl heraus erfolgen, sondern bedarf – nicht zuletzt aus juristischen Gründen – einer genauen Prüfung im Vorfeld.

Für Unternehmen mit Standort in Deutschland ist das Bundesdatenschutzgesetz (BDSG) die Grundlage für die Erhebung, Verarbeitung und Nutzung personenbezogener Daten; es setzt die

Datenschutzrichtlinie 95/46/EG des Europäischen Parlaments und des Rates vom 24. Oktober 1995 in deutsches Recht um und wird regelmäßig an die aus dem technischen Fortschritt resultierenden neuen Anforderungen angepasst. Einerseits erhalten die Unternehmen damit verbindliche Grundlagen für die Datenverarbeitung, andererseits werden aber auch enge Grenzen für den Umgang mit personenbezogenen Daten gesetzt. Umfassende Informationen zum Thema bietet der Bundesbeauftragte für den Datenschutz und die Informationsfreiheit in diversen Veröffentlichungen, etwa [2], für die Allgemeinheit zugänglich an.

Für das „As a Service“-Universum sind zwei Bereiche des Gesetzes von besonderem Belang: Erstens gibt es die Regelungen zu den technischen und organisatorischen Maßnahmen, die zum Schutz der Daten vor Missbrauch, Fehlern und Unglücksfällen zu treffen sind. Das Gesetz definiert bewusst keine konkreten Maßnahmen, sondern spezifiziert vielmehr in der Anlage zu § 9 Anforderungen, die solche Maßnahmen zu erfüllen haben, etwa die Zugangs- und Zugriffskontrolle für Datenverarbeitungsanlagen, den Schutz vor unbefugtem Lesen, Ändern und Löschen von Daten auch während der Übertragung, die Nachvollziehbarkeit von Änderungen, den Schutz der Daten im Katastrophenfall, das Verbot der Mischung von zu unterschiedlichen Zwecken erhobenen Daten sowie die Maßgabe, die Datenverarbeitung im Auftrag durch Dritte auf einen definierten Umfang zu begrenzen.

Für Datenübermittlungen innerhalb der Europäischen Union und an andere Vertragsstaaten des Abkommens über den Europäischen Wirtschaftsraum ist die Situation vergleichsweise einfach, da diese ohne weitere Detailprüfung genauso wie inländische Datenübertragungen eingestuft werden. Auch für Argentinien, Guernsey, die Isle of Man, Kanada, die Schweiz und Ungarn hat die EU-Kommission entsprechende Einstufungen vorgenommen. Jedoch war für den Datenverkehr mit den Vereinigten Staaten ein Sonderweg zu schaffen, da dort keine dem europäischen Standard nachkommende Datenschutzgesetze existieren. Daher sind Datenübertragungen – sofern überhaupt nach deutschem Recht zulässig – in die Staaten nur erlaubt, wenn sich der dortige Empfänger freiwillig den Regeln der „Safe Harbor Principles“ [e] unterworfen hat. Für andere Staaten sind

Ansicht der Datenstruktur der beiden „Custom Objects“ im Salesforce.com-Setup (Abb. 5)

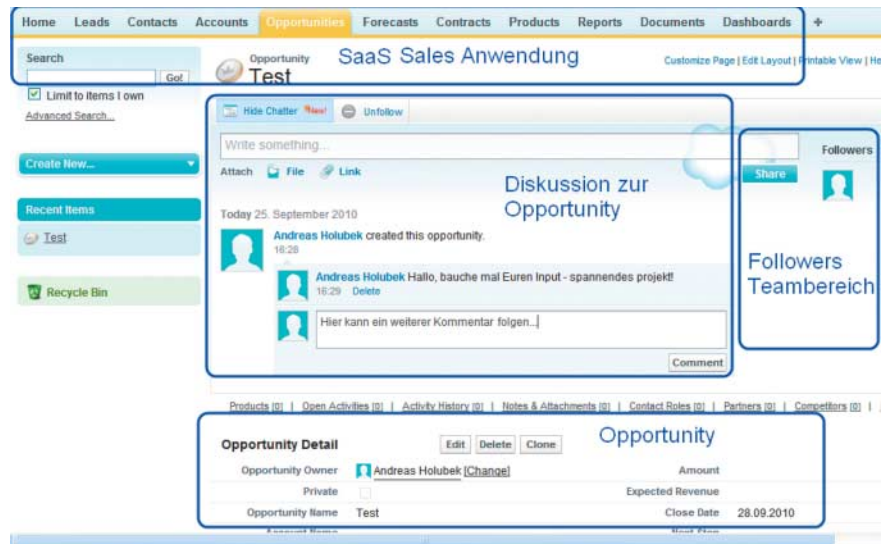
The screenshot displays the Salesforce Setup interface for two Custom Objects. The top section shows the configuration for the 'User Contact' object, including its singular and plural labels, object name, API name, and various tracking options. Below this, the 'Standard Fields' table lists fields like 'Created By', 'Last Modified By', 'Owner', and 'User Contact Number'. The bottom section shows the configuration for the 'Message' object, including its singular and plural labels, object name, API name, and deployment status. Below this, the 'Standard Fields' table lists fields like 'Created By', 'Last Modified By', 'Message Name', and 'Owner'. The bottom section also shows the 'Custom Fields & Relationships' table for the 'Message' object, listing fields like 'Date sent', 'Message Body', 'Parent Message', 'Priority Request', 'Recipient', 'Recipient Marked Deleted', 'Sender', 'Sender Marked Deleted', 'Status', 'Subject', and 'Type'.

keine vordefinierten Regeln verfügbar, das heißt, eine Einzelfallprüfung muss über die Zulässigkeit der Datenübertragung entscheiden; eine Sonderregelung kann sich zum Beispiel auf die in § 4c des Bundesdatenschutzgesetzes definierten Ausnahmen stützen. Der Betroffene kann etwa seine Einwilligung gegeben haben oder die Datenübermittlung kann für die Erfüllung vertraglicher Angelegenheiten zwischen dem Betroffenen und der verantwortlichen Stelle erforderlich sein. In jedem Fall ist die datenerhaltende Stelle durch die datenübertragende Stelle auf die Nutzungseinschränkungen hinzuweisen.

Perspektiven und Ausblick

Betrachtet man sich die nahe Vergangenheit der Plattformen und ihrer SaaS-Anwendungen, fällt eines auf: Es waren eher traditionell geprägte Umsetzungen. Darunter ist zu verstehen, dass sie zwar gut ihre jeweilige Arbeit verrichten, jedoch keinen Gebrauch von den vielfältigen Kommunikationsmitteln des Internet-Zeitalters machen. Dadurch öffnet sich jedoch eine Schere zwischen Anwendern der Systeme auf der einen und dem Kommunikationsstil vieler Anwender auf der anderen Seite. Oft ist zu beobachten, dass ein Benutzer zeitgleich mehrere Software wie Facebook, Twitter (Kommunikation) und auch SaaS-Anwendungen (CRM, Projektmanagement...) nutzt.

Demzufolge kann man gerade jetzt ein Verschmelzen beider Welten – Softwareanwendungen und Kommunikation – beobachten. Der Vorteil für den Benutzer ist enorm, nicht er muss mehr schauen, ob es etwas Neues gibt (ein neuer Account, eine neuer Auftrag oder auch eine neue Aufgabenbeschreibung), vielmehr ist es die SaaS-Anwendung, die über eine Veränderung in den Daten benachrichtigt. Abbildung 6 zeigt beispielhaft ein solches Zusammenspiel. Es handelt sich um eine klassische Anwen-



Zusammenwachsen von Anwendung und Kommunikation am Beispiel einer Sales-Anwendung mit Chatter auf Force.com (Abb. 6)

dung zur Vertriebsunterstützung. Neue Opportunities lassen sich erfassen und bearbeiten. Das ist an sich ja nichts Spannendes, wäre da nicht die Integration eines Feed. Zusätzlich zum Datensatz (Opportunity) lassen sich somit Diskussionsfäden aufbauen, die sich nun über das Kommunikationsmittel der Wahl verfolgen lassen – je nachdem was der Anwender als Lieblings-Social-Network nutzt. Allerdings sollte man aufpassen, keine sicherheitsrelevanten Informationen nach außen zu geben.

Der Artikel beleuchtet den Status quo des „As a Service“-Universums, und das Verhalten der Anwender zeigt deutlich: SaaS ist die erste Wahl bei neuen Geschäftsanwendungen. Während jedoch in der Vergangenheit häufig die Geschäftsführung und die Anwender solche Anwendungen vorangetrieben haben (nicht die IT), nimmt die Seite der IT immer mehr an Bedeutung zu. Für den Anwender der Applikation bleibt alles beim Alten – das muss auch so sein.

Für die Entwicklungsseite von SaaS geht es von einem reinen Customization-Modell der Entwicklung hin zu vollständigen Anwendungen. Das bedeutet, Services werden viel öfter von anderen Anbietern mit in die Anwendung hineingenommen. So bieten Amazon (zum Beispiel mit S3 oder EC2) und Microsoft (mit SQL Azure) ausgereifte Infrastrukturlösungen an, die man nahezu beliebig in eine SaaS-Lösung von Salesforce.com (Force.com-Plattform) einfügen kann. Solche Anwendungen sind jedoch weit vom früheren „Customizing“ entfernt. Daraus lassen sich einige Trends ableiten:
– Die SaaS-Entwicklung nähert sich der Standard-Anwendungsentwicklung an: Das erfordert neue Fähigkeiten der SaaS-Anwendungsentwickler.

– Es wird eine Zunahme von Infrastrukturkomponenten und ihren Anbietern geben.

– Entwickler von Plattformanwendungen bedienen sich immer mehr der Infrastrukturkomponenten und bauen mit diesen die Anwendung für den Kunden.

– Applikationen für den Kunden werden hochgradig an ihn angepasst.

– Die Integration erfolgt in immer stärkeren Maße über Webservices (SOAP- oder REST-Kommunikation).

– Die Entwicklungswerkzeuge werden in der Cloud zu finden sein.

– Durch Anforderungen der Anwender werden Plattformen zueinander kompatibel und Komponenten austauschbar. (ane)

ANDREAS HOLUBEK

arbeitet als VP Engineering für die arlanis Software AG und blickt auf eine langjährige Arbeit in Konzeption, Design und Programmierung von Software zurück.

GEORG SPENGLER

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 Aspekte der Datensicherheit.

Literatur

- [1] Christian Metzger, Juan Villar; Cloud Computing; Hanser Verlag, 2010
- [2] BfDI-Info 1: Bundesdatenschutzgesetz – Text und Erläuterung (14. Aufl., März 2009)

www.ix.de/ix1017137



Onlinequellen

- [a] Force.com
force.com
- [b] Amazon Web Services
aws.amazon.com
- [c] Windows Azure Plattform
microsoft.com/windowsazure
- [d] U.S. Department of Commerce;
Safe Harbor Frameworks
export.gov/safeharbor/