

WebServices - Einsatzfelder und Systemarchitektur

Roger Heutschi, Florian Leser, Rainer Alt

Bericht Nr.: BE HSG/ CCBN2/ 2

Lehrstuhl: Prof. Dr. H. Österle

Version: 1.0

Datum: 2. Oktober 2002

**Universität St. Gallen -
Hochschule für Wirtschafts-, Rechts-
und Sozialwissenschaften (HSG)**

Institut für Wirtschaftsinformatik
Müller-Friedberg-Strasse 8
CH-9000 St. Gallen
Tel.: ++41 / 71 / 224 2420
Fax: ++41 / 71 / 224 2777

Prof. Dr. A. Back
Prof. Dr. W. Brenner (geschäftsführend)
Prof. Dr. H. Österle
Prof. Dr. R. Winter

Inhaltsverzeichnis

1	Einführung	4
2	Begriff und Einsatzfelder von WebServices	4
2.1	Sichten auf WebServices	4
2.2	Outtasking und Zusatzdienste	7
2.3	Systemintegration	8
3	Architektur von WebServices	12
3.1	Gestaltungselemente von Business Networking-Lösungen.....	12
3.2	Nutzungsprozess von WebServices	13
3.3	Bestandteile einer WebService Systemarchitektur	14
3.3.1	WebService Standards.....	16
3.3.2	WebService Frameworks	22
3.3.3	WebService Plattformen	22
3.3.4	WebService Entwicklungsumgebungen.....	23
3.3.5	WebService Laufzeitumgebungen	23
3.3.6	WebService Verzeichnisse	23
3.3.7	WebService-Management	23
4	Anwendungsbeispiel: Stockquote WebService	24
5	Konsequenzen für Unternehmen	26
	Literatur.....	30
Anhang A:	Definitionen von WebServices.....	34
Anhang B:	Ziele und Ebenen der Systemintegration	36

1 Einführung

Mit WebServices haben Softwareanbieter, Beratungs- und Marktforschungsunternehmen ein neues Schlagwort geprägt. Gartner geht davon aus, dass bis Ende 2005 Software auf Basis von WebServices einen Umsatz von 21 Mrd. USD erreicht [vgl. Plummer 2002, 1]. IDC rechnet damit, dass die durch WebServices erwirtschafteten Einnahmen zwischen 2004 und 2007 von 1,6 Mrd. USD auf 34 Mrd. USD steigen [vgl. Hailstone 2002, 7]. Diese Erwartungen basieren zunächst auf den vielen technischen Spezifikationen, die in den vergangenen Jahren zu WebServices begonnen haben. Über standardisierte Schnittstellen und Protokolle sollen Applikationen künftig auf unterschiedlichen Systemen lauffähig und vor allem flexibel integrierbar sein. Fortschritte in dieser Richtung erhöhen unmittelbar die Netzwerkfähigkeit von Unternehmen und lassen Verbesserungen gegenüber den proprietären und kostenintensiven bisherigen Technologien wie etwa EDI oder EAI erkennen. Für Unternehmen ergeben sich durch die schnellere und günstigere Integration von Geschäftspartnern vor allem Möglichkeiten zur Fremdvergabe von nicht-Kernleistungen und zur Erweiterung des eigenen Leistungsangebotes durch die flexible Einbindung von Partnern.

Der vorliegende Arbeitsbericht soll den Begriff „WebServices“ klären, den derzeitigen Entwicklungsstand und Einsatzfelder aufzeigen sowie Auswirkungen auf die Unternehmensarchitektur und Gestaltungshinweise für Unternehmen ableiten. Er soll damit eine erste Beurteilung der Auswirkungen von WebServices auf die Netzwerkfähigkeit schaffen. Kapitel 2 grenzt zunächst Einsatzfelder und Potenziale von WebServices ab. Kapitel 3 beschreibt den Ablauf der Erstellung und Nutzung von WebServices und beleuchtet die Bestandteile einer WebService-Architektur. Kapitel 4 illustriert die Ausführungen an dem konkreten Anwendungsbeispiel eines Stockquote-Services. Kapitel 5 zeigt schliesslich die aktuellen Herausforderungen für Unternehmen auf und versucht, den Einfluss von WebServices auf künftige Unternehmensarchitekturen zu beurteilen.

2 Begriff und Einsatzfelder von WebServices

2.1 Sichten auf WebServices

In Praxis und Literatur finden sich unterschiedliche Begriffsverständnisse zu WebServices (siehe Anhang A). Dies zeigen die folgenden Beispiele:

- Das Logistikunternehmen Inet-Logistics bietet eine Web-basierte Logistik-Plattform, über die Unternehmen elektronische Transport- und Statusinformationen austauschen und Versandpapiere und Labels erstellen [Gizanis et al. 2002]. Als Shared Services oder externer eService übernimmt Inet-Logistics spezifische Aufgaben für seine Kunden. Dabei kommen weitgehend eigendefinierte Technologiestandards zum Einsatz.
- Die Internetsuchmaschine Google bietet die Möglichkeit, statt wie bisher üblich über den Browser, nun auch über eine Schnittstelle automatisch Suchanfragen zu starten. Suchergebnisse werden als Datensatz in einem standardisierten Format (API basierend auf SOAP und WSDL) ausgegeben und nicht wie herkömmlich auf einer Internetseite darzustellen. Dies erlaubt eine elektronische Weiterverwendung der Suchergebnisse z.B. in Unternehmensportalen [Computerwoche 2002].
- Die US-Warenhauskette Nordstrom.com verwendet WebService-Technologien zur Integration ihres ERP-System mit einem Onlineshop und einem Lagerhaltungssystem. WebServices kommen dabei als Middleware zum Einsatz, alternativ zu bestehenden Lösungen wie MQSeries von IBM, oder Webmethods. Gemäss [Patton 2002] besitzt die WebService-Lösung jedoch einen deutlichen Kostenvorteil.

Aus diesen Beispielen gehen zwei grundsätzliche, teilweise überlappende Sichten von WebServices hervor (siehe Bild 2-1 und Tabelle 2-1) [vgl. Yates/Kafka 2001]:

- Eine *betriebswirtschaftliche Sicht*, die WebServices als externe Dienstleister betrachtet, die klar abgrenzbare, standardisierte Aufgaben aus Prozessen übernehmen. Im Mittelpunkt steht die durch WebServices erbrachte Leistung und deren Auswirkung auf Geschäftsprozesse sowie das Geschäftsmodell von WebServices. Die damit verbundenen Einsatzfelder sind die Erhöhung des Kundennutzens durch die Erweiterung der Unternehmensleistungen mit externen Leistungen und die Möglichkeit zum Outtasking, d.h. der Fremdvergabe modularer Aufgaben in Geschäftsprozessen.
- Eine *technologische Sicht*, die WebServices als eine bestimmte Systemarchitektur betrachtet. WebServices sind danach selbstbeschreibende Softwarekomponenten die Funktionalitäten kapseln, die von anderen WebServices oder Applikationen plattformunabhängig über standardisierte Schnittstellen genutzt werden können (vgl. [Sleeper/Robins 2001], [UDDI.org 2001]). Die technologische Sicht konzentriert sich damit auf Mecha-

nismen zur Applikationsintegration und hat als Einsatzfeld die vereinfachte Systemintegration.

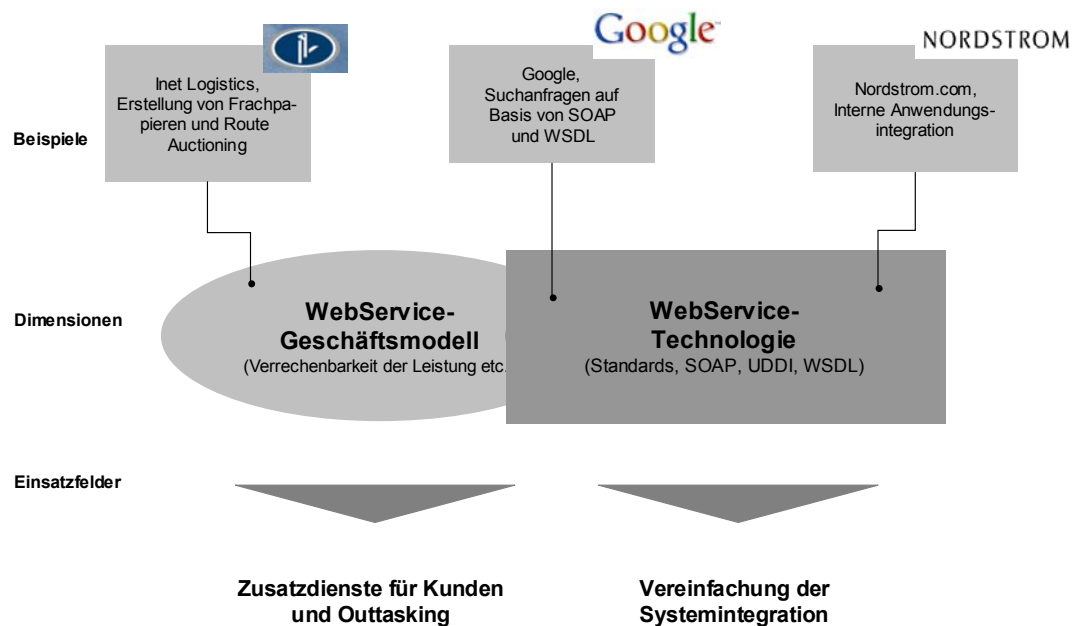


Bild 2-1: Zwei Sichten von WebServices

Geschäfts- und Prozesssicht	Technologische Sicht
WebServices...	
übernehmen eine <i>klar abgrenzbare, eigenständige Geschäftsaufgabe</i>. sind <i>wiederverwendbar</i> und können einzeln in verschiedenen Geschäftsprozessen verwendet oder mit anderen Web-Services gebündelt werden. sind <i>vollständig automatisiert</i> und <i>besitzen gegen Null strebende Grenzkosten</i> für eine einzelne Leistungserbringung. erbringen <i>standardisierte</i> und <i>modularisierte Leistungen</i>. werden <i>zeit- oder nutzungsbasiert</i> verrechnet.	sind <i>selbstbeschreibend</i>. Sie können Auskunft geben über ihre Funktionalitäten, den Input den sie benötigen und den Output den sie generieren (z.B. WSDL). basieren auf anerkannten <i>Internet-Protokollen</i>. Sie tauschen Nachrichten zum Beispiel auf Basis von TCP/IP, HTTP oder SMTP und XML (SOAP) aus. sind <i>plattform- und ortsunabhängig</i>. Sie können von unterschiedlichen Standorten und technologischen Plattformen aus verwendet werden. <i>kapseln spezifische Funktionalitäten</i> und <i>sind lose gekoppelt</i>. Als in sich geschlossene Anwendungen erbringen sie durch Parameterübergabe konfigurierbare Leistungen. Interne Änderungen beeinflussen die den Service nutzenden Komponenten nicht.
Einsatzfelder	
- Zusatzdienste für Kunden - Outtasking	Systemintegration

Tabelle 2-1: Merkmale und Einsatzfelder von WebServices

2.2 Outtasking und Zusatzdienste

Outtasking setzt die bekannten Konzepte der Kernkompetenzorientierung und des Outsourcing fort, welche die Auslagerung von Unternehmensteilen und Geschäftsfeldern bzw. von Prozessen in Geschäftsfeldern vorsehen. Outtasking zielt auf die Auslagerung einzelner Aufgaben innerhalb von Prozessen, so dass beispielsweise nicht die gesamte Logistikabwicklung, sondern lediglich einzelne Aufgaben innerhalb dieses Prozesses fremdbezogen werden (siehe Bild 2-2). Ein Beispiel ist der Schweizer Uhrwerkhersteller ETA SA, ein Tochterunternehmen der Swatch Group. Den Versand elektronisch bestellter Ersatzteile unterstützen WebServices von Inet-Logistics und 3C Systems. Wie in Bild 2-3 gezeigt, erledigt Inet-Logistics Aufgaben wie die Zolldatenaufbereitung, die Beauftragung der Transportdienstleister, die Transportpreisermittlung sowie die Erstellung von Frachtpapieren in der Transportplanung. 3C Systems übernimmt die gesamte Kreditkartenabwicklung.

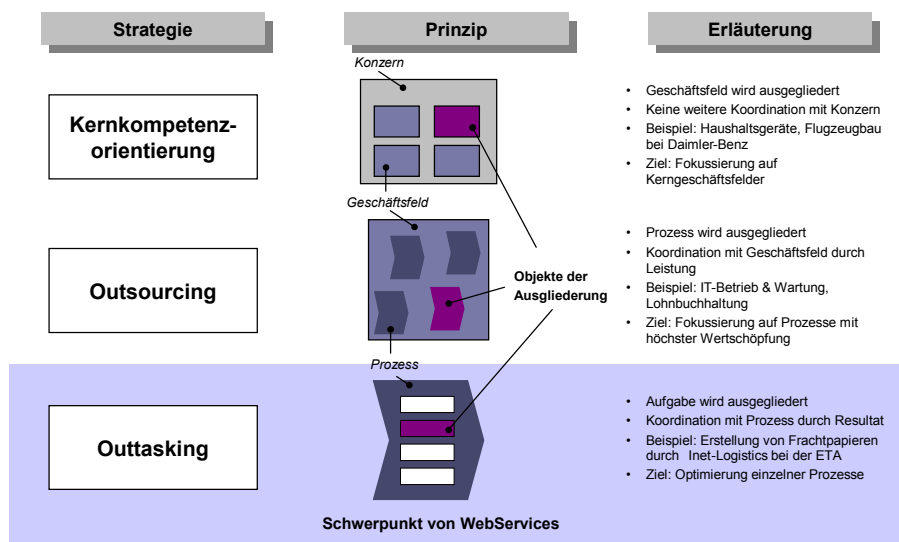


Bild 2-2: Strategien zur Ausgliederung von Unternehmensteilen

Indem ETA SA beide WebServices in seinen Online-Shop elektronisch integriert hat, stehen die Informationen der Transport- und Kreditkartenabwicklung in Echtzeit zur Verfügung. Dies betrifft einerseits die Weitergabe von Transportaufträgen oder Kreditkartenzahlungen an die WebServices, andererseits stehen vor allem Statusinformationen über die Aufträge im Online-Shop für die Kunden zur Verfügung. Dies sind Zusatzdienste für Kunden von ETA SA, die zum Kundennutzen beitragen und schliesslich positiv auf die Kundenbindung wirken können. Weitere Beispiele für die Einbindung von WebServices in Unternehmensleistungen sind:

- Cisco bietet Händlern über sein Portal einen Webservice an, der auf Basis der Produktkataloge von Cisco und den Preisdaten der Händler individuelle Preise für Ihre Kunden berechnet [vgl. Yates/Kafka 2001, 8].
- Networkcar bietet Kunden einen Webservice, der Diagnosedaten, Leistungsdaten und GPS-Daten von Fahrzeugen sammelt und in andere Anwendungen (z.B. für Flottenmanagement, Routenplanung) integriert [vgl. Fou 2002].

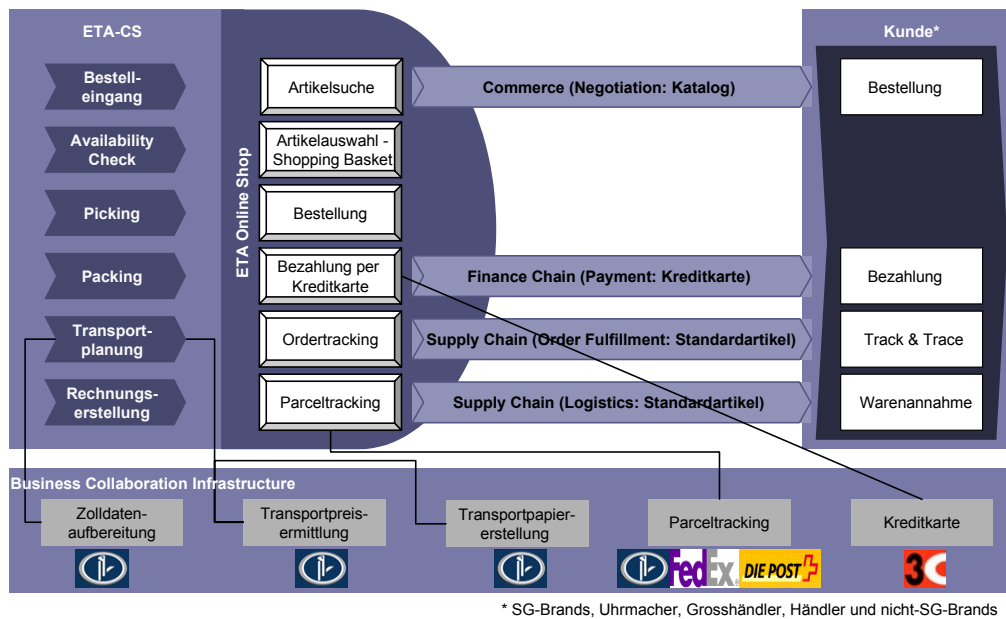


Bild 2-3 Nutzung von WebServices in der Logistikabwicklung bei ETA SA

2.3 Systemintegration

Nachdem aus betriebswirtschaftlicher Sicht nicht die technologische Realisierung im Vordergrund steht, finden sich bei den oben beschriebenen WebServices unterschiedliche Systemarchitekturen. Inet-Logistics und 3C Systems verwenden proprietäre Protokolle und Formate, während Google und Networkcar auf den Internet-Standards HTML, SOAP und XML aufbauen. Bild 2-4 zeigt die wesentlichen Unterschiede zwischen beiden Ansätzen.

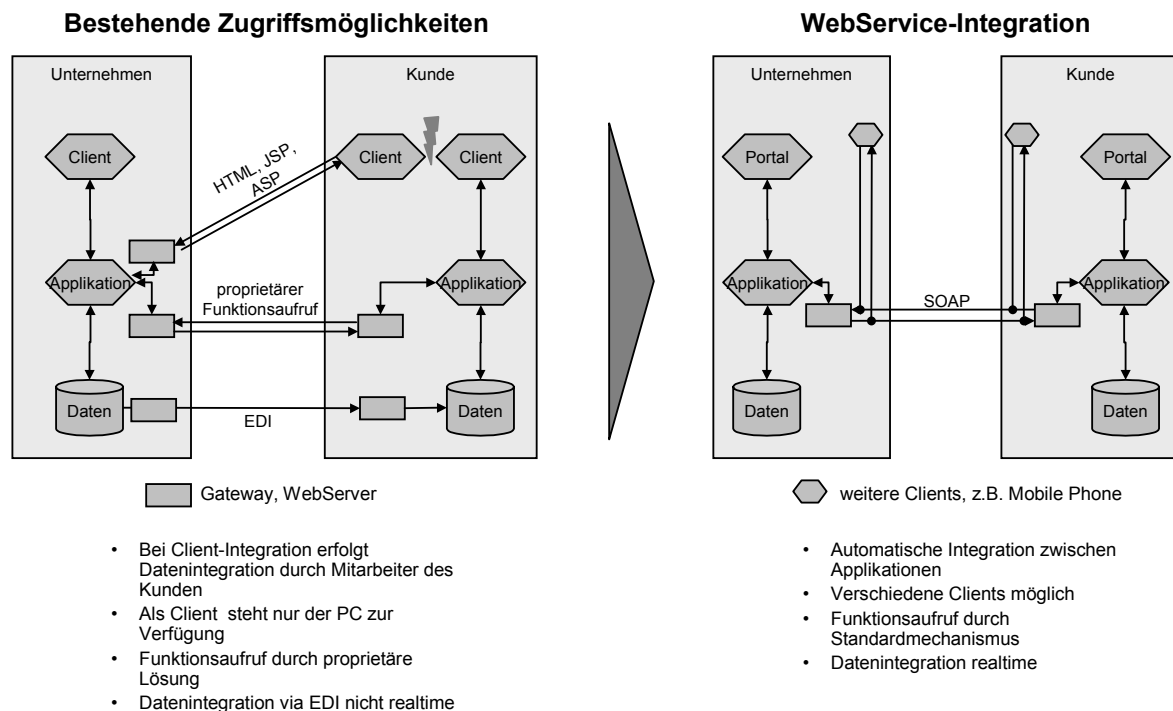


Bild 2-4: Öffnung der Unternehmenssysteme durch WebServices

Nach dem links dargestellten Zugriffsverfahren gewähren bereits viele Unternehmen ihren Kunden, Geschäftspartnern oder Mitarbeitern über Portale oder Websites einen externen Zugriff auf interne Anwendungen. Beispielsweise können Kunden des Onlinebrokers Consors über das Portal direkt auf die interne Vorgangsbearbeitung bei Wertpapiertransaktionen zugreifen. In wenigen Fällen kommen aufgrund hoher Kosten Integrationslösungen auf Datenebene (z.B. EDI) oder Applikationsebene (RFC-Aufrufe bei SAP) zum Einsatz. Mit WebServices sollen interne Funktionen über eine einheitliche SOAP-Schnittstelle für unterschiedliche Clients (Portale, PDAs etc.) [vgl. Patil/Saigal 2002], aber auch für Applikationen direkt zur Verfügung stehen [vgl. Loviscach et al. 2002, 92]. Webservice Standards regeln die Systemintegration bezüglich [Beimborn et al. 2002, 277]:

- Dem *Nachrichtentransport* durch Verwendung standardisierter Protokolle für die asynchrone und synchrone Kommunikation (SMTP, MIME, HTTP, HTTPR).
- Der *Nachrichtenkapselung und -übertragung* durch das SOAP-Protokoll, das mittels eines einheitlich strukturierten Umschlags Ein- und Ausgabeparameter überträgt.
- Der *Beschreibung* und dem *Aufruf* von Applikationen durch die WSDL. Geregelt sind z.B. Eigenschaften der Aufrufparameter, Ports oder das Format der Rückgabedokumente.

- Dem *Auffinden* von Applikationen in n:m-Netzwerken. In einem einheitlich nach UDDI strukturierten, zentralen Verzeichnis sind die Applikationen registriert.

Nordstrom setzt beispielsweise WebServices auch für die Anbindung von Zulieferern ein, wo sie bestehende EDI-Verbindungen ersetzen. Trotzdem wäre es dem Unternehmen prinzipiell möglich, auch weiterhin den EDIFACT-Standard zu nutzen. Beabsichtigt das Unternehmen beispielsweise eine elektronische Bestellung im EDIFACT-Format an seine Lieferanten zu versenden, so könnte über die SOAP-Schnittstelle des Materialplanungssystems ein geeigneter Konvertierungsdienstleister im öffentlichen UDDI-Verzeichnis gesucht werden. Findet sich ein geeigneter Dienstleister, so enthält dessen WSDL-Datei die für einen automatischen Aufruf notwendigen Parameter. Diese fließen in die SOAP-Nachricht zur ‚Beauftragung‘ des Dienstes ein, der den Anhang (die formatierte Bestellung) in das gewünschte EDIFACT-Format konvertiert und über SOAP zurück schickt.¹

Aus technischer Sicht bilden WebServices eine Alternative zur konventionellen Applikationsintegration über ERP- und Middleware- bzw. EAI-Systeme (siehe Tabelle 2-2, zu den Integrationsebenen vgl. Anhang B). WebServices ermöglichen die Kapselung von Anwendungsfunktionen, die Verwendung eines zentralen Verzeichnisses zur Registrierung von Applikationen (ähnlich CORBA oder DCOM²) und den consequenten Einsatz von Internet-Standards bzw. -Protokollen. In zunehmendem Masse setzen ERP- und EAI-Anbieter diese Merkmale ein, so dass sich die idealtypische Abgrenzung verwischt.

¹ Beispiel erstellt in Anlehnung an [Beimborn et al. 2002].

² WebServices sollen die Hersteller- und Plattformabhängigkeit und die Interoperabilität mit Internet-Clients gegenüber der Common Object Request Broker Architecture (CORBA) der Open Management Group (OMG) und dem Distributed Component Object Model (DCOM) von Microsoft verbessern [Gisolfi 2001].

	Integriertes ERP-System	EAI-Architektur	WebService-Architektur
<i>Beschreibung / Prinzip</i>	Ein übergreifend gestaltetes System mit verteilten nach Herstellervorgaben integrierten Komponenten.	Verschiedene dezentrale Applikationen von üblicherweise mehreren Herstellern sind über Adapter mit einem zentralen (EAI-) Hub verbunden.	Verschiedene dezentrale Applikationen kommunizieren über standardisierte anbieter- und plattformunabhängige Nachrichten und Schnittstellen.
<i>Integration auf Applikationsebene</i>			
- <i>Topologie</i>	Eine Applikation.	Verteilte Applikationen mit 1:1 (dezentrale Adapter) bzw. 1:n (zentraler Hub) Verbindungen.	Verteilte Applikationen mit n:m Verbindungen.
- <i>Prozesslogik</i>	Zentrale Prozesslogik.	Dezentrale Prozesslogik mit zentralen Komponenten für übergreifende Prozesse.	Dezentrale Prozesslogik, übergreifendes Prozessmodell definiert in Standards wie z.B. RosettaNet.
<i>Integration auf Datenebene</i>			
- <i>Semantische Integration</i>	Einheitliches semantisches Modell innerhalb des ERP-Systems.	Individuelle semantische Modelle je Applikation. Applikationsspezifische Umwandlung in zentrales Modell durch Konvertierungsdienste.	Individuelle semantische Modelle je Applikation. Entweder homogene Semantik durch gleiche verwendete Standards oder durch Konvertierung in zentrales Modell.
- <i>Syntaktische Integration</i>	Einheitliche proprietäre Nachrichtenstrukturen innerhalb des ERP-Systems.	Unterschiedliche Nachrichten je Applikation, Umwandlung in zentrales, i.d.R. proprietäres Format über applikationsspezifische Adapter.	Standardisierte Nachrichtenstrukturen nach einem XML-Nachrichtenset (z.B. CML).
- <i>Kommunikationsprotokolle</i>	Proprietär.	Proprietär.	Standardisierte Envelopes (SOAP) und internetbasierte Transportprotokolle.
<i>Integration auf Präsentationsebene</i>	Anbieter-spezifische Oberflächen (z.B. SAP GUI).	EAI-Anbieter treten häufig auch als Portal-Anbieter auf.	Standardisierung z.B. mit Web Service User Interface (WSUI) Spezifikation.
<i>Beurteilung</i>	Geringe Flexibilität und hoher zentraler Koordinationsaufwand. Geeignet für stabile Prozesse zwischen gleichbleibenden Geschäftspartnern mit hohen Transaktionsvolumina.	Höhere Flexibilität als bei ERP-Architekturen, da zentral lediglich Schnittstellen anzupassen sind, die Applikationen aber dezentral entwickelt werden können. Geeignet für die Realisierung komplexer Geschäftsprozesse über heterogene Applikationen.	Hohe Flexibilität aufgrund einheitlicher Transport- und Schnittstellenbeschreibungsprotokolle und keine Entwicklung applikationsspezifischer Adapter. Derzeit geeignet für die punktuelle Integration spezifischer Funktionen.
<i>Beispiel</i>	Hewlett Packard - Konzernweites SAP R/3 integriert durch ALE-Szenarios.	Ein EAI-Bus verbindet SAP R/3-Systeme mit eProcurement-, eSales- und Legacy-Systemen.	Bei Cisco findet ein Webservice zum aktuellen Produktkatalog die jeweiligen Verkaufspreise.

Tabelle 2-2: Vergleich von Architekturen zur Anwendungsintegration

3 Architektur von WebServices

3.1 Gestaltungselemente von Business Networking-Lösungen

Der Beitrag von WebServices für das Business Networking wird ermittelt anhand der allgemeinen Gestaltungselemente von Business Networking-Lösungen. Bei der Prozess- und Systemintegration zwischen rechtlich selbständigen Organisationseinheiten sind unabhängig von der verwendeten Technologie verschiedene Gestaltungselemente zu regeln. Aufbauend auf der Business Networking-Architektur [vgl. \Alt, 2001 #3442] unterscheidet Bild 3-1 drei Ebenen. Aus strategischer Sicht müssen WebService-Nutzer und -Anbieter Handelsvereinbarungen treffen (z.B. Rechtsverbindlichkeit elektronisch ausgetauschter Dokumente), Gewährleistungsabkommen (Service Level Agreements) und Verantwortlichkeiten zur Datenverwendung festlegen. Auf Prozessebene müssen sie gemeinsame Prozessspezifikationen, die eigenen Rollen und ausgetauschte Leistungen sowie für einzelne Aufgaben konkrete Ergebnisse definieren. Auf Systemebene müssen die Partner die Nachrichten festlegen, welche die bei der Leistungserbringung betroffenen Anwendungskomponenten austauschen, benötigte Datensätze müssen bestimmt und weitere Fragestellungen der Kommunikation (z.B. eingesetzte Kommunikationsprotokolle) geklärt werden.

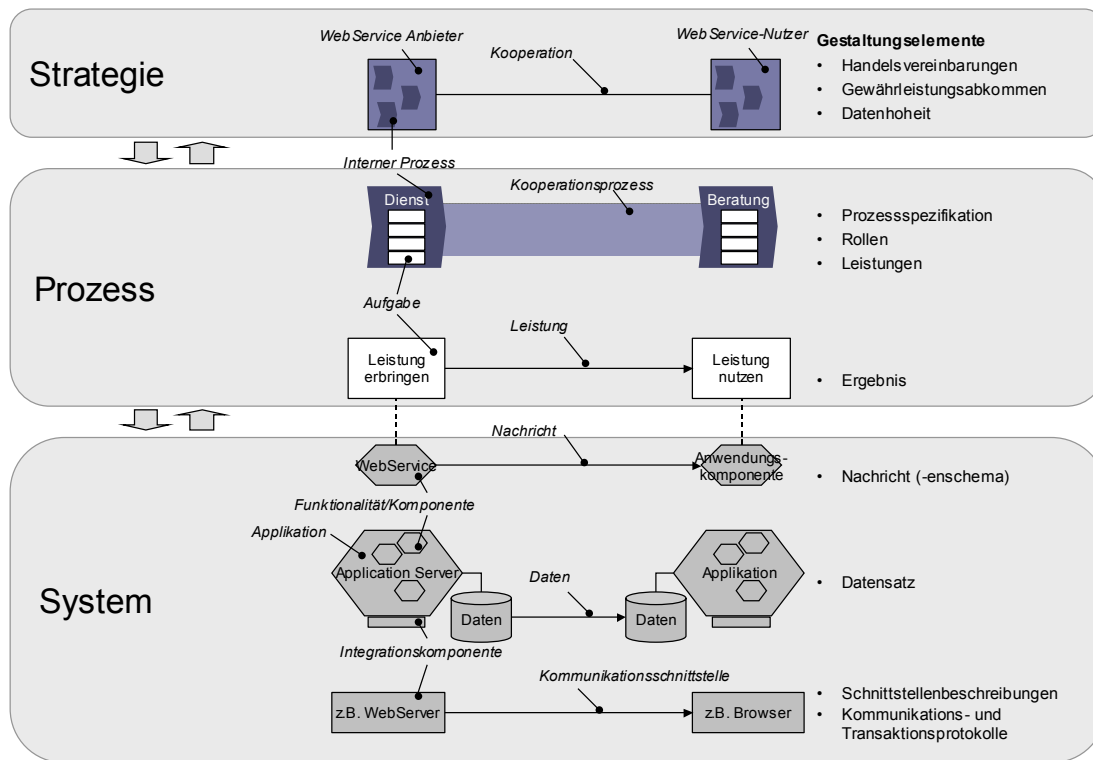


Bild 3-1: Gestaltungselemente einer WebService-Architektur

3.2 Nutzungsprozess von WebServices

Zur Entwicklung, Integration und Nutzung von WebServices bestehen verschiedene Teilprozesse: Der Entwicklungs- und Veröffentlichungsprozess, der Such- und Integrationsprozess und der Betriebs- und Nutzungsprozess. Die einzelnen Teilprozesse können innerhalb einer einzelnen Organisationseinheit durchgeführt werden oder mehrere Partner einbeziehen. In den Teilprozessen übernehmen unterschiedliche Rollen verschiedene Aufgaben. Der WebService-Anbieter entwickelt und betreibt einen WebService und publiziert ihn gegebenenfalls in einem zentralen Verzeichnis. Dieses wird von einem WebService-Verzeichnisanbieter (innerhalb einer Organisation oder als externer Dienst beispielsweise von IBM oder Microsoft) betrieben, der WebServices kategorisiert, beschreibt und so für Dritte verfügbar macht. Ein WebService-Nutzer kann über dieses Verzeichnis geeignete Dienste identifizieren, die entsprechenden Schnittstellen implementieren und den WebService nachrichtenbasiert in eine neue Anwendung integrieren. Bestehen besondere Anforderungen an Sicherheit und Verfügbarkeit oder stellt der WebService nur einen Teil eines komplexeren Prozesses dar, in dem die Dienste mehrerer WebServices gebündelt genutzt werden, kann ein spezialisierter WebService-Manager die Transaktionen überwachen und mittels einer definierten Geschäfts-

logik zusammenstellen. Eine schematische Darstellung der Teilprozesse und der Rollen zeigt Bild 3-2.

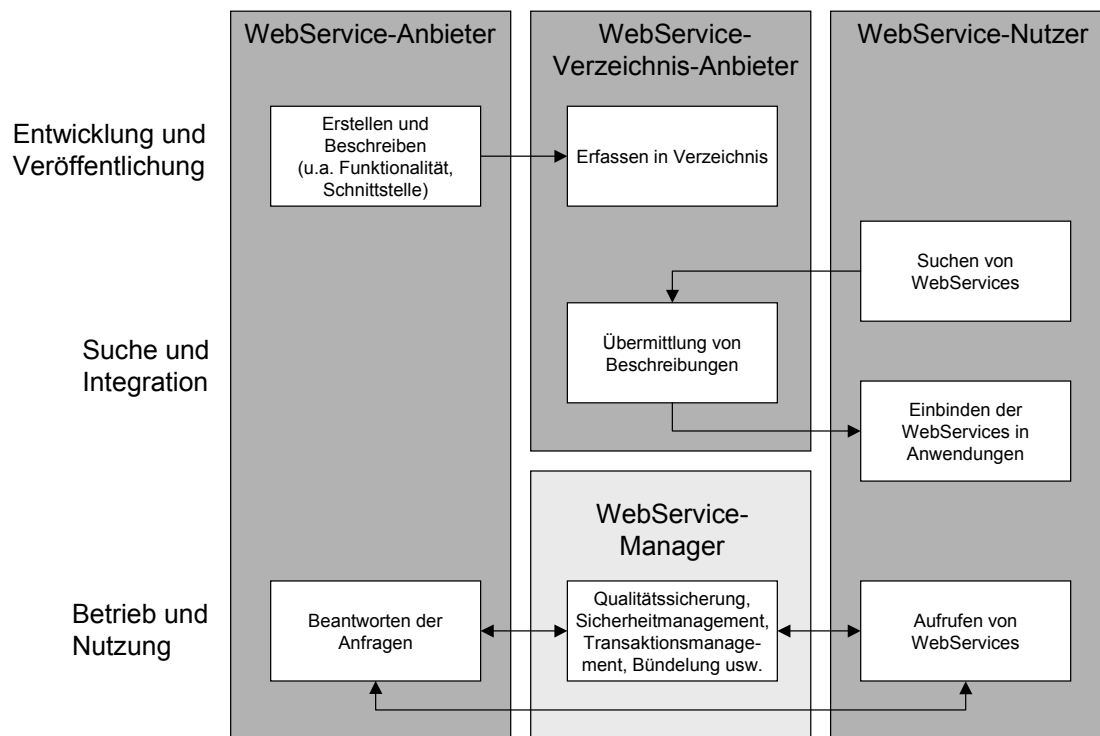


Bild 3-2: Publikation und Nutzung eines WebServices

3.3 Bestandteile einer WebService Systemarchitektur

Ausgehend vom Entwicklungs- und Nutzungsprozess von WebServices (vgl. Bild 3-2) lassen sich verschiedene Bestandteile einer WebService-Systemarchitektur identifizieren (vgl. Bild 3-3). Dazu zählen einerseits gemeinsame Standards, andererseits Plattformen für die Neuentwicklung von WebServices oder das Design von WebService-Adaptoren für bestehende Applikationen, sowie WebService-Verzeichnisse für den Entwicklungs- und Veröffentlichungsprozess. Einen weiteren Bestandteil bilden Laufzeitumgebungen für WebServices und Dienste, die im Betriebs- und Nutzungsprozess das WebService-Management unterstützen. Momentan existieren mit .Net und J2EE zwei grundsätzliche WebService-Frameworks die alle Bestandteile enthalten. Tabelle 3-1 zeigt die Bestandteile einer WebService-Architektur mit Beispielen.

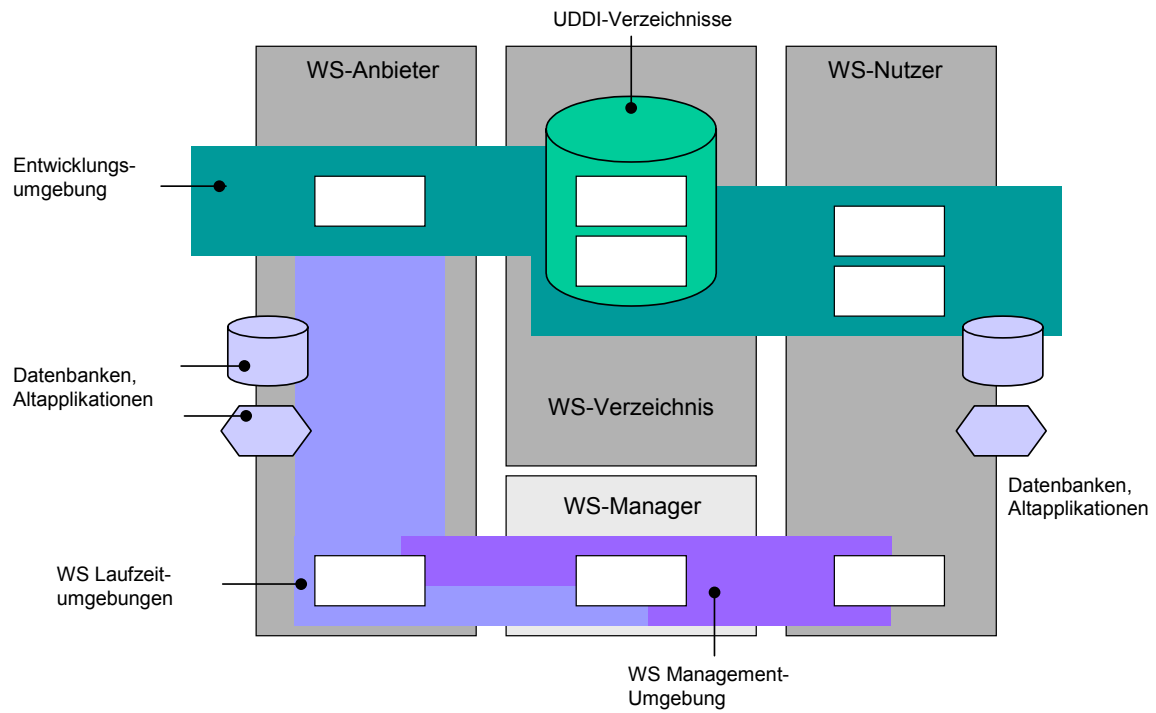


Bild 3-3: Applikationskomponenten für die Entwicklung und die Nutzung von WebServices

WebService-Architektur Bestandteil	Beschreibung	Beispiel	
<i>WebService Standards</i>	<i>WebService Standards bezeichnen die framework- und plattformübergreifenden Spezifikationen</i>	<i>UDDI, SOAP, WSDL</i>	
<i>WebService Framework</i>	<i>Frameworks bezeichnen eine Menge von aufeinander abgestimmten Regeln, Plattformen und Spezifikationen für das Erstellen und Betreiben von WebServices und Applikationen</i>	<i>J2EE</i>	<i>.Net</i>
<i>WebService Plattform</i>	<i>Plattformen bezeichnen eine abgestimmte Menge von Entwicklungsumgebung, Laufzeitumgebung und weiteren Bestandteilen wie Verzeichnisse und WebService Management Umgebungen sowie einzelne WebServices auf Basis eines WebServices Frameworks</i>	<i>IBM Websphere, Bea Weblogic, Sun ONE</i>	<i>Microsoft .Net Produktfamilie</i>
<i>WebService Entwicklungsumgebung</i>	<i>WebService Entwicklungsumgebungen bieten Funktionalitäten zum Entwickeln und Verknüpfen von WebServices</i>	<i>IBM Websphere DE, Bea Weblogic Workshop</i>	<i>Visual Studio .Net</i>
<i>WebService Laufzeitumgebung</i>	<i>WebService Laufzeitumgebungen bieten Funktionalitäten für das Betreiben von WebServices</i>	<i>IBM Websphere Application Server, Sun iPlanet</i>	<i>.Net Common Language Runtime</i>
<i>WebService Verzeichnis (oder Broker)</i>	<i>WebService Verzeichnisse bieten Funktionalitäten zum Registrieren und Auffinden von WebServices</i>	<i>Websphere UDDI Registry</i>	<i>.Net UDDI Service Development Kit</i>
<i>WebService-Management Umgebungen</i>	<i>WebService-Management Umgebungen bieten Funktionalitäten für das zuverlässige, hochverfügbare und sichere Zusammenspiel von WebServices. Wichtigste Dienste sind Integrations-, Messaging-, Authentisierungs- und Transaktionsdienste.</i>	<i>Talking Blocks, Grand Central</i>	

Tabelle 3-1: Bestandteile der WebService-Architektur

3.3.1 WebService Standards

WebServices sind plattformunabhängig und vereinfachen durch standardisierte, an Internet-Protokollen orientierten Schnittstellen die Integration verteilter Applikationen. Nach den Visionen einiger Softwareanbieter oder Autoren [vgl. Hagel 2001] soll es Unternehmen in Zukunft gar möglich sein, ganze Anwendungen dynamisch durch die Bündelung modularer WebServices verschiedener Anbieter zusammenzusetzen. Dies setzt jedoch eine deutliche umfassendere Standardisierung voraus als sie heute gegeben ist.

Derzeit zeichnen sich WebServices vor allem durch verschiedene Standards auf Transport- und Kommunikationsebene aus. Grundlegend sind allgemein akzeptierte Internetstandards und -dienste wie TCP/IP, HTTP, SMTP, HTML oder XML. Auf höherer Ebene bilden UDDI (Verzeichnisse für WebServices), SOAP (Kommunikation zwischen WebServices) und WSDL (Beschreibung von WebServices) die drei wichtigsten weiterführenden Standards für

WebServices. In den drei Standards fehlen jedoch Mechanismen zur Authentifizierung, Verschlüsselung oder zur Koordination komplexer Transaktionen [vgl. Newcomer 2002, 219]. Auch die Standardisierung von Prozessen und ihrer Beschreibung, Daten oder Handelsvereinbarungen werden nicht abgedeckt.

Hier sind andere Standards von Bedeutung, beispielsweise Electronic Business XML (ebXML), RosettaNet, Business Transaction Protocol (BTP), WS-Security oder das Liberty Alliance Project. Diese bauen zwar teilweise auf den WebService-Kernstandards auf bzw. sind zu diesen kompatibel (beispielsweise sehen einige Standards WSDL für die Schnittstellenbeschreibung oder SOAP als Kommunikationsprotokoll vor), sie weisen aber untereinander oft grosse Redundanzen auf (Tabelle 3-2). Für die Beschreibung eines Partners bestehen mit UDDI Schema, RosettaNet Directory und ebXML Core Components gleich drei unterschiedliche Stammdatensätze. BPSS in ebXML, BPEL4WS (Business Process Execution Language for Web Services, kürzlich entstanden aus IBMs WSFL und Microsofts XLANG) oder BPML (Business Process Modelling Language) sind verschiedene, teilweise inkompatible Ansätze zur Modellierung von Geschäftsprozessen aus gekoppelten WebServices. Wie sich die Standards in Zukunft gegeneinander positionieren und welche sich durchsetzen werden, ist gegenwärtig unklar. Es existieren zwar Gremien wie die WS-I¹ oder die Liberty Alliance, welche die Kompatibilität verschiedener für WebServices relevante Standardisierungsinitiativen sicherstellen wollen, das jeweilige Fehlen wichtiger Akteure (z.B. von Sun Microsystems in WS-I oder von Microsoft oder IBM in Liberty Alliance) lässt ihren Erfolg aber in einem kritischen Licht erscheinen. Wichtig ist auch die Akzeptanz von Anwendern. Viele Unternehmen experimentieren derzeit mit den Technologien SOAP, WSDL und z.T. auch UDDI. Für ebXML-Messaging bestehen erste Anwendungen in der Versicherungswirtschaft. RosettaNet findet Anwendung in High-Tech Unternehmen wie CISCO, HP und Texas Instruments. Die übrigen aufgeführten Standards haben bisher noch keine grosse Bedeutung bei Anwendern erlangt.

¹ Die Web Services Interoperability Organization (WS-I) wurde im Februar 2002, angeführt von IBM und Microsoft mit 51 weiteren Unternehmen (u.a. Intel, SAP) gegründet. WS-I versucht die Interoperabilität von WebServices über verschiedene Plattformen, Betriebssysteme und Programmiersprachen sicherzustellen und verabschiedet Spezifikationen auf Basis von SOAP, WSDL und UDDI.

Standardisierungsbereich		SOAP	WSDL	UDDI	ebXML ¹	RosettaNet ²	BTP	WS-Security ³	Liberty Alliance ⁴	BP4WS ⁵	BPM
Strategie z.B.	Handelsvereinbarungen und Gewährleistungsabkommen. Regeln Rechtsverbindlichkeiten ausgetauschter Dokumente oder Haftungsfragen bei Systemausfällen.				✓	✓					
	Semantische Prozess- und Datenstandards. Gemeinsames Verständnis über die inhaltliche Bedeutung von Prozessen und ausgetauschten Daten.					✓			✓		
Prozess z.B.	Prozessbeschreibung. Standardisierte (und damit maschinell interpretierbare) Beschreibung von Aufgaben, Aufgabenträgern und Abläufen.				✓	✓				✓	✓
	Leistungsbeschreibung. Standardisierte Beschreibung/ Klassifizierung von Diensten und Anbietern für ihre automatische Identifikation.			✓	✓	✓					
Informationssystem z.B.	Nachrichtenschemata. Legen die Nachrichten fest, die während einer Leistungserbringung ausgetauscht werden.				✓	✓			✓		
	Transaktionsmanagement. Standards die regeln, unter welchen Bedingungen eine Transaktion als abgeschlossen gilt und wann sie wie rückgängig gemacht werden muss.				✓	✓	✓			✓	✓
	Schnittstellenbeschreibung. Standardisierte Beschreibung von Operationen, die ein Dienst ausführen kann, des Inputs den er dazu benötigt und des Outputs, den er generiert.		✓		✓	✓				✓	✓
	Kommunikationssicherheit. Mechanismen für den Identitätsnachweis, die Garantie der Vertraulichkeit, Datenintegrität und Protokollierung der Kommunikation.				✓	✓		✓	✓		
	Kommunikationsprotokolle. Einheitliche Regeln und Abläufe für den Austausch von Daten zwischen zwei Applikationen.	✓			✓	✓		✓			
¹ ebXML spezifiziert in einigen Bereichen (z.B. Kommunikationssicherheit, Kommunikationsprotokolle) keine eindeutigen Standards, sondern gibt lediglich Empfehlungen für mögliche einzusetzende Standards ab. ² Die Standardisierungsbestrebungen von RosettaNet konzentrieren sich auf die High-Tech Industrie. RosettaNet spezifiziert eine fixe Menge an Prozessen und erlaubt keine eigenen Prozessdefinitionen. ³ Die WS-Security Standards sind eine Erweiterung von SOAP ⁴ Die Liberty Alliance versucht Standards für die Authentifizierung zu setzen. Semantische Prozess- und Daten-Standards beziehen sich nur auf diesen Bereich. ⁵ BP4WS sieht explizit WSDL als Standard für die Beschreibung von Service-Schnittstellen vor.											

Tabelle 3-2: Beispiele für Standardisierungsbedarfe bei der Vernetzung von Unternehmen

3.3.1.1 UDDI

Universal Description, Discovery and Integration (UDDI) ist eine von IBM, Microsoft und Ariba gegründete Initiative an der heute über 300 Unternehmen beteiligt sind. Ziel ist der Aufbau eines offenen, plattformunabhängigen und branchenneutralen Geschäfts- und Diensteverzeichnisses, das es potenziellen WebService-Kooperationspartnern ermöglichen soll, einander zu finden und Leistungen miteinander auszutauschen.

Ein UDDI-Verzeichnis besteht aus drei Teilen: den White-, Yellow- und Green-Pages. In die White-Pages trägt ein WebService-Anbieter allgemeine Unternehmensdaten wie Name und Kontaktinformationen ein. Die Yellow-Pages listen WebServices nach Kategorien wie Branche, geographische Region oder Industriecodes auf, und die Green-Pages beinhalten die technischen und geschäftlichen Spezifikationen zur Nutzung der WebServices. Verzeichniseinträge können manuell oder mittels SOAP/XML-Nachrichten automatisiert erstellt werden.

WebService-Nutzer können die Verzeichnisse entweder mittels Browser über die Web Sites der Verzeichnisbetreiber oder über spezielle Clients durchsuchen. Es sind aber auch programmierbare Schnittstellen vorgesehen, die es einer Applikation ermöglichen sollen, dynamisch geeignete Dienste zu identifizieren und zu nutzen [UDDI.org 2001].

Betreiber wie Hewlett Packard, IBM, Microsoft oder SAP bieten globale UDDI-Verzeichnisse für die zwischenbetriebliche Kooperation an. Zusätzlich kann der Einsatz einer unternehmensinternen UDDI-Datenbank sinnvoll sein, um nur innerbetrieblich genutzte WebServices zu koordinieren.

```
<businessDetail generic="2.0" operator="Microsoft UDDI Services"
  truncated="false" xmlns="urn:uddi-org:api_v2">
  <businessEntity businessKey="176a3131-0c20-45d1-b31d-efb4f61b8b14"
    operator="sample" authorizedName="sample">
    <discoveryURLs>
      <discoveryURL useType="businessEntity">
        http://sample/uddipublic/discovery.ashx?businessKey=176a3131...
      </discoveryURL>
    </discoveryURLs>
    <name>sample</name>
  </businessEntity>
</businessDetail>
```

Bild 3-4: Beispiel für einen einfachen UDDI-Geschäftseintrag [Bellwood et al. 2002, 239]

3.3.1.2 SOAP

Das Simple Object Access Protocol (SOAP) ist ein Protokoll für den Nachrichtenaustausch zwischen Applikationen. SOAP basiert auf XML, verwendet in der Regel HTTP für den Datentransport und unterstützt das Konzept der Remote Procedure Calls (RPCs): Eine Applikation kann mittels SOAP-Nachricht bei einer anderen, auf einem entfernten Server und unter Umständen auf einer anderen Plattform laufenden Applikation einen Prozess starten. Dieser verarbeitet die in der Nachricht enthaltenen Daten und liefert das Resultat des Prozesses wieder via SOAP-Nachricht zurück.

Dazu definiert SOAP eine XML-Struktur, um eine Operation einer entfernten Applikation aufzurufen und dieser Operation Parameter mitzuliefern. Ausserdem definiert SOAP eine XML-Struktur um angefragte Werte oder Fehlermeldungen zurückzuliefern. Wie die Inhalte dieser Strukturen – also die spezifischen Tags für die zu verarbeitenden Werte – genau auszusehen haben, wird durch den Webservice-Anbieter in einem WSDL-Dokument spezifiziert und in einem UDDI-Verzeichnis publiziert (vgl. Bild 3-5). Durch den Einsatz von HTTP als Datentransferprotokoll ist der Nachrichtenaustausch auch über Firewalls hinweg möglich. Zur Verschlüsselung der Nachrichten lassen sich Mechanismen wie SSL einsetzen.

Die SOAP-Spezifikation liegt momentan in der Version 1.2 vor [vgl W3C 2001a] und gilt als der zukunftsträchtigste Standard für den Nachrichtenaustausch zwischen WebServices.

```
<SOAP-ENV:Envelope
  xmlns:Soap="http://schemas.xmlsoap.org/soap/envelope">
  <SOAP-ENV:Header>
    <program:Info SOAP-ENV:mustUnderstand="1"
      xmlns:program="urn:eddys-stockomatic:program.xsd">
      <program:Name>Stockomatic</program:Name>
      <program:Version>0.83</program:Version>
    </program:Info>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <stocks:GetQuote
      xmlns:stocks="urn:eddys-stockomatics:stocks.xsd">
      <stocks:Symbol>GTW</stocks:Symbol>
      <stocks:Time>2001-09-05T16:00:00.000-05:00</stocks:Time>
    </stocks:GetQuote>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Bild 3-5: Beispiel eines Anfrage-Dokuments mit SOAP [Fitzgerald 2001]

3.3.1.3 WSDL

Die Web Service Description Language (WSDL) ist eine XML-basierte Beschreibungssprache für WebServices. Sie beschreibt die Funktionalität eines WebService und wie die Verbindung und Kommunikation mit ihm zu erfolgen hat. Diese Spezifikationen dienen als wesentlicher Input für die Green-Pages eines UDDI-Verzeichnisses.

Ein WSDL-Dokument (vgl. Bild 3-6) definiert einen WebService als eine Sammlung von Endpunkten, sogenannten „Ports“, die eine Kombination von Netzwerkadresse und Bindung („binding“) darstellen. „Bindings“ beschreiben die beim Nachrichtenaustausch verwendeten Protokolle und Datenformate. „Messages“ sind abstrakte Definitionen der ausgetauschten Daten und die sogenannten „Port-types“ spezifizieren die Operationen, die ein WebService ausführen kann [Ariba 2001].

WSDL wurde von IBM, Microsoft und anderen Unternehmen als Entwurf im W3C eingereicht [vgl. W3C 2001b], geht von einem Nachrichtenaustausch zwischen WebServices mittels SOAP aus und ermöglicht die Veröffentlichung und Weiterverbreitung der WebServices via UDDI.

```
<?xml version="1.0" encoding="UTF-8" ?>
- <definitions xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://tempuri.org/StockQuoteService" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
  name="StockQuoteService" targetNamespace="http://tempuri.org/StockQuoteService">
- <message name="SymbolRequest">
  <part name="symbol" type="xsd:string" />
</message>
- <message name="QuoteResponse">
  <part name="quote" type="xsd:string" />
</message>
- <portType name="StockQuoteService">
  - <operation name="getQuote">
    <input message="tns:SymbolRequest" />
    <output message="tns:QuoteResponse" />
  </operation>
</portType>
- <binding name="StockQuoteServiceBinding" type="tns:StockQuoteService">
  <soap:binding style="rpc" transport="http://schemas.xmlsoap.org/soap/http" />
  - <operation name="getQuote">
    <soap:operation soapAction="http://tempuri.org/GetQuote" />
    - <input>
      <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://tempuri.org/live-stock-quotes" use="encoded" />
    </input>
    - <output>
      <soap:body encodingStyle="http://schemas.xmlsoap.org/soap/encoding/" namespace="http://tempuri.org/live-stock-quotes" use="encoded" />
    </output>
    </operation>
  </binding>
</definitions>
```

Bild 3-6: Beispiel für eine WSDL-Definition eines Börsenkurs-Services

3.3.2 WebService Frameworks

WebService Frameworks stellen einen umfassenden Ansatz zur Implementierung von Web-Service-Architekturen dar. Sie beinhalten Richtlinien, Spezifikationen und Plattformen für die Entwicklung und Bereitstellung, das Management und die Nutzung von WebServices und umfassen Bestandteile wie Entwicklungsumgebungen, Applikationen die WebServices anbieten oder nutzen, Laufzeitumgebungen, Integrationsdienste, Messaging-Dienste, Prozessmanagement-Dienste, Transaktionsmanagement- oder Sicherheitsdienste [Samtani 2002b]. Die gegenwärtig verfügbaren Hauptframeworks sind Java 2 Platform, Enterprise Edition (J2EE) und das .NET-Framework.

- Das *J2EE-Framework* baut auf einem Java Application Server als Laufzeitumgebung auf und spezifiziert verschiedene Programmierschnittstellen (APIs) und -Modelle. Die WebServices werden typischerweise in Form von Enterprise JavaBeans (EJBs), unterstützt von verschiedenen Entwicklungsplattformen, implementiert.
- Das *.NET-Framework* baut auf Microsofts Windows-Plattform auf und will um diese Plattform herum entwickelte Applikationen in Form von WebServices verfügbar machen.

Die Kompatibilität zwischen diesen beiden Framework-Welten soll durch die Einhaltung der WebService Standards sowie durch Organisationen wie die Web Service Interoperability Group (WS-I) sichergestellt werden.

3.3.3 WebService Plattformen

Plattformen für das J2EE-Framework sind von einer Vielzahl von Anbietern wie BEA Systems (WebLogic), Sun Microsystems (ONE), IBM (Websphere) oder Hewlett-Packard (Net Action) erhältlich.

Produkte für das .Net-Framework stammen ausschliesslich von Microsoft und umfassen Server (etwa BizTalk Server 2000 oder SQL Server 2000), Client Applikationen (etwa Windows CE .NET), Entwicklungsumgebungen (Visual Studio .NET) und WebServices (z.B. MS MapPoint.Net). Abgesehen von den durch externe Unternehmen entwickelten WebServices stehen diese Bestandteile unter der Kontrolle von Microsoft.

Nach Schätzungen der Gartner Group wird die .NET Plattform von Microsoft 2005 auf einen Marktanteil von 30 bis 40 Prozent, J2EE-basierte Plattformen auf 40 bis 50 Prozent Marktanteil kommen [vgl. Virtel 2002].

3.3.4 WebService Entwicklungsumgebungen

Entwicklungsumgebungen unterstützen das Design von WebService-fähigen Applikationen über den ganzen Entwicklungsprozess und erlauben beispielsweise das automatische Generieren von WSDL-Dokumenten. Weiterhin unterstützen sie das Entwickeln von Anwendungen, die WebServices einsetzen.

Hersteller sind etwa Borland (JBuilder, Delphi, C++ Builder), Microsoft (Visual Studio .NET), SUN (J2SE 1.4) oder Bowstreet (Business Web Factory).

3.3.5 WebService Laufzeitumgebungen

Laufzeitumgebungen gewährleisten den Betrieb eines WebServices. Sie stellen Dienste für die Transformation von XML-Dokumenten im SOAP-Format (sog. SOAP-Listener) zum Funktionsaufruf bereit und senden XML-Dokumente zurück. Weiterhin bieten sie – insbesondere für das J2EE-Framework – Skalierbarkeit der aufgerufenen Anwendungen um flexibel unterschiedliche Lastzahlen von WebService-Aufrufen unterstützen zu können.

Typische Anbieter sind die Hersteller von J2EE Application Servern wie Bea, IBM und Microsoft mit der Common Language Runtime als Teil der .Net-Plattform.

3.3.6 WebService Verzeichnisse

WebService Verzeichnisse unterstützen die Service-Registrierung und Suche in öffentlichen oder privaten Verzeichnissen. Sie enthalten Datenstrukturen und Taxonomien zur Beschreibung von WebServices und WebService-Anbietern [vgl. Hanson 2002, 6]. Insbesondere UDDI und WSDL werden hierzu eingesetzt. Hersteller sind IBM, HP und Microsoft, sowie spezialisierte Anbieter wie Infravio.

3.3.7 WebService-Management

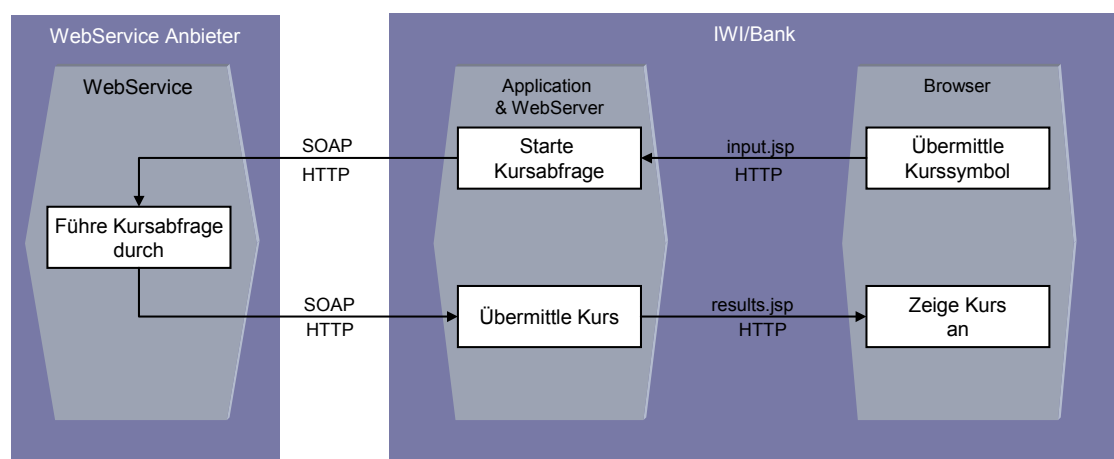
WebService-Management Lösungen übernehmen die Rolle einer spezialisierten Middleware-Schicht für WebServices, welche sich um Bereiche wie Sicherheit, Skalierbarkeit, Qualität

oder Transparenz kümmert. Sie sollen eine Trennung der Applikationslogik von allen WebServices gemeinsamen Infrastrukturdiensten ermöglichen. Zukünftig sollen auch spezielle Broker entstehen, die mehrere WebServices bündeln und komplexe Transaktionen über diese hinweg managen können (vgl. [Mikalsen 2001]; [Samtani 2002a]).

Spezialisierte Lösungen existieren beispielsweise von Primordial, Talking Blocks, Blue Titan, Grand Central, AmberPoint oder Flamenco Networks.

4 Anwendungsbeispiel: Stockquote WebService

In einem Prototyp hat das IWI-HSG von Dezember 2001 bis Februar 2002 eine Lösung für das Outtasking erstellt. Der Beratungsprozess einer Bank wird durch einen Stockquote WebService unterstützt, der aktuelle Kursdaten zu einem gewünschten Wertpapier ermittelt. Für den WebService-Nutzer (die Bank) bleibt die interne Systemarchitektur des WebService-Anbieters (Stockquote WebService) verborgen (vgl. Bild 4-1).



Die Technologie des WebService-Anbieters ist aus Sicht des WebService-Nutzers unsichtbar.

Bild 4-1: Aufgabenkette des Beratungsprozesses mit WebService (Ausschnitt)

Bild 4-2 zeigt beispielhaft die Architektur auf Geschäfts-, Prozess- und Systemebene. Aspekte wie rechtliche Vereinbarungen zwischen Leistungserbringer und Nutzer, die in einem echten Kooperationsszenario geregelt werden müssten, wurden im Prototyp nicht behandelt.

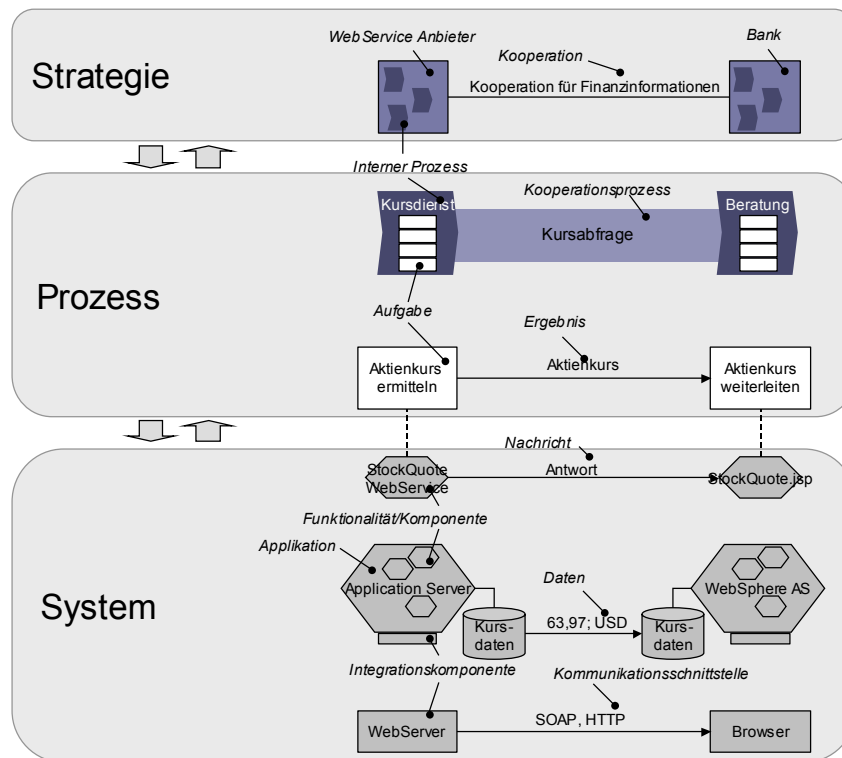


Bild 4-2: Webservice aus Geschäfts-, Prozess- und Systemsicht – Beispiel Stockquote

Aufgrund der leicht und kostenlos verfügbaren Software und Dokumentation haben – neben einem Windows2000 Server Betriebssystem von Microsoft – ausschliesslich Produkte von IBM Verwendung gefunden (Tabelle 4-1).

Bei der Realisierung des Prototyps zeigten sich Schwierigkeiten in einer mangelnden Abstimmung der eingesetzten Bestandteile und einer grossen Versionsdynamik. So konnten Erweiterungen für den Aufbau eines eigenen UDDI-Verzeichnisses trotz Support des Herstellers nicht realisiert werden. Weiterhin zeigte sich bei der Nutzung des Webservices eine hohe Störanfälligkeit des aufgerufenen Stockquote Webservices. Der Verbindungsaufbau zum Stockquote Webservice war schwerfällig und brach häufig ab. Der Prototyp hat jedoch gezeigt, dass Anwendungen mit geringen Programmierkenntnissen und begrenztem Zeitaufwand realisierbar sind. Schwierigkeiten in der Entwicklung sowie eine höhere Verfügbarkeit sind Bereiche zu denen bereits wenige Monate nach Fertigstellung des Prototypen eine Vielzahl neuer Produkte verfügbar sind. Insbesondere zum Webservice-Management bestehen neue Lösungen, die Verbesserungen für die Verfügbarkeit und Zuverlässigkeit erwarten lassen. Auch die Veröffentlichung des Visual Studio .Net – ersten Einschätzungen nach eine

benutzerfreundliche Entwicklungsumgebung – begünstigt einfachere Entwicklungsumgebungen.

WebService-Architektur Bestandteil	In der Anwendung genutzt
<i>WebService-Standards</i>	<i>SOAP, WSDL, UDDI</i>
<i>WebService-Framework</i>	<i>J2EE</i>
<i>WebService-Plattform</i>	<i>IBM Websphere,</i>
<i>WebService-Entwicklungsumgebung</i>	<i>IBM Websphere Studio Application Developer 4.0</i>
<i>WebService-Laufzeitumgebung</i>	<i>IBM Websphere Application Server 4.0</i>
<i>WebService-Verzeichnis (oder Broker)</i>	<i>IBM Websphere UDDI Registry</i>
<i>WebService-Management</i>	-

Tabelle 4-1: WebService-Architektur des Stockquote Beispiels

5 Konsequenzen für Unternehmen

Insgesamt ist die Entwicklung von WebServices in einem frühen Stadium. Noch beschränken psychologische Barrieren (Vertrauen) einerseits und technische Limitierungen (z.B. Netzgeschwindigkeit) andererseits, vor allem aber der Mangel an Standardisierung die schnellere Verbreitung von WebServices. Unternehmen, die Lösungen auf WebService-Standards aufbauen wollen, begegnen noch verschiedenen Herausforderungen:

- *Mangelnde Transaktionssicherheit bei komplexen Diensten.* Bei gebündelten Diensten, die mehrere WebServices aufrufen, existieren gegenwärtig keine akzeptierten Standards zur Kontrolle der Vollständigkeit aller durchgeführten Aktionen oder für Roll-back¹ Mechanismen.
- *Lücken bezüglich Authentifizierung und Verschlüsselung.* Standards für eine Sicherheitsinfrastruktur existieren erst in Ansätzen, z.B. Ansätze aus einer Kooperation von IBM, Microsoft und Verisign, der Liberty Alliance oder von Sicherheitsanbietern wie Netegrity. Diese Ansätze entstehen isoliert voneinander und sind teils inkompatibel. Einzig für den Bereich Authentifizierung hat das W3C in Zusammenarbeit mit der Internet Enginee-

¹ Dabei werden im Fehlerfall die Transaktionen rückgängig gemacht und der Datenbestand auf den Ursprungszustand zurückgesetzt.

ring Task Force (IETF) im Februar 2002 eine „XML Signature“ verabschiedet [W3C 2002b].

- *Fehlende geschäftliche Prozess- und Daten-Standards.* Prozessabläufe wie beispielsweise eine Verfügbarkeitsprüfung werden von verschiedenen Softwareanbietern und -Anwendern unterschiedlich definiert. Auch fehlen in den meisten Branchen semantische Standards zur Beschreibung der ausgetauschten Leistungen. Dies erschwert die Integration entsprechender Services in bestehende Applikationen und Prozesse. Initiativen wie ebXML oder RosettaNet versuchen hier ein gemeinsames Verständnis zu schaffen.
- *Geringes externes Angebot an WebServices.* Für WebServices sind erst seit kurzem Entwicklungstools zur Implementierung von Diensten auf den entsprechenden Standards erhältlich. Es finden sich daher noch kaum WebService-Anbieter, die ihre Leistungen in WSDL-Notation in UDDI-Verzeichnissen listen. Die gegenwärtigen Inhalte beschränken sich primär auf Kontaktadressen und Produktbeschreibungen.
- *Mangelndes Vertrauen.* Wegen fehlenden standardisierten Handelsvereinbarungen ist das dynamische, automatisierte Identifizieren und Nutzen von WebServices gegenwärtig für die meisten Unternehmen derzeit nicht relevant.
- *Mangelnde Zuverlässigkeit.* Unternehmen haben derzeit Bedenken, eine hohe Verfügbarkeit von WebServices zu gewährleisten oder für nicht verfügbare WebServices verantwortlich gemacht zu werden und nutzen WebService-Technologien innerhalb der eigenen Firewall.

Der Einsatz von WebServices konzentriert sich vorerst auf den unternehmensinternen Bereich oder auf existierende Partnerschaften zwischen Unternehmen. Analysten (vgl. [Mac Neela 2002], [Hailstone 2002]) rechnen jedoch mit einem wachsenden überbetrieblichen Einsatz innerhalb der nächsten zwei bis drei Jahre. WebServices könnten in Zukunft grosse Auswirkungen auf die Entstehung und Zusammensetzung der Wertschöpfungsketten und die zwischenbetriebliche Zusammenarbeit von Unternehmen haben – vorausgesetzt die beteiligten Akteure können sich zumindest branchenweit auf eine Grundgesamtheit an notwendigen geschäftlichen Standards einigen. Auch im innerbetrieblichen Bereich können WebServices Nutzen bei der Wiederverwendung und Integration von Anwendungskomponenten bringen. Softwarehersteller wie SAP, Oracle oder Microsoft zerlegen ihre monolithischen Systeme in Module, die über WebService-Technologien verbunden werden [vgl. Whiting 2002]. Für

Unternehmen macht es Sinn, bereits heute darüber nachzudenken, welche Funktionalitäten in verschiedenen Systemen mehrmals vorkommen und wie sich diese durch zentrale Dienste ersetzen lassen. Der Aufwand für ein entsprechendes Redesign der Systeme ist beträchtlich.

Um aus technischer Sicht Erfahrungen mit WebServices zu sammeln und die Koordination vieler WebServices zu erlernen, bieten sich zuerst interne Anwendungen an. Externe Partner können in einer späteren Phase einbezogen werden. Mögliche Schritte umfassen:

1. Anbinden isolierter WebServices an bestehende eBusiness-Lösungen. WebServices automatisieren einzelne, hochstandardisierte manuelle Tätigkeiten wie etwa die Adressprüfung oder die Erstellung von Transportpapieren. Mit diesem Schritt sammeln Unternehmen erste Erfahrungen mit WebService Standards und –Plattformen.
2. Verbreitung der erprobten WebServices in weiteren Unternehmensbereichen und Aufbau eines internen privaten WebService-Repository. Herausforderungen: Aufbau des zentralen UDDI-Verzeichnisses, Prozesse und Strukturen für die interne Definition, Entwicklung, Veröffentlichung und Nutzung standardisierter Anwendungskomponenten.
3. Mehrere WebServices zur Unterstützung von (Teil-) Prozessen aggregieren und zu neuen Anwendungen zusammenführen. Herausforderungen: z.B. Transaktionskoordination, Integration sich evtl. funktional überlappender WebServices oder WebServices unterschiedlicher Anbieter.
4. Ausgewählten Partnern des Geschäftsnetzwerkes über das private Repository Zugriff auf vereinzelte interne Services geben. Herausforderungen: z.B. Definition einer gemeinsamen unternehmensübergreifenden Terminologie und Sicherheitsarchitektur.
5. Komplexere eBusiness Legacy-Anwendungen (z.B. EDI/EDIFACT-Verbindungen mit Zulieferern oder Kunden) auf WebService-Technologie migrieren. Herausforderungen: z.B. Gewährleistungsaspekte, Koordination von Transaktionen über Anwendungen rechtlich autonomer Einheiten hinweg.

Zusammenfassend sind WebServices aus betriebswirtschaftlicher und technologischer Sicht noch wenig verbunden. Mangels standardisierter Lösungen auf den Ebenen Strategie und Prozess greifen WebService-Anbieter auf proprietäre Lösungen zurück. Aus technologischer Sicht beschränken sich WebServices auf die Transport- und Nachrichtenebene. Sie unterstützen mit den verwendeten Internet-Standards die Systemintegration, ohne dabei aber die

Komplexität zwischenbetrieblicher Integration zu lösen. Die wichtigsten Regelungen für eine deutliche Erhöhung der Netzwerkfähigkeit stehen hier noch aus.

Literatur

[Ariba 2001]

Ariba, I., Microsoft, Web Services Description Language (WSDL) 1.1, 2001

[Beimborn et al. 2002]

Beimborn, D., Mintert, S., Weitzel, T., Web Services und ebXML, in: Wirtschaftsinformatik, 44 (2002) 3, S. 277-280

[Bellwood et al. 2002]

Bellwood, T., Clément, L., Ehnebuske, D., Hately, A., Hondo, M., Husband, Y.L., Januszewski, K., Lee, S., McKee, B., Munter, J., von Riegen, C., UDDI Version 3.0, UDDI.org, 2002

[Bettag 2001]

Bettag, U., Web-Services, in: Informatik Spektrum, (2001) 24. Oktober 2001, S. 302-304

[Clark 2002]

Clark, M., Selling WebServices, Tect, Chicago, 2002

[Computerwoche 2002]

Computerwoche, Google: Internetsuche als Web-Service, Computerwoche, www.computerwoche.de, 15.04.2002

[Fitzgerald 2001]

Fitzgerald, M., Building B2B Applications with XML - A Resource Guide, John Wiley & Sons, Inc., New York, 2001

[Fou 2002]

Fou, J., Web Services and Logistics - Web Services and Your Car, Web Services Architect, <http://www.webservicesarchitect.com/content/articles/fou04print.asp>, 2002-04-12

[Gisolfi 2001]

Gisolfi, D., Web Services Architect: Part 1 - An Introduction to Dynamic e-business, IBM Corp., <http://www-106.ibm.com/developerworks/webservices/library/ws-arc1/>, 05.12.2001

[Gizanis et al. 2002]

Gizanis, D., Reichmayr, C., Alt, R., Logistik Web Services zur Unterstützung des Distributed Order Management, in: Alt, R., Fleisch, E., Österle, H., Real-time Business, Springer, Berlin et al., 2002

[Hagel 2001]

Hagel, J.I., Brown, John Seely, Your Next IT Strategy, in: Harvard Business Review, 79 (2001) 9, S. 105-113

[Hailstone 2002]

Hailstone, R.e.a., Web Services Adoption Timeline and Related Business Opportunities, IDC, 2002

[Hanson 2002]

Hanson, J.J..Net and J2EE Web Services: A Comparison, Tect, 2002

[IBM 2001]

IBM, Web Services PMT, IBM alphaWorks,
<http://www.alphaworks.ibm.com/aw.nsf/FAQs/wspmt>, 02.10.2002

[Intel 2002]

Intel, Web Services Readiness, Intel Corporation, 2002

[Loviscach et al. 2002]

Loviscach, J., Schulz, H., Violka, K., Sunspiration, in: c't, (2002) 4, S. 92-97

[Mac Neela 2002]

Mac Neela, A., Zwischen Hype und wahrem Nutzen: Was steckt hinter Web-Services?, SAP Info, www.sapinfo.de, 2002-03-18

[Microsoft 2001]

Microsoft, Web Services, msdn,
<http://www.microsoft.com/india/msdn/dotnet/webservices.asp>, 02.10.2002

[Mikalsen 2001]

Mikalsen, T., Rouvellou, I., Tai, S., Reliability of Composed Web Services - From Object Transactions to Web Transactions, IBM T.J. Watson Research Center, New York, 2001

[Newcomer 2002]

Newcomer, E., Understanding Web Services, Addison-Wesley, Boston, 2002

[Österle et al. 1996]

Österle, H., Riehm, R., Vogler, P., Middleware - Grundlagen, Produkte, Praxisbeispiele für die Integration heterogener Welten, Vieweg, Braunschweig/Wiesbaden, 1996

[Patil/Saigal 2002]

Patil, S., Saigal, S., When Computers Learn to Talk: A Web Services Primer, McKinseyQuarterly, www.mckinseyquarterly.com, 2002-04-10

[Patton 2002]

Patton, S., Web Services in the Real World, CIO Magazine,
http://www.cio.com/archive/040102/real_content.html, 2002-04-09

[Picot 2002]

Picot, A., Einführung, in: Picot, A., Breidler, J., Web Services - Bausteine für das e-Business, Hüthig, Heidelberg, 2002, S. 1-8

[Plummer 2002]

Plummer, D., Key Entry Points into Web Services Markets, Gartner Group, 2002

[Samtani 2002a]

Samtani, G., Sadhwani, D., Integration Brokers and Web Services - Will Web Services Support be Just Another Feature?,
<http://www.webservicesarchitect.com/content/articles/samtani03.asp>, 30.01.2002

[Samtani 2002b]

Samtani, G., Sadhwani, D., Web Services and Application Frameworks,
<http://www.webservicesarchitect.com/content/articles/samtani04.asp>, 27.03.2002

[Sleeper/Robins 2001]

Sleeper, B., Robins, B., Defining Web Services, The Stencil Group, 2001

[Sleeper/Robins 2002]

Sleeper, B., Robins, B., The Laws of Evolution: a Pragmatic Analysis of the Emerging Web Services Market, The Stencil Group, 2002

[UDDI.org 2001]

UDDI.org, UDDI Executive White Paper, 2001

[Virtel 2002]

Virtel, M., Microsoft Monopol nicht auf Internet ausdehnbar, Financial Times Deutschland, <http://www.ftd.de/tm/it/1014398985904.html>, 09.04.2002

[W3C 2001a]

W3C, SOAP Version 1.2 - W3C Working Draft 9 July 2001, W3C,
<http://www.w3.org/TR/2001/WD-soap12-20010709/>, 20.9.2001

[W3C 2001b]

W3C, Web Services Description Language (WSDL) 1.1 - W3C Note March 2001, World Wide Web Consortium, <http://www.w3c.org/TR/wsdl>, 11.05.2002

[W3C 2002a]

W3C, Web Services Activity, World Wide Web Consortium,
<http://www.w3.org/2002/ws/>, 02.10.2002

[W3C 2002b]

W3C, World Wide Web Consortium Issues XML Signature as a W3C Recommendation, <http://www.w3c.org/2002/02/xmlsignature-pressrelease.html.en>, 14.02.2002

[Whiting 2002]

Whiting, R., Web Services Take Integration To A New Level, Informationweek.com, <http://www.informationweek.com/story/IWK20020411S0009>, 15.04.2002

[WS-I 2002]

WS-I, WS-I introduction presentation, Web Services Interoperability Organization, <http://www.ws-i.org/Documents.aspx>, 02.10.2002

[WSUI 2002]

WSUI, About WSUI, Web Services User Interface Organization,
<http://www.wsui.org/about.html#what>, 02.10.2002

[Yates/Kafka 2001]

Yates, S., Kafka, S.J., The Web Services Payoff, Forrester Research, Cambridge,
2001

Anhang A: Definitionen von WebServices

[Beimborn et al. 2002, 277]	Dienste, die auf dem losen Zusammenspiel von Standards wie XML, SOAP, WSDL und UDDI beruhen und somit als Grundlage einer unternehmens- und anwendungsübergreifenden automatisierten System-zu-System-Kommunikation dienen.
[Bettag 2001, 302]	XML-basierte selbstbeschreibende Darstellung einer Anwendung oder Softwarekomponente, deren Schnittstelle und ihre Metadaten getrennt sind.
[Clark 2002, 3]	A new phenomenon of computer to computer interaction using XML, and standard internet protocols such as HTTP.
[Hailstone 2002, 7]	Internet technology-based “machine-to-machine services” that are based on the Web Services Architecture (WSA). The WSA is a standardized approach to dynamic component connectivity and interoperability that relies on self-describing components that interoperate at run time and open connectivity standards including Internet protocol (IP), simple object access protocol (SOAP), and Web services description language (WSDL). Other standards, such as extensible markup language (XML), are evolving to meet operational and business process semantic requirements and, as they mature, will become part of the WSA.
[IBM 2001]	Web services are self-describing, self-contained, modular applications running on the Web for the benefit of enterprise operators who want to create innovative processes and value chains.

[Intel 2002, 1]	Web services encompass a vision of a fully integrated computing network that include PCs, servers, handheld devices, programs, applications and network equipment, all working together. This network can perform distributed computation with the best-matched device for the task and deliver the information on a timely basis in the form needed by the user.
[Microsoft 2001]	A unit of application logic providing data and services to other applications. Applications access Web Services via ubiquitous Web protocols and data formats such as HTTP, XML, and SOAP, with no need to worry about how each Web Service is implemented.
[Picot 2002, 4]	Softwarebausteine, die eine offene Kommunikationsschnittstelle zur Verfügung stellen, um weltweit verteilte Anwendungen und Dienste, plattform-, programm- und standortunabhängig abrufen, nutzen und zusammenfügen zu können.
[Sleeper/Robins 2002, 7]	A stack of emerging standards that define protocols and create a loosely coupled framework for programmatic communication among disparate systems.
[W3C 2002a]	The programmatic interfaces made available for application to application communication using the World Wide Web.
[WS-I 2002]	Standards-based integration and interoperability across platforms, applications, and programming languages.
[WSUI 2002]	An end-user facing Web application that is exposed via standard XML technologies.
[Yates/Kafka 2001, 3]	Software designed to be used by other software via Internet protocols and formats.

Anhang B: Ziele und Ebenen der Systemintegration

Für die Integration von Systemen bestehen unterschiedliche Beweggründe:

- Kunden oder Mitarbeiter sollen auf Informationen und Transaktionen entsprechend ihren Bedürfnissen in einem Portal zugreifen können.
- Daten aus den operativen Systemen sollen in analytischen Systemen (OLAP) zeitnah ausgewertet werden und so Entscheidungen unterstützen. Kooperationsprozesse zu Partnern sollen aufgebaut werden und Systeme beispielsweise zur Produktionsplanung oder Produktentwicklung integriert werden.
- Die Systemlandschaft soll vereinfacht werden und einzelne Systeme in Integrationsbereichen zusammengeführt werden.
- Neue Geschäftsprozesse sollen schnell durch bestehende Systeme möglichst vollständig automatisch unterstützt werden.

Für die Erfüllung der unterschiedlichen Anforderungen bestehen unterschiedliche Ebenen zur Integration von Geschäftsanwendungen [Österle et al. 1996, 29]:

- *Integration auf Präsentationsebene.* Eine Integration auf Präsentationsebene hat zum Ziel, Benutzeroberflächen verschiedener Anwendungen auf einer gemeinsamen Oberfläche zusammenzufassen. Bei einer reinen Integration auf Präsentationsebene erfolgt keine Integration von tieferliegenden Ebenen. Beispielsweise erfolgt bei einer gemeinsamen Darstellung von Kundendaten und Bahnfahrplanabfrage keine Verknüpfung der Ortsangaben. Für eine Abfrage der nächsten Zugverbindung zum Kunden muss der Mitarbeiter selbst die Ortsangaben übernehmen.
- *Integration auf Applikationsebene.* Eine Integration auf Applikationsebene hat zum Ziel Funktionen von Applikationen anderen Applikationen plattformübergreifend etwa über spezielle Programmierschnittstellen (APIs) oder Remote Procedure Calls (RPCs) zugänglich zu machen. Auf dieser Ebene findet eine Prozessintegration statt, die das Steuern von komplexen Transaktionen, an denen mehrere unterschiedliche Applikationen beteiligt sind, erlaubt. Middlwaredienste, wie etwa Message Broker, standardisieren und überwachen den Informationsfluss und müssen die Transaktionssicherheit sicherstellen: Transaktionen müssen vollständig ausgeführt – das heisst über alle beteiligten Applikationen

hinweg – oder vollständig rückgängig gemacht werden. Beispielsweise könnte automatisch beim Aufrufen der Kundendaten die nächste Zugverbindung in der Bahnfahrplanabfrage ausgelesen werden.

- *Datenintegration.* Die Integration auf Datenebene hat zum Ziel, heterogene und verteilte Informationen applikationsübergreifend konsistent zur Verfügung zu stellen. Beispielsweise können verschiedene Standorte eines Unternehmens eine gemeinsame Kundenadressdatenbank pflegen, so dass leicht ein Massenversand durchgeführt werden kann.