Leitfaden Web Services

- Architektur
- Integration
- Nutzen

BITKOM
Connecting visions - creating solutions

Impressum

Herausgeber:
BITKOM
Bundesverband Informationswirtschaft,
Telekommunikation und neue Medien e.V.
Albrechtstraße 10
10117 Berlin-Mitte

Tel.: 030/27 576 – 0
Fax: 030/27 576 – 400
bitkom@bitkom.org
www.bitkom.org

Eine Arbeit des Arbeitskreises
„Application Development & Integration Technology“

Herausgegeben im Oktober 2005

Ansprechpartner:
Dr. Birgit Heinz, BITKOM e.V.
Tel: +49 (0)30 / 27576 – 243
E-Mail: b.heinz@bitkom.org

Inhaltsverzeichnis

Zielsetzung und Aufbau des Dokuments

Web Services bieten eine technologische auf Standards basierende Grundlage, Anwendungssoftware unternehmensintern und / oder unternehmensübergreifend auszutauschen und zu integrieren. Web Services haben mittlerweile einen Reifegrad erreicht, der den einfachen Einstieg und den breiteren Einsatz unabhängig von Unternehmensgrößen und über Unternehmensgrenzen hinweg fördert.

Um aus der Vielzahl der mittlerweile verfügbaren Veröffentlichungen zu diesem Thema, den schnellen Überblick und leichten Einstieg zu finden, möchten wir mit diesem Leitfaden den entstehenden Informations- und Lernbedarf erleichtern und Erfahrungen aus dem Einsatz mit anderen Unternehmen teilen.

Das vorliegende Dokument bildet einen Einstieg in das Themengebiet „Web Services“, von der zu Grunde gelegten begrifflichen Abgrenzung über die historische Entwicklung hin zum derzeitigen Forschritt der Standardisierung und Blick in die zukünftigen Entwicklungen.

Das vorliegende Informationspapier stellt ein lebendiges Dokument dar, das kontinuierlich von den Autoren und dem Arbeitskreis Application Development & Integration Technologies weiter entwickelt wird. Alle die einen weiterführenden Beitrag oder einen Erfahrungsbericht über die Einführung im eigenen Unternehmen beisteuern möchten, sind herzlich eingeladen, den Kontakt aufzunehmen.

Der Leitfaden wendet sich an alle, die sich für die Potentiale „komponentenorientierter“ Anwendungssoftware interessieren und einen Überblick zu Technologie, Standardisierung und Anwendung gewinnen möchten. Im Mittelpunkt steht dabei die kompakte und einfache Darstellung der grundlegenden Prinzipien.

Alle Leser, die sich intensiver mit dem Thema beschäftigen möchten, finden in den jeweiligen Kapiteln stets weiterführende Literaturangaben.

Das Dokument gliedert sich in fünf Abschnitte, die unterschiedlichen Zielsetzungen und Zielgruppen dienen. Der erste Abschnitt gibt eine kurze Einführung in das Thema Web Services und zeigt die historische Entwicklung als auch gegenwärtige oder zukünftige Einsatzszenarien auf.
Der zweite Abschnitt wendet sich den konzeptuellen Aspekten der Web Service Architekturen und ihren Eigenschaften zu. Dabei finden neben der Diskussion der aktuellen Standardisierungsbemühungen auch ein Vergleich alternativer Integrationsarchitekturen sowie existierender Frameworks Berücksichtigung.
Das dritte Kapitel untersucht dann die praktischen Fragen der Implementierung. Dabei bilden Fragen wie Sicherheit und Authentifizierung, Entwicklungsmanagement oder Betrieb und Abrechnung Schwerpunkte der Betrachtungen.
Der Nutzen von Web Services steht im Mittelpunkt der Betrachtungen des vierten Kapitels. Dabei diskutiert das Kapitel sowohl technische als auch Management-Aspekte.
Der letzte Abschnitt lenkt die Aufmerksamkeit des Lesers in die Zukunft und gibt einen Ausblick auf Tendenzen und zu erwartenden Entwicklungen.
Ein Glossar sowie einige Anhänge komplettieren das Dokument.

1 Überblick und Einführung [Status Dezember 2004]

Zielgruppe Manager

1.1 Was sind Web Services

Der Begriff „Web Services“ wird in verschiedenen Zusammenhängen gebraucht, deren unterschiedliche Bedeutungen wir an dieser Stelle kurz erläutern.

Umgangssprachliche Bedeutung von Web Services
Der Begriff Web Services wird bei der allgemeinen Beschreibung von Services verwendet, z.B. von neuen Kundenservices, die über das Internet zugänglich sind. Kunden erhalten die Möglichkeit, „sich selbst zu bedienen“, etwa dann, wenn es um das Beschaffen von Informationen und / oder geschäftliche oder private Transaktionen geht, z.B. Überweisungen veranlassen, Kontostände abfragen, Bestellungen aufgeben, Lieferketten verfolgen. Die Unternehmen schaffen damit zusätzlichen Komfort und effiziente Kommunikationswege für ihre, Kunden, Lieferanten und Partner.

Unter dem Schlagwort „Web Services“ wollen Unternehmen diese Entwicklungen stärken. Mit Hilfe von webbasierten Services werden die Beziehungen zwischen Kunden, Unternehmen und Partnern neu gestaltet und ausgebaut.

Technische Bedeutung von Web Services[1]
Der Begriff Web Services meint die relevanten Technologien, die zur Umsetzung der vorher beschriebenen Szenarien aus Sicht der Informationstechnologien (IT) beitragen. Web Services basieren auf technologischen Standards wie SOAP, WSDL (Web Service Definition Language), UDDI (Universal Description, Discovery and Integration, http://www.uddi.org/) und XML. Sie stellen eine moderne und innovative Möglichkeit dar, verteilte Anwendungen (Dienste) zu realisieren. Diese Technologie ist jedoch nicht ganz neu, sondern stellt eine Kombination bereits etablierter Technologien dar, die das Zusammenspiel plattform- und programmiersprachenunabhängiger verteilter Anwendungen regeln. Web Services vereinfachen somit die Kommunikation zwischen Anwendungen in heterogenen Netzwerken.

Der vorliegende Leitfaden konzentriert sich auf die technische Entwicklung und Bedeutung.

1.2 Historische Entwicklung

Die Historie der Software Applikationsentwicklung ist über die Jahre geprägt durch die Verwendung von uneinheitlichen, zum Teil nicht standardisierten Technologien basierend auf unterschiedlichen Programmiersprachen, Betriebssystemen und Hardwareplattformen. Eine Integration oder Vernetzung dieser Applikationen ist daher nicht ohne weiteres möglich. Die meisten Systeme hatten und haben für die Kommunikation untereinander proprietäre, wenig flexible Schnittstellen in denen die Parameter genau festgelegt sind. Zu einer Verbesserung führten hierbei EAI-Werkzeuge (Enterprise Application Integration). Damit war eine unternehmensweite Integration unterschiedlicher Applikationen möglich, da diese Werkzeuge eine Reihe von Standardschnittstellen und Services zur Verfügung stellen. Um aber über Unternehmensgrenzen

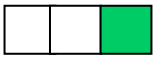
[1] Web Services ist der Oberbegriff für eine Sammlung aus den drei Technologien: SOAP, UDDI und WSDL. Web Services erlauben das maschinelle Auffinden und Nutzen von Services, bei denen es sich in der Regel um Softwaremodule handelt. Diese sind in Verzeichnissen (UDDI) beschrieben. Die Kommunikation bei der Suche und der Nutzung wird über das SOAP-Protokoll abgewickelt. Die Services (Funktionen) und deren Parameter werden in der WSDL-Sprache beschrieben.

hinweg eine einfache Integration, Kommunikation und Transaktion von Applikationen untereinander zu erreichen, waren weitere, vor allem technologische Hürden zu überwinden. Das Ziel war die Interaktion zwischen Programmen, die mit den heutigen Web Services möglich ist. Die Entwicklungsschritte hin zu Web Services lassen sich deshalb vor allem durch technologische Fortschritte charakterisieren.

Die erste große technologische Hürde auf dem Weg ‘grenzenloses’ E-Business, war der Zugriff auf das Internet. Jeder sollte auf Veröffentlichungen und Informationen im Internet einfach zugreifen können. Die Hürde wird 1995 und 1996 durch die Standardisierung von Web Servern und Web Browsern genommen. Heutzutage kann jeder schnell auf Informationen zugreifen und das Web Publishing ist alltäglich.

Ein großer technologischer Fortschritt ergab sich im E-Business mit der Einführung von Web Application Servern. Web Application Server stellen eine Vielzahl von Basisfunktionen bei der Applikationsentwicklung bereit. Diese Basisfunktionen umfassen Services, Kommunikation und auch Transaktionsmanagement. Die heutigen Application Server sind für die Integration webbasierter Lösungen optimiert und übernehmen auch Aufgaben wie Lastenverteilung etc. Die Entwicklung von Applikationen ist dadurch einfacher und schneller.

Weitere wichtige Schritte waren, einfache B2B und B2C Transaktionen über das Internet auszuführen. Sie führten zu Standards im Bereich E-Commerce und betreffen vorwiegend die Protokoll- und Applikationsebene.



Web Services bauen technisch auf den von den Web Application Servern bereit gestellten Basisfunktionen auf. Neu hinzu kommt die konsequente Nutzung von offenen, akzeptierten und von allen Anbietern unterstützen Standards, wie z. B. XML als Datenaustauschformat oder HTTP als Transportprotokoll. Diese Offenheit und Unabhängigkeit macht den eigentlichen Vorteil von Web Services gegenüber anderen Integrationstechnologien aus. Web Services funktioniert als einfaches Kommunikationsmodell zwischen Programmen, das auf der Grundlage existierender und neu aufkommender Standards arbeitet. Anwendungen sind damit schneller, einfacher und kostengünstiger zu integrieren. Die Weiterverwendung vorhandener Anwendungen ersetzt „Neu“-Entwicklung von Anwendungen.

Web Services setzen eine Serviceorientierte Architektur voraus. Eine Serviceorientierte Architektur, stellt nicht länger Daten, sondern Dienste in den Mittelpunkt. Sie, kann Anwendungen in Komponenten und damit in Diensten darstellen. Diese stehen dann über das Netzwerk bereit. Eine solche Architektur bietet auch die nötige Skalierbarkeit, um künftig neue Systeme und Anwendungen einzubinden und mit dem Wachstum des Unternehmens Schritt zu halten.

Quellen:

- IBM Software eNews (Ausgabe Juni 2002)
- GFT Whitepaper
- XML im Unternehmen - Briefing fürs IT-Management, Frank Bitzer, Galileo Computing,260 S., 2003, geb., 39,90 Euro, ISBN 3-89842-288-7

1.3 Einsatzgebiete

1.3.1 MapPoint.NET

MapPoint .NET ist ein von Microsoft betriebener, kommerzieller Web Service. Es ist Microsofts größter Web Service mit mehreren Millionen Anfragen pro Tag und bietet seinen Nutzern die Möglichkeit Navigationsfunktionen in eigenen Anwendungen zu verwenden. Dieser Dienst bietet seinen Nutzern die Möglichkeit geographische Navigation, Routenplanung und Kartenmaterial in seinen Anwendungen zu verwenden. Hierbei ist es unerheblich welche Art von Anwendung vorliegt und auch auf welcher Plattform die Anwendung laufen wird. Mobile Anwendungen auf einem Smartphone sind ebenso möglich wie Web Anwendungen, die in einem Rechenzentrum für eine große Menge von Nutzern laufen.

http://www.microsoft.com/mappoint/webservice/default.mspx

Weitere Referenzen

- "An Introduction to the Web Services Architecture and Its Specifications", Don Box, Christopher Kurt, Luis Felipe Cabrera, http://msdn.microsoft.com/webservices/default.aspx?pull=/library/en-us/dnwebsrv/html/introwsa.asp
- "Secure, Reliable, Transacted Web Services: Architecture and Composition", gemeinsames Whitepaper von IBM und Microsoft, http://msdn.microsoft.com/webservices/understanding/advancedwebservices/default.aspx?pull=/library/en-us/dnwebsrv/html/wsoverview.asp
- "Security in a Web Services World: A Proposed Architecture and Roadmap – A Joint White Paper from IBM Corporation and Microsoft Corporation", http://msdn.microsoft.com/webservices/understanding/advancedwebservices/default.aspx?pull=/library/en-us/dnwssecur/html/securitywhitepaper.asp
  Web Service Architekturen

Die Idee von Web Service Architekturen ist die Integration von Anwendungen unabhängig von ihren technologischen Vorraussetzung auf der Basis von Protokollen, die allgemein als Standards verabschiedet und eingesetzt werden. Ziele einer Web Service basierenden Architektur sind

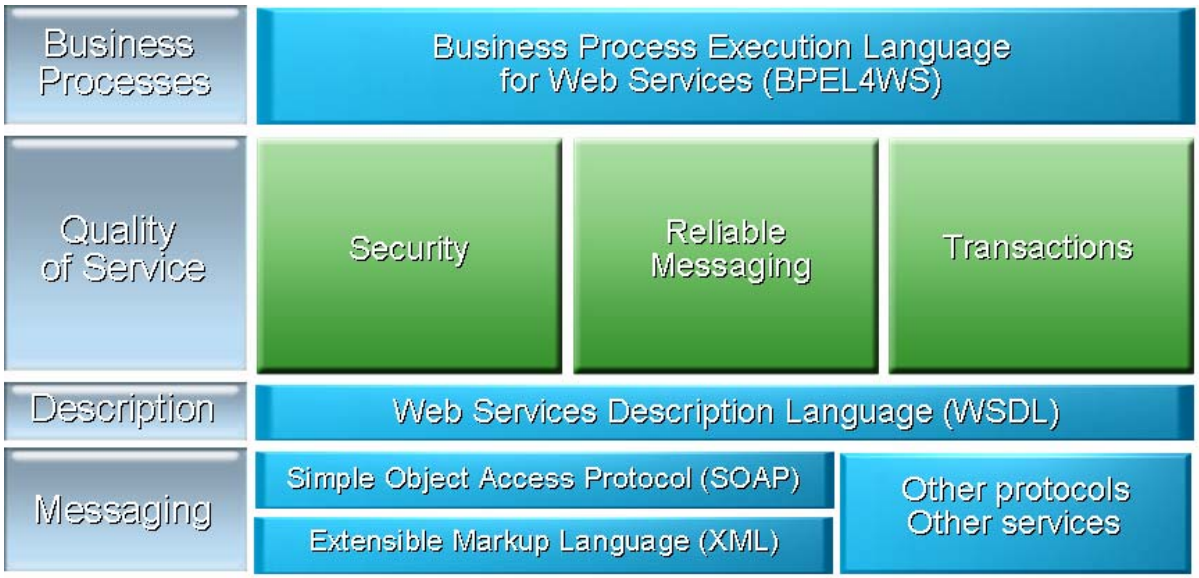
- Integration von Geschäftsprozessen – auch über Unternehmensgrenzen hinaus
- Kommunikation von Anwendungen
- Anbindungen von Anwendungen innerhalb von Organisation, von Partnern und Lieferanten über öffentliche Netze
- Sichere und zuverlässige Kommunikation der Service Partner
- Einfache und kostengünstige Verbindung von Diensten

Web Services sind inzwischen nicht mehr nur ein Hype Thema und sind bei den wichtigsten Plattformanbietern als zentrales Thema etabliert

- MSDN - Microsoft Developer Network
  http://msdn.microsoft.com/webservices
- IBM – SOA and Web services
  http://www-136.ibm.com/developerworks/webservices/?ca=drs-ws2404
- BEA – BEA SOA Resource Center
  http://www.bea.com/framework.jsp?CNT=index.htm&FP=/content/solutions/resource/soa/
- SUN Microsystems
  http://developers.sun.com/techtopics/webservices/index.html

1.4 Das Grundkonzept – oder die Standards

Als Basis für die Umsetzung der Ziele und Eigenschaften wurde eine Reihe von Protokollen basierend auf dem W3C Standard XML (http://www.w3.org/XML/) geschaffen. Hierbei werden verschiedene Ebenen der Protokolle unterschieden.

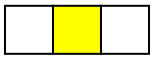


Auf der *Messaging* Ebene werden Nachrichten und ihre Meta Daten ausgetauscht. Der Nachrichtenaustausch wird durch den W3C Standard SOAP (http://www.w3.org/2000/xp/Group/) definiert. Nachrichten und ihre Nutzlast werden in Umschläge gesteckt, die durch ihre Meta Daten Auskunft über Inhalt, Absender und Adressat geben. Diese Metadaten sind erweiterbar und werden beispielsweise für erweiterte Web Services Standards, wie Sicherheit und Adressierung genutzt. Diese Protokolle sind ausgereift und sind auf breiter Basis in Produkten verfügbar.

Die Ebene der *Description* beschreibt Dienste und ihre Eigenschaften für die Nutzer. WSDL (Web Service Definition Language, http://www.w3.org/TR/wsdl) steht hier für die Beschreibung eines Web Services und seiner Dienste. Eine WSDL Beschreibung ist eine detaillierte Definition der Web Service Dienste, ihrer Parameter oder Nachrichten-Strukturen und Endpunkte, über die der Dienst erreicht werden kann. Werkzeuge nutzen diese Beschreibung für die automatische Erzeugung von Programm-Komponenten, die in der Lage sind diesen Dienst aufzurufen und zu nutzen. Die Verteilung von Software Komponenten oder die manuelle Erstellung von Client Proxies ist nicht notwendig. WSDL ist ein ausgereifter Standard.
Neben WSDL werden auch Dienste benötigt, die das Auffinden und Integrieren von Diensten erleichtern. Mit UDDI (Universal Description, Discovery and Integration, http://www.uddi.org/) wurde ein Standard und eine Infrastruktur geschaffen, die es erlaubt Dienste anhand ihrer Eigenschaften aufzufinden und durch logische Endpunkte diese Dienste dynamisch anzusprechen. IBM, Microsoft, SAP und NTT-COM betreiben ein öffentliches verteiltes Verzeichnis. Dienstverzeichnisse sind aber vor allen Dingen auch innerhalb von Organisationen sehr sinnvoll. Dort haben die Nutzer eine stärkere Kontrolle über ihre Verzeichnisse und können Dienste anbieten, die nicht der Öffentlichkeit zur Verfügung stehen sollen. Der Einsatz von UDDI steckt noch in den Kinderschuhen. Dienste werden zunächst manuell zusammengesteckt.

Mit *Quality of Service* werden Protokolle bezeichnet, die dem Nachrichtenaustausch weitere Eigenschaften, wie Vertraulichkeit, Integrität oder Zuverlässigkeit geben. Hier hat sich eine Familie von Protokollen entwickelt, die von den führenden Plattformherstellern (Microsoft, IBM, SUN, BEA, und vielen mehr) in unterschiedlichen Konstellationen erarbeitet, implementiert und zusammen mit Anwendern in der Praxis erprobt werden.

WS-Security[2] dient der Authentifizierung, Signierung und Verschlüsselung von Nachrichten, wobei die eingesetzte Sicherheitstechnologie vom Standard nicht vorgeschrieben wird. WS-Security ist bereits in einer Reihe von Produkten (Software, Netzwerk-Geräten) in verschiedenen Gütegraden realisiert.

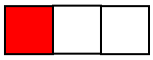


WS-Adressing[3] ist ein transportunabhängiger Standard für die Beschreibung von Dienst-Endpunkten und die transportunabhängige Übermittlung von Nachrichten durch Netzwerke, Firewalls und Gateways.

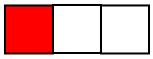
WS-ReliableMessaging[4] dient der Zuverlässigkeit des Nachrichtenaustausches auch auf nicht zuverlässigen Transportinfrastrukturen (z.B. HTTP). Dieses Protokoll kann sicherstellen, dass

[2] http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wss
[3] http://msdn.microsoft.com/webservices/understanding/specs/default.aspx?pull=/library/en-us/dnglobspec/html/ws-addressing.asp
[4] http://msdn.microsoft.com/webservices/understanding/specs/default.aspx?pull=/library/en-us/dnglobspec/html/wsrmspecindex.asp

Nachrichten genau einmal, in korrekter Reihenfolge und ohne Wiederholung beim Adressaten ankommen.
Für die Absicherung von Operationen durch transaktionales Verhalten werden eine Reihe von Spezifikationen bereitgestellt: WS-Coordination, WS-AtomicTransaction und WS-BusinessActivity. Dabei geht es darum verschiedene Partner innerhalb einer Transaktion zu koordinieren, um ein einvernehmliches Ergebnis zu erlangen. Im Kontext von Web Services können diese Transaktionen lange Zeit in Anspruch nehmen. Dadurch sind Protokolle, wie sie von traditionellen Transaktionsmonitoren verwendet werden, nicht ausreichend. Hierzu werden neue Arten von Protokollen auf der Basis von Business Activities und Compensation Logic verwendet. Diese Protokolle befinden sich zurzeit in der Implementierung.



Bei der Betrachtung von Geschäftsprozessen und deren Integration zwischen Geschäftspartnern findet die *Business Process Execution Language for Web Service*[5] (BPEL4WS) Anwendung. Sie ist entgegen ihres Namens weniger dafür gedacht, Geschäftsprozesse auszuführen. Ziel ist es, Geschäftprozesse und ihren Ablauf über Anwendungs-, Partner- oder Firmengrenzen hinweg formal zu definieren und in diese Definitionen als Grundlage für die Prozessdefinition und –Implementierung in Werkzeuge und *Enterprise Application Integration* (EAI) Plattformen importieren zu können. BPEL4WS wird bereits von einer Reihe von Produkten bereitgestellt.



1.5 Grundlegende Eigenschaften von Web Service Architekturen

1.5.1 Interoperabilität

Eines der wichtigsten Eigenschaften von Web Services ist die Interoperabilität, also die Fähigkeit der Kommunikation von Anwendungen unterschiedlicher Plattformen und Technologien. Die WS-I (Web Service Interoperability Organization, www.ws-i.org) wurde zu diesem Zweck von führenden Web Service Herstellern und Anwendern gegründet. Aufgabe ist es Web Service Interoperability durch die Erarbeitung von Richtlinien, Durchführung von Workshops und die Bereitstellung von Werkzeugen und Beispielen sicherzustellen.
In der Praxis haben sich eine Reihe von Erfahrungen etabliert, die in den folgenden Papieren niedergeschrieben sind.

- "Application Interoperability: Microsoft .NET and J2EE", Peter Laudati et.al., http://msdn.microsoft.com/webservices/building/architecture/default.aspx?pull=/library/en-us/dnpag/html/jdni.asp

1.5.2 Web Service Policies und Service Level Agreements

Web Service Policies dienen dazu technische Randbedingungen (Anforderungen, Fähigkeiten, Präferenzen) der Partner formell zu erfassen und bereitzustellen. WS-SecurityPolicy beschreibt beispielsweise die Notwendigkeit eines Dienstkonsumenten sich dem Diensterbringen gegenüber zu authentisieren, welche Art der Authentisierung notwendig ist (Kerberos, XAML, …), ob eine Signierung oder Verschlüsselung notwendig ist und vieles mehr. Diese Policies können automatisch durch die Plattform angewendet werden. Ein manueller Eingriff oder eine Berücksichtigung in der Anwendungsimplementierung ist nicht notwendig. Durch Policies hat man im Idealfall die Möglichkeit, die Qualität des Dienstes durch Deklaration zu bestimmen, ohne die Anwendung anpassen zu müssen.
WS-Policy bietet ein allgemeines Fundament für die Definition von spezifischen Policy-Frameworks, wie WS-SecurityPolicy.

[5] BPEL4WS, http://msdn.microsoft.com/library/en-us/dnbizspec/html/bpel1-1.asp?frame=true

“Understanding WS-Policy”,
http://msdn.microsoft.com/webservices/understanding/advancedwebservices/default.aspx?pull=/library/en-us/dnwebsrv/html/understwspol.asp

1.5.3 Transaktionales Verhalten

Was ist ein transaktionales Verhalten?
Typische Geschäftstransaktionen bestehen aus einer Reihe einzelner Aktionen, die alle abgearbeitet sein müssen, um die Geschäftstransaktion als erledigt beenden zu können. Eine solche Geschäftstransaktion darf auf keinem Fall halbfertig beendet werden, alle IT - Systeme stellen daher eine Reihe von Vorkehrungen zur Verfügung, die sicherstellen , dass eine Geschäftstransaktion über alle notwendigen Einzelschritte entweder komplett abgearbeitet wird oder komplett rückgängig gemacht wird.

Komponenten, die als ein Block abgearbeitet werden müssen, werden als eine IT-Transaktion bezeichnet und stehen in einem transaktionalem Zusammenhang.

Eine Komponente zeigt ein transaktionales Verhalten, wenn sie Funktionen bereitstellt, Arbeitsergebnisse bei Abbruch der Transaktion rückgängig machen zu können (Undo , roll-back Funktionen etc)

Beispiel für eine Geschäftstransaktion basierend auf Web Services

- Buchen einer Reise mit den Elementen Flug, Mietwagen, Hotel
- Wenn alle Elemente verfügbar sind, soll die Reise fest gebucht werden
- ansonsten sollen alle Reservierungen freigegeben werden

Die Buchung von Flug, Mietwagen und Hotel werden von unterschiedlichen Anbietern über Web Services als einzelne, in sich geschlossene Angeboten zur Verfügung gestellt, ebenso wie Storno und Aufheben der Reservierung.

Transaktionaler Zusammenhang zwischen Web Services

1. BPEL

Web Services sind in sich geschlossene Komponenten, die Stand heute unabhängig von einem transaktionalem Zusammenhang die definierte angebotene Leistung erbringen. Transaktionales Verhalten kann als individuelle Funktion angeboten werden, ist aber im Grunde ein weiterer Web Service (Beispiel: WebService_Reservieren, WebService_Reservierung aufheben)
Ein transaktionaler Zusammenhang zwischen unterschiedlichen Web Services muss auf einer zusätzlichen, einzelne Web Services übergreifenden und verbindenden Ebene hergestellt werden. BPEL4WS ist der technische Vorschlag des w3c Standardgremiums für diese transaktionale Verbindung von Web Services, entsprechende BPEL Tools und Laufzeitplattformen werden heute von allen führenden Software-Anbietern angeboten und haben sich in einfachen Problemstellung (siehe Beispiel) bewährt.
Es gibt gute praktische Argumente, heute für den transaktionalen Zusammenhang die BPEL Technologie zu benutzen, aufgrund der Offenheit des Web Services Standard können auch andere, ältere oder proprietäre Technologien als verbindende Transaktionsebene eingesetzt werden.

BPEL ermöglicht, das Rückgängigmachen einer Transaktion als eigenen Geschäftsprozess anzulegen (BPEL nennt das Compensation). Im obigen Beispiel würde das z. B. bedeuten, dass

bei einem Mangel an Hotelzimmern, Mietwagen- und Flug-Reservierung gemeinsam zu stornieren sind. Die Stornierung ist also in sich ebenfalls eine Geschäftstransaktion.
Auch auf diesem Pfad können verschieden Einflüsse dazu führen, dass die gemeinsame Aufhebung der Reservierung nicht oder nur teilweise durchgeführt werden kann.
Dieser mögliche Fall erfordert eine neue Compensations-Logik, die wiederum fehlschlagen kann.und so weiter. Transaktionales Verhalten der Web Services könnte diese Komplexität reduzieren.

Es gilt festzuhalten:
BPEL ist der Standard zur Herstellung eines transaktionalen Zusammenhanges zwischen an sich voneinander unabhängigen Web Services, Compensation der vorgeschlagene Weg, eine Geschäftstransaktion über alle beteiligten Web Services zu stornieren / zu annullieren.

2. Policies

Um Web Services präziser und effizienter in größere transaktionale Zusammenhänge einfügen zu können, werden zur Zeit Policies als Erweiterung des Web Services Standards definiert. Policies können genutzt werden, transaktionales Verhalten Web Service –gerecht abzubilden.

Policies sind das maschinell verarbeitbare Äquivalent zu den heutigen schriftlich vereinbarten Geschäftsbedingungen zwischen Geschäftspartnern.

Eine Policy wird zwischen Anforderer und Web Services verhandelt und vereinbart. Policies können zwischen einzelnen Web Services oder auf der BPEL Ebene für eine Transaktion über mehrerer Web Services mit einzelnen Web Services verabredet werden.

Die Funktionalität einer Policies muss immer auf Web Service Ebene abgebildet werden, d. h. bei der Entwicklung des Web Services müssen mögliche Policies bedacht und die entsprechende Geschäftslogik vorbereitet werden.

Der Policy Standard regelt in der ersten Stufe lediglich die Kommunikationsbasis. Policies können alles regeln, die Verantwortung, sinnvolle Policies vorzuschlagen liegt ausschließlich beim Entwickler des Web Services.

In unserem Beispiel könnten sinnvolle Policies sein:

- Frage → Bitte reserviere Flug xyz unter folgender Bedingung:
  - Wenn innerhalb von 10 Minuten keine Festbuchung eintrifft, bitte Reservierung aufheben.
- Antwort → OK , Reservierung gültig bis Uhrzeit xxx
  - Oder: Antwort → Bestätige Flugreservierung. Gilt für 5 Minuten, dann automatisch Storno, O.K?

Das Beispiel zeigt, dass transaktionales Verhalten durch Policies in einer neuen Art abgebildet werden kann. Policies sind im Standardisierungs–Prozess, daher fehlen praktische Erfahrungen.

1.5.4 Management von Web Services [Kapitel derzeit nicht verfügbar]

1.5.5 Sicherheit

Beim Thema Sicherheit geht es darum Angriffe von Außen und Innen durch geeignete Maßnahmen abzuwehren. Angriffe können in die folgenden Kategorien unterteilt werden

- Spoofing – stehlen von Identitäten und sich als jemand anderes ausgeben

- Tampering – Verändern von Informationen während der Übertragung
- Repudiation – Abstreiten von Transaktionen (z.B. Bestellungen) nachdem sie ausgeführt wurden
- Information Disclosure – Veröffentlichen von vertraulichen Informationen
- Denial of Service – Dienste zum Erliegen bringen
- Elevation of Privilege – unrechtmäßiges erlangen von Administrator Rechten

Diese Bedrohungen müssen beim Design und der Implementierung von Web Services in betracht gezogen werden. Eine umfassende Beschreibung findet sich im Artikel „Improving Web Application Security: Threats and Countermeasures“ (http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnnetsec/html/ThreatCounter.asp).

Grundsätzlich können folgende Möglichkeiten unterschieden werden

- WS-Security, wie oben beschrieben, als ein Web Service spezifisches Protokoll zur Absicherung von Web Service Kommunikation. WS-Security ist auf der Web Services Messaging Schicht (SOAP) angesiedelt und ist damit unabhängig vom darunter liegenden Kommunikationsprotokoll. WS-Security ist ein offenes Protokoll, welches die Art der Authentisierung, Signierung und Verschlüsselung offen lässt und einen interoperablen Rahmen bietet.
- HTTPS ist eine etablierte Technologie zur Authentifizierung und Verschlüsselung von HTTP Verbindungen (typischerweise zwischen Web Browser und Web Server). Die meisten Web Services sind über diesen Weg abgesichert. Nachteil hier: nur die Kommunikationsstrecke zwischen Client und Server ist abgesichert, siehe auch „HTTP Security and ASP.NET Web Services”, http://msdn.microsoft.com/webservices/building/frameworkandstudio/designing/default.aspx?pull=/library/en-us/dnwebsrv/html/httpsecurity.asp
- IPSEC wird vor allen Dingen im internen Bereich als Authentifizierungs- und Verschlüsselungstechnologie eingesetzt.

1.5.6 Abgrenzung zu verteilten Objekt Technologien

Eine häufig gestellte Frage ist die nach der Abgrenzung von Web Services zu *traditionellen* verteilten Objekttechnologien. Oft wird mit dem Overhead von XML basierten Protokollen argumentiert.

Die Idee der verteilten Objektarchitekturen war ein transparenter und für den Klienten unsichtbarer Aufruf von Objekten über Maschinengrenzen hinweg.
Web Services sollten nicht als neues CORBA oder DCOM gesehen werden, die eine Kommunikation zwischen Objekten verteilter Maschinen erlaubt. Bei Web Services geht es darum Anwendungen auf der Basis von (asynchronen) Nachrichten kooperieren zu lassen (siehe Service Orientierte Architekturen).
„ASP.NET Web Services or .NET Remoting: How to Choose”,
http://msdn.microsoft.com/webservices/building/frameworkandstudio/designing/default.aspx?pull=/library/en-us/dnbda/html/bdadotnetarch16.asp
„Chapter 10 —Improving Web Services Performance“,
http://msdn.microsoft.com/webservices/building/frameworkandstudio/designing/default.aspx?pull=/library/en-us/dnpag/html/scalenetchapt10.asp
„Performance Comparison: .NET Remoting vs. ASP.NET Web Services “,
http://msdn.microsoft.com/webservices/building/architecture/default.aspx?pull=/library/en-us/dnbda/html/bdadotnetarch14.asp

1.6 Service-Orientierte-Architekturen

Service-orientierten Architekturen (SOA) wird ein großes Potential für ein besseres Zusammenarbeiten von Anwendungen prognostiziert. Dabei bieten service-orientierte Architekturen Visionen, Strategien und Richtlinien für die Umsetzung auf der Ebene der Geschäftsprozesse, Anwendungsarchitekturen und Infrastruktur. Die Ebenen werden miteinander verbunden http://msdn.microsoft.com/architecture/soa/default.aspx

Die Kerngedanken sind:

1. Grenzen sind explizit – der Aufruf eines Dienstes geschieht immer mit dem Wissen, dass es sich hier um eine Kommunikation mit einem möglicherweise weit entfernten Partner handelt. Die Kosten sind jederzeit bewusst und werden beim Design und der Implementierung in Betracht gezogen. Dazu gehört auch das Design der Beschaffenheit eines Dienstes. Zu feine gegliederte Dienste können durch die hohe Latenz eines Aufrufs zu unakzeptablen Antwortzeiten führen. Zu grob gegliederte Dienste bergen die Gefahr, das sie nicht flexibel genug nutzbar sind.

2. Dienste sind autonom – jeder Dienst ist eine autonome Einheit und kann unabhängig von einem anderen Dienst genutzt werden.

3. Dienste vereinbaren Nachrichtenstrukturen und Prozessabläufe, aber keine Implementierungsdetails – dadurch wird die Plattformunabhängigkeit und Interoperabilität der Dienste sichergestellt.

4. Verträglichkeit von Diensten basiert auf Vereinbarungen – Vereinbarungen definieren unter anderem den Einsatz von Sicherheitsmaßnahmen und Prozessschritten.

Der Webcast *MSDN Architecture Webcast: Application Decomposition for SOA Based Systems* (http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032256201&Culture=en-US) beschreibt Vorgehensweise für die Entwicklung einer SOA.

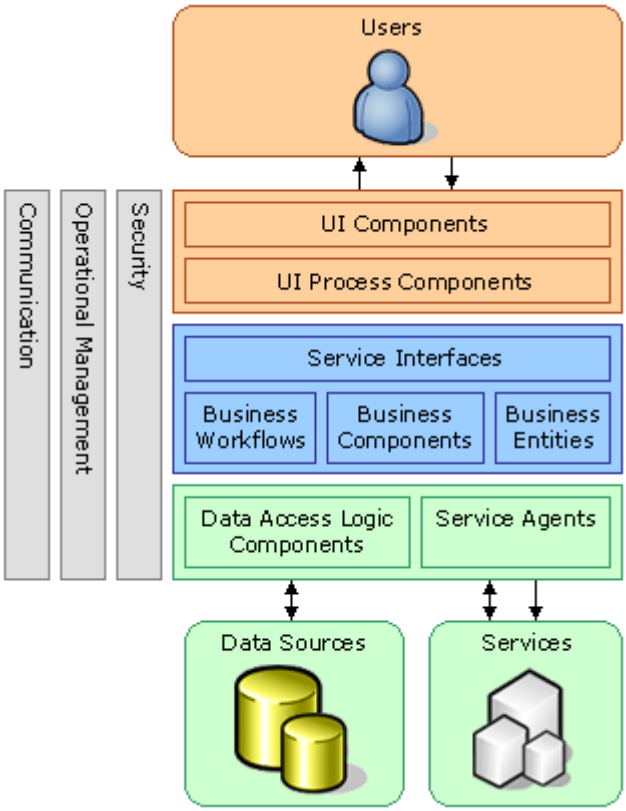
1.6.1 Typische Web Service Architekturen

Wie sehen konkrete Web Services Anwendungs-Architekturen aus? Nicht sehr viel anders als normale mehrschichtige Architekturen, also eine Trennung von Präsentationsschicht, Geschäftslogik und Datenzugriff. Grundsätzlich können Web Services als

- eine alternative Präsentationsschicht oder als
- zentrale Schnittstelle zur Geschäftslogik

gesehen werden. In der nächsten Abbildung ist die Schnittstelle zur Geschäftslogik als Web Services ausgeprägt. Die Entscheidung zwischen diesen oder anderen Varianten hängt dabei von verschiedenen Gegebenheiten ab, beispielsweise

- (Sicherheits-) Infrastruktur
- Technologie für die Kommunikation zwischen den Schichten der Anwendungs-architektur



((Achtung!! Dies Bild zeigt eine funktionale Architektur und ist damit unabhängig von der Implementierung und demnach nicht WS-spezifisch!!!))

In den folgenden Papieren werden typische Ansätze und Kriterien vorgestellt.
„Application Architecture for .NET: Designing Applications and Services", http://msdn.microsoft.com/architecture/application/default.aspx?pull=/library/en-us/dnbda/html/distapp.asp

1.6.2 Integrationsalternativen Zielgruppe Techniker / Architekten

1.6.2.1 Integration ohne Web Services

Integration von Anwendungen ist ein wichtiges Element in jeder Architektur, die aus mehreren Komponenten oder Anwendungspaketen gebildet werden soll.
Herkömmlich wurden dazu spezielle Programmpakete mit Adaptoren zu gängigen Anwendungen und Werkzeugen zur Erstellung solcher Adaptoren oder Konnektoren eingesetzt, die durch ihre eigene Systematik zwar das Integrations-Problem lösen konnten, aber in der Regel der Architektur eine weitere nicht-standardisierte Komponente hinzufügten mit Folgewirkungen für Wartung, Betrieb und mögliche zukünftige Veränderungen.

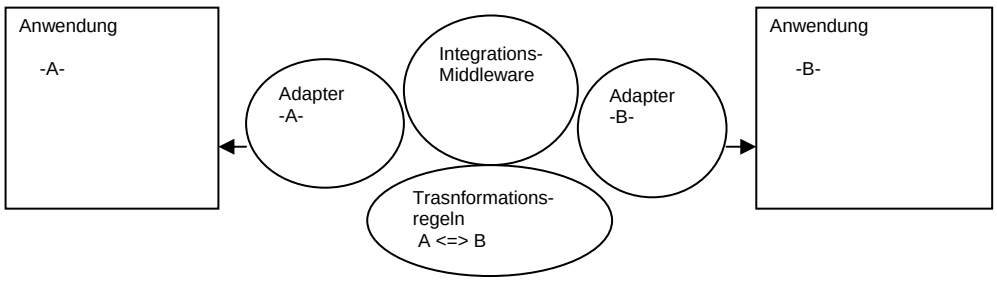


Abbildung Integration -1-

Abbildung 1 zeigt diese herkömmliche Art der Integration, die in den 90er Jahren unter verschiedenen Begriffen als eigenständige Architekturkomponente eingeführt wurde. Mit gutem Grund gilt diese Art der Gestaltung der Integration als die Königsdisziplin unter Architekten. Die Komplexität der Integration erschwerte es aus technischer wie auch wirtschaftlicher Sicht, Architekturen aus vielen Komponenten aufzubauen.

1.6.2.2 Integration mit Web Services

Web Services haben das Gebiet Integration von Anwendungen in mehrfacher Hinsicht grundlegend verändert, um wesentliche Möglichkeiten erweitert und auf eine standardisierte Form reduziert.

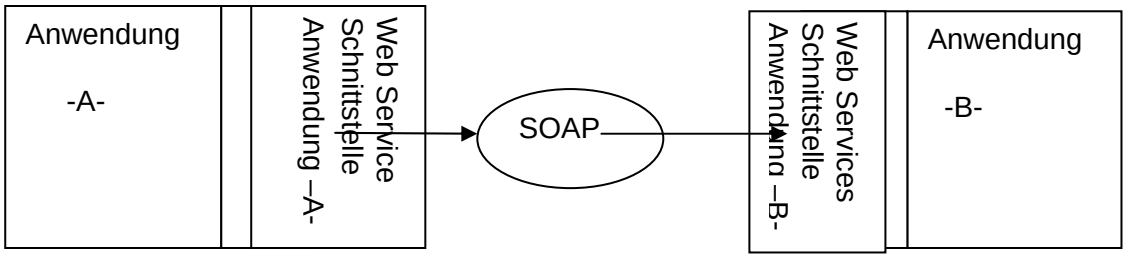


Abbildung Integration -2-

Abbildung 2 zeigt die gleiche Integrationsaufgabe wie Abbildung -1-.
Die dramatischen Vereinfachungen der Architektur sind deutlich sichtbar.
Auf Basis des Web Service Standards werden aus Komponenten zusammengesetzte Architekturen zu einer werthaltigen Alternative zu den klassischen Anwendungsarchitekturen. Das Thema Integration verliert seine Sonderstellung als zusätzliche Architekturkomponente und wird zu einer integrierten Funktion einer jeden Komponente der Architektur.

Ohne Anspruch auf Vollständigkeit sollen hier heute gängige Architekturmodelle auf dieser Basis aufgezeigt werden:

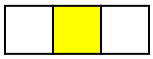
1. Feste Integration mehrerer Komponenten unterschiedlichster Technologie und Systematik innerhalb einer in sich geschlossenen Anwendungsarchitektur.
   - In jeder Komponente wird das gegebene Interface durch ein standardgerechtes Web Service Interface ersetzt oder mit einer entsprechenden Fassade überlagert.
   - Die Komponenten können nun direkt miteinander verbunden werden oder alternativ über eine Steuerungsebene. Diese Steuerungsebene kann ebenfalls im Web Service Standard BPEL4WS gestaltet werden, die Komponenten werden prozessorientiert gesteuert und entlang einer Wirkungskette (z.B Geschäftsprozess) eingebunden. Die Verbindung wird in der Zusammenbau-Phase festgelegt und ist danach statisch.
   - Zusätzlich kann der Transport durch Transformationsdienste für Daten/ Formatwandlungen ergänzt werden, um unterschiedliche technologische Inseln zu verbinden. Mit Hilfe von XML und entsprechenden Style-sheets ist dies in den Web Services zugrunde liegenden XML - Standard abzubilden.

In Summe ergibt sich:

- Die Integration ist durch einen Satz aufeinander aufbauender Standards (XML, XSD, Spath, WSDL Web Services, BPEL4WS) einfach und robust gestaltbar. Die Standards sind geprüft und in sich stabil.
- Die zeitraubende Frage, welche Art der Integration zu wählen ist, entfällt. Die Koordinierung von größeren Projekten mit mehreren Partnern etc. wird einfacher und verliert die politische Dimension, die jeweiligen Haus-Standards im Partnerkreis durchzusetzen.
- Ein großes Angebot an Tools und Laufzeitumgebungen, das den Standard unterstützt, ist verfügbar.
- Architektur und Tools können in der Regel heute als produktionsreif eingestuft werden.
- Die Evolution zu Web Services sollte mit sorgfältiger Ausbildung und einem Proof of Concept beginnen, um möglichen Risiken am Anfang von Projekten weitestgehend auszuräumen.

2. Lose Verbindung von Anwendungen verschiedener geschlossener Architekturen, gegebenenfalls verschiedener Partner (B2B), über das Netz.

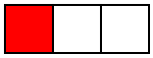
- Diese Integration von Anwendungen kann statisch, wie unter -1- beschrieben, gestaltet werden.
- oder dynamisch zur Laufzeit gestaltet werden:
  Eine dynamische Integration von Web Services in eine Architektur erfolgt über eine entsprechende Suche des Services in einem UDDI Repository und ein anschließendes, zeitweises Anbinden des gefundenen geeigneten Web Services an die Architektur. Diese Art der Integration ist im Web Services Standard beschrieben, hat ein enormes Zukunftspotenzial, ist aber in der Praxis heute noch wenig gebräuchlich.



  Einerseits ist das Angebot an freien Web Services noch relativ klein, andererseits fehlen für eine produktive Nutzung wichtige Ergänzungen im Standard, die zusätzliche Eigenschaften des Web Services verlässlich regeln.

Für beide Anwendungsstrukturen gilt:

- Der Web Service Standard hat zur Zeit noch keine einsatzfähige Lösung für die Probleme:
  Sicherheit der Verbindung , Servicelevel – Agreement für die Verfügbarkeit des Web Services, Policies zur Angleichung verschiedener Vorgehensmodelle der Partner,
- und keine verlässliche Information darüber, ob der gefundene Web Service die angebotenen Dienste auch erfüllen kann.



Integrationen im B2B Bereich benötigen heute noch ergänzende Komponenten, wobei der jeweilige Stand der Standardisierung einzusehen ist.

Status der Standardisierung heute :
Siehe Abbildung 3 im Anhang, oder unter : www.w3c.org

3. Bilden neuer Web Services durch Integration verschiedener bestehender Web Services oder Web-Anwendungen.

   - Diese Art der Vorgehensweise ist bekannt geworden durch ein Zusammenfügen von Modulen von Google, Ebay und Amazon.com zu einem neuen Service, der zu Büchern, Autoren und Sachthemen erweiterte Informationen, Kritiken mit Kaufpreisen für neue und gebrauchte Ware zusammenfügte ohne Wissen oder technische Abstimmung mit den Lieferanten der Komponenten.

     Hier zeigt sich die technische Stärke und Offenheit des Web Service Standards, die es auf der anderen Seite dringend erforderlich macht, möglichen Missbrauch angebotener Web Services durch Nutzungsvereinbarungen und lizenzrechtliche Vorkehrungen zu beschränken. Eine zum Web offene Web Service Architektur sollte vor der Indienststellung auch von der juristischen Seite geprüft werden sowie eine mögliche Integration seitens Dritter bedacht und geregelt werden.

   - Innerhalb geschlossener Architekturen lassen sich so flexibel und schnell neue Services bilden, um neuen Aufgaben gerecht zu werden oder bekannte Funktionen nach einer Umorganisation für das neue Zusammenspiel, arbeitsplatzgerecht zusammenzustellen.
     Diese Vorgehensweise hat Stand heute die Pionierphase verlassen und sollte in jedem Fall in die Architekturüberlegungen einbezogen werden.

4. Eine Service-orientierte Architektur ist eine Architektur, die Komponenten, die sich als Web Services darstellen und einbinden lassen, über einem Bus steuert und verbindet.
   Dieser Architekturansatz wird heute von allen maßgeblichen Anbietern von Komponenten und Middleware unterstützt, und ist auf dem Wege, andere Architektur-Modelle abzulösen. Entsprechende erste Produkte stehen zur Verfügung.
   Die klassische Aufgabenstellung der Integration von Komponenten ist damit zu einem allgemeinen, grundsätzlichen und standardbasierten Teil jeder grundlegenden Anwendungs-Architektur geworden.
   Das Thema Integration verliert einen großen Teil seiner zusätzlichen Komplexität und Vielfältigkeit. Die wirtschaftliche Abwägung zwischen den verschiedenen, möglichen Architekturen verschiebt sich zugunsten der komponentenbasierten Ansätze.

Einführende Literatur

- Web Services und Java : Erwin Merker Java Technologien Vieweg, ISBN 9 783528 058982
- Besonders gut : Kurze knappe Einführung, Beispiele zum Üben am PC für tieferes Verständnis.

Weitere Referenzen

- **developerWorks : Web services**
  The **Web services** section on the developerWorks Web site is your resource for tools, demos, articles, tips, and standards for creating **Web services**.
  URL: http://www-106.ibm.com/developerworks/webservices/

- **The role of private UDDI nodes in Web services, Part 1: Six species of UDDI**
  Steve Graham introduces the concepts behind **Web services** discovery and gives a brief overview of UDDI (Universal Description Discovery and Integration). He
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **The Web services (r)evolution: Part 4**
  This article describes the **Web Services** Description Language (WSDL), an XML grammar for stubs from WSDL and simplify the creation of **Web services** applications.
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **The Web services (r)evolution: Part 1**
  of building Web service applications. **Web services** may be an evolutionary step in designing a truly workable distributed system of **Web services**. This article also outlines his plan
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **Web services architect, Part 3: Is Web services the reincarnation of CORBA?**
  of the evangelism of **Web services**, customers have already ... this installment of the **Web services** architect, Dan Gisolfi
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **Web services insider,Part 1: Reflections on SOAP**
  What is the current state of the '**Web services** revolution?' In this, the first installment of my new column titled '**Web services** insider,' I'll answer this question
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **An overview of the Web Services Inspection Language**
  This is an update to the **Web Services** Inspection Language (WS-Inspection) overview in addition to an overview of the **Web Services** Inspection Language, this paper describes
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **Understanding quality of service for Web services**
  widespread proliferation of **Web services**, quality of service affecting performance of **Web services**, approaches of providing
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **Web services architect: Part 1**
  Now the hype surrounds **Web services**. In this series of well as describe the **Web services** strategy of IBM. Additionally ... the business impact of **Web services**, how to identify a
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- **The Web services (r)evolution: Part 3**
  messages are processed. It also explains how objects can be passed by value between **Web services**, and touches on performance and security issues. The **Web services** (r)evolution: Part 3
  URL: http://www-106.ibm.com/developerworks/webservices/librar...

- IBM Informationen zur SOA
  URL: http://www-306.ibm.com/software/info/openenvironment/soa/

1.7 Beispielarchitekturen und Frameworks

Neben den theoretischen Abhandlungen zu Web Services existieren auch eine Reihe von Beispielarchitekturen und Frameworks

1.7.1 EDRA

EDRA früher auch unter dem Namen *Shadowfax bekannt*, bietet Richtlinien sowie ein Framework für die Entwicklung von verteilten Anwendungen auf der Microsoft .NET Plattform.

- **GotDotNet Workspace**
  http://www.gotdotnet.com/community/workspaces/workspace.aspx?ID=9C29A963-594E-4E7A-9C45-576198DF8058
- **Channel 9 Wiki**
  http://channel9.msdn.com/wiki/default.aspx/Channel9.EDRAWiki
- **MSDN Architecture Webcast: patterns & practices Live: Shadowfax**
  http://msevents.microsoft.com/CUI/EventDetail.aspx?EventID=1032252204&Culture=en-US

1.7.2 FABRIQ

FABRIQ ist ein hochskalierbares Framework für den XML-Nachrichtenaustausch basierend auf der Microsoft .NET Plattform. Anders als EDRA, fokussiert sich FABRIQ auf eine GRID-artige Architektur für den Nachrichtenaustausch basierend auf Message-Queues.

- **GotDotNet Workspace**
  http://www.gotdotnet.com/community/workspaces/workspace.aspx?ID=B4FCF02F-3E71-4A15-A305-F0511240EEC1
- **What is FABRIQ**
  http://www.thearchitectexchange.com/asehmi/PermaLink.aspx?guid=0bf5e765-d6ed-4cc4-91d6-74db4489a426

2 Praktische Aspekte von Web Services

Wichtige praktische Aspekte sind unter anderen:

- Die Beeinträchtigung der Performanz von Web Services durch die Umwandlung der Daten in SOAP/XML ist zu beachten
- Theoretisch müssen die Abnehmer des Web Services zwar nicht bekannt sein, in der Praxis müssen jedoch alle im Falle einer Änderung des Dienstes informiert werden

2.1 Spezielle Anforderungen an das Entwicklungsmanagement [Kapitel derzeit nicht verfügbar]

2.2 Umsetzung von Sicherheit und Authentisierung [Kapitel derzeit nicht verfügbar]

2.3 Betrieb, Management und Abrechnung

Do-it-yourself oder Outtasking?

Es ist im Markt absolut üblich, dass Web Services nicht vom Entwickler betrieben werden, sondern vom Kunden oder viel typischer noch von einem auf das Hosting spezialisierten Dienstleister. Der hohe Anteil des Outtaskings in diesem Bereich resultiert aus den 365x24-Anforderungen derartiger Lösungen. Selbst wenn der eigentliche Zugriff nur zu den üblichen Geschäftszeiten erfolgen sollte, ist die Plattform doch oftmals 24 Stunden pro Tag etwaigen Risiken des Internets ausgesetzt. Denn sofern vertrauliche Daten gespeichert werden sollen oder gar eine Anbindung an die Backend-Systeme des Kunden erfolgt, genügt kein Schutz, der auf maschinellen Regeln einer Firewall basiert. Vielmehr muss ständig ein Betriebsteam zum sofortigen Eingriff bereit stehen, um auf etwaige Warnungen solcher Systeme umgehend reagieren zu können. Ein solches mit dem notwendigen Fachwissen ausgestattetes Team im Mehrschichtbetrieb einzusetzen, ist für die meisten Kunden unwirtschaftlich. Hosting-Anbieter hingegen nutzen hier interessante Skaleneffekte. Sie halten das erforderliche Know-how permanent vor, setzen dies jedoch für zahlreiche Kunden ein und ermöglichen diesen so attraktive Konditionen.

Kriterien für den erfolgreichen Betrieb von Web Services

Für den erfolgreichen Betrieb und das Management von Web Services müssen einige wichtige Kriterien erfüllt sein. Diese sind im Folgenden aufgelistet und sollten sich im Leistungsumfang des externen Serviceanbieters wieder finden, um den zuverlässigen Betrieb dieser Dienstleistungen zu gewährleisten.

Zertifizierungen, Know-how, 24/7-Überwachung

Bei der Wahl des Dienstleisters sollte der Kunde sich nicht nur auf die Ausführungen des Anbieters verlassen. Gutachten und Zertifikate unabhängiger Prüfstellen und Hersteller sind von Vorteil:

- Security-Zertifizierungen, die dem Sicherheitsbedürfnis des Kunden entsprechen
- Zertifizierungen der Prozesse, die den Geschäftsanforderungen des Kunden entsprechen

Ein weiteres wichtiges Kriterium ist das fachliche Know-how des im Rechenzentrum (RZ) eingesetzten Personals. Es muss in der Lage sein, die vom Kunden gewählten Technologien zu unterstützen. Für die Betreuung der Systeme sind besonders qualifizierte und zertifizierte Mitarbeiter notwendig. Dies gewährleistet ein hohes Maß an Sicherheit und Verfügbarkeit der eingesetz-

ten Systeme, eine schnelle Behebung von Fehlern und damit eine dauerhafte Entlastung der personellen Ressourcen des Kunden.

Zudem muss der Outsourcing-Partner über genügend Mitarbeiter verfügen, um ein professionelles Management der Rund-um-die-Uhr-Überwachung und -Sicherheit zu gewährleisten. Die Belegschaftsgröße gibt Aufschluss darüber, wie der Dienstleister die versprochene 24/7-Betreuung in die Realität umsetzt.

COLT Telecom sei hier exemplarisch für einen zertifizierten Hoster genannt, der mit ITIL-basierten Prozessen 11 Rechenzentren in Europa – wie etwa in Frankfurt/Main und Berlin – betreibt, COLT verfügt beispielsweise über folgende Zertifizierungen:

- Suntone (jährlich auditierte Prozesse für den Betrieb von Servern)
- BS 7799 (Security)
- EN ISO 9001:2000 (Qualitätsmanagement)

Bei dem europäischen Telekommunikationsunternehmen steht eine Mannschaft von 245 Administratoren und Systemingenieuren für die Realisierung eines 3-Schicht-Betriebs zur Verfügung.

Technische Ausstattung des RZ
Natürlichen spielen auch Kriterien wie die Ausstattung des RZ und die verfügbare Internet-Bandbreite eine wichtige Rolle. Abgesehen von Projekten mit außergewöhnlichen Anforderungen in diesen Bereichen erfüllen inzwischen aber nahezu alle Hoster entsprechende Mindestanforderungen.

Unterstützte Software
Es gibt aber auch klare Anforderungen an den Dienstleister, die sich auf die zu betreibende Software beziehen. Die Verwendung verbreiteter Standards ist sicherlich eine solide Grundlage für den zuverlässigen Betrieb der Lösung. In diesem Zusammenhang sind insbesondere erwähnenswert:

- Betriebssysteme: Windows-Server, Linux (RedHat oder Suse), Solaris (9/10)
- Microsoft-SQL-Server und Oracle-Datenbanken
- BEA Weblogic oder IBM Websphere

Bei Verwendung dieser Technologien kann der Kunde zu Recht von dem Dienstleister seiner Wahl fordern und erwarten, dass er über entsprechende Spezialisten verfügt, die die genannten Systeme und Applikationen professionell unterstützen können.

Betriebsführungshandbuch
In jedem Falle ist ein solides Betriebsführungshandbuch unabdingbar. Es muss erheblich umfangreicher und detaillierter ausfallen, wenn die zu Grunde liegende Technologie nicht durch entsprechendes Spezialwissen des Hosters unterstützt wird. Leider zeigt die heutige Praxis genau hier ganz erhebliche Lücken. Softwareentwickler werden an Zielen gemessen, die oft mit dem Abschluss der Tests der zu entwickelnden Applikation enden. Anforderungen an eine für den Betrieb dieser Lösung unabdingbare Dokumentation sind meist nicht definiert und schon gar nicht rechtswirksam im Entwicklungsauftrag verankert. Unter mangelhafter Dokumentation leidet jedoch der Betrieb der Lösung außerordentlich. Der Dienstleister kann die Lücken durch entsprechende Betriebsführungsworkshops, Tests und ein selbst verfasstes Betriebsführungshandbuch weitgehend kompensieren. Allerdings resultieren daraus mitunter massive Mehrkosten, die der Kunde oft nicht eingeplant hat.

Ob ein Betriebsführungshandbuch vom Softwareentwickler geliefert wird oder erst unter Federführung des Hosters entsteht – es sollte neben der Dokumentation der Systeme, Softwareversionen, Kontakte und Prozesse (vor allem für Änderungen und Eskalationen) unbedingt die Aspekte ‚Releasewechsel' und ‚Kochbücher' zuverlässig abdecken.

1. Releasewechsel

Wenn ein aufgesetztes System vom Hoster oder einem unabhängigen Dritten im Zuge der Abnahme mittels Penetration-Tests für sicher befunden wurde, spiegelt dies definitiv nur einen Momentzustand wieder, der sich möglicherweise schon kurz darauf durch neue Exploits grundlegend ändern kann. Es ist daher unabdingbar, dass der Dienstleister in diesem Falle umgehend reagiert und entsprechende Updates einspielt. Dabei sind 4 Fälle zu unterscheiden:

- Absolut dringend
  Der Hoster spielt den Patch sofort und ohne Rücksprache mit dem Kunden ein, da ohne diesen Patch die Sicherheit der Plattform maßgeblich gefährdet wäre. Der Hoster informiert den Kunden/Softwarepartner erst im Nachhinein.
- Erforderlich
  Der Hoster informiert den Kunden über den bevorstehenden Patch unter Angabe entsprechender Details, möglicher Auswirkungen und Fallback-Szenarien sowie mit Benennung eines Termins für das Einsetzen des Patches. Gibt es seitens des Kunden keine Reaktion, wird der Patch zum vorgesehenen Termin eingesetzt.
- Empfohlen
  Der Hoster informiert den Kunden über einen empfehlenswerten Patch unter Angabe entsprechender Details, möglicher Auswirkungen und Fallback-Szenarien. Der Patch wird nur mit ausdrücklicher Zustimmung des Kunden eingesetzt.
- Verbesserung
  Patches, durch die neue Leistungsmerkmale erschlossen werden, aber aus Sicht der Integrität und Stabilität nicht erforderlich sind, werden nie auf Initiative des Hosters installiert. Stichwort: ‚Never touch a running system'
  In diesem Fall sollte stets der Kunde oder dessen Softwarepartner die klare Anforderung formulieren, eine bestimmte Version einzusetzen. Die Initiative wird nie vom Dienstleister ausgehen.

Der Softwarehersteller sollte in seiner Betriebsdokumentation die im Rahmen solcher Patches relevanten Anforderungen an die Software und die getesteten Versionen darlegen sowie Hinweise über Relevanz und Auswirkung künftiger Patches geben. So können Risiken identifiziert und etwaigen Folgen angemessen vorgebeugt werden.

2. „Kochbuch"

Je länger ein Softwareentwickler bereits erfolgreich tätig ist, desto geringer ist erfahrungsgemäß seine Bereitschaft, auf Dauer an Wochenenden oder Feiertagen für Eskalationen verfügbar zu sein. Hier ergänzen sich die Interessen mit denen eines Dienstleisters ideal, für den der Rund-um-die-Uhr-Betrieb von Web Services das „tägliche Brot" ist. Ein Hoster, der Betriebsverantwortung für ihm unbekannte Software übernehmen soll, benötigt eine Form der Dokumentation, die in der Branche gemeinhin als „Kochbuch" bezeichnet wird. Es muss – ohne Applikationskenntnisse vorauszusetzen und in der Sprache des Systemadministrators – folgende Aspekte exakt beschreiben:

- Wie wird ein Dienst überwacht (Monitoring)?
- Wer wird bei einem Alarm benachrichtigt (z. B.: Helpline)?
- An wen wird der Störfall adressiert (z. B.: First Level Support)?

- Was hat dieser Supporter zu leisten?
- Wie überprüft er den Erfolg seiner Arbeit?
- Wer ist über den erfolgreichen Abschluss der Aufgabe zu informieren oder an wen ist ein Fehlschlagen zu berichten?

Eine Reihe erfahrungsgemäß erforderlicher Kochbücher sollten eindeutig zum Lieferumfang jeder guten Software gehören. Dabei ist der erforderliche Umfang bei Verwendung im Markt etablierter Standardkomponenten weitaus geringer als bei Individuallösungen. So kann beispielsweise ein Softwareentwickler durchaus voraussetzen, dass ein Dienstleister weiß, wie man den Table Space einer Oracle-Datenbank vergrößert. Im Falle einer proprietären Lösung wäre die erforderliche Vorgehensweise zu beschreiben. Sollte es dabei Limitierungen für die Maximalgröße geben, sind diese auf gleiche Weise zu dokumentieren – mit entsprechendem Monitoring, um vor diesem Zustand rechtzeitig zu warnen.

Personelle Ressourcen
Die Erfahrung lehrt, dass neben allen Prozessthemen zwei Personen für die dauerhafte Kundenzufriedenheit eine Schlüsselrolle spielen:

- Dedizierter Servicemanager
- Dedizierter Supportingenieur

Helpdesk und Support-Einheiten betreuen im Schichtbetrieb eine Vielzahl von Kunden. Deshalb kennen sie nicht immer die Historie der letzten Änderungen im Projekt, die Zielsetzung des Kunden oder sämtliche Aspekte seiner IT-Architektur. Um aber die Risiken einer Änderung zu beurteilen, angemessene Fallback-Szenarien zu identifizieren und den richtigen Zeitpunkt für deren Einsatz zu erkennen, bedarf es organisatorischer und technischer Ansprechpartner, die das Gesamtprojekt genau kennen und stets den Überblick behalten. Dies ist, wie die Erfahrung zeigt, der Schlüssel zum erfolgreichen Betrieb auch komplexer und höchstverfügbarer Web Services.

Vorgehensmodelle
Das Management von Web Services bedarf ausgefeilter Prozesse und ausgereifter Werkzeuge, da im Störungsfall eine Vielzahl von Informationen sehr schnell erschließbar sein muss. Für den erfolgreichen Betrieb von Web Services ist vor allem die Definition einer standardisierten, methodischen Vorgehensweise für folgende Aspekte entscheidend:

- Konfigurations- und Release-Management
- Change Management
- Incident Management
- Service Management

Das **Konfigurations- und Release Management** ist eine kontinuierlich fortlaufende, proaktive Maßnahme. Dabei hat es sich als extrem vorteilhaft erwiesen, die lückenlose manuelle Dokumentation etwaiger Veränderungen durch automatisierte Vorgänge zu überprüfen. So wird beispielsweise bei COLT jede Detailänderung in der Konfiguration oder der Version eines eingesetzten Produktes in einer entsprechenden Datenbank eingepflegt (CCDplus) und maschinell mit der Datenbank von TRACK abgeglichen. TRACK scannt täglich einmal alle Server und ermittelt maschinell die eingesetzten Softwareversionen, sammelt Konfigurationsdateien und viele weitere Informationen, um Veränderungen lückenlos zu protokollieren. Etwaige Abweichungen werden als Alarm behandelt und umgehend geklärt.

Im Rahmen des **Change Management** müssen für jede Änderung nicht nur Umfang, mögliche Auswirkungen sowie Zeitpunkt und Aufwand, sondern vor allem auch ein Fallback-Szenario definiert werden. Dies kommt zum Tragen, falls der Wechsel scheitern sollte. Eine Änderung sollte immer von einem erfahrenen Systemingenieur auf etwaige Risiken und Schwachstellen hin analysiert werden. Er muss die Gesamtlösung kennen und um die jüngere und künftige Entwicklung der Web Services des Kunden Bescheid wissen.

Das **Incident Management** bezeichnet in der Regel eine Helpline, die einen jederzeit kontaktierbaren Anlaufpunkt darstellt. Bei einem guten Hosting-Dienstleister ist dieser bundesweit 365x24 und kostenlos erreichbar. Die Rufnummern der Helpline sollten allen Kunden bereits während des Entscheidungsprozesses für einen Hoster mitgeteilt werden. Zudem sollten Kunden die tatsächliche Erreichbarkeit der Service-Mitarbeiter selbst testen.
Eine Helpline sollte dabei jedoch nicht nur die Problemmeldung entgegennehmen, verfolgen und den Kunden über den Fortschritt bei der Problemlösung informieren, sondern unbedingt auch selbst Zugriff auf Monitoringsysteme haben. Einem Anrufer erläutern zu können, dass man einer Störung bereits nachgeht, weil entsprechende Alarme auftraten, und wer darüber beim Kunden bereits informiert wurde, vermeidet viele Verwirrungen. Viel wichtiger aber ist der gegenteilige Fall: Wenn kein Alarm auftrat, der Kunde aber einen Ausfall reklamiert, kann in Verbindung mit dem Online-Zugriff auf das Monitoringsystem sofort wesentlich gezielter nach weiteren Details gefragt werden.

Der Hoster hat entsprechende Eskalationsprozeduren klar darzulegen, die bei Überschreiten vereinbarter Maximalzeiten für eine Störungsbehebung auch eine direkte Einbindung der Geschäftsleitung des Kunden wie des Hosters einschließen. Nur so kann sichergestellt werden, dass mit entsprechender Priorisierung alle erforderlichen Maßnahmen ergriffen werden.

Das **Service Management** umfasst nicht nur ein regelmäßiges Service-Level-Reporting mit zugehörigen Reviews, sondern dient als so genannter „Single Point of Contact“ für alle Belange des Kunden - also sowohl in vertraglicher, kommerzieller wie auch technischer Sicht. Dem Service-Manager sollte ein dedizierter Supportingenieur zur Seite stehen, der in der Lage ist, Auswirkungen geplanter Änderungen umfassend beurteilen und entsprechende Vorschläge ausarbeiten zu können.

Abrechnung von Web Services
Das Angebot der Hoster umfasst derzeit vor allem klassische Abrechnungsszenarien, bei denen Leistungen dediziert für einen Kunden erbracht und in monatlich festen Beträgen abgerechnet werden. Der Kunde muss somit keine Anfangsinvestitionen tätigen, sondern hat fest kalkulierbare monatliche Kosten und damit die optimalen Voraussetzungen für eine langfristige Budgetierbarkeit der Leistungen.

Kunden fragen jedoch zunehmend nach verbrauchsabhängigen Abrechnungskonzepten („pay as you use“). Erste Ansätze dafür sind im Markt bereits erkennbar:

- Abrechnung der Internet-Bandbreite nach Verbrauch
  Beispiel: Usage Based Billing für Internet-Verkehr
- Abrechnung von Storage-Kapazitäten und Backup/Restore/Recovery-Leistungen nach tatsächlicher Inanspruchnahme
  Beispiel: SAN-Storage on Demand
- Abrechnung von Lizenzen nach Inanspruchnahme auf monatlicher Basis (statt Kauf)
  Beispiel: Microsoft Service Provider License Agreement (SPLA)

Mit solchen Angeboten entsprechen Hoster den wirklichen Kundenanforderungen sehr gut, doch fehlen vergleichbare Offerten in vielen anderen Bereichen. So ist das obige Software-

Beispiel, das Microsoft bereits 2001 etablierte, noch eine extreme Ausnahme, und im Hardwarebereich lassen viele vollmundige „On Demand“-Versprechen noch jeden akzeptablen kommerziellen Hintergrund vermissen.

Feststellbar ist jedoch, dass nahezu alle Hoster zusammen mit ihren Dienstleistungen auch eine Finanzierung der erforderlichen Hardware anbieten, so dass der Kunde keine Anfangsinvestitionen, sondern klar kalkulierbare monatliche Leistungsraten für den erbrachten Dienst erhält.

3 Nutzen [Status Dezember 2004]

Zielgruppe Techniker / Manager

3.1 Nutzen aus technischer Sicht

„Eine wesentliche Eigenschaft von Web Services Technologie besteht darin, dass sie unterschiedliche Vorteile vereinigt und durch die Kombination dieser Vorteile alternativen Ansätzen überlegen ist“ [eine Schlussfolgerung aus der BERLECON-Studie]

Der technische Nutzen von Web Services ergibt sich im Wesentlichen aus den nachfolgenden Eigenschaften:

- Web Services sind ein offener akzeptierter Standard.
- Web Services basieren auf offenen Standards (z.B. HTTP, XML)
- Die Web Service Architektur ermöglicht Erweiterungen, die auf den vorhandenen Basistechnologien aufbauen.
- Die Web Service Schnittstelle ist maschinenlesbar.
- Die Web Service Schnittstelle ist von der Implementierung getrennt

Aus diesen Eigenschaften lassen sich zahlreiche Vorteile ableiten, die zu Produktivitätsgewinn, Flexibilität und Unabhängigkeit beim Einsatz der Web Service Technologie für die Integration von Anwendungen führen.

3.1.1 Produktivität

Aufgrund der hohen Akzeptanz im Markt, hält die Web Service Technologie nahezu in allen Middleware-Produkten und Entwicklungsplattformen Einzug. Entwickler können dadurch in ihren vertrauten Umgebungen arbeiten. Die Hersteller erreichen mit ihren Produkten hinsichtlich Definition und Implementierung von Web Services einen hohen Automatisierungsgrad. So kann z. B. aus einer WSDL-Datei der gesamte Schnittstellencode generiert werden. Darüber hinaus bieten einige Hersteller die Möglichkeit aus Programmcode die WSDL-Datei zu erzeugen. Es ist deshalb im Vergleich zu andern Integrationstechnologien verhältnismäßig einfach bestehende Legacy-Anwendungen mittels Web Service Technologie zu *wrappen* und für unterschiedliche Verwendungskontexte verfügbar zu machen.

Für die Nutzung von Web Services sind lediglich die Informationen der WSDL-Datei notwendig. Die Entwicklungsumgebungen bieten die Möglichkeit auf Basis der WSDL-Datei den Clientcode (Proxy) in der gewünschten Programmiersprache zu generieren und den Web Service darüber aufzurufen.

Im Vergleich zur Kommunikation über Binärformate, reduziert sich außerdem der Aufwand für Testen und Monitoring. Die maschinenlesbaren Daten können einfach verarbeitet und ausgewertet werden. Im Testumfeld kann die Produktivität durch Einsatz von Generatoren gesteigert werden.

3.1.2 Flexibilität

Bisher wurde Anwendungsintegration zwischen heterogenen Systemen in der Regel über Adapter realisiert, die auf den jeweiligen Nutzungskontext zugeschnitten wurden. Web Services erfüllen prinzipiell die gleiche Funktion können aber in beliebigen Kontexten eingesetzt werden. Da-

durch bieten Web Services die Voraussetzung für die nächste Entwicklungsstufe zu Serviceorientierten Architekturen hin.

Web Services zeichnen sich durch ihre Prozessunabhängigkeit aus. Dadurch wird ein hoher Grad an Widerverwendbarkeit erreicht. Sie können in unterschiedlichen Anwendungskontexten zu beliebigen Prozessen kombiniert werden. Durch die Trennung von Schnittstelle und Implementierung können einzelne Services in einem Prozessablauf durch alternative Implementierungen ausgetauscht werden, die die gleiche oder eine ähnliche Schnittstelle anbieten. Der Kommunikationspartner kann dabei dynamisch ermittelt werden. Durch die lose Kopplung der Services bleiben diese autonom und können unabhängig voneinander weiterentwickelt werden.

Durch die Verwendung des SOAP-Protokolls, das wiederum auf HTTP oder SMTP aufsetzt, entsteht zusätzliche Flexibilität. So ist neben reinen XML-Daten auch die Übertragung von Mime-Types möglich. Ebenso sind unterschiedliche Interaktionsmuster wie asynchrone Kommunikation („Fire-and-forget“) oder synchrone Kommunikation (Request-Response bzw. Remote Procedure Call) möglich.

3.1.3 Unabhängigkeit

Web Services sind weder von einer Plattform noch von einer Programmiersprache abhängig, so dass sie gerade in heterogenen IT-Landschaften bei der Integration von Anwendungen gegenüber andern Integrationsansätzen Vorteile bieten. Die Services können auf der jeweiligen Plattform in der gewünschten Programmiersprache implementiert werden. Durch die Standardisierung ist eine Kommunikation zwischen unterschiedlichen Plattformen möglich. Ein beliebtes Einsatzszenario für Web Services ist daher die Kommunikation zwischen der *Java*- und der *.net-Welt*.

Wird der Standard eingehalten, entsteht für die Kommunikationsschicht keine Abhängigkeiten von bestimmten Herstellern. Der Technologieeinstieg kann auch über frei verfügbare Open Source Tools erfolgen. Bei erfolgreicher Evaluierung kann auch auf kommerzielle Web Service Plattformen mit umfassender Funktionalität gewechselt werden.

Quellen:

- [BERLECON] Basisreport Integration mit Web Services (Konzept, Fallstudien und Bewertung), Dr. Joachim Quantz, Berlin August 2003, Berlecon Research GmbH
- Web Services, Donald Kossmann, Frank Leymann, Informatik-Spektrum 26. April 2004, Seite 117 – 128, Springer-Verlag

3.2 Nutzen aus Managementsicht – Return-On-Investment

3.2.1 Web Services – eine der schnellsten Möglichkeiten Geld zu sparen

Neue Technologien haben oft den Nimbus, dass die dafür notwendigen Investitionen in die Ausbildung der Mitarbeiter sowie in neue Hard- und Software sich erst über einen langen Zeitraum hinweg auszahlen.

Das Einsparpotential der Software-Maintenance von objektorientierten Applikationen ist verglichen mit klassischen Programmierstilen wie etwa PL1, Cobol oder RPG sehr offensichtlich. Die Initial- und Umstellungkosten von existierenden Systemen auf die objektorientierte-Technologie erfordert jedoch einen vergleichsweise langen Zeitraum bis sich ein Return-of-Invest (ROI) im Unternehmen abzeichnet.. Kaufmännisch wird daher oft gegen ein Modernisierungsprojekt entschieden, insbesondere dann, wenn für die Neu-Programmierung unterstützende Pattern-Tools fehlen.

Da Web Services auf den von den Web Application Servern bereitgestellten Basisfunktionen sowie auf offenen, von Anbietern akzeptierten und unterstützen Standards aufbauen, entsteht der essentielle Vorteil von Web Services gegenüber anderen Integrationstechnologien. Anwendungen lassen sind damit schneller, einfacher und kostengünstiger integrieren und Weiterverwendung vorhandener Anwendungen ersetzt „Neu"-Entwicklung von Anwendungen.

Damit haben sich Web Services gerade in einer zunehmend vom Kostenbewusstsein geprägten IT-Landschaft als eine Technologie herausgestellt, mit der das Management schon kurzfristig einen positiven Rückfluss der getätigten Investitionen erreichen kann.

3.2.2 Nutzenbetrachtungen aus der Sicht einiger Unternehmen

NIIT Technologies gehört durch die strikte Umsetzung des *3-Horizon-Concepts* von McKinsey zu den *early adaptors* dieser Technologie und setzt sie in zahlreichen globalen Projekten erfolgreich ein.

Beispiel 1:
Ein Auftraggeber von NIIT erhält ca. 60% seiner Aufträge per Electronic Data Interchange (EDI) und weitere elektronische Schnittstellen. Die Aufträge seiner Kunden, die die restlichen 40% ausmachen, kommen auf klassische Art und Weise in schriftlicher oder telefonischer Form. Bemühungen, diese Kunden ebenfalls zur elektronischen Auftragserteilung zu bewegen, waren aufgrund deren Größenstruktur erfolglos. Auch die Möglichkeit, die Aufträge in einem zur Verfügung gestellten Web-Interface zu erfassen, wurde durch diese Kunden nicht wie erhofft angenommen. Der Grund hierfür war, dass die Mehrzahl dieser Kunden die Aufträge in eigenen, meist PC orientierten Lösungen erfassten. Eine zusätzliche Erfassung im Web-Interface bei den Kunden hätte einen zu hohen Aufwand verursacht hätte. So blieb dem Auftraggeber von NIIT nur die sehr zeit- und kostenintensive manuelle Erfassung seiner Kundenaufträge.

Die Lösung des Problems erreichte NIIT, indem den Kunden zusätzlich zum Web-Interface ein Web Service zur Verfügung gestellt wurde, dessen Aufruf sich problemlos in die meisten Systeme der Kunden integrieren ließ. Bereits im ersten Jahr sank der manuelle Aufwand bei dem Auftraggeber von NIIT um 40%. Daraus resultierte ein ROI von gerade einmal 4 Monaten!

Beispiel 2:
Durch zunehmend oligopolistische Tendenzen in den einzelnen Branchen kommt es zwangsläufig immer häufiger zu Situationen, in denen sehr unterschiedliche EDV-Systeme und Programmiertechnologien miteinander kombiniert oder gar fusioniert werden müssen. In den meisten Fällen ist das Ziel die EDV-technische Integration des neu erworbenen Unternehmenszweigs in die Strukturen des Mutterkonzerns. Dass eine solche Integration ein zeitaufwendiges und häufig in Dauer, Kosten und Risiken schwer abschätzbares Unterfangen darstellt, wird kein EDV-Leiter, der jemals diese Aufgabe zu bewältigen hatte, bestreiten. Durch die fortlaufenden hohen Kosten eines solchen zwei- oder gar mehrgleisigen Betriebes unterschiedlicher Systeme entsteht ein hoher Druck auf die mit der Integration beauftragten Unternehmensteile, die oftmals an anderen Stellen Aufgaben des täglichen Betriebs wahrnehmen müssen.

Web Services stellen hier eine Möglichkeit dar, den Zeitdruck für eine vollständige Integration zu reduzieren und die laufenden Kosten für den Parallelbetrieb der Systeme zu minimieren. Zahlreiche NIIT-Kunden gehören in ihren Marktsegmenten zu den Top Global Playern, die immer wieder mit derartigen Integrationsaufgaben konfrontiert werden. In der Business Line „Application Integration" nutzt NIIT bei diesen Kunden sehr häufig die technologische Basis der Web Services, die eine automatisierte Verständigung verschiedener Systeme innerhalb der Konzerne über Hardware-Grenzen hinweg ermöglichen. So wurde bei einem weltweit operierenden Reiseveranstalter eine vollständige Integration der auf unterschiedlichen Hardwareplattformen lokalisierten Systeme zu einem unternehmensweit einheitlichen System realisiert. Un-

terschiedlichste Komponenten wie traditionelle, Mainframe basierte Systeme kommunizieren mit modernsten Java-basierten Applikationen und funktionieren als ein einheitliches Unternehmenssystem. In einem mehrstufigen Prozess wurden Herausforderungen wie doppelte Datenpflege und ähnliches Schritt für Schritt reduziert ohne die Risiken einer „Big-Bang“-Umstellung eingehen zu müssen.

Der ROI wurde durch die konsequente Verwendung von Web Services als Integrationsmedium in einem Bruchteil der Zeit erreicht, die ansonsten für eine „klassische“ Form der Integration der Systeme notwendig gewesen wäre. Es konnten durch diese Integrationstechnologie außerordentlich große Einsparungen erzielt werden, so dass der Kunde zur Zeit sogar eine weitere Kosten-Nutzen-Analyse durchführt, um zu prüfen, ob eine weitere Zusammenführung der unterschiedlichen Systeme auf eine einheitliche technologische Basis überhaupt noch einen wirtschaftlichen Sinn macht.

Weitere Informationen
zu den Erfahrungen im Bereich von Web Services befinden sich unter

- info@niit-tech.de
- http://www.niit-tech.de/
- http://www.niit.com/

4 Tendenzen und Entwicklungen

Die Informationstechnik hatte stets die Aufgabe, Unternehmen dabei zu unterstützen, Aufgaben und Tätigkeiten effizienter und wettbewerbsfähiger zu gestalten..

Wenn die Strukturen der Wirtschaft sich globalisieren, Unternehmen sich durch Fusionen, Auslagern von Funktionen, Konzentration auf Kernkompetenzen, Flexibilisierung der Prozesse und neue Organisationsstrukturen an die neuen Realitäten der Märkte anpassen, muss die Informationstechnik diese Veränderungen in den Geschäftsprozessen optimal unterstützen.

Dies gilt umso mehr, als durch schnelle Kommunikation und wettbewerbsintensive Märkte der Bedarf nach Anpassungen und Veränderungen sich beschleunigt, die Innovationszyklen werden kürzer, auch für die Geschäftsprozesse und damit für die sie unterstützende IT.

Der heutige Kunde will seine gewünschte Leistung zum besten Preis in höchster Qualität möglichst sofort, mit bester Innovation. Stillstand führt zum Abstieg. Die Schnellen und Innovativen gewinnen, wenn sie die Kosten im Griff behalten.
In den Finanzmärkten sind Quartalabschlüsse so wichtig wie im letzten Jahrhundert Jahresbilanzen.
Automobile kommen nach 2 bis 3 Jahren Entwicklung anstelle von 7 bis 9 Jahren auf den Markt. Innovationen wie Mobile Telefone und Digitale Fotografie werden schneller zum Allgemeingut als es andere Technologien jemals zuvor wurden.

Wir leben mehr und mehr in einer Welt mit einem Zeitraffer-Effekt, in der jede Art von Nachfrage sofort, ohne große Verzögerung, direkt und unmittelbar befriedigt sein will. Einer Kultur also, die unter dem Motto „Leistung auf Wunsch sofort“ oder englisch „On Demand“ lebt.
Die heutigen klassischen IT-Strukturen und Technologien werden in einer solchen On-Demand Kultur mehr und mehr zum Hindernis für notwendige Anpassungen der Geschäftsprozesse, da sie aufwendiger, schwieriger und langsamer zu verändern sind als das Tempo des Wandels es erfordern würde.

Die Vision, die sich auf den heutigen Web Services aufbaut, ist der Versuch einer Antwort auf die Frage, wie die IT diese neue On Demand Welt entsprechend pro-aktiv unterstützen kann, ohne alle getätigten Investitionen in Frage zu stellen.

Die Vision ist, einen Geschäftsprozess firmenübergreifend zu definieren und den Prozess jederzeit mit seiner IT –Komponente gemeinsam verändern, anpassen, managen, kontrollieren, vernetzen und erweitern zu können. Damit lassen sich dann Geschäftsprozesse in allen Abhängigkeiten und Vernetzungen mit anderen Geschäftsprozessen zu jeder Zeit effizient und wettbewerbsgerecht unterstützen. Die Geschäftsprozesse werden dabei durch eine gemeinsame Architektur, die Service Oriented Architecture (SOA), unterstützt und zusammengehalten.

Komponenten werden über das Web angeboten, und über den Web Service Standard verbindbar. Das Web ermöglicht es, physisch entfernte Komponenten mit lokalen Komponenten zu koppeln und die Leistung anschließend im Web bereitzustellen. Prozessmodellierungs-Tools unterstützen die Entwicklung neuer Geschäftsprozesse. Des Weiteren können sie bestehende Geschäftsprozesse auf der Geschäftsebene und anschließend auf der IT-Ebene durch die Auswahl, Einbindung und Montage geeigneter Komponenten verändern. Der Ablauf des Geschäftsprozesses kann einfach nachvollzogen werden. Das gibt dem für den Geschäftsprozess Verantwortlichen die Möglichkeit zu zeitnahen Überprüfungen, Anpassungen, Eingriffen und Korrekturen.
Eine SOA mit einem Enterprise Bus sorgt für die nötige feste Infrastruktur als Rahmen für die dynamischen Teile der Geschäftsprozess-Unterstützung. Die SOA mit allen Komponenten ih-

rerseits setzt auf einer durch Grid Standard virtualisierten und damit an den aktuellen Bedarf anpassbaren IT-Hardware–Struktur auf.

Aus den klassischen statischen IT-Verfahren werden dynamische, vernetzte Abläufe, aus IT–Verfahren im Unternehmen werden IT-Verfahren, die nahtlos die ganze Wirkungskette firmenübergreifend unterstützen. Die klassische Frage, ob Anwendungen von der Stange gekauft oder für den Eigenbedarf maßgeschneidert selber entwickelt werden sollen, weicht einem feingranularerem Konzept, das die Nutzung vorhandener Komponenten mit einem Angebot von Komponenten zur Nutzung durch andere nahtlos verbindet.

Standards wie Web Services, XML und Internet sind die grundlegenden Bausteine dieser Vision. Die Richtung und die Notwendigkeit der Veränderung werden nahezu einstimmig und allgemein von den Kunden und der Industrie geteilt. Es wird eine dauerhafte Kooperation der IT-Industrie im Bereich der Standards und erhebliche Entwicklungsarbeit erfordern, diese Vision in die Realität von Produkten, Serviceangeboten und Verfahren zu überführen.

Die Vision ist eine wichtige Orientierung, um heute mit dem Machbaren und Nutzbaren in der richtigen Richtung zukunftssicher zu investieren.

Weitere Referenzen

- http://www-306.ibm.com/software/info/openenvironment/
- http://ftp.software.ibm.com/software/info/openenvironment/CBDI-Report-SOA-Introduction-for-Managers-2004-July.pdf
- http://www-306.ibm.com/software/solutions/webservices/
- Service Orientation and Its Role in Your Connected Systems Strategy - http://msdn.microsoft.com/Longhorn/understanding/pillars/Indigo/default.aspx?pull=/library/en-us/dnbda/html/srorientwp.asp

Anhang

Abbildung 3 Web Services Stack und Stand der Standardisierung

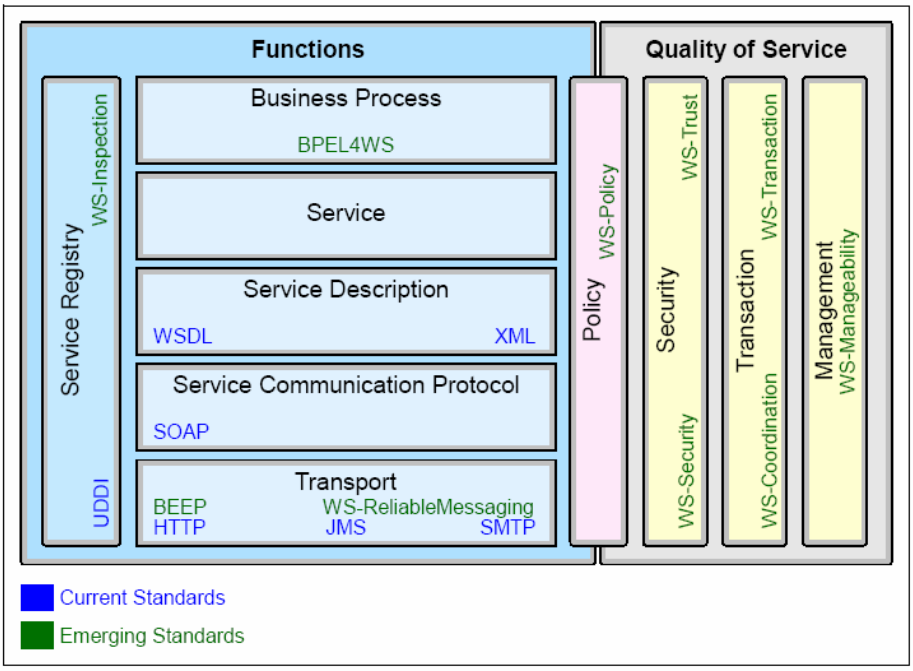


Figure 5-1 Web services and the service-oriented architecture stack

Glossar

Begriffe

SOAP:	Simple Object Access Protocol
WSDL:	Web Service Definition Language
UDDI:	Universal Description, Discovery and Integration, http://www.uddi.org/
XML:	Extensible Markup Language
EAI:	Enterprise Application Integration
W3C:	Word Wide Web Consortium
BPEL4WS :	Business Process Execution Language for Web Service
Kerberos :	ist ein verteilter Authentifizierungsdienst (Netzwerkprotokoll) zur Authentifizierung, der für offene und unsichere Computernetze (z. B. das Internet) von Steve Miller und Clifford Neuman entwickelt wurde.
XAML :	eXtensible Application Markup Language
IPSEC:	IP Security
CORBA.	Common Object Request Broker Architecture
DCOM :	Distributed Component Object Model
MSDN:	Microsoft Developer Network

Danksagung

Der vorliegende Leitfaden entstand in dem BITKOM-Arbeitskreis „Application Development & Integration Technology“.

Wir danken allen Mitgliedern des Arbeitskreises für die konstruktive und kontinuierliche Mitarbeit sowie die hilfreichen Verfeinerungsvorschläge. Besonderer Dank gilt den Autoren der vorliegenden Kapitel Hans-Peter Brill, Solution Consulting, Colt Telekom GmbH, Joachim Moser, Managing Director, GFT Technologies AG, Matthias Neugebauer, Architekturberater, Microsoft Deutschland GmbH, Felix Ott, Gruppenleiter Softwareentwicklung, DATEV eG, Michael Starck, Project Manager, NIIT Technologies AG, Maria Sundermann, Inubit AG, Jan Thielscher, Geschäftsführer, profit architects, Friedrich Vollmar, Manager Tech Sales Composite Applications, IBM Deutschland GmbH, Dr. Birigt Heinz, Leiterin Kompetenzbereich Software, BITKOM e.V. für das außerordentliche Engagement und die z. T. wissenschaftlichen Textbeiträge, die diesen Leitfaden erst ermöglichten.

Der Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V. vertritt 1.300 Unternehmen, davon gut 700 als Direktmitglieder mit etwa 120 Mrd. Euro Umsatz und etwa 700.000 Beschäftigten. Hierzu zählen Produzenten von Endgeräten und Infrastruktursystemen sowie Anbieter von Software, Dienstleistungen, neuen Medien und Content. BITKOM setzt sich insbesondere für bessere ordnungsrechtliche Rahmenbedingungen, eine Modernisierung des Bildungssystems und eine innovationsorientierte Wirtschaftspolitik ein.

Bundesverband Informationswirtschaft,
Telekommunikation und neue Medien e.V.
Albrechtstraße 10
10117 Berlin-Mitte

Tel.: 030/27 576 - 0
Fax: 030/27 576 - 400

bitkom@bitkom.org
www.bitkom.org