Community, Modularität und Empowerment in der Softwarebereitstellung
White Paper
Mai 2007

Rational. software

Die Zukunft der Softwareentwicklung

Danny Sabbah, PhD
General Manager, Rational Software

Highlights

Einführung

„*Jede Generation belächelt die alten Sitten und Gebräuche, hält sich aber peinlich genau an die neuen.*“

– Henry David Thoreau.

Spezialisten im Bereich Softwareentwurf und -implementierung können heutzutage wichtige aktuelle Trends nicht ignorieren: Serviceorientierte Architektur (SOA), Community-Software (auch bekannt als Open-Source-Software, OSS) und dezentrale Entwicklung in globalem Maßstab (Globally Distributed Development, GDD).[1] Ein Analyst schätzt, dass Ende 2006 in etwa 62 Prozent der Global 2000-Unternehmen eine SOA implementiert war,[2] ein anderer setzt diese Zahl noch höher an: „Zum Jahresende 2006 führen 9 von 10 Unternehmen serviceorientierte Architekturen ein oder haben diesen Prozess bereits abgeschlossen, sodass diese Unternehmen über Erfahrung mit SOA-Planung, -Entwurf und -Programmierung verfügen.“[3] Es wird erwartet, dass OSS bis zum Ende dieses Jahrzehnts in den unternehmenskritischen Softwareportfolios von über 75 Prozent der großen internationalen Unternehmen eingesetzt wird. Viele Umfragen deuten darauf hin, dass bereits heute die Mehrheit von uns OSS in der Produktion verwendet.[4]

SOA, OSS und GDD mögen den Eindruck unvorhergesehener Phänomene erwecken, trotzdem muss daran erinnert werden, dass sie nicht aus dem Nichts entstanden sind. Bevor man sich einer neuen Architektur, einem neuen Softwaremodell oder einer neuen Methode zur Entwicklung und Pflege von Software zuwendet, ist es ratsam, sich sorgfältig damit zu beschäftigen, worum es sich bei SOA, OSS und GDD handelt, was sie darstellen und woher sie stammen. Auf diese Weise können wir uns besser darauf vorbereiten, dass sie eines Tages massenhaft eingeführt werden.

Highlights

SOA ist nicht gänzlich neu. Bereits bei der Entwicklung der Enterprise JavaBeans-Technologie von Java™ und auch der CORBA-Architektur davor ging es wesentlich um die Bereitstellung homogener Services. Eine gute SOA ist offen, standardisiert, Community-basiert, Governance-fähig, modular und einfach verfügbar – einige dieser Konzepte gibt es bereits seit Langem. Für OSS gilt dies ebenso. Tatsächlich kann man im Zusammenhang mit der Softwareentwicklung eigentlich nicht von einer Erfindung von OSS sprechen.

Der Prozess ist der Schlüsselfaktor.

Open-Source-Software ist größtenteils in der akademischen Welt entstanden und wird bereits seit 25 Jahren verwendet. Während der letzten drei bis fünf Jahre hat sie sich aber zu einem wichtigen Faktor entwickelt, und das hauptsächlich wegen eines noch bedeutenderen Trends: Der Community-basierten Software. Die sichtbarsten Merkmale von OSS – Standortunabhängigkeit, Innovation und geringe Kosten – sind allgemein bekannt. Echten Mehrwert erhält die Softwareentwicklung und -bereitstellung über einen dreistufigen Prozess:

1. *Ausreichend gute Softwaremodule schaffen den Anreiz, dass eine breite Community bei der Erstellung von OSS mitwirkt.*
2. *Eine offene Entwicklungsumgebung begünstigt Innovation und Komponentenstandardisierung.*
3. *Die Standardisierung führt zu branchenweiter Einführung.*

Auch wenn das Produkt also OSS ist – der Prozess ist der Schlüsselfaktor, der Community-basierte Software ausmacht.

Es geht also eigentlich nicht um SOA, OSS, GDD oder weitere moderne Akronyme, sondern um die zugrunde liegenden Prinzipien und Prozesse: Interessengemeinschaften, modulare Systeme und Empowerment. Wenn wir uns nicht auf die Schlagwörter selbst, sondern auf die Qualitäten und Potenziale dieser Trends konzentrieren, können wir schon heute beginnen, diese Vorteile in unsere Software zu integrieren (soweit dies nicht bereits erfolgt).

Highlights

IBM verfolgt das Ziel, immer mehr Fachleute aus immer mehr Bereichen einzubeziehen.

Den Schwerpunkt der vorliegenden Abhandlung bildet die Integration der gegenwärtig rapide zunehmenden Anforderungen und Vorgehensweisen in die eigene Softwarebereitstellung. Das Ziel liegt darin, bewährte Verfahren gemeinsam mit heutigen Innovationen herauszufiltern, um exakt unsere Bedürfnisse befriedigen zu können. Die IBM Rational Group hat diese Erfahrungen bereits in die Erstellung und Pflege vorhandener Produkte und Lösungen einfließen lassen. IBM unternimmt alles, um diese Prinzipien in zukünftigen Produktreleases von Rational immer mehr Expertencommunitys zugutekommen zu lassen.

Die folgende Abhandlung gliedert sich in vier Abschnitte: Eine Kurzbeschreibung von Fragen der Softwareentwicklung und -bereitstellung aus der Vergangenheit, die sich auf die gegenwärtige Umgebung auswirken, die Identifizierung der wichtigsten Trends, die unsere Branche heutzutage maßgeblich beeinflussen, einen Entwurf der zukünftigen Softwarebereitstellungsstrategie der Rational Group sowie eine kurze Zusammenfassung.

Traditionelle Softwareentwicklung

Bei der Entwicklung von Software und Systemen sieht die Realität gegenwärtig so aus, dass es schneller möglich ist, eine neue Fertigungsstätte zu entwickeln und zu errichten, als ein neues System für Enterprise-Resource-Planning (ERP) zu implementieren. Es geht anscheinend schneller, einen Teilelieferanten in eine physische Lieferkette als einen Lieferanten in eine IT-Lieferkette zu integrieren.

Toyota legte zum Beispiel im Mai 1986 in Georgetown, Kentucky, den Grundstein für sein neues Montagewerk in Nordamerika. Nach nicht einmal zwei Jahren (um genau zu sein, nach 23 Monaten) rollten bereits die ersten Fahrzeuge vom Band. Die Installation eines ERP-Systems von SAP dauert im Durchschnitt fast drei Jahre (33,6 Monate).[5] Die Schaffung einer Struktur für das Routing digitaler Arbeitsergebnisse, Ideen und Prozesse zu einem IT-System dauert länger als die Errichtung eines Montagewerks für Fahrzeuge – vom ersten Spatenstich über den Antransport der benötigen Teile aus aller Welt bis zu dem Zeitpunkt, an dem das erste fertige Fahrzeug das Werk verlässt.

Highlights

Die Softwarebranche ist noch lange nicht erwachsen.

Mittlerweise ist die Softwarebranche über 50 Jahre alt, und die Diskrepanz zwischen physischer Implementierung und Softwareimplementierung ist immer noch groß. Das Problem liegt darin, dass sowohl die Softwarebranche selbst als auch die Softwareprozesse nicht voll ausgebildet sind – sie sind noch nicht ausgereift. Die Softwarebranche ist noch lange nicht erwachsen – obwohl sie gerade einen Wachstumsschub erfährt. Eine der Hauptursachen für diese Unbeständigkeit ist die starke Abhängigkeit von dem Fundament, auf dem wir unsere Software ausführen: Die Basis unserer IT-Infrastruktur entwickelt sich immer schneller. Die schnelleren CPUs haben uns ein hohes Maß an Ineffizienz bei der Entwicklung und Implementierung unserer Software gestattet.

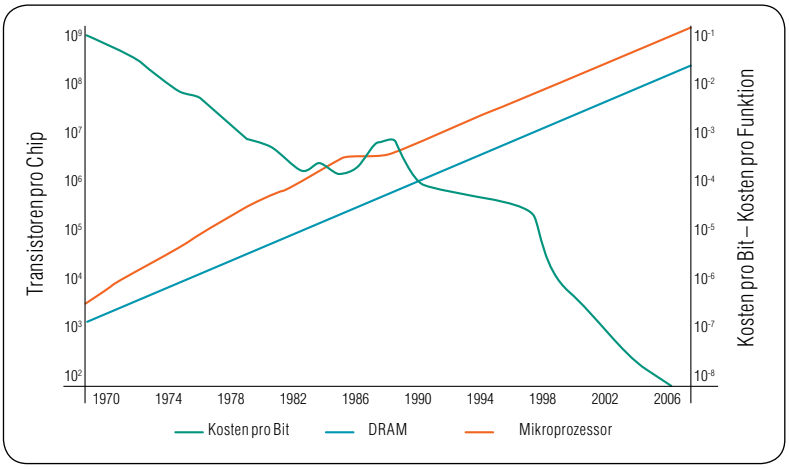

Abbildung 1. Mit der Zunahme an Speicher und Verarbeitungsleistung sinken die Kosten.

Abbildung 1 zeigt das Wachstum der Prozessorgeschwindigkeit gemäß dem Moore'schen Gesetz[6] sowie den Sturzflug der Kosten der Rechenleistung. Die schnelle Entwicklung von Hardware und Software erschließt der Technologie neue Anwendungsbereiche; wir sind heute in der Lage, Software zu erstellen und zu implementieren, die noch vor wenigen Jahren undenkbar war. Heute stehen allen Entwicklern Prozessoren und Hauptspeicher preiswert zur Verfügung, was zu einer unübersehbaren Zahl neuer Software-Frameworks geführt hat, die diese Vorteile nutzen. Traditionelle Anwendungen, die nicht in diese neueren Frameworks integriert werden können, haben außer ihrem eingeschränkten Anwendungsbereich einen weiteren entscheidenden Nachteil: Sie werden selbst zu einem Hindernis für die Entwicklung.

Highlights

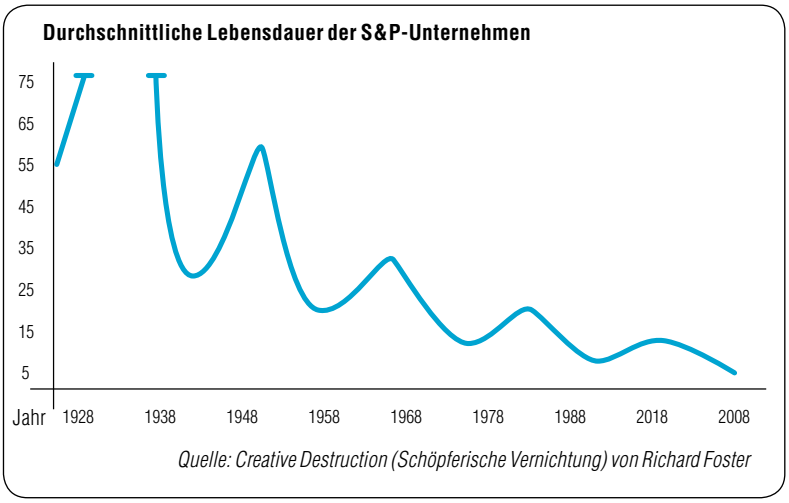

Abbildung 2. Die durchschnittliche Lebensdauer der S&P-Unternehmen ist seit 1930 drastisch gesunken.

Die Märkte von heute unterscheiden sich ganz wesentlich von denen vor 5 oder 10 Jahren.

Mit der Beschleunigung der Datenverarbeitungs- und Kommunikationstechnologien hat sich die durchschnittliche Lebensdauer der S&P 500-Unternehmen deutlich verkürzt. Um 1930 betrug die durchschnittliche Lebensdauer 75 Jahre, heute sind es 15 Jahre. Die sich rapide entwickelnden globalen Märkte und die Prinzipien der Marktkapitalisierung liefern uns einige wertvolle Erkenntnisse. Die Märkte von heute unterscheiden sich ganz wesentlich von denen vor 5 oder 10 Jahren. Das könnte uns zu der Annahme verleiten, alle Regeln hätten sich geändert. Aber dies ist nicht der Fall.

Highlights

Denken wir nur an die „New Economy“ in den späten 90er-Jahren und an die Kurs-Gewinn-Verhältnisse im Jahr 2000 – dort schien sich eine neue Realität in der Unternehmensbewertung anzukündigen. Und blicken wir dann in das Jahr 2001, in dem dies auf beeindruckende Weise widerlegt wurde. Unternehmen werden in immer schnellerer Folge gegründet und wieder aufgelöst, trotzdem sieht es – zumindest gegenwärtig – so aus, als würden für die Marktbewertung immer noch die alten Regeln gelten.

Dieselben Erkenntnisse können auch genutzt werden, um zu verstehen, wie mit dem Ansturm an technologischen Fortschritten umgegangen werden sollte, der gegenwärtig zu beobachten ist. Wir befinden uns gegenwärtig mitten in einem Orkan der Computertechnologie, in dem wir sehr leicht unsere fundierten Erkenntnisse in den Bereichen Wirtschaft, Konstruktion, Softwareentwicklung und Systemintegration aus dem Blick verlieren können. Wenn wir nur für einige Augenblicke aus diesem Orkan heraustreten würden, könnten wir leicht erkennen, was in seinem Gefolge verloren gegangen ist.

Methodiken der Softwareentwicklung, Konzepte des Projektmanagements, Programmiersprachen und scriptbasierte Sprachen sowie Unternehmensframeworks sind in ständiger Bewegung.

Die Softwarearchitektur hat noch nicht den Reifegrad der traditionellen Architektur in der physischen Welt erreicht. Auch die Technologie im Bereich der Lieferketten für Software befindet sich noch nicht auf dem Niveau an Funktionalität und Zuverlässigkeit, das die physischen Lieferketten im Bereich der Fertigung aufweisen. Methodiken der Softwareentwicklung, Konzepte des Projektmanagements, Programmiersprachen und scriptbasierte Sprachen sowie Unternehmensframeworks sind in ständiger Bewegung. Wie wir aber von den Aktienmärkten in den vergangenen zehn Jahren gelernt haben, neigen die Prinzipien, denen die Prozesse echter Wertschöpfung unterliegen, dazu, unverändert zu bleiben.

Die Wertschätzung vorhandener Prozesse und Methodiken, das Verständnis dafür, worauf wir unsere Anstrengungen konzentrieren sollten, und das Wissen, wann wir auf dem richtigen Weg sind – all dies sind außerordentlich wichtige Einsichten, mit deren Hilfe wir entscheiden können, welche der vielen Möglichkeiten, die uns heute zur Verfügung stehen, wir beibehalten und welche wir verwerfen sollten. Die IBM Rational Group hat eine Reihe von Trends identifiziert, die eine langfristige Vision ermöglichen und eine genauere Untersuchung rechtfertigen.

Highlights

Der kosteneffiziente Breitbanddatenzugriff und die soziale Vernetzung haben nachhaltige Auswirkungen.

Wichtige Trends von heute

Wenn wir auf den Anfang des 21. Jahrhunderts zurückblicken, dann gilt es für die kommenden Jahre als sicher, dass eines der bedeutendsten Phänomene, das nicht nur die Software-Community, sondern die Welt insgesamt beeinflusst, aus dem kosteneffizienten Breitbanddatenzugriff und der damit verbundenen sozialen Vernetzung besteht, die durch das Internet sowie durch die Mobilfunk- und Wireless-Technologie möglich werden. Die Verfügbarkeit von Mobiltelefonen in aufstrebenden Nationen hat zum Beispiel bereits nachhaltige Auswirkungen auf die Ökonomie vieler aufstrebender Länder. Und dieser Fortschritt gründet sich ausschließlich auf das gesprochene Wort. Nun stellen Sie sich einmal vor, was geschieht, wenn diese Gesellschaften ihre kommerziellen Aktivitäten auf das Internet ausweiten und beginnen, ohne die traditionellen Grenzen hinsichtlich Zeit, Raum und Kapital Wissensressourcen zu entwickeln.

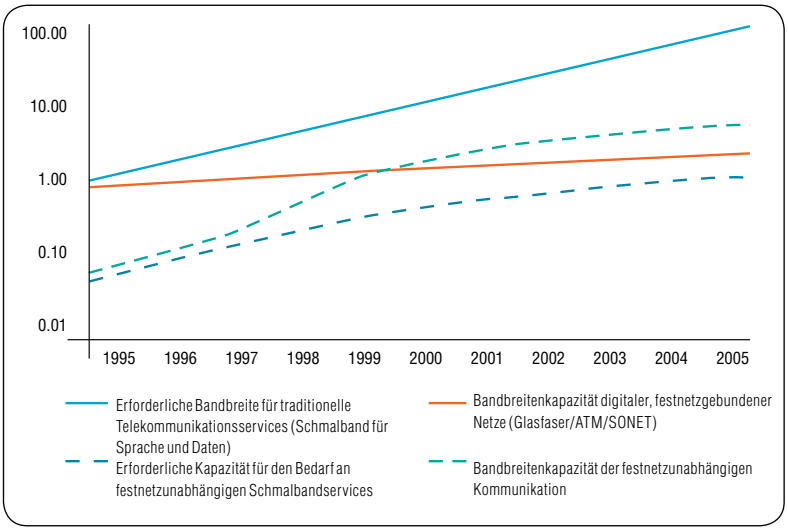

Abbildung 3. Die Bandbreite nimmt im Verlauf der Zeit ständig zu.

Highlights

Die Breitbandkapazität nimmt auch weiterhin Jahr für Jahr zu, das heißt, wir sind eigentlich nur Zeugen des Beginns dieser Entwicklung. Ein Ergebnis der gestiegenen Bandbreite bei unterschiedlichen Medien ist die Erkenntnis, dass die übermittelten Daten bzw. der Inhalt der Daten wichtiger ist als der Übermittlungsmechanismus. Die Telekommunikationsbranche hat bereits eine Konvergenz zwischen Daten- und Informationsströmen erlebt; Fernsehen, Internet und Telefonie sind nicht mehr voneinander getrennt. Es geht um die Geschwindigkeit, Zuverlässigkeit und Verfügbarkeit der Datenübertragung, wobei dem Endbenutzer die Entscheidung bezüglich Standort, Format und Gerät für das Hosting der Inhalte überlassen bleibt.

Die vielen verschiedenen Möglichkeiten des Datenzugriffs verändern unser Leben. Das Metcalfe'sche Gesetz[7] besagt, dass der Wert eines Netzes proportional dem Quadrat der Anzahl der Benutzer des Systems ist: $\frac{n(n-1)}{2}$. Mit anderen Worten: Je mehr Personen angeschlossen sind, desto größer ist der aus den Verbindungen generierte Wert. Durch die soziale Vernetzung wird dies bereits heute erwiesen.

Neue Communitys oder Untergruppen bilden sich im Umfeld interessanter und innovativer Ideen.

So wie Communitys im Bereich der sozialen Vernetzung jeweils eine eigene Agenda besitzen, besitzen auch Software-Communitys gleichermaßen transparente gemeinsame Ziele: Linux®-Entwickler sind an ihren Bibliotheken interessiert; Eclipse-Entwickler sind an ihren Projekten interessiert. Die Softwareentwickler haben heute eine Vielzahl an Möglichkeiten zur Mitwirkung an vorhandenen Communitys oder zur Erstellung eigener Communitys. Bei fehlenden formalen Governancestrukturen werden diese Communitys häufig durch rücksichtslose Meritokratien hervorgebracht und gesteuert. Neue Communitys oder Untergruppen bilden sich im Umfeld interessanter und innovativer Ideen. Jeder, der bisher mit der Überwachung dieser Diskussionsgruppen zu tun hatte, kennt die scharfe Kritik an demjenigen, von dem unvollkommene Ideen oder mangelhafter Code stammen.

Highlights

Technologiecommunitys sind einflussreich: Die Rational-Software-Community, Microsoft®-Benutzer, Systemadministratoren und andere. In der Tat sind solche mitwirkungsfreien Communitys allein durch ihre Größe die stärkste Kraft bei der Festlegung von De-facto-Standards. Denken wir nur an die Communitys der C-Entwickler, der Java-Entwickler, der PHP-Benutzer (PHP – persönliche Homepage), der Websiteentwickler usw., die nicht nur eigene Domänen gebildet, sondern auch die Art und Weise vorgegeben haben, in der wir Software erstellen. Der Unterschied besteht in der Geschwindigkeit, mit der sich diese Communitys heutzutage bilden können. Während die C-Technologie noch zehn Jahre gebraucht hatte, um allgemeine Popularität zu erlangen, brauchte die Java-Technologie dafür nur fünf Jahre. Und die relativ neue Subversion-Technologie ist dabei, die zwanzig Jahre alte Technologie Concurrent Version System (CVS) bereits in den ersten zwei Jahren ihrer Existenz zu ersetzen.

Der Unterschied besteht in der Geschwindigkeit, mit der sich diese Communitys heutzutage bilden können.

Wenn es um Communitys geht, ist Geschwindigkeit jedoch nicht die einzige Dimension. Wie immer, so ist auch hier der neuralgische Punkt erreicht, sobald eine kritische Masse an Führungspersonen und Anwendern erkennt, dass eine bestimmte Technologie alles Benötigte besitzt, und sich dafür entscheidet.[8] Wirklich spannend in der sich rapide entwickelnden Softwarelandschaft von heute ist die Tatsache, dass sich bereits eine sehr kleine Gruppe von Personen als diese kritische Masse erweisen kann. Das bedeutet, dass die Veränderung in der Community durch die Bedürfnisse des Einzelnen gelenkt werden kann.

Werfen wir nun einen Blick auf eine Reihe von Trends, die im Zuge einer leicht verfügbaren Bandbreite und der Vielzahl an Verbindungsmöglichkeiten entstanden sind: Serviceorientierte Architektur, Community-basierte Software und die Umgebung, die für deren Erfolg verantwortlich ist. Innerhalb dieser Trends finden wir eine ganze Reihe an Prinzipien, wie zum Beispiel den wichtigen Prozess der Destillation, der eine komplexe Technologie auf das Wesentliche reduziert und sozusagen eine Anwendung des Ökonomieprinzips auf Softwareprobleme darstellt. Wichtig ist dabei, sich zum Beispiel daran zu erinnern, dass Linus Torvalds kein Betriebssystem erfunden, sondern lediglich vorhandene UNIX®-Module vereinfacht hat. SOA, die viele komplexe Technologien abdeckt, ist in ihrem Kern ein sehr einfaches Konzept: Atomare homogene Services. Dieser gemeinschaftlich gesteuerte Entwicklungstrend, von der Komplexität zur Klarheit, treibt alle Trends an, um die es hier geht. Die Rational Group hat schon lange erkannt, dass erfolgreiche Softwaretrends dieselben drei Grundeigenschaften aufweisen: Community, Modularität und Empowerment, denen wir uns einzeln zuwenden werden.

Highlights

Die Programmierung im Großen hat sich weiterentwickelt.

Community-basierte Softwareentwicklung
– Weiterentwicklung der Programmierung im Großen[9]

Falls Sie sich schon immer einmal an einer philosophischen Debatte ähnlich der Streitfrage beteiligen wollten, wie viele Engel auf den Kopf einer Stecknadel passen – die Blogosphäre ist voll von Beispielen. Googeln Sie doch einmal Folgendes: „Woraus besteht echte Open-Source-Software?“ Sie werden schnell feststellen, dass es extrem wenige Frameworks gibt, die von allen als „reine“ OSS angesehen werden. Sogar die Linux-Technologie als die vermutlich weltweit erfolgreichste Open-Source-Software gibt es (nach jüngster Zählung[10]) in mehr als 350 Versionen mit unterschiedlichem Open-Source-Charakter. Ruby wird (hauptsächlich) durch eine Person in Japan kontrolliert, und jedermann sucht nach einer Finanzierung. Die kommerziell orientierten Open-Source-Entwickler von heute haben einen langen Weg von der nicht gewinnorientierten Free Software Foundation (FSF) hinter sich. Wir wollen uns an dieser Stelle nicht in Glaubensfragen verlieren, sondern stattdessen jedes Software-Framework und Softwareprodukt anhand seiner Attribute beurteilen. Beginnen wir mit der grundlegendsten Definition von OSS, und arbeiten wir uns von dort aus vorwärts.

Offen

Die beste Definition von Open-Source-Software ergibt sich bereits aus dem Namen: Bei OSS steht Ihnen der Quellcode offen. Mit dem direkten Zugriff auf den Originalcode sind bestimmte Vorteile verbunden, jedoch nicht garantiert, wie wir noch feststellen werden.

Highlights

Einfache Änderung und Korrektur

Diese Änderung wird nicht durch jedes OSS-Framework garantiert, da es für OSS mehr als drei Dutzend Lizenzierungsmodelle gibt, die sich jeweils darin unterscheiden, was Sie tun müssen, nachdem Sie Änderungen am Code vorgenommen haben. Vorgenommene Änderungen können sich auch negativ auf Unterstützungsverträge auswirken. Außerdem ist die Freiheit, den Code verändern zu können, ein zweischneidiges Schwert: Während die Möglichkeit, Software einem neuen Zweck zuzuführen, für Entwickler ein Segen ist, stellt sie für IT-Manager ein Problem dar, wenn sie dann die neue Software pflegen sollen.

Einfach verfügbar

Jahrzehntelang haben die kommerziellen Softwareanbieter ihren Quellcode unter Verschluss gehalten, und das hauptsächlich deswegen, weil angenommen wurde (und in großem Umfang immer noch wird), dass sie ihre Wettbewerbsvorteile verlieren würden, wenn jedermann Zugang zu ihren internen Arbeitsergebnissen hätte. Diese Geheimhaltung hatte nicht nur negative Auswirkungen auf die Softwareeinführung, sondern machte auch einen hohen Vertriebsaufwand und eine Unmenge an Marketingliteratur erforderlich, um den Schleier zu lüften. Sobald ein Quellcode öffentlich zugänglich ist, kann er sehr schnell allen zur Verfügung stehen, was den Verkaufspreis gegen Null treibt und jedem die Möglichkeit gibt, selbst einzuschätzen, was die Software wirklich tut. Open-Source-Software ist niemals Standardsoftware.

Kostenfrei

Der eigentliche Aufwand beginnt bei OSS nach der Installation.

Aus praktischer Erfahrung wissen wir, dass Open-Source-Software ungefähr so kostenfrei ist, wie es Hundewelpen sind. Die wirklichen Kosten fallen bei OSS erst nach der Installation im Rahmen der Entwicklung, Implementierung und Wartung an. Dies ist aber nicht neu, es gilt für sämtliche Software. Wegen des einmaligen Charakters solcher Lösungen sind die Kosten für modifizierte OSS im Zusammenhang mit diesen Aktivitäten sogar noch größer. Wir haben bereits festgestellt, dass das vorherrschende Geschäftsmodell erfolgreicher OSS-Unternehmen darin besteht, ihr Geld mit dem Angebot von Unterstützung, Services und Schulung zu verdienen; das ergibt auch Sinn. Mit anderen Worten: Der Aufwand für die Welpen liegt in der Pflege und Fütterung. Und wie bei einem Welpen bedeutet bei OSS eine schlechte Wahl im Nachhinein einen höheren Aufwand.

Highlights

Modular

Modularität ist nicht automatisch eine Eigenschaft von OSS, sondern generell ein Markenzeichen eines guten Softwaredesigns. Dazu gehören saubere Trennungslinien, durchdachte Anwendungsprogrammierschnittstellen (Application Programming Interfaces (APIs)) und ein erweiterbares Framework, das modular aufgebaute Komponenten enthalten kann. Die besten OSS-Frameworks sind modular, wie es auch die besten kommerziellen Softwarepakete sind.

Communitybasiert

Dies ist nicht nur das Herz und die Seele erfolgreicher OSS, sondern das Herz und die Seele des Entwurfs und der Ausführung erfolgreicher Software überhaupt. Communitys und offene Standards existieren nebeneinander, da die vorherrschende Community auch den De-facto-Standard diktiert. Das Problem hierbei ist die Gewährleistung des bestmöglichen Standards, und der Schlüssel dafür ist Governance auf mehreren Ebenen. Die Gewährleistung eines Fehlerverfolgungssystems in einwandfreiem Zustand, das Aufrechterhalten einer lebhaften Diskussion, die Entwicklung nützlicher Standards – all dies ist das Ergebnis einer vernünftigen Governance auf jeder Ebene.

All dies ist das Ergebnis einer vernünftigen Governance auf jeder Ebene.

Zusammenfassend kann festgestellt werden, dass OSS durchaus viele bekannte Attribute von Qualitätssoftware besitzen kann (aber nicht notwendigerweise besitzen muss): Modularität, Verfügbarkeit, kalkulierbare Kosten (Anschaffungspreis plus Wartungskosten), offene Standards, Zuverlässigkeit, Leistung und eine florierende Community an Benutzern mit guter Governance. Der eigentliche Punkt dabei ist, dass diese Qualitätsmerkmale nicht ausschließlich auf Open-Source-Software beschränkt sind, sondern das Ideal für sämtliche Software bilden. In dieser Hinsicht können wir dem Grundgedanken der obigen Diskussion der OSS-Entwickler zustimmen. Wir alle suchen Software, die diese positiven Qualitätsmerkmale besitzt, doch die Branche hat sich noch nicht darauf geeinigt, wie diese Qualitätsmerkmale genau realisiert werden sollen.

Highlights

Serviceorientierte Architektur

Bei SOA finden wir viele der Probleme aus der obigen Diskussion um OSS wieder. SOA bezieht sich jedoch auf ein gesamtes Unternehmen und nicht nur auf ein einzelnes Framework. Im Kern ist SOA eine Form der Architektur für Informationssysteme, die die Erstellung von Anwendungen ermöglicht, die aus flexibel verbundenen und funktionell aufeinander abgestimmten Services bestehen. Diese Services interagieren auf der Basis einer formalen Definition (bzw. eines Vertrags), die von der zugrunde liegenden Plattform und Programmiersprache unabhängig ist.

SOA-Services sollten sorgfältig definiert (in sich geschlossen), erkennbar und flexibel angebunden sein, damit sie ausreichend autonom und abstrakt (wiederverwendbar) sind und in komplexere Services eingebunden werden können, die selbst die obigen Attribute besitzen.

Es ist leicht, den Denkfehler zu begehen, dass eine SOA einer bestimmten Technologie entspricht.

Da sich SOA-Standards noch in der Definitionsphase befinden, ist es leicht, den Denkfehler zu begehen, dass SOA einer bestimmten Technologie oder einer bestimmten Lösung entspricht. Noch vor einigen Jahren dachten viele, dass *SOA = Web-Services*, da Web-Services dazu gedacht sind, die primären Anforderungen von SOA-Services zu erfüllen. Dass *SOA ≠ Web-Services* gilt, dürfte mittlerweile jedem klar sein. Trotzdem wiederholen diesen Irrtum all jene, die denken: *SOA = ESB* (Enterprise-Service-Bus). Eine andere Technologie, eine ebenso falsche Gleichsetzung. In der Tat sind Web-Services und ESBs nützliche Werkzeuge innerhalb einer sorgfältig ausgebildeten SOA. Mangelhafte Web-Services und über einen ESB instanziierte, schlecht konzipierte Strukturen bei gleichzeitigem Fehlen einer eleganten und durchdachten SOA – dieser Weg führt ganz einfach schneller ins Chaos.

Highlights

SOA besteht zu 1 Prozent aus Services und zu 99 Prozent aus Governance.

Der Irrtum liegt natürlich in der Annahme, eine einzelne Technologie oder eine Kombination aus Standards würde ausreichen, Ihr Unternehmen auf SOA umzustellen. Nichts könnte falscher sein. Die Realität liegt näher an der berühmten Maxime von Thomas Edison, die da lautet: „Genie ist 1 Prozent Inspiration und 99 Prozent Transpiration." Deshalb besteht SOA zu 1 Prozent aus Services und zu 99 Prozent aus Governance.

Eine erfolgreiche SOA ist *harte Arbeit*, und sie erfordert den gemeinsamen Willen vieler voneinander unabhängiger Abteilungen und Organisationen, ihre eigenen Interessen dem Wohle des Ganzen unterzuordnen. Diese gemeinsame Zielorientierung ist schon in einem kleinen Team einzelner Personen schwierig zu erreichen. Noch entmutigender ist es, wenn man die geometrische Zunahme der Wendepunkte zwischen allen Teams und allen Organisationen innerhalb eines Unternehmens berücksichtigt, die zusammenarbeiten müssen, um eine SOA zu realisieren.

Schwierig, aber nicht unmöglich. Eine SOA besitzt eine größere Erfolgswahrscheinlichkeit, wenn die Vorteile von offenen Standards, Modularität und einer Community mit guter Governance genutzt werden. Und wie bei OSS ist Governance das Attribut, das für den Gesamterfolg erforderlich ist. Governance kontrolliert die Einhaltung von Standards, die Integration modularer Komponenten und letztlich die Fähigkeit Ihrer SOA-Community zu einer sinnvollen Zusammenarbeit.

Highlights

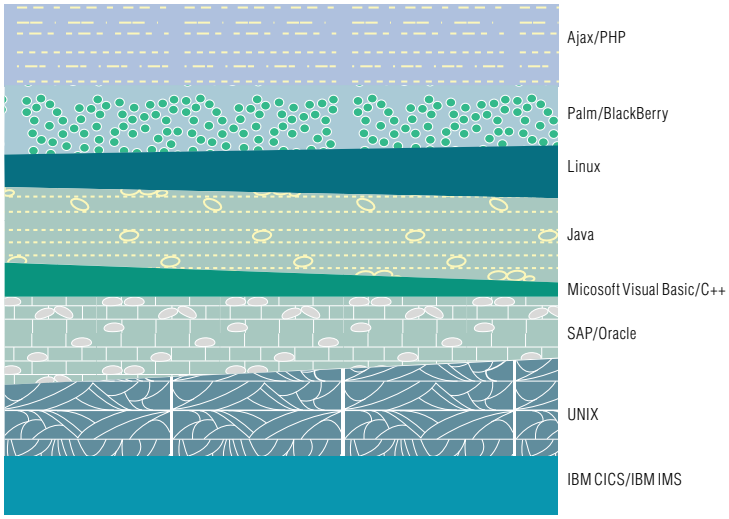

Abbildung 4. Daten, Services und Prozesse auf verschiedenen Ebenen.

Denken Sie daran, was Banken und Finanzwirtschaft bereits mit Daten in einer SOA anstellen können.

Um die Services, Daten und Prozesse, die unter vielen Softwareebenen vergraben sind, freilegen zu können, müssen die Produkte, die für Ihr Unternehmen von Wert sind, in gemeinsamer Anstrengung extrahiert und wiederverwendbar gestaltet werden. Denken Sie daran, was Banken und Finanzwirtschaft bereits mit Daten in einer SOA anstellen können. Konten, Investitionsvorgänge, Kreditkarten, Hypotheken und Darlehen können jetzt über eine einheitliche Benutzerschnittstelle verwaltet werden. Diese Services sind mit variablen Zahlungsplänen, automatischen Ein- und Auszahlungen, Dividendenausschüttungen, Finanzanalysen und Steuerrechnern verbunden – die Liste ist nahezu endlos. Diese Services repräsentieren eine Vielzahl offener Artefakte, die sicher und nahezu fehlerfrei in die ganze Welt übertragen werden können. Und dies sind nur die Dinge, die heute im Bankwesen möglich sind; vieles steht erst noch bevor.

Highlights

Wie wir sehen werden, macht die Rational Group Fortschritte in Richtung auf eine Vereinfachung Ihres SOA-Entwicklungsprozesses. IBM weiß um die Bedeutung der Entwicklung im Rahmen von Communitys, der gemeinsamen Informationsnutzung und einer angemessenen Governance während des gesamten Lebenszyklus von Anwendungen.

Governance und Empowerment

Zu oft führt die Erwähnung des Begriffs „Governance“ unmittelbar zu Gedanken an die Einhaltung von Vorschriften: Einhaltung von Vorschriften des Sarbanes-Oxley-Gesetzes (Sarbanes-Oxley Act) und des Health Insurance Portability and Accountability Act (HIPAA). Gleichzeitig führt sie zu Überlegungen hinsichtlich der Nichteinhaltung von Vorschriften und hinsichtlich harter Strafen für Gesetzesübertretungen. Das Netz der Governance deckt im Bereich der Softwareentwicklung und -bereitstellung viel mehr ab als nur die Implementierung vorhandener rechtlicher Vorgaben. Governance ist nicht nur ein Framework zum Zweck der Aufsicht, sondern ein Satz von Standards, deren Umsetzung ein Softwareteam fördern kann, statt es zu behindern. Zum Beispiel hat das Framework der Projektgovernance, das in den Arbeitsergebnissen des Eclipse-Projekts sichtbar ist, eine der kreativsten und am schnellsten wachsenden Open-Source-Communities maßgeblich gefördert.[11]

Governance ist nicht nur ein Framework zum Zweck der Aufsicht, sondern ein Satz von Standards, deren Umsetzung ein Softwareteam fördern kann, statt es zu behindern.

Die Einhaltung von Vorschriften entsteht dann aus erfolgreicher Governance und nicht aus der Orientierung an den Maßnahmen zur Softwarebereitstellung. Ob C, Java, das Internet, XML, Linux, Open-Source-Software oder SOA – nichts kann ohne Governance existieren. Alle diese Technologien, Architekturen und Frameworks haben sich gerade deshalb so gut entwickelt, weil Standards vorhanden sind, Normen, die dem Verhalten, der Diskussion und der Implementierung Grenzen vorgeben. Sie erlauben einer immer größer werdenden Zahl an Benutzern, sich auf die Stabilität der Technologien zu verlassen und sie effektiv zu nutzen. Ohne die Einhaltung der Regelwerke des American National Standards Institute (ANSI) und der International Organization for Standardization (ISO) wäre C niemals die Verkehrssprache der PC-Welt geworden. Auch gäbe es ohne World Wide Web Consortium (W3C) und Internet Corporation for Assigned Names and Numbers (ICANN) kein Internet und ohne C und das Internet keine Linux-Lösungen. OSS und SOA stehen auf diesen breiten Schultern und sind mindestens ebenso stark an die Anforderungen einer richtigen Governance gebunden wie ihre Vorgänger. Ohne Governance lassen Sie eine Menge Gelegenheiten ungenutzt.

Highlights

Ein Tempomat ermöglicht einem Auto die Einhaltung der vorgegebenen Geschwindigkeit – anders als bei einem einfachen „Abregler“, also einem Gerät, das lediglich eine Obergrenze festlegt. Der Tempomat ähnelt im Wesentlichen einer wirksamen Governance, da er einerseits den Betrieb mit der maximal zulässigen Geschwindigkeit und damit ein Maximum an Effizienz gewährleistet und gleichzeitig die Fahrzeit und die Anzahl der einfachen Vorgänge minimiert. Wenn wir diese Analogie weiterführen möchten, können wir uns dasselbe Fahrzeug vorstellen, wie es zeitlich abgestimmte Ampeln nutzen kann, die eine computergestützte Governance erfordern. Dies veranschaulicht, wie wir Governance zu unserem Vorteil nutzen können, um unsere Ziele mit einem Maximum an Geschwindigkeit und Effizienz zu erreichen. Das Gegenteil zu diesem Szenario würde entstehen, wenn wir uns vor der Einhaltung von Vorschriften fürchteten – wenn wir hinter jedem Baum einen Verkehrspolizisten sähen. Der Unterschied könnte nicht größer sein.

Wer Governance zum eigenen Vorteil nutzen möchte, muss sich mit anderen auf einen Standard, ein Framework oder ein Produkt einigen, der bzw. das unnötigen Aufwand minimiert, und die implizite Kontrolle nutzen, um ein Projekt in die richtige Richtung zu lenken. Governance ist ein entscheidender Bestandteil jeder erfolgreichen Softwarebereitstellung, OSS-Integration und SOA-Implementierung; eigentlich ist sie ein entscheidender Bestandteil jedes Aspekts bei der Erstellung, Implementierung und Pflege von Software. Es ist die Pflicht jedes Einzelnen und jeder Organisation, diese Kontrollen vernünftig einzusetzen.

Governance ist der Klebstoff, der alle Communitys unsichtbar miteinander verbindet.

Governance ist der Klebstoff, der alle Communitys unsichtbar miteinander verbindet. Es muss einen bestimmten Grund dafür geben, dass eine Community überhaupt existiert: Gemeinsame Ziele, gemeinsame Überzeugungen, gemeinsames Eigentum – es ist immer ein impliziter Vertrag vorhanden, der für die Bindung sorgt. Viele Communitys werden geschaffen, aber viele von ihnen scheitern bereits nach kurzer Zeit.

Highlights

Die wichtigsten Merkmale einer erfolgreichen Community sind Governance und Empowerment.

Damit eine Community lange genug existiert, um den breiteren Markt beeinflussen zu können, muss sie in einem solch guten Zustand sein, dass sie das intellektuelle Potenzial ihrer Mitglieder aktiv nutzen kann. Wir haben herausgefunden, dass die wichtigsten Merkmale einer erfolgreichen Community Governance und Empowerment sind und dass Empowerment des Einzelnen durch Modularität bei der Auswahl einer der Schlüssel für den zukünftigen Erfolg in der Softwareentwicklung und -bereitstellung ist.

Softwarebereitstellung in der Zukunft

> *„Die Leute können das Modell T in jeder Farbe haben – solange es schwarz ist."*
>
> *– Henry Ford.*

Das zu Beginn des 20. Jahrhunderts eingeführte Modell T von Ford hatte den richtigen Preis, war in großen Mengen verfügbar und eignete sich laut Werbung für fast alles. Es war das richtige Produkt zur richtigen Zeit, konnte aber nicht umkonfiguriert werden – es war nur in einer Größe, in einer Farbe und ohne jegliche Optionen verfügbar.

Gehen wir ins Jahr 2007, und sehen wir uns an, wie sich die Automobilindustrie verändert hat. Sie können mittlerweile Ihre Spezifikationen direkt an das Werk übermitteln und Ihr neues Auto online anpassen. Zu den Optionen gehören GPS-System, Satellitenradio, CD- oder MP3-Player, Klimaanlage, Regensensoren, Regulierung des Kraftstoffverbrauchs, Hybridantrieb und alternative Antriebe, Navigationssysteme, Einparkhilfen, Umgebungsvisualisierung und viele andere, die eine Menge Rechenleistung unter der Motorhaube benötigen. Ach ja, die Farbe können Sie natürlich auch wählen. Softwareintegration finden Sie in der Automobilindustrie sowohl im Fahrzeug als auch in der gesamten Fertigungslieferkette, so dass Produkte bedarfsgerecht geliefert werden können.

Highlights

Wir möchten unsere Technologie gemäß unseren Bedürfnissen konfigurieren können.

Sehen wir uns auch den Erfolg von Just-in-time-Einzelhandelsunternehmen wie Amazon, Netflix und anderen an, die heute tausende von Produkten auf Lager haben – viel mehr, als ein normales Geschäft bewältigen könnte – und selbst ausgefallene Wünsche erfüllen können. Wir leben in einem Zeitalter der kundenspezifischen Anpassung. Wir möchten unsere Technologie gemäß unseren Bedürfnissen konfigurieren können, und es gibt keinen Grund, etwas anderes zu erwarten, wenn es um Systemintegration oder Softwareentwicklung und -bereitstellung geht. Wir wissen, dass eine solche Anpassung ihren Preis hat, und haben nichts dagegen, ihn zu zahlen.

Lieferketten für Wissensressourcen

Willkommen im Zeitalter des Rattenschwanzes, des „Long Tail". Die „Long Tail"-Theorie entstammt dem gleichnamigen Buch von Chris Anderson (das übrigens über Websites verkauft wird, was Andersons Theorie bestätigt). Sie besagt, dass die Zukunft der Geschäftswelt darin besteht, weniger Artikel zu verkaufen, die sich an eine starke Nachfrage wenden, und dafür mehr Artikel, die auf sehr kleine Gruppen oder sogar Einzelpersonen zugeschnitten sind.[12] Für Softwareanbieter bedeutet dies, weniger Artikel für den Massenmarkt und dafür mehr individuell anpassbare Komponenten zu entwickeln. Kurz gesagt, generische Frameworks mit einer Vielzahl angepasster Plug-ins.

Highlights

Das „Long Tail"-Argument konzentriert sich mehr auf die Angebotsseite. Die Technologie ermöglicht eine Verschiebung im Kostenmodell der Angebotsseite – aus dem Schaufenster des Einzelhändlers zum Vertriebszentrum, wo wiederum ein größerer Bestand unterschiedlichster Artikeln möglich ist. Dies ermöglicht dann dem Anbieter die Nutzung des „Rattenschwanzes", also des „Long Tail"-Abschnittes der Nachfragekurve, durch die Bereitstellung eines breiteren Angebots.

In Anwendung dieses Phänomens auf die Softwarebereitstellung bedeutet dies die Übertragung der Auswirkungen von der Nachfrageseite auf die Softwarelieferanten auf der Angebotsseite. In diesem Sinne ist SOA sowohl ein Ausdruck für den Wunsch nach Auswahl als auch ein Mechanismus zum Ausgleich für den Mangel an Flexibilität im Zusammenhang mit den vielen Ebenen in den Softwarearchitekturen. Wir reden dabei über nichts Geringeres als die Offenlegung von Informationen, Daten und Produkten beim Aufbau von Lieferketten für Wissensressourcen.

In Bezug auf die Softwareentwicklung bedeutet dies, dass die meisten von uns den „Rattenschwanz" des Verwendungsmusters für verschiedene Entwicklungsaufgaben nicht effizient und effektiv nutzen. Allzu oft wedelt der Schwanz mit dem Hund: Unternehmen werden gezwungen, sich beim Entwickeln und Testen an die verschiedenen Merkmale und Funktionen der verwendeten Werkzeuge zu halten. Die Anbieter von Softwarewerkzeugen beklagen die Tatsache, dass sich die Tester zu sehr auf den Test der Leistungsmerkmale konzentrieren statt auf den Geschäftszweck der Anwendungen. Die Softwarelieferanten sind jedoch mit dafür verantwortlich, da sie Tool-Suites anbieten, deren Merkmale das funktionale Schubladendenken fördern.

Die beste Möglichkeit zur Gestaltung dieser Produkte erfährt man von der Community selbst.

Für den bestmöglichen Weg nach vorn müssen die Anbieter auf der Nachfrageseite der Gleichung beginnen und sich anschließend klarmachen, wie die Angebotsseite aussehen muss, um die Auswahl und die Flexibilität zu bieten, die für dieses Problem erforderlich sind. Die beste Möglichkeit zur Gestaltung dieser Produkte erfährt man von der Community selbst.

Highlights

Mehr als 200 IBM Produkte enthalten Eclipse-Technologie.

Öffnung der Community für die Teilnahme

Mit der Freigabe von Eclipse an die Open-Source-Community im Jahr 2001 beabsichtigte IBM, die Entwickler näher an die offenere Java-basierte Middleware heranzubringen. Dahinter stand die Vorstellung von einer Welt, in der die Entwicklungsumgebung eines Kunden aus einer heterogenen Kombination von Werkzeugen besteht. Dazu wurden ein leistungsfähiges Thick Client-Framework geschaffen und alle Services für Plug-ins zugänglich gemacht – Plug-ins, die als vorhandene Eclipse-Projekte, Produkte anderer Anbieter oder Produkte unabhängiger Entwickler vorliegen konnten, die aber alle für eine einheitliche Plattform erstellt worden waren und so ein direktes Geschäftsumfeld aus Softwarewerkzeugen bildeten.

Eclipse wurde sehr schnell die weltweit beliebteste Java-Entwicklungsplattform für Unternehmen und wird heute durch 65 bis 75 Prozent der Java-Entwickler genutzt.[13] Dies spricht für die große Popularität von Eclipse bei Anbietern und Benutzern. Mehr als 200 IBM Produkte enthalten Eclipse-Technologie; die Standardisierung hin zu einer einzigen Entwicklungs- und Implementierungsplattform hat zu einer besseren Interoperabilität der Produkte beigetragen.

Die erklärten Ziele des Eclipse-Projekts waren Transparenz, Vorhersagbarkeit und ständiges Feedback. Diese Ziele haben zu einem beeindruckenden Grad an Fehlerfreiheit des Projekts geführt – einem der wichtigsten Merkmale bei der Erstellung von Qualitätssoftware. Eine gesunde Community, starker Teamgeist, leistungsfähige Builds, solide Meilensteine, das Angebot von Betaversionen und die ständigen Tests sind Faktoren, die ihren Anteil am hervorragenden Zustand des Eclipse-Projekts haben.

Highlights

Die Mitwirkung der Rational Group am Eclipse-Projekt in den vergangenen sechs Jahren hat bewiesen, dass der fehlerfreie Zustand eines Projekts letztlich wichtiger ist und entscheidend zur Softwarequalität beiträgt. Da sich der Zustand eines Projekts ständig verändert, wird die Gewährleistung der Fehlerfreiheit zu einer gemeinsamen Aufgabe des gesamten Teams. Eine gesunde Entwicklercommunity – eine, die Offenheit, Verantwortung und Beteiligung schätzt – kann besser auf den normalen Stress reagieren, der mit der Arbeit an sich ständig weiterentwickelnder Software verbunden ist: Die unerwarteten Änderungen, die neuen Anforderungen und die ständige Bewegung des Zielobjekts.

Rational denken

Für die IBM Rational Group bedeutet die Zukunft der Softwareentwicklung die Einbeziehung offener, dynamisch definierter Communitys, die in ihrer Arbeit mit einem Maximum an Freiheit für die Konfiguration und Anpassung ausgestattet sind, um modulare Lösungen zu erstellen, die bewegliche, reaktionsfähige Organisationen unterstützen. Governance – die Fähigkeit zur aktiven Nutzung der Energie unterschiedlichster Gruppen und heterogener Technologien – ist der Schlüssel zur Schaffung der Synergie, die notwendig ist, um diese Dinge fest miteinander zu verbinden. Kein einzelner Standard und keine Technologie kann die Geschwindigkeit vorgeben, mit der Entwicklungsabteilungen Geschäftsinitiativen voranbringen. Weder SOA noch OSS werden der treibende Faktor sein, sondern die wirksame Governance aus mehreren treibenden Faktoren wird zu gesunden Communitys führen und die Flexibilität unterstützen.

Governance ist der Schlüssel zur Schaffung der Synergie, die notwendig ist, um diese Dinge fest miteinander zu verbinden.

Diese wirksame Governance wird die Form einer einheitlichen Datenstruktur annehmen, die eine gemeinsame, echtzeitorientierte Nutzung der Informationen ermöglicht – anwendungs- und plattformübergreifend. Es ist eine Möglichkeit zur Integration der Kommunikation: Projektteammitglieder können im Zusammenhang mit dem aktuellen Projekt und den verwendeten Anwendungen untereinander Nachrichten austauschen – von den Unternehmensanalysten bis zu den Spezialisten für die Produktion. Es ist die Möglichkeit zur Softwareentwicklung in weltweit verteilten Teams und zur nahtlosen Integration dieser Produkte. Es ist die Möglichkeit, staatliche Prüfungen ohne zusätzlichen Aufwand zu bestehen, da die Einhaltung von Vorschriften bereits im Framework enthalten ist. Es ist eine grundlegende Verschiebung von rein rollenorientiertem zu teamorientiertem Denken. Der Schlüssel ist die stärkere Beteiligung des Teams an allen Phasen des Projekts.

Highlights

Die Zukunft der Softwareentwicklung und -bereitstellung ähnelt stark dem Jazz-Projekt von IBM Rational.

Die Zukunft der Softwareentwicklung und -bereitstellung ähnelt stark dem Jazz-Projekt von IBM Rational, einer Technologie, die Team-Tasks in den gesamten Bereitstellungszyklus für Software und Systeme integriert. Die Organisationen, die das Jazz-Framework verwenden, werden einschätzen können, wie sich die Änderung einer Anforderung auf Builds auswirkt, und können dadurch noch genauer bestimmen, was durch wen geändert werden muss. Mit dieser Integration in den gesamten Lebenszyklus wird eine werkzeuggestützte Prozessführung erreicht, bei der die Werkzeuge selbst den Entwicklungsprozess verstehen, für den sich das Team entschieden hat. Und durch Automatisierung kann mit der Integration in den Lebenszyklus sichergestellt werden, dass die Teammitglieder dem Prozess von außen folgen können.

Um den vollen Nutzen aus einer teamorientierten Governanceplattform für den gesamten Lebenszyklus ziehen zu können, müssen die Organisationen außerdem flexible modulare Komponenten auf der Basis offener Standards bereitstellen, die auf der Plattform ausgeführt werden. Das eine geht nicht ohne das andere. Dekomposition und Aufteilung der vorhandenen Merkmale und Funktionen in Komponenten sind parallele und gleichermaßen wichtige Arbeitsaufgaben.

Dies ist eine Aufgabe für die gesamte Community – wie bei Eclipse. Deshalb hat die IBM Rational Group die Community-basierte Software auf die nächste Stufe gebracht – nicht nur mit Blick auf die Verantwortung für Linux, Eclipse, Geronimo und weitere OSS-Projekte, sondern auch mit Blick auf die Schaffung von Communitys innerhalb der eigenen Softwareorganisation des Unternehmens. Die Funktionalität der IBM Rational Group zur Prozessanleitung wird entwickelt, um von flexiblen Verfahren zu strukturierten, weniger komplexen Konzepten zu gelangen. Sie wird Entwicklungsumgebungen jeglicher Größe unterstützen, von sehr kleinen Teams bis zu großen, dezentralen Organisationen. Die Organisationen werden die Möglichkeit haben, Werkzeuge und Technologien für die Softwareentwicklung und -bereitstellung nach Bedarf zusammenzustellen, um in jeder Phase der Transformation ihrer Entwicklungsumgebungen die optimale Lösung zu erreichen.

Highlights

Schlussfolgerung

Wir leben in einer aufregenden Zeit rapider Veränderungen, und unsere Software wird ständig weiterentwickelt, um den wachsenden Bedürfnissen gerecht zu werden. Die harte Realität zwingt uns, mit der Zeit zu gehen. Ansonsten würden wir beiseite gedrängt und liefen Gefahr, nicht erst in Jahren, sondern bereits in Monaten als veraltet auf dem Abstellgleis zu landen. Das heißt aber nicht, dass sich alles verändert hat, ganz im Gegenteil. Die Softwarehersteller stehen immer noch denselben Realitäten gegenüber: Die Softwaremodellierung und die Zusammenstellung der Geschäftsanforderungen sind immer noch von entscheidender Bedeutung, und hochwertige Software ist stets besser als die Alternative. Was sich natürlich geändert hat, sind die Geschwindigkeit, der Umfang und die Reichweite.

Die Antwort liegt in der Flexibilität.

Die Lösung ist nicht gänzlich neu, aber sie enthält neue Ideen. Von entscheidender Bedeutung wird die Anpassung Ihrer Softwareentwicklung und -bereitstellung an immer stärker segmentierte Kundenanforderungen werden. Wenn Sie sich ausschließlich auf die Bereitstellung immer neuer Merkmale und Funktionen konzentrieren, werden Sie scheitern. Die Antwort liegt vielmehr in Ihrer Flexibilität – der Fähigkeit, sinnvolle Daten und relevante Ressourcen und Produkte durch Ihre derzeitigen Ebenen an Wissensressourcen an die Oberfläche zu bringen: Ihre Lösung vereinigt sicherlich viele unterschiedliche Typen, Marken und Ebenen von Software.

Highlights

Die Softwareentwicklung und -bereitstellung der Zukunft ist durch drei Hauptprinzipien gekennzeichnet: Community, Modularität und Empowerment.

Wenn Ihre Daten für Sie arbeiten sollen, müssen Sie vorhandene Produkte in ihre Bestandteile zerlegen und im gesamten Unternehmen wiederverwenden. Sie müssen beweglich und anpassungsfähig bleiben, damit Ihre Kunden ihre Interaktion mit Ihrem Unternehmen individualisieren können. Der Erfolg dabei hängt davon ab, wie gewissenhaft Sie die drei Hauptprinzipien der Softwareentwicklung und -bereitstellung der Zukunft im Auge behalten: Community, Modularität und Empowerment.

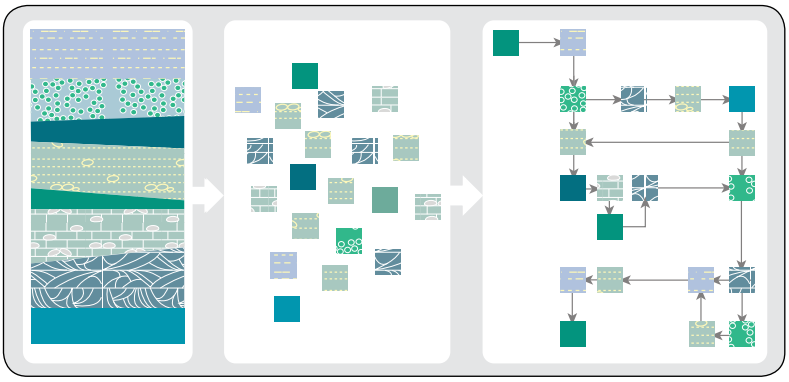

Abbildung 5. Offenlegung und Nutzbarmachung der Ebenen aus Daten, Services und Prozessen.

In Zukunft wird die Softwareentwicklung und -bereitstellung mehr einer Lieferkette ähneln: Einer horizontalen Einrichtung, die flexibel verbunden ist und einer wirksamen Governance unterliegt, um Ihren SOA- und GDD-Anforderungen gerecht zu werden. Außerdem müssen die Daten, Prozesse und Services auf rationelle Art extrahiert werden, um eine positive Kapitalrendite zu erzielen. In Zukunft geht es weniger um einzelne Entscheidungen hinsichtlich Anwendungen, Betriebssystemen oder Software. Stattdessen wird es mehr um die Bereitstellung des Service, die Modularität des Systems und die Einbeziehung Ihrer definierten und flexiblen Community gehen, um die richtigen Entscheidungen zu treffen.

Highlights

Die IBM Rational Group unternimmt bedeutsame Schritte.

Die IBM Rational Group unternimmt bedeutsame Schritte, um diese Zukunftsvision Realität werden zu lassen. Community, Modularität und Empowerment sind gleichzeitig Leitfaden und Bestandteil der aktuellen Rational-Produkte, des Jazz-Frameworks und aller zukünftigen Rational-Produktangebote.

Die Zukunft der Softwareentwicklung und -bereitstellung wird spannend. Sie können sicher sein, dass die IBM Rational Group auch weiterhin eine führende Rolle spielen wird.

Weitere Informationen

Weitere Informationen dazu, wie Sie IBM Rational-Produkte im Rahmen der Softwareentwicklung und -bereitstellung Ihres Unternehmens einsetzen können, erhalten Sie von Ihrem Vertriebsbeauftragten oder auf folgender Website:

ibm.com/software/rational

Endnoten

[1] Friedman, Thomas L., *The World is Flat: A Brief History of the Twenty-first Century*; 2005; Farrar, Straus and Giroux.

[2] Lo Giudice, Diego, *Service Oriented Architecture: The Foundation for Digital Business;* Forrester Research; 2006.

[3] *Enterprise Service Bus and SOA Middleware;* Aberdeen Group; Juli 2006.

[4] *Open Source in Global Software: Market Impact, Disruption, and Business Models;* IDC; Juli 2006.

[5] Nucleus Research; verschiedene ROI-Berichte; 2003 (http://nucleusResearch.com).

[6] http://en.wikipedia.org/wiki/Moore's_Law

[7] http://en.wikipedia.org/wiki/Metcalfe's_law

[8] Gladwell, Malcolm, *The Tipping Point;* 2002; Little, Brown & Co.

[9] http://en.wikipedia.org/wiki/Programming_in_the_large

[10] http://distrowatch.com

[11] Erich Gamma und John Wiegand, *The Eclipse Way;* 2006.

[12] Weitere Beispiele zum „Long Tail"-Postulat finden Sie unter www.longtail.com/the_long_tail

[13] Dieser Wert schließt die Verwendung weiterer Produkte ein, in deren kommerziellen IDEs Eclipse-Komponenten eingebettet sind, wie zum Beispiel IBM Rational Application Developer.

IBM Deutschland GmbH
70548 Stuttgart
ibm.com/de

IBM Österreich
Obere Donaustraße 95
1020 Wien
ibm.com/at

IBM Schweiz
Vulkanstrasse 106
8010 Zürich
ibm.com/ch

Die IBM Homepage finden Sie unter:
ibm.com

Hergestellt in den USA
05-07

RAW10983-DEDE-00