

# **Service-orientierte Anwendungsintegration im Intra- und Internet**

**Inauguraldissertation zur Erlangung  
des akademischen Grades eines Doktors der  
Wirtschaftswissenschaften an der  
Wirtschaftswissenschaftlichen Fakultät  
der Universität Passau**

**Michael Götzfried**

**Passau**

**2006**

Gutachter:

Prof. Dr. Peter Kleinschmidt, Universität Passau

Prof. Dr. Franz Lehner, Universität Passau

Tag der letzten Fachprüfung des Rigorosums:

09. November 2006

# Inhaltsverzeichnis

<b>Abkürzungsverzeichnis.....</b>	<b>V</b>
<b>Abbildungsverzeichnis.....</b>	<b>IX</b>
<b>Tabellenverzeichnis.....</b>	<b>XI</b>
<b>I. Motivation und Problemstellung .....</b>	<b>1</b>
<i>I.1 Begriffliche Abgrenzung .....</i>	<i>3</i>
<i>I.2 Aufbau der Untersuchung.....</i>	<i>7</i>
<i>I.3 Motivierendes Szenario aus der Personalwirtschaft.....</i>	<i>11</i>
<i>I.4 Ziele und Einordnung der Arbeit.....</i>	<i>13</i>
<b>II. Die service-orientierte Architektur .....</b>	<b>15</b>
<i>II.1 (Software-) Architektur.....</i>	<i>15</i>
<i>II.2 Nutzenpotenziale aus der Beschäftigung mit Softwarearchitekturen... </i>	<i>16</i>
<i>II.3 Architektonische Grundlagen.....</i>	<i>16</i>
II.3.1 Verteilte Systeme .....	17
II.3.2 Das Composite Computing Model (CCM) .....	17
<i>II.4 Charakterisierung der service-orientierten Architektur.....</i>	<i>18</i>
II.4.1 Akteure und Rollen.....	22
II.4.2 Interaktion und Stufen der Interaktion .....	26
II.4.3 Verfeinerung der Architektur .....	28
<i>II.5 Konsequenzen für die Implementierung.....</i>	<i>30</i>
<i>II.6 Praktische Umsetzung der SOA.....</i>	<i>31</i>
II.6.1 Fördernde und hemmende Faktoren für den Einsatz der SOA .....	32
II.6.2 Lösungsansätze im Bereich der Standardsoftware .....	33
II.6.3 Service-Design .....	36
<b>III. Die Web Service-Technologie.....</b>	<b>41</b>
<i>III.1 Definition .....</i>	<i>41</i>
<i>III.2 Merkmale von Web Services.....</i>	<i>42</i>
<i>III.3 Abgrenzung zu anderen Technologien.....</i>	<i>44</i>
III.3.1 Common Object Request Broker Architecture (CORBA) .....	44

III.3.2	Java Remote Method Invocation (RMI) .....	46
III.3.3	Sun Remote Procedure Call (Sun RPC) .....	47
<i>III.4</i>	<i>Ablauf eines Web Service-Aufrufs</i> .....	<i>47</i>
<i>III.5</i>	<i>Der Protocol Stack</i> .....	<i>49</i>
III.5.1	Übertragung .....	50
III.5.2	Beschreibung .....	52
III.5.3	Verzeichnisdienst .....	53
III.5.4	Sicherheit .....	55
III.5.5	Prozesse und Transaktionen.....	57
III.5.6	Semantik und Ontologien .....	58
III.5.7	Unterstützung von Zuständen bei Web Services .....	59
III.5.8	Herausforderungen aus der Vielzahl an Protokollen .....	61
<i>III.6</i>	<i>Verwandte Technologien</i> .....	<i>64</i>
III.6.1	Grid-Computing.....	65
III.6.2	Peer-to-Peer Computing .....	67
III.6.3	Konvergenz.....	68
<b>IV.</b>	<b>Typen von IT-Sourcing-Szenarien</b> .....	<b>75</b>
<i>IV.1</i>	<i>Auswahl der zu betrachtenden Szenariotypen</i> .....	<i>76</i>
<i>IV.2</i>	<i>Innerbetriebliche Service-Ausrichtung</i> .....	<i>77</i>
IV.2.1	Erweiterung vorhandener Anwendungen .....	78
IV.2.2	Integration von Neuanwendungen .....	80
IV.2.3	Einbindung von Kunden und externen Geschäftspartnern .....	81
<i>IV.3</i>	<i>Shared Services Center (SSC)</i> .....	<i>82</i>
<i>IV.4</i>	<i>Out-/ In- und Backsourcing</i> .....	<i>85</i>
IV.4.1	Der Begriff des Outsourcing .....	86
IV.4.2	Charakterisierung und Abgrenzung des Outsourcing.....	87
IV.4.3	Outsourcingformen.....	88
IV.4.4	Bedeutung und Gestaltung von Schnittstellen bei Outsourcingprojekten .....	91
IV.4.5	In- bzw. Backsourcing .....	93
IV.4.6	Problemstellung bei der Reintegration von ausgelagerten Anwendungen und der Beendigung von Outsourcingprojekten .....	94
IV.4.7	Die Rolle von Web Services bei Insourcing-Projekten .....	95

---

<i>IV.5 Provisioning</i> .....	96
IV.5.1 Definitionen .....	97
IV.5.2 Diensteeinteilung.....	98
IV.5.3 Provisioningszenariotypen .....	100
IV.5.4 Abrechnungsvarianten .....	102
<b>V. Prototypische Umgebungen .....</b>	<b>108</b>
<i>V.1 Enterprise Resource Planning</i> .....	108
V.1.1 Rahmenbedingungen.....	109
V.1.2 Datenaustausch mit einem Integrationsserver .....	110
V.1.3 Auswertung .....	113
<i>V.2 Web Service-Umgebung</i> .....	114
V.2.1 Technische Architektur und Technologie .....	114
V.2.2 Programmiersprachenunabhängigkeit .....	118
V.2.3 Integration von Web Services in ein Portal .....	121
V.2.4 Zusammenfassung der Ergebnisse.....	123
<i>V.3 Grid-Umgebung</i> .....	124
V.3.1 Praxisrelevante Anforderungen.....	125
V.3.2 Technologie.....	126
V.3.3 Anwendung des Web Services Resource Frameworks .....	129
V.3.4 Analyse der Ergebnisse .....	131
<i>V.4 Integrierte Umgebung</i> .....	131
V.4.1 Auswahl geeigneter Systemumgebungen.....	132
V.4.2 Technische Architektur.....	134
V.4.3 Metadaten in der integrierten Provisioning-Umgebung .....	137
V.4.4 Migrationsszenarien .....	139
V.4.5 Ergebnisauswertung .....	141
<i>V.5 Die prototypischen Umgebungen im Kontext des IT-Sourcing</i> .....	142
<b>VI. Schlussbetrachtung .....</b>	<b>145</b>
<i>VI.1 Resümee</i> .....	145
<i>VI.2 Ausblick</i> .....	150
<b>Literaturverzeichnis.....</b>	<b>XII</b>

---

## Abkürzungsverzeichnis

ACID	Atomicity, Consistency, Isolation, Durability
ASAP	Asynchronous Service Access Protocol
ALE	Application Link Enabling
B	Broker
BAPI	Business Application Programming Interface
BC	Business Connector
BITKOM	Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V.
BPEL	Business Process Execution Language
BPO	Business Process Outsourcing
bzw.	beziehungsweise
CCM	Composite Computing Model
CIO	Chief Information Officer
CORBA	Common Request Broker Architecture
DTD	Document Type Definition
DSAG	Deutschsprachige SAP Anwendergruppe
DV	Datenverarbeitung
EAI	Enterprise Application Integration
ebXML	electronic business XML
EDI	Electronic Data Interchange
EGA	Enterprise Grid Alliance
ERP	Enterprise Resource Planning
ESA	Enterprise Service Architecture
etc.	et cetera
f.	folgende

---

FTP	File Transfer Protocol
GB	Gigabyte
GIOP	General Inter-ORB Protocol
GT4	Globus Toolkit Version 4
GRAM	Grid Resource Allocation and Management
HTTP	Hypertext Transfer Protocol
Hrsg.	Herausgeber
hrsg. v.	herausgegeben von
IDL	Interface Description Language
IDoc	Intermediate Document
IEFT	Internet Engineering Task Force
i. e. S.	im engeren Sinne
IIOB	Internet Inter-ORB Protocol
IP	Internet Protocol
IT	Informationstechnologie
i. w. S.	im weiteren Sinne
JDBC	Java DataBase Connectivity
JINI	Java Intelligent Network Infrastructure
JMS	Java Message Service
JNDI	Java Naming and Directory Interface
LDAP	Lightweight Directory Access Protocol
MOWS	Management of Web Services
MUWS	Management Using Web Services
NIC	Network Interface Card
o. A.	ohne Autor

---

OASIS	Organization for the Advancement of Structured Information Standards
o. J.	ohne Jahr
OMG	Object Management Group
openLDAP	open Lightweight Directory Access Protocol
openSSL	open Secure Socket Layer
ORB	Object Request Broker
P	Provider
P2P	Peer-to-Peer
PHP	PHP Hypertext Processor
R	Requestor
RFC	Remote Function Call
RMI	Remote Methode Invocation
RMI-IIOP	Remote Methode Invocation-Internet Inter-Object Request Broker Protocol
RPC	Remote Procedure Call
S.	Seite
SaaS	Software as a Service
SCM	Supply Chain Management
SQL	Structured Query Language
SSC	Shared Services Center
SSL	Secure Sockets Layer
SMTP	Simple Mail Transfer Protocol
SOA	service-orientierte Architektur/ service-oriented architecture
SOAP	Simple Object Access Protocol
TCP/IP	Transmission Control Protocol/ Internet Protocol
tRFC	transaktionaler Remote Function Call
u. a.	unter anderem
UBR	Universal Business Registry



---

UDDI	Universal Description, Discovery and Integration
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
USA	United States of America
US-\$	US-Dollar
US	United States
vgl.	vergleiche
vNIC	virtuelle Network Interface Card
W3C	World Wide Web Consortium
WAN	Wide-Area-Network
WebMDS	Web Frontend für den Meta Data Service
WS-*	Web Service-Standards der 2. Generation
WS-I	Web Service Interoperability Organization
WSDL	Web Services Description Language
WSDM	Web Services Distributed Management
WSRF	Web Services Resource Framework
WSRP	Web Services for Remote Portlets
XDR	External Data Representation
XSD	XML Schema Definition
XML	Extensible Markup Language
z. B.	zum Beispiel

## Abbildungsverzeichnis

Abbildung 1: Emerging Technologies Hype Cycle 2004 [Quelle: Gartner zitiert nach HERRMANN (2004), S. 5] .....	2
Abbildung 2: Maschinell unterstützte Kommunikation.....	6
Abbildung 3: Betrachtungsebenen der Untersuchung.....	8
Abbildung 4: Personalabrechnungssystem in einer "gewachsenen" Systemlandschaft .....	12
Abbildung 5: Rollen und Beziehungen innerhalb der SOA [Quelle: BURBECK (2000)] .....	21
Abbildung 6: Ebenen der Dienstenutzung.....	27
Abbildung 7: Erweiterung der SOA [Quelle: MYERSON (2004)].....	29
Abbildung 8: Schematische Darstellung der Enterprise Service Architecture [Quelle: WOODS (2004), S. 34 und 46 f.].....	34
Abbildung 9: Ablauf eines Web Service-Aufrufs.....	48
Abbildung 10: Der Web Service Protocol Stack [Quelle: BOOTH u. a. (2004)] .....	49
Abbildung 11: Aufbau einer SOAP-Nachricht [Quelle: MITRA (2003)].....	51
Abbildung 12: Aufbau eines WSDL-Dokumentes [Quelle: ERL (2005), S. 134].....	52
Abbildung 13: Interaktion von UDDI Registries der Version 3 [Quelle: OASIS (2004), S. 10].....	54
Abbildung 14: Adaptionprozess der Standards des Web Service Protocol Stack [Quelle: WILKES (2005)].....	63
Abbildung 15: Konvergenz zwischen Technologien.....	69
Abbildung 16: Konvergenz in den Entwicklungsrichtungen bei Web Services und Grid Computing [Quelle: GLOBUS ALLIANCE (o.J.)] .....	70
Abbildung 17: Einbindung von Partnern im Unternehmensumfeld im Rahmen unternehmensübergreifender Prozesse [Quelle: WATT (2002), S. 144].....	82
Abbildung 18: Shared Services Center .....	84

---

Abbildung 19: In- bzw. Outsourcing .....	86
Abbildung 20: Traditionelles ASP-Szenario .....	100
Abbildung 21: Next Generation Service Provider.....	102
Abbildung 22: Schichtenaufteilung bei Application Link Enabling [eigene Darstellung nach LINTHICUM (2001), S. 358 f.].....	111
Abbildung 23: Architektur und Einsatzbereiche des SAP Business Connectors [Quellen: SAP 2005a und SAP 2005b].....	112
Abbildung 24: Schematischer Aufbau des AXIS Servers auf einem Applikationsserver [Quelle: DEITEL u. a. (2003), S. 111].....	115
Abbildung 25: Konfiguration der Web Service-Umgebung aus Apache AXIS und jUDDI auf einem Apache Tomcat Server mit Anbindung an eine PostgreSQL-Datenbank.....	116
Abbildung 26: Verwaltungswerkzeug für Webapplikationen auf dem Apache Tomcat Server .....	117
Abbildung 27: Quellcode-Beispiel für einen mit NuSOAP implementierten PHP Web Service .....	119
Abbildung 28: Quellcode-Beispiel für einen JAVA-Web Service.....	120
Abbildung 29: Integration eines PHP Web Service Clients in das Apache Cocoon-Portal.....	121
Abbildung 30: Quellcode-Beispiel zu einem Web Service-Aufruf mit PHP.....	122
Abbildung 31: Schematischer Aufbau des Globus Toolkit (Version 4) [Quellen: FOSTER (2005), S. 4 und 6].....	127
Abbildung 32: Nutzungsmöglichkeiten eines Containers im Globus Toolkit [Quelle: FOSTER (2005), S. 10] .....	129
Abbildung 33: Nutzung des WSRF für stateful resources [Quelle: SOTOMAYOR (2005)].....	130
Abbildung 34: Schematischer Aufbau eines XEN-basierten Systems zur Virtualisierung von Betriebssystemressourcen [eigene Darstellung nach XEN (2006), S. 3].....	133
Abbildung 35: Integrierte Web Service-/ Grid-Umgebung .....	135
Abbildung 36: Metainformationen zu den Diensten innerhalb einer Hosting-Umgebung des Globus Toolkit .....	138
Abbildung 37: Migration eines XEN-Servers über eine gemeinsame SAMBA-Freigabe .....	140

---

## Tabellenverzeichnis

Tabelle 1: Phasen des Enterprise Community Process [vgl. HUBER (2006), S. 65] .....	36
Tabelle 2: Richtlinien für das Service-Design [vgl. ERL (2005), S. 416 - 421] .....	40
Tabelle 3: Ausgewählte Spezifikationen zur Gewährleistung der Sicherheit innerhalb einer SOA [vgl. ERL (2005), S. 257] .....	56
Tabelle 4: Spezifikationen des Web Services Resource Frameworks [vgl. CZAJKOWSKI u.a. (2004), S. 6 f.] .....	61
Tabelle 5: Bedeutende Standardisierungsgremien für Web Service-Standards [vgl. HAUSER/ LÖWER (2004) S. 20-23] .....	62
Tabelle 6: Vergleich von Grid und Peer-to-Peer Systemen [vgl. FOSTER/ IAMNITCHI (2003), S. 2-4].....	72
Tabelle 7: Einteilung der IT-Sourcing-Formen [vgl. JOUANNE-DIEDRICH (2005)] .....	76
Tabelle 8: Gliederung des anwendungsnahen Outsourcing [vgl. DITTRICH/ BRAUN (2004), S. 4] .....	90

## I. Motivation und Problemstellung

Die Integration von Anwendungen unterschiedlicher Art stellt seit geraumer Zeit eine der großen Herausforderungen in der Unternehmens-IT dar. Die Integrationsanforderungen ergeben sich beispielsweise aus der Einbindung von externen Geschäftspartnern in die Prozesse im Unternehmen. Dabei entstehen durch die erforderliche Flexibilität bei der Kommunikation neue und gestiegene technische Anforderungen. Integration ist jedoch nicht primär eine technologische Fragestellung, sondern wird durch betriebswirtschaftliche Notwendigkeiten bestimmt. Eine rein technische Herangehensweise greift hier zu kurz und bietet keine tragfähige Lösung für die Weiterentwicklung einer unternehmensweiten (Software-) Architektur.

An dieser Stelle setzt die service-orientierte Architektur ein und ermöglicht die Gestaltung einer offenen und flexiblen Unternehmensarchitektur. Diese Architektur kann sowohl als Rahmen für den Umbau als auch für den Aufbau der einzelnen Anwendungsarchitektur dienen.

Technisch gesehen stehen hierfür mehrere Implementierungsansätze zur Verfügung. Web Services scheinen sich dabei nach einer anfänglichen Hype-Phase zu etablieren, wenn es auch noch großer Anstrengungen im Rahmen des Standardisierungsprozesses zur Steigerung der Interoperabilität und der Erfüllung aller an die Technologie gestellten Anforderungen bedarf.

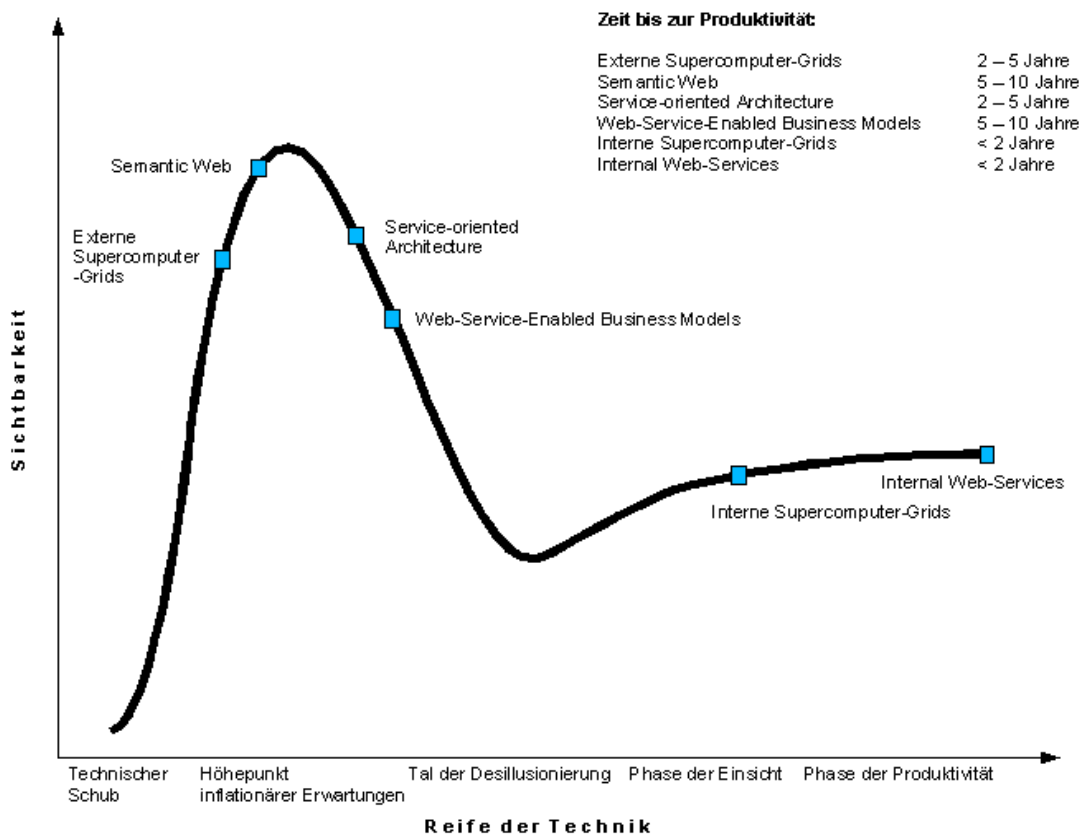


Abbildung 1: Emerging Technologies Hype Cycle 2004 [Quelle: Gartner zitiert nach HERRMANN (2004), S. 5]

Laut dem von Gartner herausgegebenen Emerging Technologies Hype Cycle 2004 sind die internen Web Services in Bezug auf den Reifegrad der Technik in die Phase der Produktivität einzuordnen, wenngleich die Sichtbarkeit der Technologie in den Medien deutlich abgenommen hat.<sup>1</sup>

Verlässt man die Unternehmensgrenzen, so scheint das Web Service-Konzept einige der Ideen des Application Service Providing mit Leben füllen zu können. „Software aus der Steckdose“ verspricht die Anbindung von GRIDs über eine Web Service Schnittstelle. Nach Gartner (2004) wird dieses Netz oberhalb der Internetinfrastruktur in einem Zeithorizont von 2-5 Jahren ebenso im produktiven Einsatz sein wie service-orientierte Architekturen.<sup>2</sup>

Um tatsächlich genutzt zu werden, wird neben einer technologischen Aus-

<sup>1</sup> Vgl. HERRMANN (2004), S. 5

<sup>2</sup> Vgl. HERRMANN (2004), S. 5

gereiftheit jedoch auch ein positives ökonomisches Umfeld benötigt. Hier bietet es sich an, bewusst den Blickwinkel der Technik etwas in den Hintergrund zu stellen, und zuerst aus dem Blickwinkel der Softwarearchitektur nach dem wirtschaftlich sinnvollen Vorgehen zu fragen.

Die Arbeit untersucht vor dem Hintergrund der bisher dargelegten Aspekte Typen von IT-Sourcing-Szenarien, die vom innerbetrieblichen Aufbau einer service-orientierten Architektur, über die Einbindung von Partnern und Serviceunternehmen im Rahmen von Outsourcing bis hin zum Provisioning von Anwendungen reichen. Dabei wird die Möglichkeit einer Entkopplung der betriebswirtschaftlichen Entscheidung von der technischen Realisierung aufgezeigt. Durch die steigende Komplexität der Szenarien und durch eine damit einhergehende abnehmende Eingriffsmöglichkeit seitens des Managements wird die Tragfähigkeit der Architektur getestet, um komplexe Einmalnutzungsszenarien im Sinne des Service Providing vor dem Hintergrund gestiegener technologischer Möglichkeiten zu beleuchten. Ziel der Arbeit ist es also nicht, die Frage des Aufbaus von Geschäftsprozessen aus Services zu erörtern, sondern die Auswirkung der SOA auf Sourcing-Prozesse zu zeigen. Das Sourcing bezieht sich dabei auf Teile eines Geschäftsprozesses genauso wie auf einzelne Services innerhalb des Prozesses.

## ***1.1 Begriffliche Abgrenzung***

Der Begriff Service wird in der Arbeit aus unterschiedlichen Blickwinkeln betrachtet. Nach der Verwendung im Rahmen der betriebswirtschaftlichen Betrachtung wird auf abstrakter Architekturebene ebenso wie auf der Technologieebene davon gesprochen. Allen Verwendungen gleich ist das grundlegende Verständnis von Service als einer elektronischen Dienstleistung. Dabei ist es im Rahmen der Arbeit nicht von großer Bedeutung, ob die Dienste entgeltlich, oder unentgeltlich erbracht werden. Obwohl beide Varianten Kosten im leistenden Unternehmen verursachen, ist die Frage der Gegenleistung individuell zu beurteilen. Im Falle der Unentgeltlichkeit gilt es, verschiedene Formen im Rahmen der marktorientierten Preissetzung zu unterscheiden:

- 1) Die Leistung ist zwar für den Dienstleister auf längere Sicht nicht kostenlos, verursacht aber kurzfristig keine zusätzlichen Kosten, da freie Kapazitäten vorhanden sind, die nicht anders genutzt werden können. Die damit verbundenen Kosten sind beispielsweise im Rahmen einer Flatrate für die Internetanbindung oder eines Dauermietverhältnisses für einen dedizierten Server schon entstanden und können bzw. sollen kurzfristig nicht abgebaut werden. Für das Angebot des Dienstes gilt jedoch die Annahme, dass der Ressourcenverbrauch nur über einen begrenzten Zeitraum stattfindet und einen geringen Umfang hat. Die Leistung muss dabei positive Auswirkungen z. B. Verbesserung des Images der Unternehmung in der Öffentlichkeit haben, sonst wäre die Nichterbringung der Leistung die wirtschaftlich sinnvolle Alternative.
- 2) Die Leistung wird im Rahmen eines Leistungsbündels erbracht. Dabei erfolgt eine Bewertung des Leistungsbündels als Ganzes auf der Seite der leistungserbringenden Unternehmung. Die unentgeltliche Teilleistung kann beispielsweise unterstützend im Rahmen einer Geschäftsanbahnung eingesetzt werden (z. B. Angebot von elektronischen Warenverzeichnissen mit dem Ziel, zukünftige Bestellungen zu generieren), oder als Zusatzleistung den Wert einer anderen Dienstleistung oder Ware steigern und so als Verkaufsargument herangezogen werden. Ein Beispiel hierfür wäre der kostenlose Web Service für Aktienkurse, der zusätzlich zu einer Chartanalyse-Software angeboten wird.
- 3) Die Leistung ist nur über einen gewissen Zeitraum kostenlos, wird jedoch später kostenpflichtig. Hier könnte man sich das Beispiel eines Software-Herstellers vorstellen, der im Rahmen der Entwicklung eines neuen Produktes, Teile des Produktes schon als Web Service kostenlos anbietet. Nach der Integration von Vorschlägen zum Produkt aus einer sich bildenden Nutzer-Community werden jedoch weitere komplementäre Dienste mit weitergehenden Funktionalitäten kostenpflichtig angeboten oder die gesamten Web Services sind ab einem bestimmten Zeitpunkt für alle Nutzer kostenpflichtig. Neben



einer Markteintrittsstrategie wird hiermit auch die Community der Nutzer für den internen Entwicklungsprozess des Software-Herstellers nutzbar.

Die in der Arbeit zu betrachtenden Dienste lassen sich nach unterschiedlichen Kriterien einteilen. Für das weitere Vorgehen werden folgende Arten verwendet:<sup>3</sup>

- „Fire and Forget“: Hierunter fallen Informationsdienste an eine größere Anzahl von Adressaten. Der Erhalt der einzelnen Nachricht ist hierbei nicht als kritisch einzustufen, da eine erneute Benachrichtigung in absehbarer Zeit erfolgt. Als Beispiel kann hierfür ein Aktientickerdienst angeführt werden.
- „Request-Response“: Sowohl Anfrage als auch Antwort sind bei dieser Serviceart wichtig. Ein Beispiel wäre eine Bestellabwicklung, wobei es sich hierbei sogar um eine transaktionsgesicherte Abwicklung handeln sollte.
- Dienste als Teil von Geschäftsprozessen: Ein Dienst wird in einen Geschäftsprozess eingebunden, der aus weiteren Diensten bestehen kann. Eine Kreditwürdigkeitsprüfung durch einen externen Dienstleister innerhalb eines Vertriebsprozesses soll hier als Beispiel dienen. Ein Prozessbestandteil kann dabei aus verschiedenen Diensten zusammengesetzt werden. Entscheidend für die erfolgreiche Integration des Services ist Versorgung mit Daten aus vorhergehenden Prozessschritten in ausreichender Qualität. Dienste mit sehr hohem Datenbedarf und damit unter Umständen auch hoher Spezifität können unter Umständen weiterhin nur sinnvoll innerhalb der Unternehmung erbracht werden.
- Rechendienste: Hierbei geht es um die Durchführung aufwendiger Berechnungen und um die Nutzung von verteilten Ressourcen.

---

<sup>3</sup> Die Aufteilung wurde in Anlehnung an die Diensteeinteilung des W3C entwickelt. [Vgl. HE u. a. (2004)]

Der Dienst steht hier nicht als solcher im Mittelpunkt, sondern gewährt nur Zugang zu Ressourcen.

Betrachtet man die vorgeschlagene Aufteilung, so fällt eine steigende Komplexität der Dienste auf. Sie drückt sich entweder in höherwertigen Eigenschaften des Dienstes aus, oder führt zu einer flexibleren Einsatzmöglichkeit.

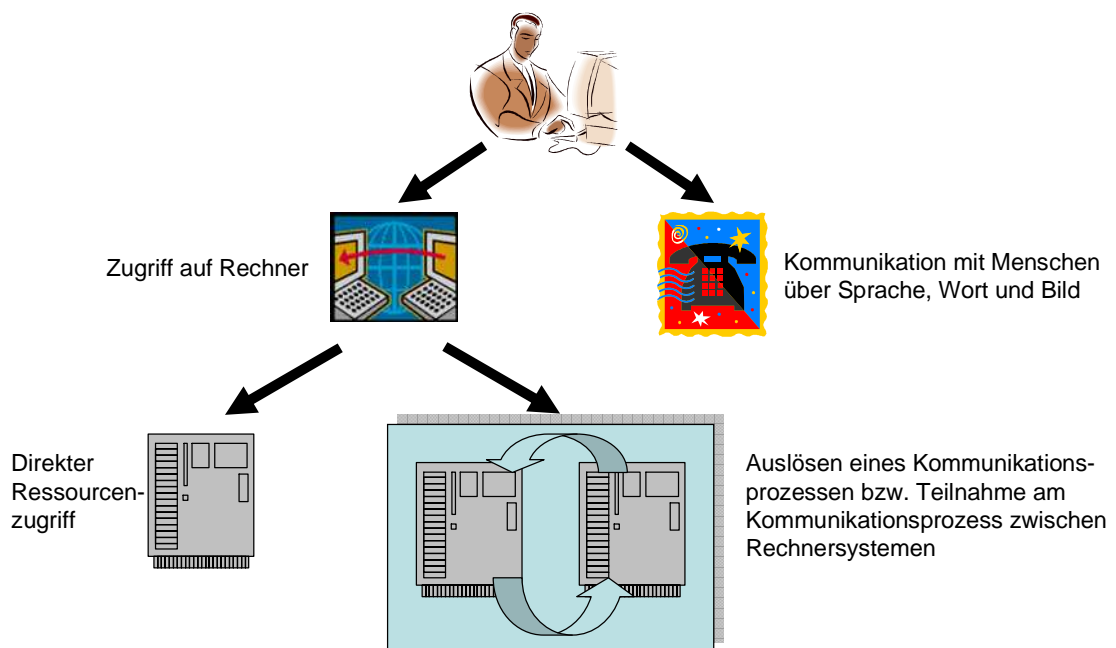


Abbildung 2: Maschinell unterstützte Kommunikation

Neben der begrifflichen Konkretisierung der unterschiedlichen Dienstarten, die außer in den betriebswirtschaftlichen auch in den informationstechnischen Betrachtungen eine Rolle spielt, muss noch der Begriff der maschinell unterstützten Kommunikation näher beleuchtet werden.

Im Rahmen der Arbeit wird die Kommunikation immer von Menschen initiiert, die an Rechnersystemen arbeiten. Dabei kann es neben der direkten maschinell unterstützten Kommunikation mit anderen Menschen unter Verwendung von Sprach-, Text- und Bildinformationen auch zu einer Kommunikation mit anderen Rechnern kommen. Nun kann entweder ein

direkter Zugriff auf die vom entfernten Rechner zur Verfügung gestellte Information (z. B. Zugriff auf einen Web Server) erfolgen, oder es kann zur Teilnahme an einer Kommunikation des Rechnersystems mit weiteren Rechnersystemen kommen. Letztere Art der Kommunikation ist Untersuchungsgegenstand der Arbeit. Ein Beispiel dafür wäre der Zugriff auf ein Enterprise Resource Planning System (ERP-System), das im Rahmen eines Beschaffungsprozesses bei einem Lieferanten eine Preisanfrage an dessen Rechnersystem macht und das Ergebnis dem Anwender an seinem Client-Arbeitsplatz mitteilt.

Bei der Betrachtung der maschinellen Kommunikation wurde von einer rein hardwareorientierten Sichtweise auf eine Client-Server Umgebung ausgegangen. Dabei wird eine Aufgabe bzw. ein Dienst einem Rechner zugeordnet. Bezieht man die softwareorientierte Sichtweise mit ein, so ist es natürlich möglich, dass auf einer physischen Hardwareplattform mehrere (Software-) Server betrieben werden. Im Zuge der Virtualisierung der Ressourcen kommen noch Szenarien hinzu, in denen auf einer physischen Hardware mehrere Rechner konsolidiert werden, oder eine Partitionierung der Hardwareressourcen stattfindet. Diese Erweiterungen sind für die weitere Arbeit von hoher Bedeutung und werden später weiter thematisiert.

## ***1.2 Aufbau der Untersuchung***

Die zu untersuchende Fragestellung liegt im Spannungsfeld zwischen der Strategie der Unternehmung und dem IT-System, das zur Unterstützung der Geschäftstätigkeit eingesetzt wird.

Zur Reduktion der Komplexität wird ein Ebenenmodell aus Geschäftsprozess-, Softwarearchitektur- und Technologieebene eingeführt, das zwischen Strategie und IT-System eingefügt wird. So wird eine Verbindung von betriebswirtschaftlich motivierten strategischen Entscheidungen bis zu den Umsetzungen auf der technischen Ebene zumindest erleichtert. Es werden nacheinander für drei verschiedene Typen von Geschäftsszenarien Betrachtungen über die eingeführten Ebenen hinweg angestellt. Ergebnisse einer

Ebene können dabei von den anderen genutzt werden, indem sie bei der Bottom-up-Betrachtung abstrahiert bzw. bei der Top-down-Sichtweise konkretisiert werden.

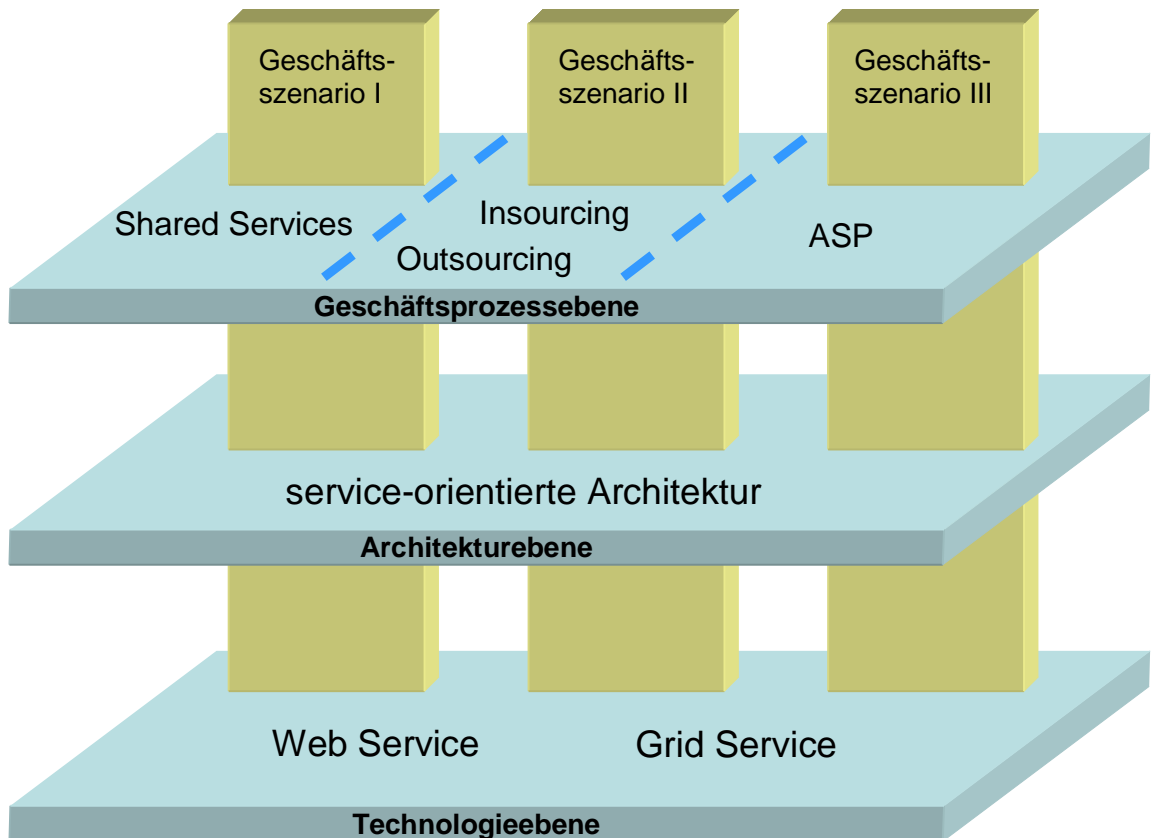


Abbildung 3: Betrachtungsebenen der Untersuchung

Service-orientierte Architekturen (SOA) bilden den Bezugsrahmen für die Untersuchung und werden in Kapitel II betrachtet. Nach einer allgemeinen Beschäftigung mit (Software-) Architekturen und den Grundlagen der service-orientierten Architektur erfolgt eine Charakterisierung der SOA. Durch den relativ hohen Abstraktionsgrad auf dieser Betrachtungsebene werden in einem ersten Schritt betriebswirtschaftlich orientierte Beschreibungen der beteiligten Akteure (Service-Provider, Service-Broker, Service-Requestor) möglich. Die Akteure werden hierbei mit Unternehmen bzw. mit Organisationseinheiten gleich gesetzt. Diese Prämisse wird in der später folgenden technologischen Untersuchung durch eine software-technische Sichtweise der verschiedenen Rollen bzw. Akteure ergänzt.

Neben der Beschäftigung mit den Rollen und Beziehungen innerhalb der SOA wird eine Analyse der Konsequenzen aus der unterschiedlichen organisatorischen Zuordnung der Akteure auf die entstehenden Dienste-Szenarien vorgenommen. Dies ist möglich, da es in einer SOA aus architektonischer Sicht nicht entscheidend ist, ob Dienste nur innerhalb einer Unternehmung angesiedelt werden, oder ob ein Dienst für eine potenzielle Nutzergruppe im Internet zur Verfügung gestellt wird.

Anschließend werden Konsequenzen für die Implementierung einer SOA abgeleitet und Hinweise für die praktische Umsetzung erarbeitet. Hierbei sind konkrete Ansätze auf der Ebene der SOA-Plattform genauso wichtig wie das Service-Design.

Die in Kapitel III zu untersuchende technologische Grundlage für die Implementierung einer service-orientierten Architektur sind Web Services auf XML-Basis, die zur innerbetrieblichen Anwendungsintegration genutzt werden können und auch die Anbindung von externen Anwendungen erlauben. Web Services bilden dabei entweder die Schnittstelle(n) zu komplexen Anwendungen, wie ERP-Systemen, oder stellen selbst eine abgegrenzte Funktionalität bereit, die unter Umständen erst durch geeignete Kombination mit anderen Diensten oder Anwendungen eine gegebene Aufgabe realisiert. Die Realisierung einer Schnittstelle mit einem Web Service ist dabei nicht automatisch mit der service-orientierten Gestaltung der jeweiligen Anwendung gleich zu setzen. Durch die Beschreibung und Abgrenzung zu anderen Technologien in Kapitel III.3 wird deutlich, dass eine SOA auch mit alternativen Technologien zu implementieren ist. Allerdings werden hierbei Schwachstellen einzelner Implementierungsansätze deutlich.

Einen weiteren Schwerpunkt des Kapitels bildet die Betrachtung der für die Web Service-Technologie relevanten Protokolle und Standards. Zusätzlich werden Konsequenzen aus dem Ablauf des Standardisierungsprozesses vorgestellt. Aus der Konvergenz der Web Service-Spezifikationen mit Standards aus dem Grid Computing ergeben sich zudem weitere Szenarien für verteiltes Rechnen.

Nachdem die Architektur und die Technologie analysiert wurden, werden in

Kapitel IV konkrete Typen von IT Sourcing-Szenarien vorgestellt. Als Referenzsituation und Voraussetzung für folgende Szenarien dient eine SOA innerhalb eines Unternehmens.

Davon ausgehend wird der Weg von der innerbetrieblichen Serviceorientierung über die Installierung eines Shared Services Centers und die Unterstützung von In-/ Outsourcing durch Web Services hin zur Integration von Diensten externer Serviceprovider aufgezeigt.

Im Provisioning-Szenario bietet sich für ein Unternehmen nicht nur die Möglichkeit, die Dienste im "pay-as-you-use"-Verfahren zu nutzen, sondern unter Umständen auch als Anbieter von konkreten Anwendungsdiensten aufzutreten oder Rechendienste aus dem Bereich des Grid Computing anzubieten.

Je weiter die Szenarien externe Dienstleister in Geschäftsprozesse einbinden, desto wichtiger werden Fragen des Billing bzw. der Abrechnung:

Während Outsourcingbeziehungen typischerweise auf längere Dauer angelegt sind, ist bei Provisioning-Szenarien die Möglichkeit der Einmalnutzung eines Dienstes gegeben. Es besteht somit keine enge Geschäftsbeziehung des Anbieters zu seinen Kunden, die im Falle von Outsourcing die praktische Realisierung der Abrechnung erleichtert. Darüber hinaus ermöglicht die Höhe der Zahlungen an den Outsourcer im Vergleich zu den Transaktionskosten eine wirtschaftlich sinnvolle Abwicklung durch seine Kunden.

In Kapitel V werden prototypische Umgebungen präsentiert, die für das Sourcing von IT-Leistungen wichtig sind. Als Ausgangssituation wird die Einbindung eines Enterprise Resource Planning-Systems in eine Systemlandschaft über einen Integrationsserver gezeigt. Exemplarisch wird hierzu das SAP R/3 Release 4.6c und der SAP Business Connector verwendet. In Analogie zu den schon vorgestellten verschiedenen Implementierungstechnologien lassen sich hier auch ohne die Verwendung von Web Services wichtige Aspekte der SOA aufzeigen.

Die Web Service Umgebung bedient sich ausschließlich Open Source Komponenten. Es wird hierbei zuerst die Programmiersprachenunabhängigkeit demonstriert, die für das Outsourcing- und Shared Services-Szenario wichtig

ist. Darüber hinaus wird auf die einfache Gestaltung von Integrationsmöglichkeiten von Web Services in Portale hingewiesen.

Die Grid-Umgebung steht für das Outsourcing von komplexen Anwendungen. Für die nahtlose Integration mit Web Services wird das Web Services Resource Framework verwendet. Der Abschnitt zeigt eine weitere wichtige Komponente der integrierten Hosting Umgebung neben der dedizierten Web Service-Umgebung.

Die integrierte Umgebung steht als exemplarische Hosting-Umgebung für den Next Generation Service Provider. Beide schon vorgestellten Hosting-Umgebungen werden hierzu zusammengeführt. Es werden zwei für den Hosting-Prozess entscheidende Aspekte aufgezeigt:

1. die Gewinnung von Metadaten für administrative und abrechnungstechnische Zwecke und
2. die Migration von Hosting-Umgebungen.

Hierbei wird die Wirkung von Virtualisierungstechniken präsentiert.

Den Abschluss bildet eine zusammenfassende Auswertung der verschiedenen Umgebungen bevor mit einem Resümee und einem Ausblick die Arbeit abgeschlossen wird.

### ***1.3 Motivierendes Szenario aus der Personalwirtschaft***

In der Abbildung 4 wird ein idealtypisches Szenario aus dem Bereich der Personalwirtschaft dargestellt. Der Funktionsbereich Personal wurde wegen seiner branchenübergreifenden Gültigkeit gewählt. Die Erkenntnisse können jedoch leicht auf andere Funktionsbereiche oder spezielle Branchen übertragen werden.

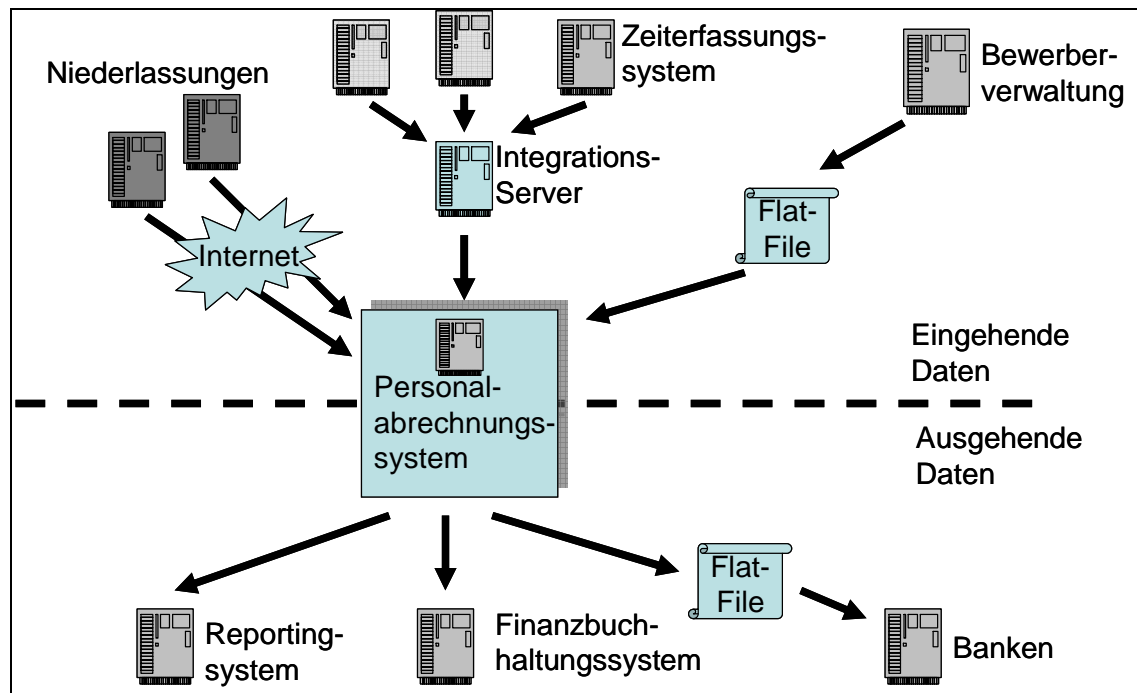


Abbildung 4: Personalabrechnungssystem in einer "gewachsenen" Systemlandschaft

Es wird von einer gewachsenen Systemlandschaft ausgegangen, die sich im Laufe mehrerer Jahre über verschiedene Rechner- und Softwarearchitekturen hinweg entwickelt hat. Parallel dazu wurden unter Umständen auch mehrere Wechsel der IT-Strategie vollzogen. Es sind verschiedene Hardwareplattformen unterschiedlicher Bauart und Betriebsdauer genauso zu finden, wie verschiedenste Betriebssystem- und Anwendungssoftware-Produkte. Selbst gleiche Systeme sind in verschiedenen Konfigurationen und Releaseständen installiert. Des Weiteren ist das Szenario auf verschiedene Unternehmensstandorte in Form von Niederlassungen verteilt und bindet externe Geschäftspartner wie Banken mit ein.

Bei den Servern lassen sich 3 verschiedene Gruppen identifizieren:<sup>4</sup>

<sup>4</sup> Im Rahmen der Beschreibung wird nicht zwischen einer Software- und Hardwaresicht unterschieden, da die Konsequenzen der Aussagen sich auf beide Sichten beziehen bzw. leicht übertragen lassen.



- 1) Server, die ständig Daten liefern bzw. Daten empfangen. (z. B. Zeiterfassungssysteme)
- 2) Server, die nur zu bestimmten Zeiten Daten beziehen, oder verschicken. (z.B. Systeme in den Niederlassungen)
- 3) Server, die nur sporadisch im Einsatz sind. (z. B. Server für spezielle Reporting-Anforderungen)

Bei der Entscheidung ein vorhandenes System aus der Systemlandschaft herauszulösen, oder im Falle der Installation eines neuen Systems muss die Systemumgebung ermittelt werden. Bei den ersten beiden Systemarten ist dies meist kein Problem, da sie bewusst im Blickfeld der Mitarbeiter sind. Schwierig wird die Erhebung jedoch bei der Kombination der zweiten mit der dritten Gruppe, also bei Systemen, die auf verschiedene Standorte verteilt sind, und/oder nur für sporadische Aufgaben verwendet werden.

Die dargestellte Situation kann sich schon bei wenigen Systemen durch eine hohe Anzahl an technischen Schnittstellen einstellen. Hierbei kann analog zu den Servern eine Einteilung in ständig genutzte, temporär genutzte und sporadische Schnittstellen getroffen werden. Die Problematik der Erhebung bleibt ebenfalls erhalten.

Vor diesem Hintergrund wird klar, dass eine betriebswirtschaftlich motivierte Entscheidung ein System aus dem bestehenden Systemkontext herauszulösen bzw. in diesen Kontext einzubringen, durch technische oder technologische Hürden erschwert und im Einzelfall sogar verhindert werden kann.

#### ***1.4 Ziele und Einordnung der Arbeit***

Ausgehend von einer betriebswirtschaftlich relevanten Fragestellung werden aktuelle Technologien als Lösungsansätze präsentiert und prototypische Umgebungen evaluiert.

Insbesondere werden konkrete Anwendungsszenarien identifiziert, die die hohe Bedeutung von Schnittstellen zeigen und eine immer stärkere Einbindung

---

externer Akteure demonstrieren. Dabei werden Gemeinsamkeiten identifiziert, die für die Lösung auf technischer Ebene eine einheitliche Herangehensweise erlauben.

Für praktische Anwendungen im Unternehmen soll der dargestellte Migrationspfad einen Anhaltspunkt bieten und Problemfelder aufzeigen.

Auf der technischen Ebene werden weitestgehend Lösungsansätze aus dem Bereich der Open Source Software verwendet. Neben Kostenvorteilen gewährleistet dieser Ansatz den Zugang zu neuen Entwicklungen, die unter Umständen später Einzug in kommerzielle Produkte finden oder Anregung für den Entwicklungsprozess von Software-Herstellern bieten.

Als weiteres Ziel wird ein neuer Typ des Application Service Providers dargestellt, dessen Angebot an Services sich durch die innerbetrieblich über Serviceorientierung gewonnene Flexibilität einfacher in die Systemlandschaft einbinden lässt. Speziell für diesen Fall wird eine geschlossene Hosting-Umgebung für Grids und Web Services präsentiert und ein Vorschlag für die Verwendung von Virtualisierungstechniken gegeben.

## II. Die service-orientierte Architektur

Als Einstieg in die Betrachtung der service-orientierten Anwendungsintegration wird die Perspektive der Softwarearchitektur gewählt. Die so gewonnene Abstraktion lässt Raum für betriebswirtschaftliche und zukunftsgerichtete Betrachtungen, die nicht durch technische Möglichkeiten einer spezifischen Implementierungstechnologie eingeengt werden.

### II.1 (Software-) Architektur

Unter einer Softwarearchitektur (kurz Architektur) eines Programms bzw. eines Systems wird im Folgenden die Beschreibung der Komponenten, ihrer sichtbaren Eigenschaften und ihrer Interaktion, also die Struktur(en) eines Systems verstanden.<sup>5</sup>

Die Softwarearchitektur stellt neben der Anwendungsarchitektur, der Datenarchitektur oder auch der Systemarchitektur einen Teilbereich der Informationsarchitektur eines Unternehmens dar.<sup>6</sup> Die Informationsarchitektur bildet dabei das „grundlegende konzeptuelle Architekturmodell“<sup>7</sup>.

Wie die Begriffe Programm bzw. System als Bezugssystem für die Betrachtung der Architektur nahe legen, lässt sich nicht nur die Architektur einer einzelnen Anwendung (= Application Architecture) beschreiben, sondern auch auf der Ebene des Unternehmens eine Gesamtarchitektur (= Enterprise Architecture) entwickeln.<sup>8</sup> Die Application Architecture kann dabei nach ERL (2005) unterschiedlich detailliert abgefasst werden und fügt sich bei mehreren nebeneinander gültigen Architekturen in den Rahmen einer gemeinsamen Enterprise Architektur ein, die durch die dort festgelegten Prinzipien jeweils darauf Einfluss

---

<sup>5</sup> Vgl. WANG u. a. (2004), S. 33 und BASS/ CLEMENTS/ KAZMAN (1998), S. 23

<sup>6</sup> Vgl. HEINRICH/ LEHNER (2005), S. 53

<sup>7</sup> HEINRICH/ LEHNER (2005), S. 52

<sup>8</sup> Vgl. ERL (2005), S. 87

nimmt.<sup>9</sup>

## **II.2 Nutzenpotenziale aus der Beschäftigung mit Softwarearchitekturen**

Die Beschäftigung mit (Software-) Architekturen ermöglicht, aus einer abstrahierenden Sichtweise heraus auf ein zukünftiges oder schon entwickeltes System zu blicken. Es können neben den notwendigen taktischen und zielorientierten Aspekten eines Entwicklungsprojektes auch strategische Überlegungen aus der Sicht des gesamten Unternehmens einbezogen werden.<sup>10</sup> Mit anderen Worten entkoppelt die Architektur im Rahmen der strategischen Planung die Anpassung von konkreten Anwendungssystemen von strategischen Zielen der Unternehmung.<sup>11</sup>

Eine gemeinsame Architektur erleichtert die Interoperabilität von Anwendungen, gewährleistet das Erkennen von Synergien im Entwicklungsprozess und kann bei mehrfachem Einsatz durch die Harmonisierung der Systeme positive Auswirkungen auf den Wartungsaufwand einer Systemlandschaft haben.<sup>12</sup>

## **II.3 Architektonische Grundlagen**

Die service-orientierte Architektur (SOA) ist keine komplette Neuentwicklung, sondern trägt klare Züge anderer Architekturen in sich. So ordnet sich die SOA in den Bereich der verteilten Systeme ein, wobei hier nur der generelle Ausgangspunkt zu sehen ist und noch keine konkreten Ansatzpunkte für die speziellen Eigenschaften zu finden sind. Engere Beziehungen bestehen jedoch zum Composite Computing Model (CCM). Man kann die SOA sogar als eine Konkretisierung dieses Modells sehen.

---

<sup>9</sup> Vgl. ERL (2005), S. 87

<sup>10</sup> Vgl. KRAFZIG/ BANKE/ SLAMA (2005), S. 5

<sup>11</sup> Vgl. HEINRICH/ LEHNER (2005), S. 50

<sup>12</sup> Vgl. FOEGEN (2003), S. 58 f.

### II.3.1 Verteilte Systeme

Ein verteiltes System lässt sich durch seine Bestandteile, eine Vielzahl an atomaren, über ein Netzwerk kommunizierenden Software-Agenten charakterisieren, die erst durch ihr Zusammenwirken über die eigenen Ablaufumgebungen hinaus eine gestellte Aufgabe erfüllen.<sup>13</sup>

Aus dieser Definition lassen sich Schlüsse auf die grundsätzliche Systemumgebung ziehen. Es erfolgt eine Kommunikation zwischen den Software-Agenten, die über den eigenen Adressbereich hinausgeht, wobei bis auf die Nachrichtenschnittstelle des empfangenden Agenten nichts über dessen Ablaufumgebung bekannt ist.<sup>14</sup> Bricht man die hier angesprochene softwaretechnische Sichtweise auf Hardwareplattformen herunter, kann nicht mit Sicherheit festgestellt werden, ob die Kommunikationspartner auf der gleichen Plattform rechnen – ein Informationsdefizit, das weit reichende Auswirkung auf ein anvisiertes Implementierungsmodell hat.<sup>15</sup>

Es lassen sich vier Bereiche identifizieren, denen deshalb eine besondere Aufmerksamkeit geschenkt werden muss:<sup>16</sup>

- Latenz durch die Zeit, die für die Nachrichtenübermittlung gebraucht wird
- Speicherzugriff bzw. Speicherbereiche
- Ausfall von Teilbereichen des verteilten Systems aus Software-Agenten
- Probleme der Gleich- bzw. Nebenläufigkeit

### II.3.2 Das Composite Computing Model (CCM)

Auslöser für die Entwicklung des Composite Computing Models (CCM) waren Anforderungen aus bestimmten Geschäftsprozessen, die Komponenten eines

---

<sup>13</sup> Vgl. BOOTH u. a. (2004)

<sup>14</sup> Vgl. WALDO u. a. (1994), S. 2

<sup>15</sup> Vgl. WALDO u. a. (1994), S. 2

<sup>16</sup> Vgl. WALDO u. a. (1994), S. 5

Softwaresystems, die die Geschäftslogik enthalten, dynamisch finden zu können und mit möglichst geringen Vorarbeiten (z. B. Planung, Installation), gemeinsam in einer verteilten Umgebung ausführen zu können.<sup>17</sup> Dabei sollen menschliche Eingriffe auf ein Minimum beschränkt sein und die Beschreibung des logischen Ablaufs der Bestandteile losgelöst von der konkreten Implementierung ermöglicht werden.<sup>18</sup> Zusammengefasst lässt sich das CCM somit als eine verteilte Laufzeitumgebung charakterisieren, in der dienstartige Softwaregüter (einzelne Methoden oder Komponenten) verwaltet, veröffentlicht und dynamisch gefunden werden können.<sup>19</sup> Im Vergleich zum verteilten System kommt hier die Forderung nach der dynamischen Entdeckung hinzu, was komplexere Szenarien ermöglicht.

## **II.4 Charakterisierung der service-orientierten Architektur**

Die service-orientierte Architektur oder kurz SOA ist ein Versuch, die weit reichenden Vorstellungen des Composite Computing Model zu konkretisieren, und ein Stück näher an die technologische Realisierung zu bringen.<sup>20</sup> BURBECK (2000) grenzt von der service-orientierten Architektur (= service-oriented architecture) die service-basierte Architektur (= service-based architecture) durch die dynamische Entdeckung der Dienste ab, die einem reinen Austausch von Nachrichten zwischen Diensten gegenübersteht.<sup>21</sup>

Auf sehr abstrakte Weise lässt sich die SOA folgendermaßen definieren:

*„SOA is an architectural style whose goal is to achieve loose coupling among interacting software agents.“<sup>22</sup>*

---

<sup>17</sup> Vgl. CHAPPEL/ TYLER (2003), S. 15

<sup>18</sup> Vgl. CHAPPEL/ TYLER (2003), S. 15

<sup>19</sup> Vgl. CHAPPEL/ TYLER (2003), S. 15

<sup>20</sup> Vgl. CHAPPEL/ TYLER (2003), S. 16

<sup>21</sup> Vgl. BURBECK (2000)

<sup>22</sup> HE (2003)

Hierbei wird klar auf verteilte Systeme Bezug genommen und die Eigenschaft der losen Kopplung herausgestellt, also die Vermeidung einer zu engen Bindung der Applikation an einen bestimmten Kommunikationspartner.<sup>23</sup>

Die Basis für die dynamische Interaktion bildet der Dienst bzw. Service, der als modularer Baustein der Architektur, eine wohl definierte Aufgabe übernimmt und die Beschreibung seiner Funktionsweise liefern kann.<sup>24</sup> Hinter dem Dienst kann jedoch entweder nur ein atomarer Verarbeitungsschritt, ein kleiner Teilprozess oder ein komplexer Geschäftsprozess stehen.<sup>25</sup> Aus dieser Aussage lässt sich schon ableiten, dass die konkrete Implementierung des Dienstes und die Einzelheiten der darin gekapselten Geschäftslogik für den Verwender des Dienstes keine Rolle spielen.

Um den Dienst herum entsteht ein Dreieck aus dem Provider, dem Diensteanfragenden (= Requestor) und der Registrierungsstelle (= Broker).<sup>26</sup> Die einzelnen Bestandteile dieses Dreiecks sind zunächst einmal Programme.

Der Provider ist der Eigentümer der in Form des Dienstes implementierten Geschäftslogik. Der Broker hält Informationen über den angebotenen Dienst und seinen Anbieter in elektronischer Form vor und ermöglicht es dem Requestor, die gewünschte Geschäftslogik zu entdecken und in geeigneter Weise in den Zielgeschäftsprozess einzubinden.<sup>27</sup> Bei der Darstellung ist zu beachten, dass es sich bei den verschiedenen Rollen jeweils gleichermaßen um Unternehmen bzw. Organisationseinheiten und um Applikationen handelt, wenn dies auch nicht explizit angesprochen wird.

Bei der Verwendung einer service-orientierten Architektur fließen eine Reihe von service-orientierten Prinzipien ein, die sich herausgebildet haben, ohne dass sie laut ERL (2005) normativen Charakter erlangt haben.<sup>28</sup>

- Dienste sollten auf eine potenzielle Wiederverwendbarkeit hin ausgelegt

---

<sup>23</sup> Vgl. HE (2003)

<sup>24</sup> Vgl. CHAPPEL/ TYLER (2003), S. 5

<sup>25</sup> Vgl. ERL (2005), S. 34

<sup>26</sup> Vgl. CHAPPEL/ TYLER (2003), S. 17

<sup>27</sup> Vgl. für die letzten beiden Sätze CHAPPEL/ TYLER (2003), S. 18 f.

<sup>28</sup> Vgl. ERL (2005), S. 291 f.

sein.

- Für die Interaktion von Diensten muss eine formale Vereinbarung ausreichen, die neben der Schnittstelle auch die Bedingungen für den Nachrichtenaustausch festlegt.
- Dienste sollen nur lose miteinander gekoppelt werden und möglichst keine Abhängigkeiten mit anderen Diensten aufweisen.
- Ein Dienst kann als Blackbox gesehen werden. Die Art der Implementierung der Logik ist für die Kommunikation mit dem Dienst und für die Nutzung des Dienstes nicht wichtig.
- Dienste sollten mit anderen Diensten innerhalb eines Geschäftsprozesses verwendet werden können.
- Dienste sind innerhalb der Grenzen der implementierten Funktionalität autonom und von keinen anderen Diensten abhängig.
- Dienste sollten zustandslos implementiert werden. Informationen zum aktuellen Zustand können das Prinzip der losen Kopplung gefährden.
- Dienste sollen anhand ihrer Beschreibung gefunden werden können. Die Beschreibung soll dabei sowohl von Menschen als auch von den Softwarekomponenten interpretiert werden können.

Die genannten Prinzipien sind mehr oder weniger bindend für die praktische Umsetzung. Am Beispiel der Forderung nach einer zustandslosen Implementierung eines Services kann dies gezeigt werden. Im Rahmen der Verwendung eines Grid innerhalb einer service-orientierten Architektur besteht zwingend die Forderung nach stateful web services (vgl. Abschnitt III.5.7 und Abschnitt III.6.1). Ohne diese Eigenschaft sind Ressourcen des Grids in eine SOA gar nicht integrierbar.



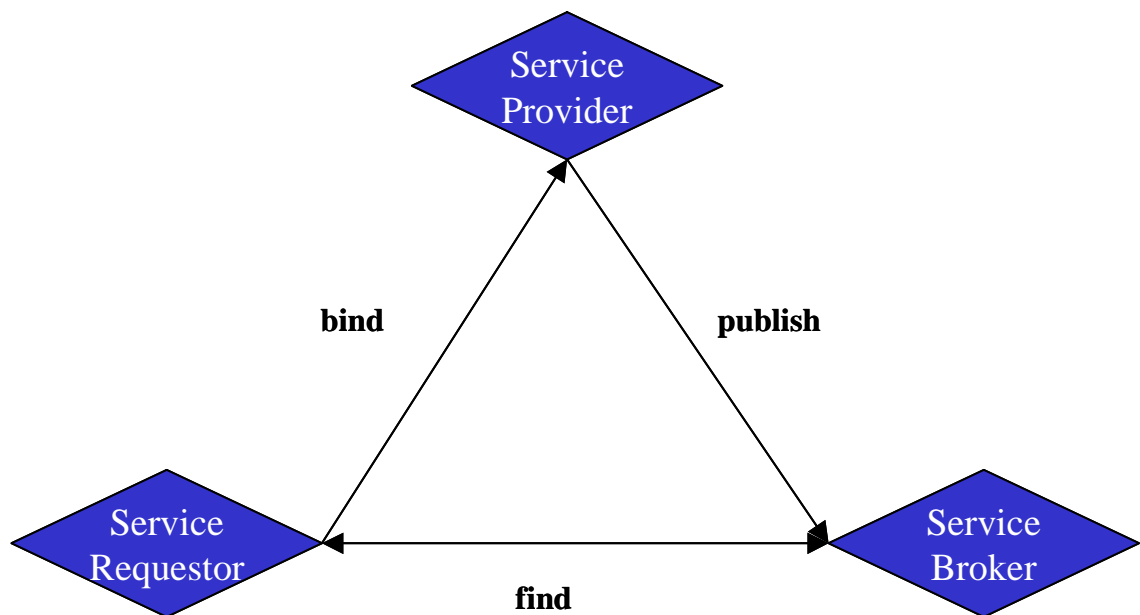


Abbildung 5: Rollen und Beziehungen innerhalb der SOA [Quelle: BURBECK (2000)]

Einzelne oder alle Funktionen können unterschiedlichen organisatorischen Einheiten zugeordnet werden, aber auch simultan von einer Organisation übernommen werden. Aus einer interorganisationalen kann so eine intraorganisationale Zusammenarbeit werden und umgekehrt – eine Tatsache, die jedoch an der architektonischen Konzeption nichts ändert.

Zwischen den drei Rollen entsteht ein Beziehungsgeflecht.<sup>29</sup>

- Der Service Provider veröffentlicht die Service-Beschreibung (Aktivität „publish“ in Abbildung 5) beim Service Broker. Der Broker wiederum ordnet den Dienst innerhalb einer von ihm entwickelten Taxonomie ein und stellt Suchmöglichkeiten zur Verfügung.
- Anschließend kann der Requestor einen Dienst auffinden (Aktivität „find“ in Abbildung 5).
- Wurde ein Dienst gefunden, findet die weitere Interaktion mit dem Service Provider statt. Der Dienstenachfrager bindet den Dienst in den Zielprozess bzw. die Zielanwendung ein. (Aktivität „bind“ in Abbildung 5).

<sup>29</sup> Vgl. BURBECK (2000)

Der Prozess des Findens und Einbindens eines Services kann unterschiedlich ausgestaltet und mehr oder weniger dynamisch angelegt sein. Das Ausmaß der Dynamik wird durch den Grad der Determinierung der möglichen Serviceanbieter als Kommunikationspartner gekennzeichnet.

Im Falle einer sehr geringen Dynamik wird eine Servicebeschreibung schon während der Entwicklungsphase zur festen Anbindung einer Anwendung an einen Dienst genutzt.<sup>30</sup> Um die Fehlerhäufigkeit zu verringern, kann der Broker in einem nächsten Schritt dazu benutzt werden, den schon bekannten Service durch gezielte Anfragen zu finden, um Details zum Datenaustausch (z. B. Art des Transportprotokolls, URI des Dienstes) einzuholen oder abzugleichen.<sup>31</sup> Ist einer Anwendung die Semantik der Dienstschnittstelle eines Dienstes bekannt, so kann die Anfrage an den Broker zum Finden von alternativen Anbietern genutzt werden.<sup>32</sup> Mit zunehmend geringerer Vorbestimmung des Service-Providers steigt jedoch auch die Komplexität des Prozesses. Je dynamischer die Bindung angelegt ist, desto höher ist die Gefahr des Versagens der Service-nutzung und daraus resultierend der Aufwand, der für die Vermeidung dieser Situation zu treffen ist. Das Versagen kann durch das Scheitern des Findens verursacht werden bzw. falls der Service gefunden wird, besteht die Gefahr des Versagens des Dienstes als solchen.

## II.4.1 Akteure und Rollen

Im Zuge der Einführung des Begriffs der SOA bilden sich drei Determinanten der Betrachtungsweise der Akteure und Rollen innerhalb der Architektur heraus:

- die Assoziation der Dienste mit Programmen bzw. mit den hinter den Diensten stehenden Organisationseinheiten
- die Art der Einbindung der Services in einen Organisationskontext

---

<sup>30</sup> Vgl. BURBECK (2000)

<sup>31</sup> Vgl. BURBECK (2000)

<sup>32</sup> Vgl. BURBECK (2000)

- die Dynamik in der Dienstenutzung

Der Begriff des Akteurs kann dabei nicht mit dem Begriff der Rolle in der SOA synonym verwendet werden. Während beispielsweise in einem Szenario innerhalb eines Unternehmens der Akteur, also die Organisationseinheit Unternehmen, immer gleich bleibt, nimmt die Organisationseinheit in ihren Untergliederungen (z. B. Abteilungen) jeweils alle drei Rollen (Provider, Requestor und Broker) simultan war.

#### **II.4.1.1 Der Service Provider**

Beim Begriff des Service Providers lassen sich die interorganisationellen von den innerorganisationellen Szenarien abgrenzen. Innerorganisationell werden Teilfunktionen von Anwendungen bzw. ganze Anwendungen als Dienste gestaltet. Durch diese innerhalb der Unternehmensgrenzen angebotenen Services lassen sich über definierte, selbstbeschreibende Schnittstellen Anwendungsfunktionalitäten für andere Anwendungen bereitstellen.

Sind diese Funktionalitäten auch für andere Unternehmen interessant, da sie sich beispielsweise durch übertragbares Prozess-Know-how auszeichnen, so bietet sich die Öffnung der Unternehmung für Externe im Rahmen der zur Verfügungsstellung des Dienstes an. Es entsteht ein interorganisatorischer Prozess, dessen Ursprung jedoch in Bemühungen der Anwendungsintegration im Sinne einer losen Kopplung liegt.

Losgelöst von existierenden Geschäftsprozessen können unabhängige Softwarehersteller als Provider auftreten. Sie stellen ihre Anwendung in Form von Diensten für einen unbestimmten Personenkreis im Internet zur Verfügung und behalten somit die Hoheit über ihre Entwicklung und haben die Möglichkeit einer gezielten Programmpflege.

#### **II.4.1.2 Der Broker**

Aus Sicht der Software stellt der Broker Informationen über Dienste und

insbesondere über deren Schnittstellen bereit. Schnittstellen werden typischerweise während ihrer Designphase dokumentiert und finden Eingang in Konzepte zu größeren Anwendungssystemen, oder werden beispielsweise im Rahmen von Integrationsprojekten erneut ermittelt und im Zweifel angepasst. Das Vorhalten eines ständig aktualisierten Schnittstellenkataloges, der neben der fachlichen Bedeutung auch technische Informationen über die Erreichbarkeit (z. B. IP-Adresse) enthält, wird jedoch eher selten zu finden sein. Gerade einen solchen Wissenspool kann ein Broker zur Verfügung stellen.

Bewegen wir uns im Unternehmenskontext, so lassen sich anfänglich Dienste in etwa mit Schnittstellen zu Anwendungssystemen gleichsetzen. Diese rein technische Gestaltung der Schnittstellen ist nur bedingt vorteilhaft und verursacht eine unübersichtliche Menge an Diensten in einem Service-Verzeichnis. Durch die service-orientierte Herangehensweise tritt die konkrete Anwendung jedoch zusehends in den Hintergrund und das Service-Design orientiert sich eher an betriebswirtschaftlich sinnvoll gebildeten Diensten. Dies ist der Übergang von der *service-based architecture* zur *service-oriented architecture*. Je stärker die Funktionalität betont wird, desto mehr tritt die Anwendung, die die Funktionalität umsetzt, in den Hintergrund.

Verlässt man die rein softwaretechnische Sicht und bezieht den Organisationskontext mit ein, so kann ein Broker innerhalb des Unternehmens der erste Schritt auf dem Weg zur Nutzung von externen Verzeichnisdiensteanbietern sein. Unternehmensexterne Kataloge werden anfänglich von Herstellern service-orientierter Software als Showcase bzw. Verkaufsargument vorgehalten. Falls keine unabhängigen Organisationen diese Dienstleistung kostenlos erbringen, ist der nächste Schritt jedoch das kommerzielle Angebot von Informationen über Dienste. Die Art der Refinanzierung kann dabei beispielsweise über die gebührenpflichtige Listung von Providern geschehen oder aber durch kostenpflichtige Anfragen der Dienstesuchenden. Der Broker reduziert die Distanz zwischen Provider und Requestor, die durch relativ hohe Transaktionskosten bei der Suche entsteht, durch die Erstellung einer geeigneten Taxonomie und die Gewährleistung eines genügend breiten und tiefen Providerangebotes in seinem Verzeichnis. Neben dieser Funktion können bei dem Broker als zentrale Anlaufstelle entscheidende Informationen über die

Nutzung der Dienste zusammenlaufen (z. B. Dienstqualität, Geschwindigkeit, Fehleranfälligkeit) und über deren Aggregation und Auswertung ein zusätzliches Hilfsmittel beim Suchprozess sein.

Fallen bei der Dienstenutzung Gebühren an, so wäre er auch für die Bankenfunktion im Sinne einer Clearingstelle prädestiniert. Dies wird vor allem deshalb nötig, da die anfallenden Beträge im Vergleich zu den Abwicklungskosten in keinem ökonomisch sinnvollen Verhältnis stehen und somit unter Umständen sogar prohibitiv für die Inanspruchnahme sein können. Eine im Extremfall nur einmalige Nutzung eines kostenpflichtigen Dienstes würde sich ohne Mittler eventuell ausschließen.

### **II.4.1.3 Der Requestor**

Beim Requestor bietet sich ebenfalls die organisatorische Sicht als Grundlage für die nähere Charakterisierung der Rolle an.

Innerhalb des service-orientierten Unternehmens ist mit einer Reduzierung der Komplexität zu rechnen. Es entstehen zwar unter Umständen durch den offeneren Aufbau der Anwendungskommunikation mehr Schnittstellen. Die Schnittstellen sind jedoch grundsätzlich vom selben Typ und werden in einem unternehmensweit zugänglichen Verzeichnis registriert. Ändert sich also beispielsweise der Standort einer Anwendung, so ist nur eine Anpassung der Adresse der Anwendung im Verzeichnis nötig. Von dieser Schnittstelle abhängende Anwendungen, die nicht mit einem hart kodierten Zielpunkt versehen wurden, brauchen nicht angepasst werden. Durch eine einfache Anfrage an den Verzeichnisdienst kann die korrekte Adressierung der Zielanwendung gewährleistet werden.

Verlässt man nun die softwaretechnisch geprägte Sicht innerhalb einer Organisation und erweitert den Blickwinkel auf interorganisationelle Prozesse, so rückt die Make-or-buy Entscheidung in den Mittelpunkt. Unter der Voraussetzung, dass die unternehmenseigene Datenverarbeitung zumindest zum Teil serviceorientiert ausgerichtet wurde und für Auslagerungen relevante Anwendungen oder die mit solchen Anwendungen kommunizierenden Anwendungen mit

geeigneten Schnittstellen ausgestattet sind, kann ein solcher Fremdvergabe-prozess beschleunigt abgewickelt werden.

Durch den service-orientierten Ansatz besitzen Anwendungen innerhalb eines Unternehmens Eigenschaften von Anwendungen, die über das Internet zur Verfügung gestellt werden. Der Standort einer mit mehreren Schnittstellen aus-gestatteten Anwendung ist für den Requestor dadurch genauso irrelevant, wie die in einer Anwendung stattfindenden Verarbeitungsschritte. Diese Abstraktion ermöglicht neben der erleichterten Fremdvergabe des Applikationsbetriebes den einfacheren Austausch einer alten gegen eine Anwendung mit unter Um-ständen erweiterten Funktionalitäten.

## **II.4.2 Interaktion und Stufen der Interaktion**

Bei der Interaktion zwischen den drei Rollen innerhalb einer SOA sind zwei verschiedene Dimensionen zu unterscheiden: Eine Dimension betrifft den Grad der Beteiligung Externer bei der zur Verfügungsstellung und Nutzung von Diensten bzw. bei der Bereitstellung von Verzeichnisdienstleistungen. Die andere Dimension betrifft die schon angesprochene Dynamik, die sich im Ausmaß des Entdeckungsprozesses äußert, der zu einer Diensteinteraktion führt.

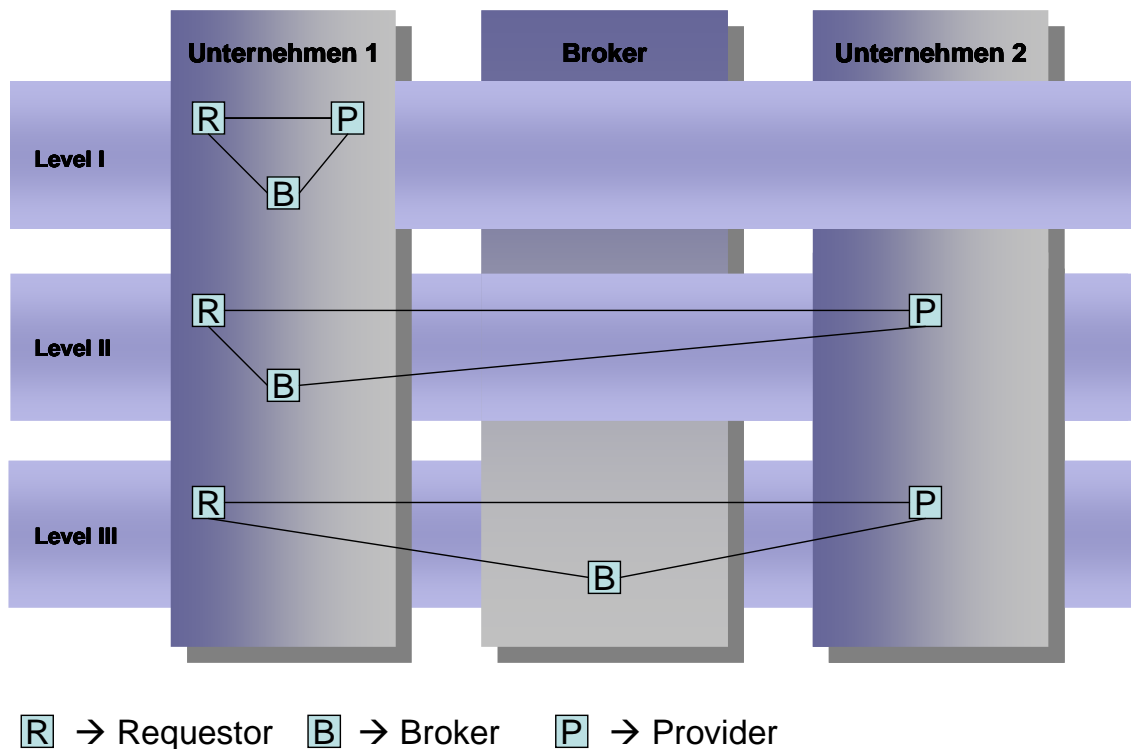


Abbildung 6: Ebenen der Dienstnutzung

In Abbildung 6 ist dies in Form einer Matrix verdeutlicht. Es entstehen drei Level bzw. Ebenen der Diensteszenarien. Das Level I umfasst alle innerorganisationellen Interaktionen. Über den Grad der Dynamik kann hierbei nur die Aussage getroffen werden, dass in Szenarien zumindest kein externer Akteur eine Rolle spielt. Eine dynamische Dienstesuche ist jedoch trotzdem möglich. Dieses Szenario kann als Ausgangspunkt für den Aufbau einer service-orientierten Architektur dienen. Es lassen sich alle Bestandteile in Vorbereitung späterer interorganisationeller Prozesse testen und auf die späteren Szenarien übertragen.

Level II ist der nächste logische Schritt auf dem Weg zur vollen Nutzung der SOA. Hier wird der Dienst eines externen Providers in die unternehmenseigene DV-Landschaft integriert. Je nach dem Grad der dynamischen Ausrichtung kann der Dienst statisch an die Anwendungen gebunden werden, oder falls der Dienst von mehreren Anwendungen genutzt werden soll, schon in einem unternehmenseigenen Verzeichnisdienst registriert werden. Sollte der externe Diensteanbieter ausgewechselt werden oder sich der Standort der Services ändern, so muss nur hier eine geringfügige Änderung vorgenommen werden,

um die korrekte Dienstenutzung zu gewährleisten. Der interne Verzeichnisdienst kann hierbei als Auflistung von bewährten Diensten genutzt werden, die der Sicherung eines Qualitätsniveaus durch den Dienstenachfragenden zuträglich sind und Ausdruck einer geeigneten Vorselektion ist.

Hat sich die service-orientierte Ausrichtung des Unternehmens in den letzten beiden Stufen bewährt, so kann im Level III der komplette Umfang eingesetzt werden. Obwohl die Darstellung auf die Aufnahme von unternehmenseigenen internen Verzeichnisdiensten verzichtet, können bestehende Verzeichnisse weiterhin geführt und für die internen Dienste genutzt werden, während die öffentlichen Dienste über die externen Verzeichnisse abgefragt und gefunden werden können. Es können hier also immer noch von Unternehmen zu Unternehmen unterschiedliche Adaptionsgeschwindigkeiten realisiert werden, was sich auch in dem Umfang der von unternehmensexternen Providern nachgefragten Dienste widerspiegelt.

### **II.4.3 Verfeinerung der Architektur**

Die Darstellung der SOA in den vorangegangenen Abschnitten spiegelt die in der Literatur vorherrschende Darstellungsweise wider und findet sich schon in frühen Veröffentlichungen zu dem Thema. Durch zunehmende Adaption im Rahmen konkreter Projekte ergeben sich jedoch Anforderungen, die es schon auf architektonischer Ebene abzufangen gilt.



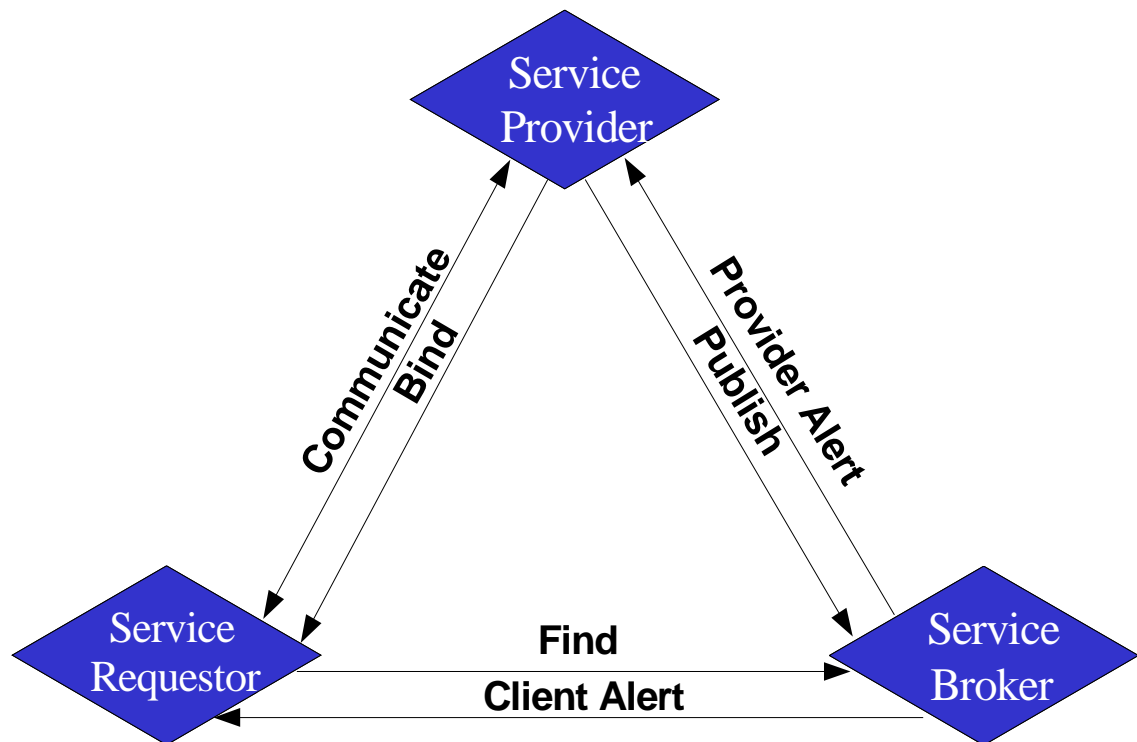


Abbildung 7: Erweiterung der SOA [Quelle: MYERSON (2004)]

Eine für die weitere Arbeit bedeutende Verfeinerung betrifft die zwischen den Diensten ausgetauschten Informationen. So bedarf es neben der Kommunikation bei den Aktivitäten „publish“, „find“ und „bind“ (siehe Ausführungen weiter oben) bei kostenpflichtigen Diensten in Verbindung mit Service Level Agreements eines weiteren Informationsaustausches, wobei hier nicht nur die dynamische Interaktion mit mehreren Diensten, sondern auch die Möglichkeit des Versagens von Aktionen Berücksichtigung finden.<sup>33</sup>

In der von MYERSON (2004) vorgeschlagenen Erweiterung werden zusätzlich zu den in der ursprünglichen Architektur schon vorhandenen Kommunikationskanälen noch Statuskanäle hinzugefügt. Wie in Abbildung 7 dargestellt, lassen sich folgende beispielhafte Interaktionen unterscheiden:<sup>34</sup>

- Kommunikation zwischen Requestor und Broker: Der Broker liefert über den Rückkanal entweder eine Nachricht über die erfolgreiche Suche

<sup>33</sup> Vgl. MYERSON (2004)

<sup>34</sup> Vgl. MYERSON (2004)

bzw. über das Scheitern der Suchanfrage zurück. (= Client Alert in Abbildung 7)

- Kommunikation zwischen Broker und Provider: Der Broker verständigt den Provider nach dessen Publikationsversuch seines Dienstes, ob die Publikation erfolgreich war oder nicht. (= Provider Alert in Abbildung 7)
- Kommunikation zwischen Requestor und Provider: Der Provider teilt dem Requestor auf diesem Wege mit, welche weiteren Dienste gebraucht werden und welche Schritte einzuleiten sind. (= Communicate in Abbildung 7)

Die Erweiterungen lassen sich auch erst in der Realisierung des Dienstes in konkreten Anwendungen berücksichtigen. Jedoch bietet die Propagierung der Erweiterung im Rahmen der Architektur die Chance, dass sich eine neue Standardvorgehensweise bei der Implementierung herausbildet, was die Interoperabilität steigert.

## ***II.5 Konsequenzen für die Implementierung***

Eine (Software-) Architektur soll nicht nur die Kommunikation zwischen den am Entwicklungsprozess eines Softwaresystems beteiligten Parteien fördern, sondern auch ein abstrahiertes und damit übertragbares Modell eines Softwaresystems sein.<sup>35</sup> Ziel ist also ein auf wesentliche Aussagen reduziertes Abbild eines Systems, welches auf mehrere konkrete Systeme bzw. Implementierungen Anwendung finden kann. Es können zusätzliche Freiheitsgrade in Bezug auf die Konsequenz der Umsetzung bestimmter Merkmale des Systems bestehen bleiben.

Bezogen auf die SOA bedeutet dies, dass es durchaus mehr als einen Implementierungsansatz mit unterschiedlich hoher Ausprägung der Idee der Serviceorientierung gibt. So stehen neben Web Services auch CORBA, Java RMI und Sun RPC für konkrete Implementierungen zur Verfügung.

---

<sup>35</sup> Vgl. BASS/ CLEMENTS/ KAZMAN (1998), S. 28

Grundlegende Gemeinsamkeit der Ansätze ist die Realisation eines verteilten Systems mit eigenem Kommunikationsprotokoll und dem Einsatz einer Schnittstellenbeschreibungssprache (engl.: Interface Description Language, IDL), die die Plattformunabhängigkeit fördert.<sup>36</sup> Ausgewählte Ansätze der Umsetzung einer service-orientierten Architektur werden später bei der Vorstellung der Technologie gesondert betrachtet. Zuvor werden jedoch Merkmale von Web Services herausgearbeitet.

Wichtig bei der Auswahl der vorzustellenden Implementierungen ist die Eigenschaft der Portierbarkeit und somit die Einsetzbarkeit im Rahmen heterogener Systemumgebungen. Die Forderung ergibt sich aus der Ausrichtung der Arbeit auf die Integration von Systemen im Internet bzw. in Intranets. Hier kann eine homogene Systemlandschaft nicht als gegeben angesehen werden. DCom von Microsoft wird deshalb nicht weiter dargestellt, da hier die Portabilität nur begrenzt gegeben ist.

## ***II.6 Praktische Umsetzung der SOA***

Im Rahmen der Diskussion der praktischen Umsetzung der service-orientierten Architektur wird zunächst die Frage nach der Akzeptanz und der sie beeinflussenden Faktoren geklärt, bevor Lösungsansätze im Bereich der kommerziellen Software vorgestellt werden. Ist die Entscheidung für den Einsatz einer SOA getroffen, stellen sich weitere spezielle Anforderungen beim Design der Services auf die anschließend eingegangen werden.

Neben der Betrachtung aus Sicht der Softwarehersteller gibt es noch die Perspektive des Unternehmens, das die Architektur umsetzt. Ausgangspunkt für die Implementierung service-orientierter Architekturen in größeren Unternehmen ist meist die Steigerung des Automatisierungsgrades – eine Optimierung bestehender Prozesse steht dabei erst an zweiter Stelle.<sup>37</sup> Mit steigendem Reifegrad der umgesetzten Architekturen ist jedoch ein Wechsel des

---

<sup>36</sup> Vgl. EBERHART/ FISCHER (2003), S. 81 und S. 83 f.

<sup>37</sup> Vgl. REITER (2005), S. 1

Schwerpunktes hin zur Prozessorientierung zu erwarten.

## **II.6.1 Fördernde und hemmende Faktoren für den Einsatz der SOA**

In einer von der Experton Group bei 31 Unternehmen im Rahmen einer branchenübergreifend durchgeführten Befragung zum Thema SOA und ERP haben sich 52 % der befragten Unternehmen zumindest schon mit SOA auseinandergesetzt, wobei nur wenige eine entsprechende Lösung im Einsatz bzw. in der Erprobung haben.<sup>38</sup> Es ist also eine starke Sensibilisierung für das Thema in Unternehmen zu beobachten.

Den wichtigsten Grund sich mit dem Thema zu beschäftigen, stellen die Chancen aus der Integration von Systemen dar, die jedoch erst innerhalb des Unternehmens ausgenutzt werden, da die Integration von Partnern das Vorhandensein einer entsprechenden Infrastruktur auf der Gegenseite voraussetzen würde.<sup>39</sup>

Hat sich eine Integrationslösung in einer heterogenen und komplexen Systemlandschaft etabliert, muss der erwartete Nutzen aus der Ablösung meist nicht nur strategisch, sondern auch operativ sichtbar sein. Als Beispiel lässt sich das zögernde Upgrade in Unternehmen von einer SAP Business Connector-Lösung auf die SAP Exchange Infrastructure trotz den zu erwartenden sinkenden Schnittstellenwartungskosten und dem zu erwartenden Zeitgewinn bei Implementierungs- und Migrationsprojekten anführen.<sup>40</sup>

Daneben ist mit einer neuen Softwarearchitektur wie der SOA meist auch eine gestiegene Komplexität verbunden, die z. B. aus den neuen Protokollen, dem anderen zeitlichen Verhalten der Applikationen durch das Ersetzen von synchronen durch asynchrone Aufrufe und der stärkeren Abhängigkeit von Latenzzeiten des Netzwerkes erwächst, die erst durch Erfahrungswerte und

---

<sup>38</sup> Vgl. FRIEDMANN (2005), S. 39

<sup>39</sup> Vgl. FRIEDMANN (2005), S. 39

<sup>40</sup> Vgl. NIEMANN (2005), S. 17

Best Practices aus ähnlichen Projekten beherrschbar werden.<sup>41</sup>

## II.6.2 Lösungsansätze im Bereich der Standardsoftware

Im Bereich der betriebswirtschaftlichen Standardsoftware liegt der Schwerpunkt auf der Auflösung der bestehenden Systeme in möglichst autonome Services, die durch eine möglicherweise externe Orchestrierungslösung koordiniert werden sollen.<sup>42</sup> Durch die Orchestrierung werden Geschäftsprozesse möglich, die sich aus Services aufbauen. Neben der Flexibilität in Bezug auf die Herkunft der Dienste tritt die Wiederverwendbarkeit der Dienste hierbei im Vordergrund.

Voraussetzung für die flexible Kombination der Services zu einer geschlossenen Funktionalität ist ein Portal, das nicht nur unterschiedliche Daten- und Informationsquellen auf einer Oberfläche vereint und diese rollenspezifisch dem Benutzer zur Verfügung stellt, sondern zusätzlich die Ausdehnung des Rollenbegriffes auf aus Diensten zusammengesetzte Funktionalitäten erlaubt.<sup>43</sup> Dabei stellt die SOA die Entwickler vor einige zu lösende Fragen. Insbesondere muss geklärt werden, wie die Konsistenz und Verfügbarkeit von Daten für Services gewährleistet werden kann, oder auch wie die Granularität der Dienste zu gestalten ist, und vor allem sind Analyse- und Designtechniken für die Dienste zu erarbeiten.<sup>44</sup> Steht wie bei der Granularität keine passende Metrik für die Entscheidung bezüglich der Designgüte eines Services zur Verfügung, müssen, falls vorhanden, Erfahrungen aus früheren Projekten genutzt werden, um daraus Best Practices ableiten zu können. Im Allgemeinen kann die fachliche Zusammengehörigkeit als ausschlaggebendes Moment für die Zusammenfassung von Funktionalitäten in einem Service dienen.

Im Folgenden soll die Betrachtung auf einen konkreten Ansatz eines Softwareherstellers eingeschränkt werden.

---

<sup>41</sup> Vgl. GRASMANN (2005), S. 16 f.

<sup>42</sup> Vgl. OEHLER (2005), S. 41

<sup>43</sup> Vgl. OEHLER (2005), S. 41

<sup>44</sup> Vgl. HUBER (2006), S. 63

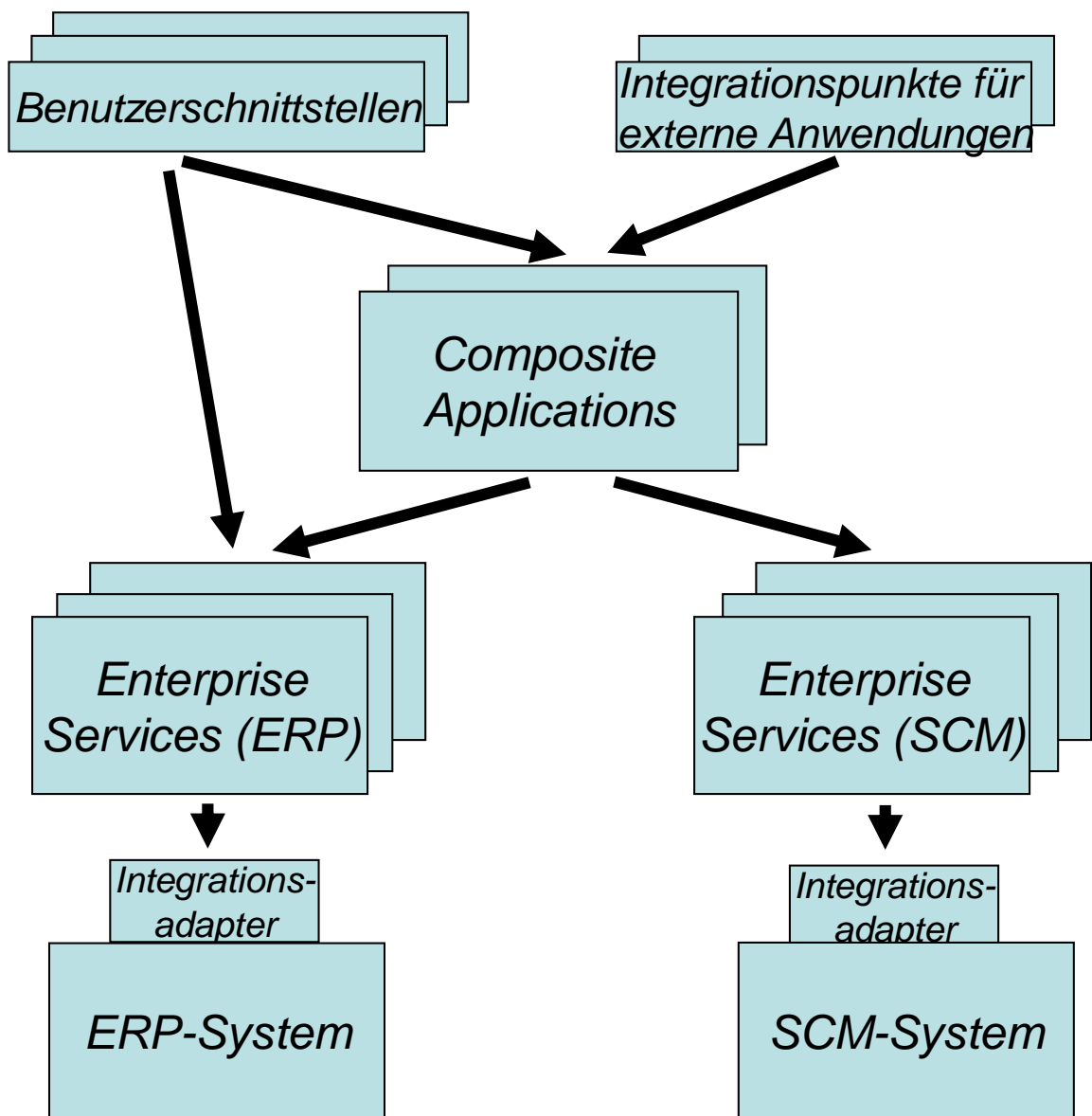


Abbildung 8: Schematische Darstellung der Enterprise Service Architecture

[Quelle: WOODS (2004), S. 34 und 46 f.]

Die Enterprise Service Architecture (ESA) stellt die von SAP vorgeschlagene Umsetzung einer SOA dar, die jedoch mithilfe einer SAP-Plattform erst individuell im Unternehmen zu implementieren ist.<sup>45</sup>

Wie die Abbildung 8 zeigt, ist es das Grundprinzip der ESA, eher monolithische Anwendungen zumindest zum Teil in Services (so genannte Enterprise

<sup>45</sup> Vgl. OEHLER (2005), S. 42

Services) zu überführen, die dann durch Kombination mit Enterprise Services anderer Systeme Composite Applications bilden.<sup>46</sup> In der Abbildung 8 stammen die Dienste aus einem ERP-System und einem Supply Chain Management System. Benutzerschnittstellen greifen sowohl auf Composite Applications oder direkt auf Enterprise Services zu und stehen zusätzlich zu den Benutzerschnittstellen innerhalb der Quellsysteme zur Verfügung.<sup>47</sup>

Bei der Einführung in einem Unternehmen erfordert insbesondere die Aufspaltung der existierenden Anwendungen, die zukünftig den Prozessen als Dienste zur Verfügung stehen sollen, ein hohes unternehmensspezifisches Wissen in Bezug auf die Interdependenz der ursprünglichen Geschäftsprozesse und der angebotenen Funktionalitäten.<sup>48</sup> Die Dienste sollen in ihrem alten Kontext weiterhin funktionieren und dabei jedoch auch den neuen Anforderungen aus der Kombination mit anderen Diensten genügen. Die zu entwickelnden Dienste für das Repository sollten folgenden Prinzipien folgen:<sup>49</sup>

- Dienste sollten in unterschiedlich strukturierten Geschäftsprozessen wieder verwendet werden können.
- Dienste sollten in geeigneter Granularität entwickelt werden. Insbesondere fraglich ist die Sinnhaftigkeit von individuell zugeschnittenen Diensten trotz hoher Ähnlichkeit der Anforderungen in unterschiedlichen Branchen. Falls hier eine Standardisierung möglich ist, kann eine höhere Anzahl von Nutzern aus unterschiedlichen Branchen erzielt werden. Dies trifft genauso auf die Frage der Generierung rein technischer Dienste zu, die aufgrund der Vielzahl nur über eine geeignete Einteilung in einem Verzeichnis noch beherrschbar wären.
- Eine geeignete Ausgestaltung von Hierarchien für die Dienstenutzung begrenzt die Komplexität des entstehenden Prozesses. Die Frage tritt vor allem bei der Nutzung von geringwertigeren Diensten durch in der Wertigkeit höher stehende Dienste auf. Die Hierarchie hilft ähnlich wie

---

<sup>46</sup> Vgl. WOODS (2004), S. 34

<sup>47</sup> Vgl. WOODS (2004), S. 34

<sup>48</sup> Vgl. OEHLER (2005), S. 42

<sup>49</sup> Vgl. OEHLER (2005), S. 42 f.

die geeignete Granularität die Implementierung zu vereinfachen, da geeignete Services leichter gefunden werden können und die entstehenden Strukturen bei der Kombination der Services weniger komplex werden. Der Begriff der Wertigkeit eines Dienstes ist Ausdruck der angebotenen Funktionalität bzw. der Komplexität der Funktion des Dienstes.

Zum Design der Enterprise Services schlägt SAP eine Methodik mit Namen Enterprise Service Community Process vor.<sup>50</sup> Die einzelnen Schritte werden in Tabelle 1 näher erläutert.

<b><i>Discovery Phase</i></b>	Berechnung des Return on Investment
<b><i>Evaluation Phase</i></b>	Projektplanung für die Umsetzung der Enterprise Service Architecture
<b><i>Implementation Phase</i></b>	Umsetzung der ESA
<b><i>Operation Phase</i></b>	Betrieb und Weiterentwicklung der ESA

*Tabelle 1: Phasen des Enterprise Community Process*

*[vgl. HUBER (2006), S. 65]*

Bei der Abrechnung der Nutzung der Softwarelizenzen stellen sich bei der service-orientierten Architektur neue Herausforderungen, die beispielsweise durch die Orientierung des Preismodells an den Nutzen, den die Software für das Unternehmen bringt, begegnet werden sollen.<sup>51</sup> Die klassische Berechnung der Lizenzkosten anhand Benutzerzahlen und Benutzertypen kann hier nicht angewendet werden, da unter Umständen die Anzahl der Nutzer aus dem Internet nicht oder nur schwer ermittelbar ist.

### **II.6.3 Service-Design**

Im Bereich des Service-Design lassen sich neben eher allgemeinen Aussagen

<sup>50</sup> Vgl. HUBER (2006), S. 65

<sup>51</sup> Vgl. BAYER (2005), S. 7



bedingt durch die Komplexität der Aufgabe weitere Richtlinien erst nach der Einführung eines Konstrukts zur Systematisierung der Dienste treffen.

Ausgangspunkt für die Betrachtung der Gestaltung von Services kann eine Reihe von allgemeinen Richtlinien sein. So ist hier zuerst die Festlegung der Granularität zu nennen. Sie sollte nicht zu fein gewählt werden, um die Anzahl der Aufrufe von Diensten für nur eine Aktion überschaubar und aus Sicht der gesamten SOA beherrschbar zu halten.<sup>52</sup> Die individuelle Beurteilung der Granularität muss im konkreten Projekt erfolgen.

Bei der Auswahl der über die Schnittstelle ausgetauschten Werte muss auf die fachliche Relevanz für die zu erfüllende Geschäftsfunktion geachtet werden, wodurch sich automatisch eine Transparenz der Implementierungsdetails aufgrund fehlender technischer Parameter ergibt.<sup>53</sup> Servicespezifische technische Parameter erschweren darüber hinaus den problemlosen Wechsel zwischen Diensten mit ansonsten ähnlicher Funktionalität.

Ein Service muss eine mehrmalige Ausführung mit identischer Wertübergabe erkennen und denselben Zustand herbeiführen, als ob nur ein Aufruf erfolgt wäre.<sup>54</sup> Diese Aussage trifft jedoch nur für zustandslose Dienste zu. Für die Durchführung des Dienstes gilt dabei die Regel, dass die implementierte Logik entweder ganz angewendet wird, oder alle vorgenommenen Änderungen nach dem Transaktionsprinzip wieder rückgängig gemacht werden.<sup>55</sup> Auf die Verwendung von zusätzlichen Nachrichten in Fehlerfällen wurde schon im Rahmen der Verfeinerung der SOA in Abschnitt II.4.3 hingewiesen.

Der Dienst soll unabhängig von dem Kontext ausgeführt werden, der den Aufruf ausgelöst hat, und somit Nebeneffekte ausschließen und für verschiedenartige Anwendungen eingesetzt werden können.<sup>56</sup>

Zur Operationalisierung und zur Vereinfachung des Service-Design löst ERL (2005) die Architekturebene in die Schichten der *Orchestration Services*, der

---

<sup>52</sup> Vgl. GUGEL (2006), S. 26

<sup>53</sup> Vgl. GUGEL (2006), S. 26

<sup>54</sup> Vgl. GUGEL (2006), S. 26

<sup>55</sup> Vgl. GUGEL (2006), S. 26

<sup>56</sup> Vgl. GUGEL (2006), S. 26

*Business Services* und der *Application Services* auf.<sup>57</sup>

Die einzelnen Ebenen bauen hierarchisch aufeinander auf. Dies bedeutet jedoch keineswegs, dass beim Zusammenbau eines höherwertigen Services nur Services aus einer direkt darunter liegenden Ebene verwendet werden dürfen.<sup>58</sup>

*Application Services* lassen sich weiter in *Utility Services*, als möglichst generisch angelegte Dienste für unterschiedliche Einsatzgebiete bei der Umsetzung der Geschäftslogik, und *Wrapper Services*, für die Kapselung eines Teils einer größeren Anwendung, unterteilen.<sup>59</sup> In der Wertigkeit höher angesiedelt sind die *Business Services*. Sie repräsentieren eine Teilfunktion der Geschäftslogik in möglichst reiner Form und entfernen sich damit weiter von der technologischen Ebene.<sup>60</sup> Die *Application Services* werden hier dazu benutzt, entweder einen aufgabenspezifischen *Business Service* für die Verwendung in einem Geschäftsprozess mit mehreren Aufgaben oder einen entitätsbezogenen Business Service (Beispiele für Entitäten wären Kunde oder auch Bestellung) zu erstellen.<sup>61</sup> Auf der letzten und komplexesten Ebene befinden sich die *Orchestration Services*, die falls sie eingesetzt werden, als Komponenten eines non-human Workflowprozesses zur Umsetzung eines ganzen Geschäftsprozesses dienen.<sup>62</sup>

Verwendet man dieses Modell als Hintergrund für das Design der Services, so ergeben sich einige Empfehlungen, die in Tabelle 2 zusammengefasst sind.

---

<sup>57</sup> Vgl. ERL (2005), S. 337

<sup>58</sup> Vgl. ERL (2005), S. 347 - 353

<sup>59</sup> Vgl. ERL (2005), S. 339

<sup>60</sup> Vgl. ERL (2005), S. 341

<sup>61</sup> Vgl. ERL (2005), S. 342

<sup>62</sup> Vgl. ERL (2005), S. 344

<b>Maßnahme beim Service-Design</b>	<b>Beschreibung/ Anwendungsgebiet</b>
Prozessübergreifende Wiederverwendung	Dies kann für aufgabenspezifische Business Services zutreffen. Um die Wiederverwendung zu fördern, sollten die Services möglichst grobkörnig modelliert werden.
Wiederverwendung innerhalb eines Prozesses	Bei mehrfachen Auftreten eines Teils an Geschäftslogik innerhalb eines Geschäftsprozesses ist das Standardisieren des Teils in Form eines Services sinnvoll.
Berücksichtigung von Abhängigkeiten in einen Prozess	Obwohl dieselbe Geschäftslogik in mehreren Prozessen verwendet werden kann, ist auf Abhängigkeiten des Dienstes von der Datenversorgung zu achten. Unterschiedliche Datenkonstellationen lassen sich unter Umständen erst durch genaue Analysen elementarer Prozessschritte erkennen.
Applikationsübergreifende Wiederverwendung	Um die Wiederverwendung von Application Services zu erhöhen, empfiehlt es sich, diese eher generisch zu halten.
Anforderungen an die Granularität	Business Services sollten daraufhin getestet werden, ob nicht ein Teil der Geschäftslogik auch für andere Einsatzbereiche benötigt wird. Im Zweifelsfall sollte der Service aufgeteilt werden.
Bildung logisch abgegrenzter Einheiten	Jeder Service sollte genau logisch abgegrenzt werden können und ein hohes Maß an Autonomie bei der Verwendung ermöglichen.
Emulierung der Orchestrierung	Zur Vorbereitung auf den Einsatz von Orchestrierungslösungen sollten schon hierarchisch übergeordnete Business Services eingesetzt werden.
Ausgewogene Gestaltung	Bei der Umsetzung der Services ist eine Abwägung zwischen Anforderungen aus dem Unternehmen, Standards und Branchenforderungen zu treffen.

---

Verhinderung des mehrfachen Implementierens der identischen Logik	Bei der Implementierung von Diensten über einen größeren Zeitraum hinweg sollten Mehrfachimplementierungen (gleiche Logik, überlappende Logik, Varianten der Logik) vermieden werden. Gegenmaßnahmen sind Analysen der Entwicklungsdokumente und Standards für das Servicedesign.
---	---

*Tabelle 2: Richtlinien für das Service-Design*

*[vgl. ERL (2005), S. 416 - 421]*

Ob die einzelnen Ebenen innerhalb der Architekturbetrachtung eine Rolle spielen, hängt vom konkreten Anwendungsfall im Service-Design ab. Des weiteren ist zu bedenken, dass die einzelnen Empfehlungen auf ihre individuelle Anwendbarkeit zu prüfen sind.

### III. Die Web Service-Technologie

Die technologische Betrachtung von Web Services beschäftigt sich zuerst mit der Charakterisierung durch eine Definition und die Identifikation von Merkmalen. Es wird außerdem der Ablauf eines Web Service Aufrufs beschrieben. Anschließend erfolgt die Einordnung von Web Services in ebenfalls für SOA-Implementierungen geeignete Technologien.

Zentrale Bedeutung haben XML und davon abgeleitete Sprachen, die zusammen im so genannten Web Service Protocol Stack für andere Anwendungen sichtbare Kommunikationsmittel darstellen. Hier wird der Grundstein für die Interoperabilität gelegt. Dies ist insbesondere für die Einbettung der Web Services in den Bereich von P2P- und Grid-Computing wichtig. Aus diesem Grund erfolgt eine überblicksmäßige Vorstellung der relevanten XML-Dialekt-Gruppen und der sich daraus ergebenden Herausforderungen.

Den Abschluss bildet die Darstellung ausgewählter angrenzender bzw. verwandter Technologien, die als Ergänzung bzw. Erweiterung der Web Service-Technologie zu bezeichnen sind.

#### III.1 Definition

Das World Wide Web Consortium als zentrale Standardisierungsstelle für Web Standards definiert einen Web Service wie folgt:

*“A web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact with the web service in a manner prescribed by its description using SOAP*

*messages, typically conveyed using HTTP with an XML serialization in conjunction with other web-related standards.*<sup>63</sup>

Die Definition betont Web Services als ein plattformunabhängig angelegtes Software-System für die Kommunikation zwischen Maschinen. Aufnahme in die Definition finden das SOAP-Protokoll zum Nachrichtenaustausch genauso, wie die WSDL für die Beschreibung der Schnittstellen. Für die Übertragung der Nachrichten wird HTTP als Trägerprotokoll explizit genannt.<sup>64</sup> Die Entdeckung der Services über UDDI wird nicht direkt erwähnt, kann aber in die angesprochenen Standards im Web eingeordnet werden. Die gegebene Definition des W3C umfasst sowohl die Nutzung von Web Services in einer *service-based architecture* und damit die reine Nutzung als andersartigen Schnittstellentyp wie auch im Rahmen der *service-oriented architecture*. Letztere fordert auch Änderungen im Design-Prozess und betont die dynamische Entdeckung der Dienste.

Als direkter Vorgänger der Web Services kann die von Hewlett-Packard entwickelte E-Speak Technologie bezeichnet werden, die Anfang 2001 als Produkt mit dem Ziel veröffentlicht wurde, eine Plattform zur Interaktion, Veröffentlichung und Entdeckung von Diensten im Internet auf XML-Basis zur Verfügung zu stellen.<sup>65</sup> Daneben sind noch BizTalk von Microsoft oder Jini von SUN Microsystems zu nennen.<sup>66</sup>

### **III.2 Merkmale von Web Services**

Web Services lassen sich mit einer Anzahl charakteristischer Eigenschaften beschreiben:<sup>67</sup>

- Sie lassen sich auf XML zurückführen. XML bildet zusammen mit XML

---

<sup>63</sup> BOOTH u. a. (2004)

<sup>64</sup> Vgl. für die letzten 3 Sätze BOOTH u. a. (2004)

<sup>65</sup> Vgl. GRAUPNER u. a. (2001), S. 279

<sup>66</sup> Vgl. BURBECK (2000)

<sup>67</sup> Vgl. CHAPPEL/ TYLER (2003), S. 2 f.

Schema und DTD die Grundlage der verwendeten Protokolle. Der Transport von Daten über XML erfolgt netzwerk-, plattform- und betriebs-systemneutral.

- Durch die Verwendung von Web Services lässt sich eine lose Kopplung realisieren. Hierbei ist nicht wie bei der engen Kopplung eine Anwendung mit der Logik genau einer anderen Anwendung verbunden, sondern es ist durchaus das Nutzen verschiedener Web Services mit leicht voneinander abweichenden Schnittstellen möglich.
- Web Services sind grobkörnig angelegt. Dies ermöglicht eine Modellierung von Diensten, die auch noch aus der Perspektive des gesamten Unternehmens eine sinnvolle Einheit bilden. Der einzelne Service besteht dann unter Umständen wieder aus fein gegliederten Methoden, die Teil der Geschäftslogik sind, die sich hinter dem Dienst verbirgt.
- Die Kommunikation kann synchron und asynchron erfolgen. Dabei ist die synchrone Kommunikation meist Zeichen der engen Kopplung zweier Anwendungen. Ein Aufruf einer entfernten vorhandenen Funktionalität bzw. Methode blockiert die Anwendung. Für lose gekoppelte Anwendungen ist die asynchrone Kommunikation jedoch zwingend erforderlich.
- Remote Procedure Calls sind über Web Services möglich. Enterprise Java Beans und .NET-Komponenten sind wichtige Bestandteile vorhandener Architekturen in Unternehmen und stellen ihre Methoden in Form von RPC-Schnittstellen zur Verfügung. Web Services bilden hierbei entweder Wrapper-Dienste durch Kapselung für oben genannte Komponenten an, oder bieten RPC-Dienste selbst an.
- Web Services ermöglichen den transparenten Austausch sowohl von einfach strukturierten Daten, als auch von komplexen Dokumenten bzw. Datenstrukturen.

Vergleicht man die Merkmale der Web Services mit den Merkmalen anderer Technologien so wird deutlich, dass es sich hierbei nicht um Alleinstellungsmerkmale handelt. Dies wird in der folgenden Betrachtung der verwandten

Technologien im Rahmen der Umsetzung der SOA deutlich.

### **III.3 Abgrenzung zu anderen Technologien**

Web Services sind eine Möglichkeit zur Implementierung einer SOA aber nicht eine Forderung, die sich aus der Architektur ergibt. Es existieren weitere mehr oder weniger geeignete Ansätze. Ausgewählte Technologien sollen im Folgenden untersucht werden.

#### **III.3.1 Common Object Request Broker Architecture (CORBA)**

Die Common Object Request Broker Architecture wurde gemeinsam mit der dazugehörigen Spezifikation von der Object Management Group (OMG) bereits im Jahre 1991 in der Version 1.0 veröffentlicht und richtet sich an Softwarehersteller, die interoperable verteilte Objektsysteme auf Basis des OMG standardisierten Object Request Brokers (ORB) entwickeln wollten.<sup>68</sup> CORBA präsentiert sich also nicht als ein fertiges Produkt, sondern als Spezifikation, die erst innerhalb eines Produkte umzusetzen ist. Als Zielsetzung wird dabei die Betriebssystem- und Programmiersprachenunabhängigkeit von Objekten bei der Realisierung verteilter Objektsysteme innerhalb einer heterogenen Systemumgebung verfolgt.<sup>69</sup>

Der Object Request Broker übernimmt die Rolle des direkten Kommunikationspartners für die Anwendungsobjekte und ermöglicht die Übertragung von Methodenaufrufen zwischen den Laufzeitumgebungen über das General bzw. Internet Inter-ORB Protocol (GIOP/ IOP) inklusive der nötigen Serialisierung bzw. Deserialisierung.<sup>70</sup> Der ORB fungiert als Hintergrundobjekt in Applikationen und stellt die verteilte Infrastruktur für eine darauf aufbauende Anwen-

---

<sup>68</sup> Vgl. OBJECT MANAGEMENT GROUP (2004)

<sup>69</sup> Vgl. EBERHART/ FISCHER (2003), S. 81 f.

<sup>70</sup> Vgl. EBERHART/ FISCHER (2003), S. 82



derung zur Verfügung.<sup>71</sup> Neben der so gewährleisteten weitgehenden Plattformunabhängigkeit begünstigt die Abstraktion der Schnittstellenbeschreibungssprache von der zu verwendenden Programmiersprache die weit reichende Unterstützung der CORBA-IDL in Form von programmiersprachenabhängigen IDL-Übersetzungstools mit Quellcode-Generatoren für die kommunikationsspezifischen Programmteile.<sup>72</sup> Um das Auffinden von durch Objekten angebotenen Funktionalitäten zu erleichtern, steht der Name Service zur Verfügung, der komplexe Adressinformationen für den Nutzer transparent gestaltet.<sup>73</sup>

Betrachtet man den von CORBA gebotenen Funktionsumfang, so entsteht die Frage, warum man daneben noch eine Technologie wie Web Services für die Realisierung einer service-orientierten Architektur benötigt? Zum einen ist hier die Spezifikationsdichte bei CORBA ins Feld zu führen, die einen hohen Komplexitätsgrad mit sich bringt und neben dem hohen Aufwand für die Implementierung oft auch zu inkompatiblen Lösungen führte.<sup>74</sup> Zum anderen hat die Dominanz der Internet-Infrastruktur, die nun auch ins Intranet Einzug gehalten hat, zum Siegeszug des Uniform Resource Identifiers (URI) gegen den CORBA-eigenen Name Service Ansatz geführt.<sup>75</sup>

Vorhandene CORBA-Lösungen weisen unter Umständen schon einen hohen Grad an Serviceorientierung auf. Diese Vorarbeit kann bei Verwendung der Web Service-Technologie über die Kapselung der CORBA-Komponenten durch Wrapper-Dienste genutzt werden. Soll in einer Lösung zwar eine SOA umgesetzt werden ohne aber mit Web Services interagieren zu wollen, so spricht nichts gegen eine reine CORBA-Lösung. Dies trifft umso mehr dann zu, falls umfangreiches CORBA Know-how im Unternehmen verfügbar ist.

---

<sup>71</sup> Vgl. LINTHICUM (2001), S. 190

<sup>72</sup> Vgl. EBERHART/ FISCHER (2003), S. 82

<sup>73</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

<sup>74</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

<sup>75</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

### III.3.2 Java Remote Method Invocation (RMI)

Java Remote Method Invocation (RMI) zielt gleichsam wie CORBA auf verteilte Objektsysteme ab, ohne aber die Heterogenität sowohl auf Plattform bzw. Betriebssystem und Programmiersprachen auszudehnen, da die Kommunikation von Programmen in JAVA Virtual Machines zentrales Anliegen des Ansatzes ist.<sup>76</sup> Die Plattformunabhängigkeit wird also durch die JAVA-Umgebung ermöglicht, die für eine Vielzahl an Betriebssystemumgebungen verfügbar ist – bei der Programmierung steht dagegen die Homogenität im Vordergrund und somit brauchen keine plattformspezifischen Releases von Produkten hergestellt werden.<sup>77</sup> Die Interface Defining Language (IDL) ist deshalb folgerichtig ein Teil des Java-Quellcodes in Form einer Abwandlung der Interface Technik für Remote-Aufrufe.<sup>78</sup>

Unter Verwendung der Schnittstellenbeschreibung kann auch bei Java RMI eine automatische Generierung der client- und serverseitigen Codeteile (STUBS und SKELETONS) vorgenommen werden, die an der Kommunikation über das Protokoll RMI-IIOP beteiligt sind.<sup>79</sup>

Ein weiterer wichtiger Baustein einer SOA ist das dynamische Suchen von Diensten. Bei Java-RMI steht dafür neben einem eigenen Registrierungsdienst in Form der so genannten RMIregistry, über das Java Naming and Directory Interface (JNDI) auch für anspruchsvolle Anwendungen mit Zugriff auf LDAP-Verzeichnisdienste eine geeignete Verzeichnisanbindung zur Verfügung.<sup>80</sup>

Auch bei RMI finden sich geeignete Ansätze einer SOA-Umgebung. Als Vor- und gleichzeitig auch als Nachteil stellt sich die Konzentration auf JAVA heraus. Zwar wird ein auf eine Vielzahl von Plattformen übertragbares Programmiermodell angeboten, die Umsetzung bleibt aber immer auf JAVA beschränkt, was den Einsatz als Standard in Bezug auf eine einheitliche Kommunikations-

---

<sup>76</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

<sup>77</sup> Vgl. WATT (2002), S. 193 f.

<sup>78</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

<sup>79</sup> Vgl. EBERHART/ FISCHER (2003), S. 83

<sup>80</sup> Vgl. EBERHART/ FISCHER (2003), S. 84 f.

infrastruktur jedoch deutlich begrenzt.<sup>81</sup>

### **III.3.3 Sun Remote Procedure Call (Sun RPC)**

Als weiteres Beispiel für Technologien, die die Umsetzung von Systemen mit service-orientierten Elementen unterstützen, kann auch die von SUN Microsystems entwickelte SUN RPC genannt werden.

Ziel war die Erweiterung des Prinzips des lokalen Funktionsaufrufs auf Funktionsaufrufe, die die eigene Systemumgebung verlassen können.<sup>82</sup> Durch die Abstraktion von der eigentlichen Übertragung über ein Netzwerk können nicht nur unterschiedliche Transportarten realisiert werden, sondern es ist auch möglich, einen Aufruf auf dem lokalen System ähnlich wie den Aufruf einer Funktion auf einem entfernten System zu behandeln.<sup>83</sup> Die Kommunikation basiert nicht mehr auf Interprozesskommunikation durch Sockets, sondern verwendet schon TCP/IP, um entfernte Schnittstellen aufzurufen, die durch die Interface Description Language External Data Representation (XDR)<sup>84</sup> beschrieben werden.<sup>85</sup>

## **III.4 Ablauf eines Web Service-Aufrufs**

In Analogie zum Aufbau der service-orientierten Architektur findet sich auch bei der konkreten Umsetzung mit Web Services das Dreieck aus Provider, Broker und Requestor. Wie die Abbildung 9 zeigt, gliedert sich die Nutzung eines Web Services unter der Voraussetzung der Nutzung des UDDI-Verzeichnisdienstes in 4 Phasen. Phase 1 wird durch den Provider initiiert, der einen Web Service bei einem Broker in dessen UDDI-Registry registriert. Die Registrierung kann

---

<sup>81</sup> Vgl. EBERHART/ FISCHER (2003), S. 84 und LINTHICUM (2001), S. 217

<sup>82</sup> Vgl. SUN MICROSYSTEMS (1995), S. 9

<sup>83</sup> Vgl. SUN MICROSYSTEMS (1995), S. 9

<sup>84</sup> Vgl. SUN MICROSYSTEMS (1995), S. 4

<sup>85</sup> Vgl. EBERHART/ FISCHER (2003), S. 85

beispielsweise unter Verwendung von JAVA Klassenbibliotheken wie UDDI4J aus einer Anwendung heraus erfolgen. Die selben Klassenbibliotheken erlauben natürlich auch das Deregistrieren des Dienstes.

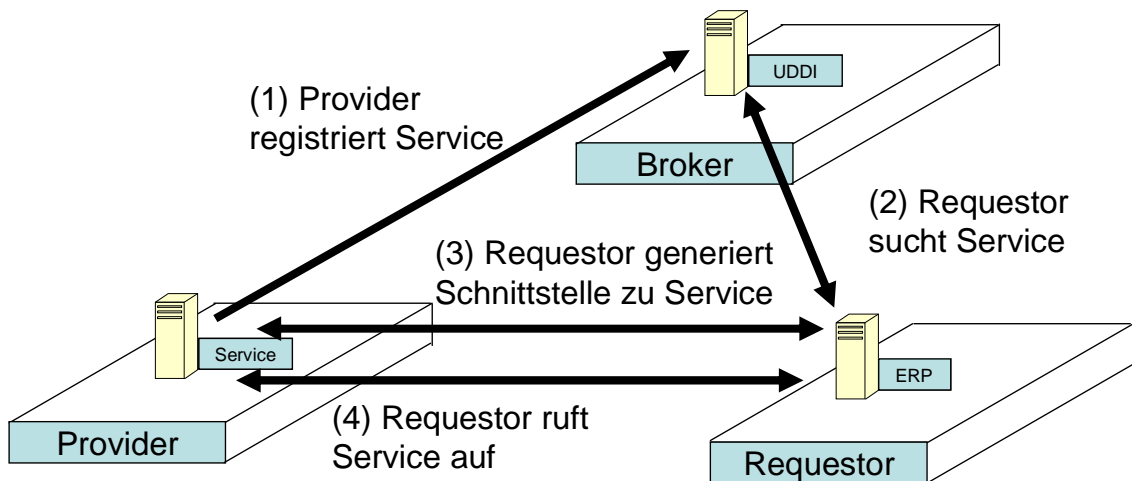


Abbildung 9: Ablauf eines Web Service-Aufrufs

Von dieser Aktion getrennt sucht ein Requestor nach einem bestimmten Dienst innerhalb der UDDI-Registry. Wurde der geeignete Dienst lokalisiert, extrahiert der Requestor technische Informationen zur Lage der WSDL-Datei, die zur Generierung der Web Service-Schnittstelle in Phase 3 verwendet werden kann. Die WSDL-Datei kann unter Verwendung geeigneter Tools wie z. B. `wsdl2java` bei der JAVA-Entwicklung zur weitgehenden Generierung der Kommunikationsinfrastruktur für einen Client verwendet werden. In einem Anwendungssystem können gleichermaßen die Datenstrukturen für den Dienst erzeugt werden und in einem als Mapping bezeichneten Vorgang mit den existierenden Datenstrukturen verbunden werden.

Die letzte Phase ist der eigentliche Aufruf des Services aus der entwickelten Anwendung bzw. aus der Web Service-Komponente eines Anwendungssystems heraus. Analog zu den Ausführungen im Rahmen der Betrachtung der SOA kann die Entdeckung des Dienstes über die UDDI-Registry entfallen, falls der Web Service oder die Lage der WSDL-Datei schon bekannt sind. Zu Zwecken der Ausfallsicherheit bei Verlagerung des Dienstes innerhalb der IT-Landschaft des Providers kann dennoch eine Anfrage an den Verzeichnisdienst

sinnvoll sein.

### III.5 Der Protocol Stack

Die Web Service-Technologie ist durch den Austausch von Nachrichten über standardisierte Protokolle gekennzeichnet. Bei der Einteilung der Protokolle wird in vielen Publikationen der Begriff des Protocol Stack verwendet. Der Stack wird dabei aus verschiedenen Protokollgruppen gebildet, wobei nicht alle Gruppen bei jeder Web Service-Implementierung genutzt werden. Die Ausgestaltung der getroffenen Klassifikationen ist dabei durchaus unterschiedlich.<sup>86</sup>

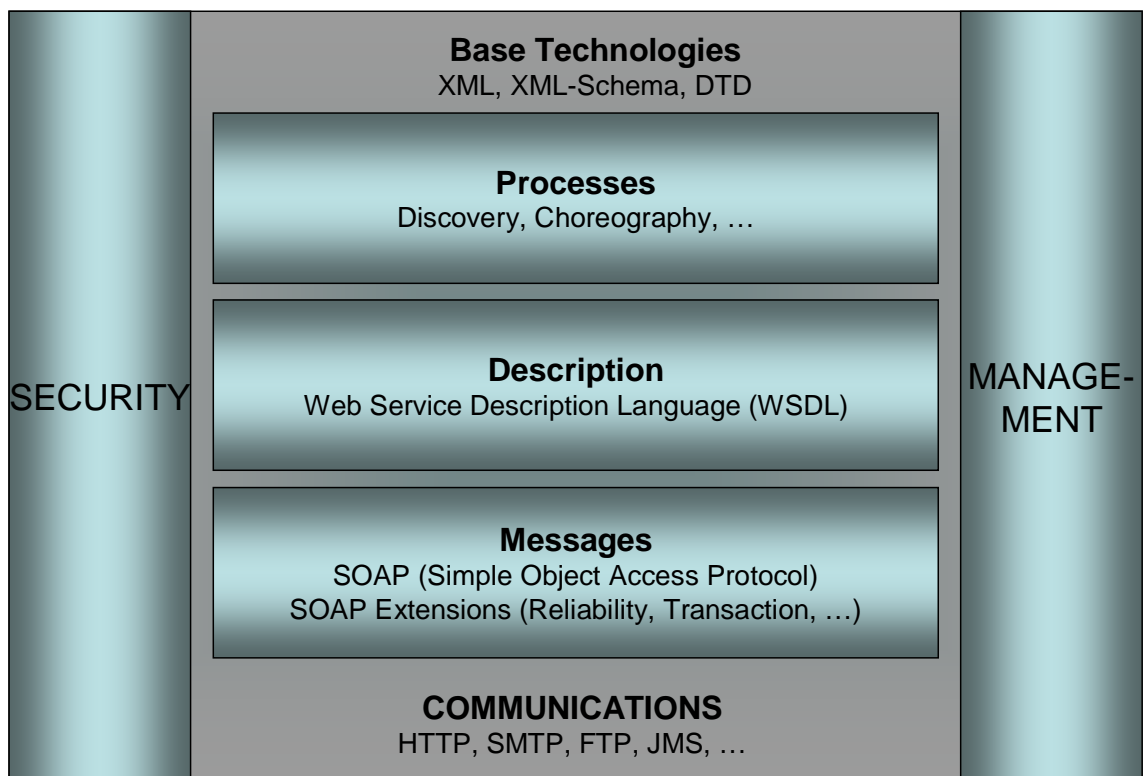


Abbildung 10: Der Web Service Protocol Stack

[Quelle: BOOTH u. a. (2004)]

Das W3C unterscheidet, wie in Abbildung 10 dargestellt, in seiner Aufglie-

<sup>86</sup> An dieser Stelle sei exemplarisch die Einteilung nach BOOTH u. a. (2004) angeführt.

derung lediglich 7 Gruppen von Protokollen. Die Grundlage für jede Web Service-Interaktion bilden Kommunikationsprotokolle (= Communications in der Abbildung 10) für die physische Durchführung der Übertragung. Daneben stehen die XML-Basistechnologien XML, XML-Schema und Document Type Definition (DTD), als technologische Grundlage für den Aufbau der restlichen Protokolle. Die Protokolle für Nachrichten (= Messages in der Abbildung 10) und Schnittstellenbeschreibung (= Description in der Abbildung 10) bilden den Kern für die Kommunikation. Flankierend stehen Protokolle aus den Bereichen des Managements von Web Services, der Sicherheit und für die Unterstützung von Geschäftsprozessen zur Verfügung.

Ausgehend von dieser Einteilung lassen sich verschiedene Technologien identifizieren, die nun einer näheren Untersuchung unterzogen werden. Im Einzelnen sind dies der Bereich der Übertragung von Nachrichten, die Beschreibung von Schnittstellen, der Verzeichnisdienst, die Unterstützung von Geschäftsprozessen und Transaktionen, der Bereich der Sicherheit und schließlich die Rolle der Semantik im Rahmen der Web Services. Der Verzeichnisdienst wird hierbei als wichtige Technologie untersucht, auch wenn er sich nicht im Protocol Stack befindet. Um den Bogen zu Web Services im Rahmen von Grid-Computing spannen zu können, wird zusätzlich das Web Services Resource Framework als Umsetzung von zustandsbehafteten Web Services erläutert. Der Bereich des Web Service Management als Folgefunktion nach der Etablierung von Web Services wird nicht ausführlich thematisiert. Es liegt hier die Management of Web Services (MOWS)-Spezifikation im Rahmen von Web Services Distributed Management (WSDM) vor, die die allgemeine Spezifikation Management Using Web Services (MUWS) für die Verwaltung von Web Services adaptiert.<sup>87</sup>

### III.5.1 Übertragung

Zur Übertragung von Nachrichten wird SOAP als zentrales Übertragungs-

---

<sup>87</sup> Vgl. MURRAY/WILSON/ ELLISON (2006), S. 3

protokoll eingesetzt.

SOAP stand ursprünglich für Simple Object Access Protocol, wird jedoch aktuell als Eigenname verwendet, da neben dem Einsatz bei Remote Procedure Calls, ein zweiter Schwerpunkt als allgemeines XML-Dokumentenaustauschformat entstanden ist.<sup>88</sup> Das W3C verfolgt mit SOAP Version 1.2 insbesondere das Ziel eines einfachen und erweiterbaren Protokolls für den Austausch von strukturierten Informationen über unterschiedliche zugrunde liegende Transportprotokolle, ohne damit ein spezielles Programmiermodell und eine Semantik für Implementierungen vorzugeben.<sup>89</sup>

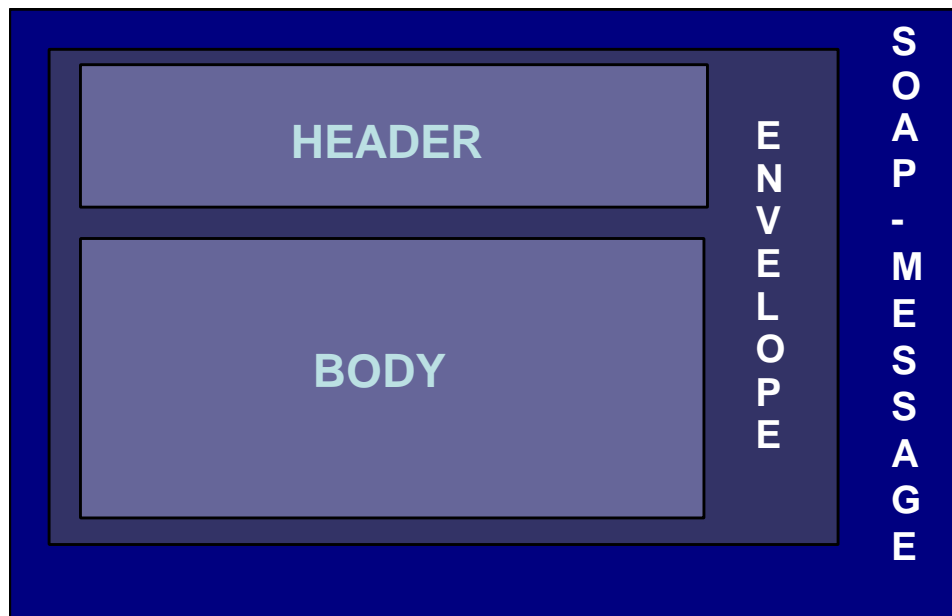


Abbildung 11: Aufbau einer SOAP-Nachricht

[Quelle: MITRA (2003)]

Der Aufbau einer SOAP-Nachricht (vgl. Abbildung 11) aus einem Envelope, der neben einem optionalen Header den Body mit den eigentlichen Nachrichten enthält, ist schon auf Nachrichtenwege ausgerichtet, die mehrere Zwischenstationen (= Intermediaries) enthalten, bis der eigentliche Empfänger erreicht

<sup>88</sup> Vgl. KÜSTER (2003), S. 7

<sup>89</sup> Vgl. GUDGIN u. a. (2003)

wird.<sup>90</sup>

### III.5.2 Beschreibung

Als Interface Description Language fungiert beim Web Service-Ansatz die Web Services Description Language (WSDL).

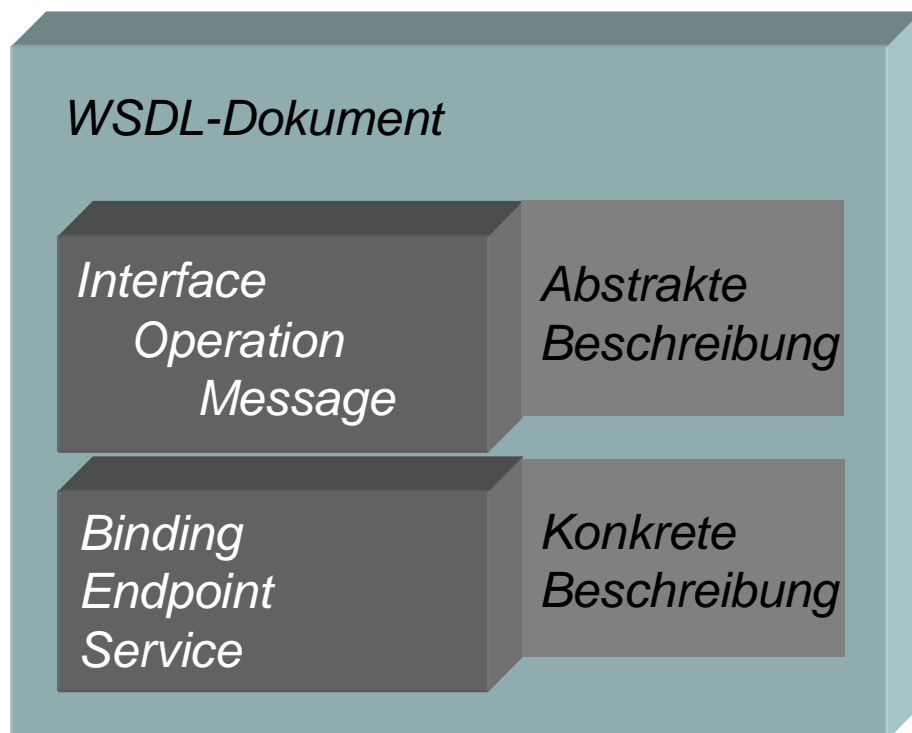


Abbildung 12: Aufbau eines WSDL-Dokumentes

[Quelle: ERL (2005), S. 134]

Ein WSDL-Dokument lässt sich dabei, wie Abbildung 12 zeigt, in einen abstrakten und einen konkreten Teil der Beschreibung einer Schnittstelle gliedern:<sup>91</sup>

- Der abstrakte Teil beschreibt das *Interface* eines Web Services losgelöst von der Technologie. Für jedes Interface werden im Abschnitt *Operations*

<sup>90</sup> Vgl. W3C (2003)

<sup>91</sup> Vgl. ERL (2005), S. 134



die verschiedenen Methoden oder Funktionen aufgelistet, die zur Verfügung stehen. Anschließend werden die eingehenden und ausgehenden Nachrichten bei *Message* spezifiziert.

- Der konkrete Teil ist für die Verbindung zur tatsächlichen Implementierung des Dienstes nötig. *Binding* beschreibt dabei die Verbindung zum Dienst in Form einer Technologie zum Transport von Nachrichten wie z. B. SOAP. Der Endpoint stellt die physische Adresse des Dienstes dar unter der mit dem Dienst kommuniziert werden kann. Das Service Element dient zur Herstellung von Querverbindungen zu anderen Endpoints.

Neben WSDL kommen im Rahmen des *Service Contracts* bei der Nutzung des Dienstes unter Umständen noch XSD Schema zur formaleren Beschreibung des Nachrichtenaufbaus, WS-Policy zur Festlegung von Regeln und Einschränkungen für automatische Geschäftsfunktionen und weitere Dokumente wie z. B. Service Level Agreements hinzu.<sup>92</sup>

### III.5.3 Verzeichnisdienst

Für die Realisierung von lose gekoppelten Diensten wird ein Verzeichnisdienst benötigt. Web Services verwenden dazu Universal Description, Discovery and Integration (UDDI). Die aktuelle Version des Standards der Organization for the Advancement of Structured Information Standards (OASIS) ist 3.0.2.

UDDI besteht aus der Beschreibung einer Struktur für ein Web Service-Verzeichnis und einer Festlegung von Programmierschnittstellen zum Publizieren, Auslesen und Verwalten von Informationen über Dienste. Bei konkreten UDDI-Verzeichnissen stellen die Schnittstellen selbst wieder Web Services dar, die mit Hilfe von SOAP-Nachrichten angesprochen werden. Ein UDDI-Verzeichnis unterstützt dabei die Suche und Katalogisierung von Organisationen, Firmen und sonstigen Web Service-Anbietern genauso wie die Bereit-

---

<sup>92</sup> Vgl. ERL (2005), S. 137 f. und S. 242

stellung von Informationen zu den von ihnen angebotenen Web Services und den dazugehörigen technischen Schnittstellen.<sup>93</sup>

Wie die Abbildung 13 zeigt, gibt es bei der Interaktion zwischen den Verzeichnisdiensten eine gewisse Hierarchie. An oberster Stelle stehen die öffentlichen Universal Business Registries (= UBR), die nicht nur für eine eindeutige Identifikation der dort publizierten Dienste sorgen, sondern die Daten auch mit allen anderen UBRs abgleichen.<sup>94</sup> Zusammengeschlossene Registries (= Affiliated in Abbildung 13) publizieren ihre Dienste genauso wie die halb-privaten Registries (= Semi-Private in Abbildung 13) von Firmen, mit dem Unterschied, dass zusammengeslossene Verzeichnisse noch nach eigenen Regeln Datenaustausch vornehmen.<sup>95</sup> Daneben existieren weiterhin rein private Registries (= Private in Abbildung 13), die keine direkte Verbindung aus dem firmeneigenen Netz heraus haben.<sup>96</sup>

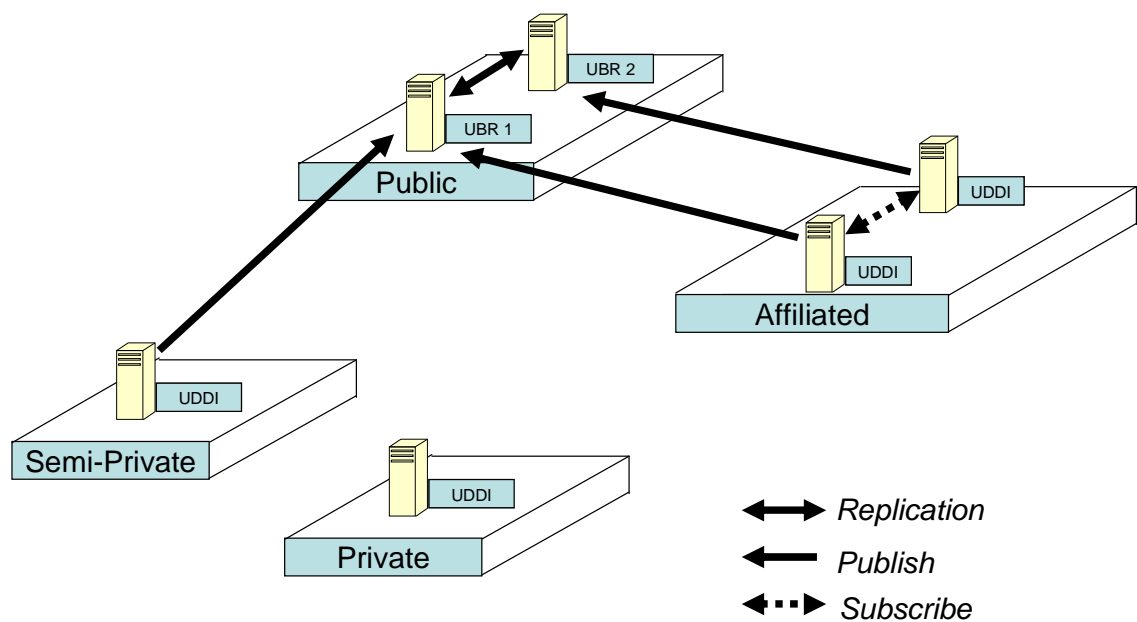


Abbildung 13: Interaktion von UDDI Registries der Version 3

[Quelle: OASIS (2004), S. 10]

<sup>93</sup> Vgl. für diesen Abschnitt OASIS (2004), S. 5

<sup>94</sup> Vgl. HAUSER/ LÖWER (2004), S. 102

<sup>95</sup> Vgl. OASIS (2004), S. 8

<sup>96</sup> Vgl. OASIS (2004), S. 8

Auf der Ebene des Datenmodells unterscheidet man bei UDDI zwischen vier großen Entitytypen:<sup>97</sup>

- Die *Business Entity* bildet die Organisationseinheit ab, die Dienste anbietet. Hierbei können über so genannte *Publisher Assertions* Verbindungen zwischen mehreren Business Entities hergestellt werden.
- Der *Business Service* stellt hierarchisch in einer 1 : N Beziehung zur Business Entity stehend, die eigentliche abstrakte Schnittstelle zum Dienst dar.
- Das *Binding Template* stellt die konkrete technische Schnittstelle zu einem Business Service dar. Hiermit wird der Dienst eindeutig adressiert.
- Mithilfe des *Technical Models* kann nun eine konkrete Schnittstellenausgestaltung in Bezug auf Datenstruktur und Operationen ebenso abgebildet werden, wie die Zuordnung zu einer Taxonomie. Es besteht eine N : M Beziehung zum Binding Template, um gleiche Schnittstellen zu Diensten abzubilden, die jedoch von verschiedenen Business Entities angeboten werden.

### III.5.4 Sicherheit

In verteilten Systemen wird Sicherheit allgemein anhand der Kriterien Vertraulichkeit, Authentizität, Integrität, Nicht-Anfechtbarkeit und Verfügbarkeit operationalisiert.<sup>98</sup> Um diesen Kriterien zu entsprechen, können bei Web Services verschiedene Maßnahmen eingesetzt werden, die wegen der Eigenschaften von Web Services an der Nachricht und der Übertragung der Nachricht ansetzen.

Bei dem Transport von SOAP Nachrichten sind dazu Protokolle zur Sicherung des Transportweges von Protokollen zur Sicherung der Nachricht zu unterscheiden. In Punkto Transportsicherheit können konventionelle Methoden wie Secure Sockets Layer (SSL) zur Absicherung der http-Übertragung verwendet

---

<sup>97</sup> Vgl. TILKOV (2004), S. 38

<sup>98</sup> Vgl. EBERHART/ FISCHER (2003), S. 260

werden, was bei einem mehrstufigen Transportvorgang dennoch zu einer Kompromittierbarkeit der Nachrichten bei den Intermediären führen kann.<sup>99</sup> Um diese Sicherheitslücke zu schließen, werden auf Nachrichtenebene Standards zur Verschlüsselung und Signierung mit digitalen Unterschriften verwendet:<sup>100</sup>

- XML-Encryption kann für die Verschlüsselung von Teilen einer Nachricht oder des gesamten Inhaltes verwendet werden.
- XML-Signature kann zur Verifizierung der Authentizität einer Nachricht durch eine in der Nachricht mitgesendete digitale Signatur verwendet werden.

Technisch gesehen bedeutet dies beispielsweise, dass der Einsatz konventioneller Firewall-Systeme nur dann ausreichend Schutz bietet, falls auch eine Filterung des Datenverkehrs auf Nachrichtenebene stattfindet.

Neben den erwähnten Spezifikationen können zur Gewährleistung der Sicherheit noch weitere herangezogen werden.

Name der Spezifikation
WS-Security
WS-SecurityPolicy
WS-Trust
WS-SecureConversation
WS-Federation
Extensible Access Control Markup Language
Extensible Rights Markup Language
XML Key Management
Security Assertion Markup Language
WS-I Basic Security Profile

*Tabelle 3: Ausgewählte Spezifikationen zur Gewährleistung der Sicherheit innerhalb einer SOA [vgl. ERL (2005), S. 257]*

<sup>99</sup> Vgl. ERL (2005), S. 262

<sup>100</sup> Vgl. ERL (2005), S. 263 f.

Eine Auswahl von Spezifikationen ist in Tabelle 3 zusammengefasst.

### III.5.5 Prozesse und Transaktionen

Bei der Gestaltung von Geschäftsprozessen aus mehreren Web Services kann die Business Process Execution Language (BPEL bzw. WS-BPEL) als wichtiger Standard angesehen werden. BPEL gestattet die Orchestrierung, also den koordinierten Aufruf mehrerer Web Services unterschiedlicher Provider, und stellt den entstehenden Prozess wieder als Web Service mit eigenen WSDL-Dokument zur Verfügung.<sup>101</sup> Es werden dabei sowohl asynchrone wie auch synchrone Aufrufe von Web Services unterstützt und für spezielle Anwendungsgebiete können Erweiterungen von BPEL angelegt werden.<sup>102</sup>

Es fehlt jedoch die Einbindung von menschlichen Prozessschritten im Rahmen des Standards, was mit dem von IBM und SAP vorgeschlagenen BPEL4People nachgeholt wird, ohne aber im Moment an die Bedeutung der vorherrschenden proprietären Standards von Business Process Management-Lösungen heranzureichen.<sup>103</sup>

Eng mit der Umsetzung von Geschäftsprozessen aus einzelnen Services verbunden ist die Frage nach der Unterstützung von Transaktionen. Transaktionen können als unterstützende Technologie für die Umsetzung der Web Service-Orchestrierung mit WS-BPEL gesehen werden.<sup>104</sup> WS-Coordination stellt dazu ein erweiterbares Framework für die Koordination von verteilten Services in heterogenen Umgebungen dar und hilft einen für die Transaktion geschaffenen Kontext zwischen den kooperierenden Services weiterzugeben.<sup>105</sup> Innerhalb des WS-Coordination Framework ermöglicht WS-BusinessActivity die Koordination von langlaufenden Transaktionen innerhalb von Geschäftstransaktionen und WS-AtomicTransaction stellt Mechanismen für kurz dauernde

---

<sup>101</sup> Vgl. MAY/ SAVUR (2006)

<sup>102</sup> Vgl. MAY/ SAVUR (2006)

<sup>103</sup> Vgl. HERRMANN (2005), S. 9

<sup>104</sup> Vgl. ERL (2005), S. 581

<sup>105</sup> Vgl. IBM (2005)

Transaktionen nach ACID-Eigenschaften bereit.<sup>106</sup>

Die ACID-Eigenschaft besagt dabei, dass auch eine verteilte Transaktion nur als Gesamtheit oder gar nicht durchgeführt wird, die Integrität und Konsistenz nicht verletzt wird, die Transaktion zurückgesetzt werden kann ohne weitere Abhängigkeiten zu beachten und die innerhalb der Transaktion vorgenommenen Änderungen dauerhaft sind.<sup>107</sup>

### III.5.6 Semantik und Ontologien

Im Rahmen von semantischen Web Services sind die Begriffe der Ontologie und der Semantik wichtig. Eine Ontologie ist in diesem Zusammenhang die explizite und formale Spezifizierung einer Konzeptualisierung im Sinne einer abstrakten Modellierung eines Phänomens aus der realen Welt durch die Identifizierung der relevanten Konzepte, die in einer Nutzergruppe verwendet wird und sich für die Verarbeitung durch Maschinen eignet.<sup>108</sup>

Ein Web Service, der den Konventionen entspricht, die in der Ontologie festgelegt wurden, wird in einer so genannten *domain of discours* in der Lage sein mit anderen Web Services zu kommunizieren, die auch der Ontologie entsprechen.<sup>109</sup>

Im Bereich der Semantik lassen sich verschiedene Ebenen bei der Nutzung in Verbindung mit Web Services unterscheiden:<sup>110</sup>

- Die Semantik auf der Ebene der Funktionalität lässt sich auf Basis der ein- und ausgehenden Nachrichten zumindest ungefähr festlegen.
- Die Semantik auf der Datenebene geht über die jeweils schon bekannten Datentypen und Bezeichnungen der Datenelemente hinaus. Insbesondere ist die Bedeutung der einzelnen Daten für das Mapping der

---

<sup>106</sup> Vgl. IBM (2005)

<sup>107</sup> Vgl. DADAM (1996), S. 185 f.

<sup>108</sup> Vgl. GRUBER (1993), S. 1 f. und CARDOSO/ SHETH (2005), S. 3

<sup>109</sup> Vgl. CARDOSO/ SHETH (2005), S. 3

<sup>110</sup> Vgl. CARDOSO/ SHETH (2005), S. 5 f.

Datenstrukturen zwischen dem aufrufenden System und dem Service wichtig.

- Die Semantik in Bezug auf Quality of Service ermöglicht die Auswahl eines Dienstes in Abhängigkeit von den Anforderungen.
- Die Semantik bei der Ausführung des Dienstes ermöglicht beispielsweise das Festlegen und Prüfen der Nachrichtenfolge, der Kommunikationsmuster, der Abfolge der Aktionen und Voraussetzungen für den Aufruf eines Dienstes.

Zur Nutzung von semantischen Informationen im Rahmen der halbautomatischen Komposition von Web Services sei auf das Adaptive Service Grid-Projekt verwiesen.

### III.5.7 Unterstützung von Zuständen bei Web Services

Mit dem Ziel einer standardisierten Unterstützung der Umsetzung von zustandsbehafteten (= stateful) Web Services, wurde das Web Services Resource Framework (WSRF) entwickelt.

Um die Beziehung von Zuständen zu einem Service zu verdeutlichen, lassen sich drei Fälle unterscheiden:<sup>111</sup>

1. Bei einem zustandslosen Service (*stateless service*) werden nur Daten für die Durchführung des Dienstes verwendet, die innerhalb der gesendeten Nachrichten enthalten sind. Der Dienst bezieht keinerlei Wissen aus früheren Aufrufen und liefert nur Daten zurück, die sich aus der Verarbeitung der gelieferten Daten ergeben.
2. Bei einem *conversational service* hängt die Reaktion bzw. die durchgeführte Operation von der Nachricht und den Nachrichten davor ab. Das Verhalten des Dienstes wird also von jeder einzelnen Nachricht aus einer logisch geordneten Kette von Nachrichten beeinflusst. Dies entspricht beispielsweise einer Session beim Einkauf in einem Online-

---

<sup>111</sup> Vgl. FOSTER u. a. (2004), S. 8

Shop.

3. Ein Dienst, der auf einer zustandsbehafteten Ressource aufbaut, erlaubt neben dem Zugriff auch die Manipulation der Ressourcen über Nachrichten. Der Zustand wird dabei ausschließlich über die Ressource abgebildet.

Grundsätzlich ist die zustandslose Gestaltung der Services aus Gesichtspunkten der Verlässlichkeit zu begrüßen, da keine Rücksicht auf Daten aus früheren Aufrufen genommen werden muss.<sup>112</sup> Weiterhin kann der Aufruf eines Service als abgeschlossene Einheit betrachtet werden, was die Skalierbarkeit stärkt.<sup>113</sup> Dennoch sind für bestimmte Anwendungsgebiete die Abbildung des Zustandes (*state*) wichtig.

Eine zustandsbehaftete Ressource lässt sich wie folgt beschreiben:<sup>114</sup>

- Die Daten, die den Zustand repräsentieren, lassen sich als XML-Dokument darstellen.
- Es existiert ein Lebenszykluskonzept für die Ressource.
- Ein oder mehrere Web Services haben von der Existenz der Ressource Kenntnis und greifen darauf zu.

Im einfachsten Fall ist eine zustandsbehaftete Ressource schon für die Umsetzung einer Session erforderlich und ermöglicht so eine effektivere Kommunikation, da nicht die gesamte Informationshistorie bei jedem Aufruf mitgesendet werden muss.<sup>115</sup>

Zur Umsetzung von *stateful web services* innerhalb des Web Services Resource Framework dienen 5 Spezifikationen, die in Tabelle 4 dargestellt werden. Zusätzlich dazu wird das durch den Standard WS-Adressing eingeführte Konstrukt der *Endpoint Reference* benötigt, der eine eindeutige Verbindung zu der Bereitstellungsadresse eines Web Service und hierbei im

---

<sup>112</sup> Vgl. FOSTER u. a. (2004), S. 10

<sup>113</sup> Vgl. HE (2003)

<sup>114</sup> Vgl. FOSTER u. a. (2004), S. 10

<sup>115</sup> Vgl. HE (2003)



Speziellen zu einer WS-Resource liefert.<sup>116</sup> Als WS-Resource wird die Kombination aus einem Web Service und einem XML-Dokument mit der Zustandsbeschreibung bezeichnet.<sup>117</sup>

<b>WS-ResourceLifetime</b>	... spezifiziert Mechanismen und Nachrichten, um eine WS-Resource, die durch eine so genannte <i>WS-Resource Factory</i> erstellt wurde, sofort bzw. zu einem bestimmten Zeitpunkt zu zerstören. Als WS-Resource Factory wird dabei ein Web Service bezeichnet, der Instanzen einer WS-Resource erzeugen kann.
<b>WS-ResourceProperties</b>	... spezifiziert die Definition der WS-Resource und legt das Auffinden, Ändern und Löschen von Eigenschaften der WS-Resource fest.
<b>WS-RenewableReferences</b>	... ermöglicht das Auffinden einer Endpoint Reference für den Fall, dass die bisherige Endpoint Reference ungültig geworden ist.
<b>WS-ServiceGroup</b>	Schnittstelle zu einer heterogenen Zusammenfassung von Web Services
<b>WS-BaseFaults</b>	... regelt die standardisierte Fehler-rückmeldung beim Nachrichtenaustausch.

*Tabelle 4: Spezifikationen des Web Services Resource Frameworks  
[vgl. CZAJKOWSKI u.a. (2004), S. 6 f.]*

### III.5.8 Herausforderungen aus der Vielzahl an Protokollen

Bei der Beschäftigung mit Web Services fällt zuerst die große Zahl an sich ergänzenden, aber auch konkurrierenden Protokollen auf. Dabei gilt es weiterhin, zwischen Spezifikationen, also nur Beschreibungen einer Technologie, und

<sup>116</sup> Vgl. CZAJKOWSKI u. a. (2004), S. 7

<sup>117</sup> Vgl. CZAJKOWSKI u. a. (2004), S. 7

Standards, also Spezifikationen, die sich im Wettbewerb mit anderen Spezifikationen durchgesetzt haben, zu unterscheiden.<sup>118</sup> Charakteristisch für die Festlegung der Standards ist die Rolle von Standardisierungsorganisationen, die zumeist vonseiten namhafter Softwarehersteller getragen werden. Das Interesse der Hersteller an diesen Gremien begründet sich in der Beeinflussung bzw. Erarbeitung von Standards, die für die Interoperabilität von Softwareprodukten wichtig sind. Des Weiteren ermöglicht die Mitarbeit den Erwerb von Know-how. Spezifikationen für spezielle Anwendungsbereiche werden auch im Rahmen der universitären Forschung erarbeitet. Die Tabelle 5 zeigt eine Auswahl der wichtigsten Standardisierungsgremien.

Standardisierungsgremium	URL	Wichtige Standards
World Wide Web Consortium (W3C)	<a href="http://www.w3c.org">http://www.w3c.org</a>	SOAP, WSDL
Internet Engineering Task Force (IETF)	<a href="http://www.ietf.org">http://www.ietf.org</a>	HTTP
Organization for the Advancement of Structured Information Standards (OASIS)	<a href="http://www.oasis-open.org">http://www.oasis-open.org</a>	ebXML, UDDI
Web Service Interoperability Organization (WS-I)	<a href="http://www.ws-i.org">http://www.ws-i.org</a>	Basic Profile

*Tabelle 5: Bedeutende Standardisierungsgremien für Web Service-Standards  
[vgl. HAUSER/ LÖWER (2004) S. 20-23]*

Jede dieser Organisationen klassifiziert den Reifegrad einer Spezifikation nach eigenen Gesichtspunkten. Das W3C beispielsweise verwendet dazu eine Skala, die von der *Note* über *Working Draft*, *Candidate Recommendation* und *Proposed Recommendation* bis zu der *Recommendation* als höchster von einer Spezifikation zu erreichenden Stufe reicht, die sie damit zu W3C-Standard

<sup>118</sup> Vgl. HAUSER/ LÖWER (2004), S. 17

erhebt.<sup>119</sup>

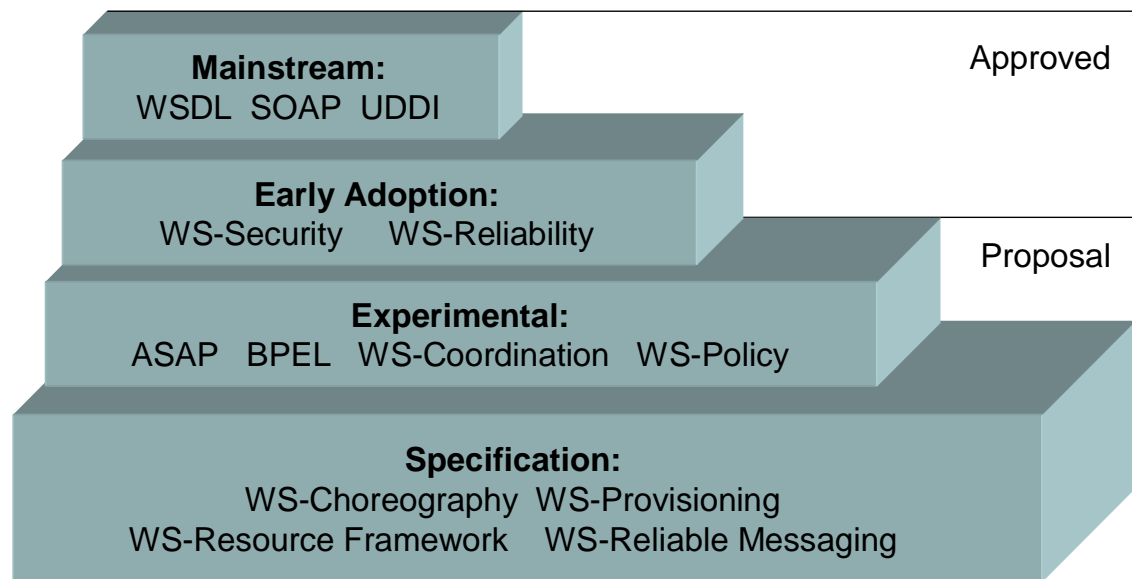


Abbildung 14: Adaptionprozess der Standards des Web Service Protocol Stack [Quelle: WILKES (2005)]

Um ein einheitliches Bild über den Reifegrad von Spezifikationen im Standardisierungsprozess, bzw. im Rahmen des Adaptionprozesses in der Praxis zu ermöglichen, unterscheidet WILKES (2005) zwischen Proposals, also noch im Standardisierungsprozess stehende Spezifikationen und Approved, also von einem Standardisierungsgremium schon verabschiedeten Spezifikationen.<sup>120</sup> Abbildung 14 zeigt ausgewählte Standards des Web Service Protocol Stack im Klassifikationsschema nach WILKES (2005).

Proposals können weiter in reine Dokumente (= Specification) und in Spezifikationen mit Proof-of-Concept Implementierungen (= Experimental) unterschieden werden. Für die Praxis relevant sind eher Spezifikationen, die im Rahmen der Early Adoption Phase von mindestens einem Hersteller in Produkte integriert, oder von einem bedeutenderen Unternehmen im Rahmen eines Projektes umgesetzt wurden. Mainstream charakterisiert schließlich einen

<sup>119</sup> Vgl. HAUSER/ LÖWER (2004), S. 20

<sup>120</sup> Vgl. WILKES (2005)

allgemein anerkannten und verabschiedeten Standard.<sup>121</sup>

Soll also ein Web Service-Projekt angegangen werden, ist zuerst eine Analyse der notwendigen Technologien auf Nachrichtenebene nötig, bevor man sich einen Überblick über deren Reifegrad macht. Selbst nach einer Implementierung bleibt ein ständiges Weiterbeobachten des Standardisierungsprozesses nicht aus, da sich bei einer Änderung an einem akzeptierten Standard eine neue Version desselben herausbilden kann, die auf Relevanz für das Produkt zu prüfen ist.

Wird ein Standard von einem Produkt unterstützt, so kann es immer noch zu Problemen bei der Interoperabilität kommen. Die Schwierigkeiten können durch Ungenauigkeiten bei der Implementierung genauso bedingt sein, wie durch herstellerspezifische Adaptionen, die unter Umständen durch spezielle Erfordernisse des Produktes bedingt sind. Umfangreiche Tests mit allen an der Kommunikation beteiligten Systemen sind hier unabdingbar. Diese Tests müssen eine Überprüfung der Ergebnisse der Nachrichten bei der Verarbeitung innerhalb der nach außen als Blackbox implementierten Logik des Web Services umfassen.

### ***III.6 Verwandte Technologien***

Im Umfeld von Web Services finden sich einige weitere Technologien, die entweder Einzug in die Entwicklung von Web Service-Lösungen halten, oder die mit Web Services zusammenwachsen. Namentlich werden hier Grid-Computing und Peer-to-Peer-Computing angesprochen und auf die Beziehung zu Web Services untersucht. Weitere Anknüpfungspunkte beispielsweise durch das Semantic Web werden in der Arbeit nicht betrachtet. In Bezug auf die Zielsetzung ist zuerst die Umsetzung von Basistechnologien wichtig. Der Grad an Dynamik, der durch die automatische Komposition von Diensten aufgrund von semantischen Eigenschaften und durch das automatische Mapping von Datenstrukturen entsteht, sollte erst nach der zuverlässigen Einführung einer

---

<sup>121</sup> Vgl. für diesen Absatz WILKES (2005)

service-orientierten Architektur angegangen werden. Hierbei sind neben technischen Herausforderungen auch Fragestellungen zu beantworten, die sich aus der Mentalität im Land des Unternehmens ergeben. Auf die Bedeutung der Mentalität auf die Entwicklung eines neuen Marktes weisen CUNNINGHAM/FRÖSCHL (1995) für den Bereich des Outsourcing hin. Unterschiede in der Mentalität bzw. der Outsourcingbereitschaft haben die Entwicklung des Outsourcing in Europa im Vergleich zur USA in den 90er Jahren des letzten Jahrhunderts behindert.<sup>122</sup> Festmachen ließen sich die unterschiedlichen Sichtweisen an der positiven Haltung zu Outsourcing aufgrund langjähriger Erfahrung in den USA und der Regulierungsdichte in einer wichtigen Branche wie der Telekommunikation auf dem europäischen Markt.<sup>123</sup>

Exemplarisch ist hier weiterhin die unterschiedliche Akzeptanz von autonomen Entscheidungen durch technische Systeme ohne menschliche Eingriffe hinzuweisen. In diesem Zusammenhang betont SESTER (2004) die notwendige Entwicklung einer gesicherten Rechtslage im Bereich der Vertragsabschlüsse durch Software-Systeme.<sup>124</sup>

### III.6.1 Grid-Computing

Foster/ Kesselmann (2004) definieren ein Grid folgendermaßen:

*„We define a GRID as a system that coordinates distributed resources using standard, open, general-purpose protocols and interfaces to deliver nontrivial qualities of service.“*<sup>125</sup>

Bei der Definition sind einige Punkte noch zu präzisieren:<sup>126</sup>

- Die verteilten Ressourcen gehören zu verschiedenen Verantwortungsbereichen, etwa Bereichen eines Unternehmens, mit getrennter Admini-

---

<sup>122</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 18

<sup>123</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 18

<sup>124</sup> Vgl. SESTER (2004), S. 311

<sup>125</sup> FOSTER/ KESSELMANN (2004), S. 46

<sup>126</sup> Vgl. FOSTER/ KESSELMANN (2004), S. 46

stration oder sogar unterschiedlichen Unternehmen. Die Koordination erstreckt sich beispielsweise auf Fragen der Sicherheit, der Abrechnung und der Zuordnung der Ressourcen.

- Der Begriff der `nontrivial qualities of service` bezieht sich auf die Annahme, dass das Zusammenwirken der Bestandteile des Grids einen höheren Nutzen für den Nachfrager des Dienstes hat, als wenn er die einzelnen Komponenten getrennt nutzen würde.

Ohne die obige Definition zu verlassen, kann man zumindest drei Arten von Grid Implementierungen unterscheiden:<sup>127</sup>

- Für die Durchführung rechenintensiver Aufgabenstellungen ist ein Computational Grid geeignet. Als Beispiel hierfür lassen sich interaktive Simulationen oder Berechnungen aus dem Bereich der Ingenieurwissenschaften anführen.
- Das Data Grid unterstützt die Anwender bei Aufgaben wie z. B. der Auswertung komplexer Experimente, die große Datenmengen erfordern bzw. mit großen Datenmengen umgehen müssen.
- Collaboration Grids adressieren die Zusammenarbeit im Rahmen der Nutzung verteilter Ressourcen, etwa bei der gemeinsamen Nutzung technischer Einrichtungen (z. B. Mikroskope, Röntgengeräte) oder auch bei Experimenten aus dem ingenieurwissenschaftlichen Bereich zur Messung der Strukturfestigkeit von Bauwerken.

Die gebildeten Kategorien sind durch die Herkunft der Grids aus dem wissenschaftlich, technischen Bereich geprägt. Auf Unternehmen lassen sich die Anwendungsgebiete leicht im Kontext von Forschungs- und Entwicklungsabteilungen übertragen. Für sonstige Einsatzgebiete im Unternehmensumfeld wurde von der Enterprise Grid Alliance (EGA) der Begriff der Enterprise Grids geprägt. Ein Enterprise Grid zeichnet sich dabei dadurch aus, dass es sich um ein Grid aus vernetzten unternehmenseigenen oder fremden Ressourcen handelt, das von einer Unternehmung betrieben wird, wobei sich die Grenzen des Grids durch den direkten Einflussbereich der Unternehmung auf die

---

<sup>127</sup> Vgl. GLOBUS ALLIANCE (o.J.)

Ressourcen ergeben.<sup>128</sup>

### III.6.2 Peer-to-Peer Computing

Peer-to-Peer-Systeme sind durch die Eigenschaft jedes Peers gekennzeichnet, sowohl Client als auch Server sein zu können.<sup>129</sup> Daneben lassen sich abhängig von der Ausgestaltung des P2P-Systems weitere charakteristische Merkmale ableiten:<sup>130</sup>

- Ein P2P-Netz ist dezentral organisiert. Es werden tendenziell weder Daten zentral vorgehalten, noch koordinierend aus Sicht eines übergeordneten Knotens auf die restlichen Knoten eingegriffen.
- Die Peers organisieren sich innerhalb vorgegebener Verhaltensweisen in ihrer Nachbarschaft ohne regelnden Eingriff selbst und bestimmen dadurch das Verhalten des gesamten Netzes.
- Weder der einzelne Peer noch die Verbindung zwischen den Peers kann garantiert werden. Dadurch müssen ausgereifte Replikationsmechanismen eingesetzt werden, um das Netz vor dem kompletten Ausfall zu schützen.
- Die Daten eines P2P-Netzes müssen zwar nicht bei jedem Knoten verfügbar sein, aber jeder Knoten muss einen Knoten erreichen können, der auf die gewünschten Daten zugreifen kann.

Für den Einsatz von P2P-Technologien sprechen positive Eigenschaften, die aus den vorgestellten Merkmalen erwachsen:<sup>131</sup>

- In einem P2P-Netz ergeben sich seltener Engpässe bei bestimmten Ressourcen, da gleichartige Ressourcen an mehreren Stellen angeboten werden.

---

<sup>128</sup> Vgl. BECKERLE (2005), S. 11

<sup>129</sup> Vgl. HAUSWIRTH/ DUSTDAR (2005), S. 5

<sup>130</sup> Vgl. HAUSWIRTH/ DUSTDAR (2005), S. 5 f.

<sup>131</sup> Vgl. CHAPPEL/ TYLER (2003), S. 27 f.

- Die Ressourcen werden von mehreren Peers angeboten und weisen somit eine höhere Ausfallsicherheit in Bezug auf das gesamte System auf.
- Durch das höhere Angebot an gleichartigen Ressourcen kann auf Peers mit besserer Verbindungsgeschwindigkeit zurückgegriffen werden.

Die vorher genannten Eigenschaften und Charakteristika müssen nicht bei jedem P2P-Netz in gleicher Weise ausgeprägt sein.<sup>132</sup> Um tatsächliche P2P-Systeme einteilen zu können, können Merkmale wie der Grad der Strukturierung (= Umfang der Informationen über benachbarte Peers in den einzelnen Knoten), die Ausgestaltung der Hierarchie (= Grad der Gleichartigkeit der Peers) und den Kopplungsgrad (= die Anzahl der möglichen von Peers gebildeten Populationen) herangezogen werden.<sup>133</sup>

### III.6.3 Konvergenz

Die vorher vorgestellten Technologien wurden wegen der Eigenschaft der Konvergenz zwischen ihnen ausgewählt. CARR (2005) betrachtet dies aus dem strategischen Blickwinkel. Er vergleicht den Weg der Web Service und Grid Technologien in Analogie zu der Entwicklung, die bei Stromnetzen nach der Erfindung der Umspannwerke, und damit zur besseren Transportierbarkeit von Strom, eingesetzt hat.<sup>134</sup> Er kommt zu folgendem Schluss:

*„Three advances – virtualization, grid computing and Web Services – are of particular importance, although their significance has often been obscured by arcane terminology to describe them. In different ways, these three technologies play a role similar to that of the early current converters: They*

---

<sup>132</sup> Vgl. HAUSWIRTH/ DUSTDAR (2005) S. 6

<sup>133</sup> Vgl. HAUSWIRTH/ DUSTDAR (2005), S. 7

<sup>134</sup> Vgl. CARR (2005), S. 71



*enable a large, tightly integrated system to be constructed out of heterogeneous and previously incompatible components.*<sup>135</sup>

Seiner Meinung nach werden die verbindenden Punkte durch ein Begriffsdickicht verschleiert. Durch die Verbindung der Ressourcen entsteht ein neues Netz oberhalb der derzeit schon bestehenden Infrastruktur aus bis dato nicht kompatiblen Technologien. Somit wird es möglich, in einer heterogenen Systemlandschaft, die ja für das Internet kennzeichnend ist, nicht nur zu kommunizieren, sondern sogar bisher inkompatible Ressourcen zu verbinden. Die konvergierenden Technologien profitieren dabei voneinander selbst dann, wenn sie nur einige Züge der anderen Technologien übernehmen und so eine existente Lücke schließen.<sup>136</sup>

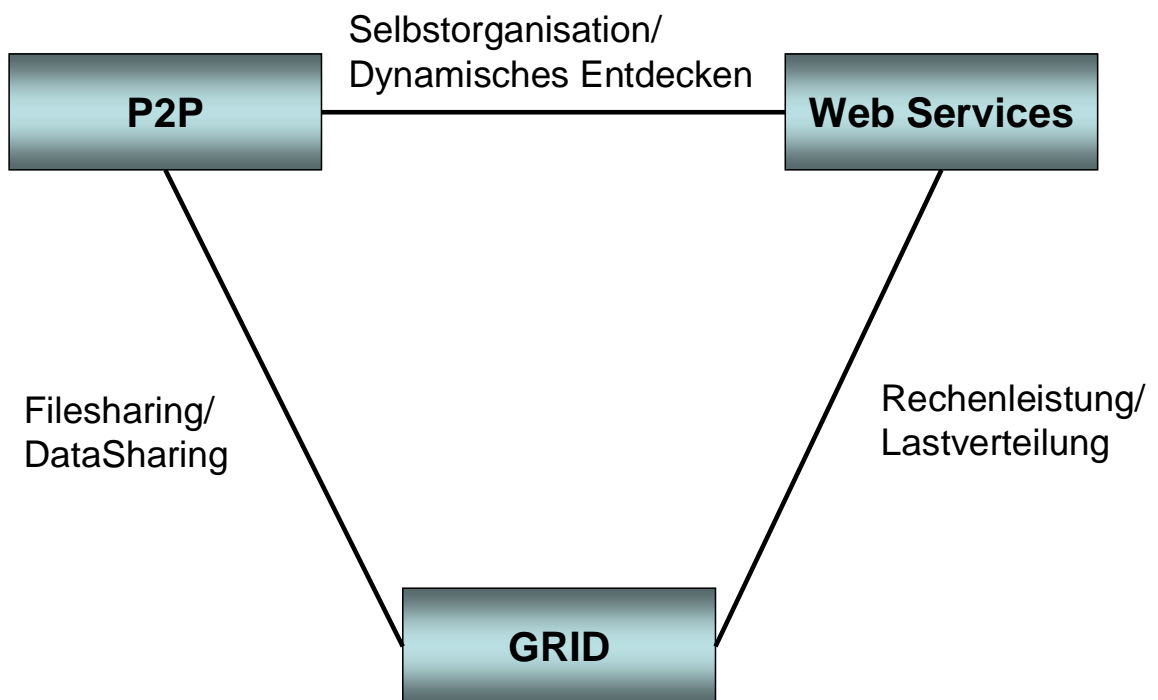


Abbildung 15: Konvergenz zwischen Technologien

In Abbildung 15 finden sich einige Konvergenzpunkte zwischen den Technologien Grid-Computing, Web Services und Peer-to-Peer. Anknüpfungspunkte

<sup>135</sup> CARR (2005), S. 71

<sup>136</sup> Vgl. für diesen Absatz CARR (2005), S. 71

punkte zum Semantic Web werden, wie schon angesprochen, nicht in die weitere Betrachtung einbezogen.

### III.6.3.1 Annäherung zwischen Web Services und Grid Computing

Die Chance aus der Konvergenz zwischen Grid-Computing und Web Services liegt einerseits auf Seiten der Web Services in der Nutzung größerer, wenn auch verteilt verfügbarer Rechenleistung begründet, die durch ein Grid koordiniert werden. Andererseits profitieren die Grid-Computing-Umgebungen beispielsweise von der Möglichkeit der Integration auf Basis von Standards, die in Unternehmen mit dem Ziel des Aufbaus einer service-orientierten Architektur schon vorhanden sind.

Der Verlauf der Annäherung lässt sich am Beispiel der Entwicklung verschiedener Web Standards und ihrer Anwendung im Globus Toolkit als einer Referenzimplementierungsplattform eines Grids zeigen:

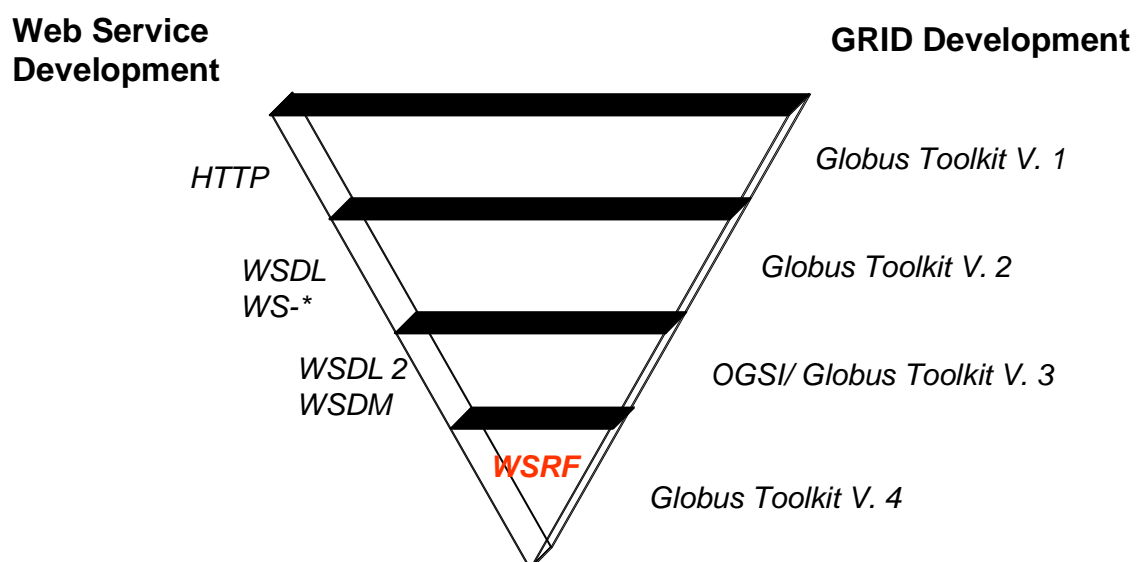


Abbildung 16: Konvergenz in den Entwicklungsrichtungen bei Web Services und Grid Computing [Quelle: GLOBUS ALLIANCE (o.J.)]

Während die ersten Versionen des Globus Toolkits sich nur teilweise der Standards aus dem Web Service-Bereich bedienen, ohne dass die Entwicklung

beider Bereiche koordiniert vorangetrieben wurde, wurde mit der gemeinsamen Entwicklung des Web Services Resource Frameworks die Basis für die enge Interoperabilität von Grid- und Web Service-Umgebungen geschaffen.<sup>137</sup> Wie die Abbildung 16 zeigt, ist die Annäherung mit dem Web Service Resources Framework nahezu abgeschlossen.

### III.6.3.2 *Berührungspunkte von Grid-Computing und P2P-Netzen*

Integrationspunkte zwischen Grid-Computing und P2P zeigen sich deutlich, da beide Ansätze:<sup>138</sup>

- sich mit der Organisation von Ressourcen in virtuellen Organisationen beschäftigen,
- dieses Problem mithilfe einer übergeordneten Struktur angehen, die die schon vorhandenen Strukturen innerhalb des Netzes nicht beeinflusst,
- und dabei bei beiden Ansätzen eigene Stärken und Schwächen deutlich werden, die aber in ihrer Ausprägung durchaus als komplementär zu bezeichnen sind.

Wie folgende Tabelle zeigt, werden von den beiden Ansätzen spezielle Punkte durchaus unterschiedlich angegangen:

	<b>GRID</b>	<b>Peer-to-Peer</b>
<b>Community</b>	Die Community aus dem wissenschaftlichen und kommerziellen Umfeld ist an der Schaffung der Infrastruktur interessiert und versucht Vertrauen und Sanktionsmechanismen aufzubauen.	Die Community besteht aus Individualisten mit wenig Neigung sich an kooperativen Verhaltensweisen zu beteiligen

<sup>137</sup> Vgl. GLOBUS ALLIANCE (o.J.)

<sup>138</sup> Vgl. FOSTER/ IAMNITCHI (2003), S. 1

<b>Ressourcen</b>	Es werden zumeist leistungsfähige Ressourcen in einem engeren Netz verbunden.	Weniger leistungsfähige Ressourcen werden auf unterschiedlichste Dauer mit dem Netz verbunden.
<b>Anwendungen</b>	Anwendungen stammen aus dem wissenschaftlichen und kommerziellen Umfeld.	Die Anwendungen dienen zur Lösung ganz spezieller Probleme der gemeinsamen Nutzung von Ressourcen.
<b>Umfang der Community</b>	Grids entstehen meist aus einer begrenzten Zahl von Netzknoten, die aber einen hohen Aktivitätsgrad aufweisen (z. B. hohen Datentransfer).	Die Community kann Millionen von Knoten umfassen, die durch die Selbstorganisation verwaltet werden können.
<b>Infrastruktur</b>	Es wird auf den Aufbau einer dauerhaften und robusten Infrastruktur für unterschiedlichste Anwendungszwecke gearbeitet.	Ziel ist die Integration von einfachen Ressourcen für ganz spezielle Zwecke.

*Tabelle 6: Vergleich von Grid und Peer-to-Peer Systemen [vgl. FOSTER/ IAMNITCHI (2003), S. 2-4]*

Im Bereich des verteilten Rechnens haben P2P-Anwendungen wie SETI@home massiv verteilte Rechenleistung genutzt. Im Unterschied zur Grid-Herangehensweise werden hier jedoch die Aufgaben in kleine Aufgabenpakete zerlegt, die auf vergleichsweise leistungsschwachen Rechnern gelöst werden. Ein Grid würde die Rechner zusätzlich unter eine gemeinsame Kontrollinstanz stellen.

Besondere Bedeutung erlangen im Bereich des Data Sharing dabei die Eigenschaft der guten Skalierbarkeit von P2P-Netzen und die dynamische Gestalt durch die Möglichkeit des Hinzufügens und Entfernens von Peers, ohne

das restliche Netz zu gefährden.<sup>139</sup> Data Sharing bezeichnet dabei den Austausch bzw. die gemeinsame Nutzung von großen Datenbeständen im Unterschied zum File Sharing, wo der Austausch einzelner Dateien wichtig ist. Bei Grid-Anwendungen spielt das Data Sharing beispielsweise im Rahmen der Auswertung komplexer Versuche eine Rolle.

### **III.6.3.3 Beziehungen von P2P-Umgebungen und Web Services**

Sowohl im P2P-Ansatz wie auch bei Web Services spielt die Selbstorganisation der Netze eine große Rolle. Selbstorganisation wird als Bildung eines Systems und Entwicklung eines Systemverhaltens verstanden, das durch das Verhalten der individuellen Komponenten des Systems hervorgerufen wird, die dynamisch miteinander agieren.<sup>140</sup>

Bei der Gestaltung der Selbstorganisation werden jedoch unterschiedliche Ansätze verfolgt. Während bei Web Services, abgeleitet aus der service-orientierten Architektur, Diensten verschiedene Rollen zugeordnet werden (Provider, Requestor und Broker), wird bei P2P gerade darauf verzichtet – jeder Peer übernimmt eine ihm bekannte Funktion, oder nutzt die Funktionen von erreichbaren Peers.<sup>141</sup> Eine direkte Übertragung der hinter den Web Services stehenden SOA könnte somit ein Service sein, der sowohl einen Dienst anbietet als auch Dienste nutzen kann. Zusätzlich hält der Dienst ein Verzeichnis mit angrenzenden Services vor, auf das andere Dienste zugreifen können.

Steigt in einem P2P-Netz die Nachfrage nach einer bestimmten Leistung, so wird auch die Anzahl der Peers steigen, die diese Leistung anbieten – umgekehrt kann dies dazu führen, dass wenig nachgefragte Dienste wenig bis gar nicht mehr angeboten werden, was sich negativ auf den Grad der Verfügbarkeit bzw. auf die Verlässlichkeit auswirkt.<sup>142</sup> Dies spricht die Anpassung des Netzes in Abhängigkeit vom Nutzerverhalten an.

---

<sup>139</sup> Vgl. STEINBACH/ SEITTER (2004), S. 35

<sup>140</sup> Vgl. HAUSWIRTH/ DUSTDAR (2005), S. 8

<sup>141</sup> Vgl. CHAPPEL/ TYLOR (2003), S. 27

<sup>142</sup> Vgl. WOJCIECHOWSKI/ WEINHARDT (2002), S. 111

Web Services können durch Anfrage an eine von mehreren öffentlichen Registries gefunden werden. Im Unterschied zu diesem eher zentralistisch organisierten Ansatz werden die Informationen zu Peers dezentral verteilt vorgehalten, spiegeln dafür aber ein aktuelleres Bild der tatsächlich verfügbaren Peers wieder.<sup>143</sup> Wie im UDDI-Standard vorgeschlagen, findet zwischen den Universal Business Registries über einen Replikationsmechanismus ein Datenaustausch bezüglich der in den einzelnen Registries verzeichneten Dienste statt. Es existiert also zumindest vor der Replikation zuerst einmal dezentral vorhandenes Wissen über neu registrierte Dienste. Einen alternativen Weg zur Verwaltung von Informationen über Web Services mit Hilfe eines P2P-Ansatzes schlagen FORSTER/ DE MEER (2004) vor.<sup>144</sup>

---

<sup>143</sup> Vgl. WOJCIECHOWSKI/ WEINHARDT (2002), S. 112

<sup>144</sup> Vgl. FORSTER/ DE MEER (2004), S. 90

## IV. Typen von IT-Sourcing-Szenarien

Unter Sourcing wird allgemein die Frage nach make-or-buy, also Selbstherstellung versus Fremdbezug von Gütern oder Dienstleistungen verstanden. Im Rahmen der Arbeit wird der Betrachtungsrahmen auf das Sourcing von IT-bezogenen Dienstleistungen, dem so genannten IT-Sourcing eingeschränkt. Unter IT-Sourcing kann dabei die statische Sicht der Beschreibung der aktuellen Aufteilung der Erbringung von Leistungen durch Unternehmensinterne oder Externe genauso verstanden werden, wie in der dynamischen Sicht der vom Management durchzuführende Entscheidungsprozess für die Gestaltung der Beschaffung von Leistungen.<sup>145</sup> Hierfür liefert JOUANNE-DIEDRICH (2005) in seiner IT-Sourcing-Map (vgl. Tabelle 7) unterschiedliche Einteilungskriterien, die einen Eindruck von der Anzahl der verschiedenen Sourcingformen geben.

Einteilungskriterium	IT-Sourcing-Form
Grad des externen Leistungsbezugs	Totales Outsourcing Selektives/ Smart Sourcing Totales Insourcing
Finanzielle Abhängigkeit	Internes/ Captive Outsourcing Externes Outsourcing
Zeitliche Ordnung	Insourcing Outsourcing Backsourcing
Grad der Geschäftsorientierung	Business Process Outsourcing Application Outsourcing Infrastructure Outsourcing
Anzahl der Leistungsersteller	Single-Sourcing Multi-Sourcing

<sup>145</sup> Vgl. DIBBERN (2004), S. 14 f.

Standort	Offshore Sourcing Nearshore Sourcing Domestic Sourcing
Strategische Aspekte	Value-added Outsourcing Transitional Outsourcing Co-Sourcing

*Tabelle 7: Einteilung der IT-Sourcing-Formen  
[vgl. JOUANNE-DIEDRICH (2005)]*

Besondere Bedeutung kommt dem IT-Sourcing durch die von JOUANNE-DIEDRICH/ ZARNEKOW/ BRENNER (2005) postulierte Verschiebung der Sourcing-Schnittstelle von der IT-Abteilung zu den Fachabteilungen zu, deren Fokus auf dem Geschäftsprozess liegt und eine zunehmende Transparenz der technologischen Schnittstellen bedarf.<sup>146</sup> Dies impliziert die Forderung nach der Trennung einer betriebswirtschaftlich motivierten Entscheidung von der technischen Umgebung, die die Entscheidung blockieren oder behindern würde.

#### **IV.1 Auswahl der zu betrachtenden Szenariotypen**

Im Rahmen der Fragestellung nach der Möglichkeit der Trennung einer betriebswirtschaftlich motivierten Entscheidung von technischen Beschränkungen im Rahmen von Sourcing-Entscheidungen sollte keine zu feine Untergliederung der Sourcing-Varianten vorgenommen werden. Es werden deshalb nur wenige Szenariotypen beschrieben, die jedoch gemeinsame Eigenschaften aufweisen und service-orientierte Ansätze verdeutlichen lassen.

Als erste Eigenschaft ist die Kunden-Dienstleister-Beziehung entscheidend für die Auswahl. Daneben liegt ein Schwerpunkt bei Anwendungen, bei denen technische Schnittstellen eine große Bedeutung haben. Damit einhergehend ist die Notwendigkeit der Standardisierung ein wichtiges Merkmal. Schließlich

<sup>146</sup> Vgl. JOUANNE-DIEDRICH/ ZARNEKOW/ BRENNER (2005), S. 19



sollen die Szenariotypen die Einflussmöglichkeit des Managements auf die Erbringung der Leistung zeigen.

Vor diesem Hintergrund wird das Sourcing von IT-Dienstleistungen in verschiedene Segmente aufgeteilt. Segmentierungsvariablen stellen hierbei die wirtschaftliche Abhängigkeit des Dienstleisters vom Auftraggeber und auf einer zweiten Stufe die zeitliche Reihenfolge bzw. die Dauer und Intensität des Leistungsverhältnisses dar. Dadurch ergibt sich zunächst eine Unterteilung in die interne Leistungserbringung über den Shared Services-Ansatz und den externen Bezug der Leistung. Hierbei wird auf der ersten Stufe zwischen dem Outsourcing als solchen und In- bzw. Backsourcing unterschieden. Als Sonderform der Outsourcing-Beziehung wird anschließend (Application) Service Providing betrachtet und der Next Generation Service Provider vorgestellt. Hier ist die Intensität der Geschäftsbeziehung als geringer bzw. sehr gering einzustufen.

Die Betrachtung beginnt mit einem rein innerbetrieblichen Szenario ohne Beteiligung von Dienstleistern, das als Referenzpunkt für die weitere Einbindung unternehmensextern erbrachter Leistung dient. Der zweite Komplex beschäftigt sich mit dem Shared Services-Ansatz, bevor in einem weiteren Abschnitt In- und Outsourcing beschrieben wird. Der letzte Szenariotyp widmet sich dem Provisioning von Anwendungen. Hier wird zwischen dem konventionellen Application Service Providing und den durch den technologischen Fortschritt möglich gewordenen Next Generation Service Providing in Form der Nutzung von kleineren Diensten und der Einmalnutzung von Diensten unterschieden.

## ***IV.2 Innerbetriebliche Service-Ausrichtung***

Startet die service-orientierte Ausrichtung mit der unternehmenseigenen IT, so werden hier Probleme des Schnittstellenmanagements auf eine Weise behandelt, die es erlaubt, schrittweise auch externe Dienste einzubinden.

Eine mögliche Roadmap für die Einführung einer SOA wird den Ausgangspunkt bei der Erweiterung vorhandener Anwendungssysteme haben. Ein weiterer Schritt wird die Integration neuer Anwendungen sein, bevor an die Einbindung

von externen Geschäftspartnern gedacht wird. Diesen Migrationspfad hin zu einer SOA greifen die folgenden Ausführungen auf.

#### **IV.2.1 Erweiterung vorhandener Anwendungen**

Sollen vorhandene Anwendungen in eine Web Service-Umgebung integriert werden, so gilt es eine individuelle Lösung zu finden, die die schon vorhandenen Schnittstellen weiterhin bedient und zusätzlich Web Service Schnittstellen anbietet.<sup>147</sup> Insbesondere aus betriebswirtschaftlicher Sicht geht es dabei um Investitionsschutz der bereits geleisteten Implementierungsarbeit, wobei trotzdem die Öffnung des Unternehmens über das Internet vorangetrieben werden kann.<sup>148</sup>

Die Vorbereitung der Migration der Altanwendungen beginnt dabei mit der Analyse der Anwendungslandschaft, um unternehmenskritische Anwendungen zu identifizieren und auf elementare Operationen herunterzubrechen, für die sich der aufwendige Prozess lohnt, da sie unternehmensindividuelle Funktionen enthalten.<sup>149</sup> Die primäre Zielsetzung ist hier im Unterschied zu Enterprise Application Integration (EAI) die Ablösung der Applikation durch Komponenten, die sich je nach Bedarf dynamisch zu neuen Applikationen rekombinieren lassen.<sup>150</sup> Weiterhin liegt der Fokus nicht auf der Schaffung vieler rein technischer Services, sondern auf Diensten, die an betriebswirtschaftlichen Funktionalitäten ausgerichtet sind.

Bei der Findung der speziell auf die Anwendung zugeschnittenen Lösung reicht das Spektrum dabei von der Entwicklung eines Web Services, der Aufrufe als Wrapper für die dahinter liegende Anwendung übersetzt, bis zu neuen Komponenten auf Applikationsservern, die die Web Service-Funktionalität leicht transparent für die aufrufende Anwendung integrieren lassen.<sup>151</sup> Führen beide

---

<sup>147</sup> Vgl. LANGNER (2003), S. 276

<sup>148</sup> Vgl. LANGNER (2003), S. 277 und WATT (2002), S. 185

<sup>149</sup> Vgl. SNEED (2006), S. 349

<sup>150</sup> Vgl. SNEED (2006), S. 354

<sup>151</sup> Vgl. LANGNER (2003), S. 306 und S. 342

Wege nicht zum gewünschten Ergebnis, so kann auch eine spezielle Middleware<sup>152</sup> eines auf Integration spezialisierten Software-Anbieters eingesetzt werden.<sup>153</sup> Dabei ist bei jeder Anwendung zu prüfen, auf welcher Stufe des Softwarelebenszyklus sie sich befindet. Steht eine Anwendung schon kurz vor der Ablösung, so stellt sich die Frage, ob die Anwendung auf eine neue Plattform migriert werden soll, ob sie durch ein Standardsoftwareprodukt zu ersetzen ist, oder ob es einer kompletten Neuentwicklung bedarf.<sup>154</sup> Die beiden letzten Alternativen bergen die Gefahr in sich, dass über lange Zeit in der Altanwendung eingeflossenes Fachwissen nicht vollständig identifiziert und damit nicht in die neue Anwendung transferiert werden kann.<sup>155</sup>

Wichtig ist die Frage der Art des Übergangs von der ursprünglichen zur Web Service-Zielarchitektur. Hier scheint der vollständige Umstieg mit allen Anwendungen weniger geeignet als der evolutionäre Weg der schrittweisen Umstellung.<sup>156</sup> Ein möglicher Migrationspfad führt hierbei von der Web Service Schnittstelle für einzelne Funktionalitäten einer Legacy-Anwendung,<sup>157</sup> über die Separation der Funktionalitäten im Rahmen einer Modularisierung, bis hin zur Migration der einzelnen Module auf eine andere Plattform.<sup>158</sup>

Entscheidet man sich für den Verbleib der Funktionalität innerhalb der Legacy-Anwendung, muss eine exakte Analyse des Sourcecodes erfolgen. Denn leider befinden sich die Funktionalitäten unter Umständen über einen großen Bereich des Quellcodes verteilt bzw. werden einzelne Bereiche nicht nur von einer Funktionalität benötigt, sodass neben Experten auch Tools zum Einsatz kommen müssen, um die beteiligten Abschnitte zu identifizieren und

---

<sup>152</sup> Middleware wird hierbei als Softwareprodukt verstanden, das die Kommunikation zwischen mehreren Softwaresystemen ermöglicht bzw. sicherstellt. (Vgl. LINTHICUM (2001), S. 128)

<sup>153</sup> Vgl. LANGNER (2003), S. 342

<sup>154</sup> Vgl. SLAMA (2006), S. 24

<sup>155</sup> Vgl. SLAMA (2006), S. 24

<sup>156</sup> Vgl. OEHLER (2005), S. 43

<sup>157</sup> Unter Legacy-Anwendungen werden Altanwendungen verstanden, die entweder noch mit einer veralteten Technologie entwickelt wurden, oder auf einer mittlerweile veralteten Systemplattform betrieben werden. (Vgl. SNEED (2006), S. 345)

<sup>158</sup> Vgl. SLAMA (2006), S. 24

anschließend in einem Modul mit einem eigenen Interface zu vereinzeln.<sup>159</sup> Hat man passend zu dem gefundenen Interface ein WSDL-Dokument modelliert, so muss ein Stück Software, der so genannte Wrapper, entwickelt werden, der die Anfragen durch SOAP-Dokumente verwendet und damit die Datenstrukturen des Interfaces versorgt und die Antworten der Applikation wieder in Form eines SOAP-Dokumentes verpackt und an den Anfragenden leitet.<sup>160</sup>

Bei der Vereinzlung der Funktionalität in dem Anwendungssystem bleibt der Quellcode weitestgehend erhalten und somit wird ein geringes Risiko bezüglich des Verlustes an implementierter Geschäftslogik eingegangen. Die Konsequenzen einer vollständigen Neuentwicklung müssen jedoch deutlich stärker analysiert werden. Neben dem Entwicklungsaufwand muss entweder mit dem Verlust an Wissen aus der Altanwendung gerechnet werden oder, falls dieser vermieden werden soll, kommen Kosten aus aufwendigen Analysen des Quellcodes auf das Unternehmen zu. Daneben gilt es bei der neuen Anwendung mit Aufwendungen aus der Integration die vorhandene Systemlandschaft zu rechnen. Neuentwicklungen ziehen hier möglicherweise Folgeprojekte für Systeme nach sich die Anwendungssysteme betreffen, die die Alt-systeme mit Daten beliefert haben oder Daten aus dem System bezogen haben. Hierbei kann es jedoch zur Ablösung komplexer Schnittstellentypen kommen.

## IV.2.2 Integration von Neuanwendungen

Die Integration von neuen Anwendungen in eine Web Service-Infrastruktur bietet flexiblere Herangehensweisen. Handelt es sich bei der Anwendung um eine Individualentwicklung, so kann die Art der Schnittstellen schon explizit in das Pflichtenheft übernommen werden. Hierbei ist die fachlich bestimmte Gestaltung von Diensten entscheidend. Darüber hinaus sollte die Anwendung als Ganzes schon service-orientiert angelegt werden. Dies bedingt beispielsweise, dass die Anwendung selbst aus Services aufgebaut ist. Die rein tech-

---

<sup>159</sup> Vgl. SNEED (2006), S. 351 f.

<sup>160</sup> Vgl. SNEED (2006), S. 353

nische Umsetzung der Schnittstellen als Web Service reicht hier nicht, da sie die Komposition der Services erschwert. (vgl. Abschnitt II.6.3)

Ebenso wird der Auswahlprozess einer Standardsoftwarelösung in der Frage der geeigneten Schnittstellen nur erweitert. Insbesondere muss natürlich eine Abwägung der Bedeutung der Schnittstellen in Vergleich zur gebotenen Funktionalität getroffen werden. Selbst wenn keine geeigneten Schnittstellen geboten werden, kann auf die bewährte Technik des Nutzens eines Middleware-Servers zurückgegriffen werden, der diese Aufgabe übernimmt. Dann ist allerdings eine genaue Analyse auf die Eignung der Anwendung im Rahmen einer SOA zu prüfen.

### **IV.2.3 Einbindung von Kunden und externen Geschäftspartnern**

Bei der Einbindung von Kunden und externen Geschäftspartnern ist die gesamte Supply- und Demand-Chain eines Unternehmens angesprochen.<sup>161</sup> Durch die service-orientierte Gestaltung von Anwendungen können sowohl Dienste des externen Geschäftspartners leichter integriert werden, als auch eigene Dienste zur Verfügung gestellt werden. Ohne die Nutzung von den auf standardisierten Technologien basierten Web Services bestünde hier ein weit schwierigeres Integrationsproblem. Dies ist einerseits durch die Heterogenität der eingesetzten Applikationen in den verschiedenen Unternehmen und andererseits durch die Vielzahl oft auch nur kleiner Zulieferer bzw. Kunden bedingt.<sup>162</sup>

---

<sup>161</sup> Vgl. WATT (2002), S. 142

<sup>162</sup> Vgl. WATT (2002), S. 145

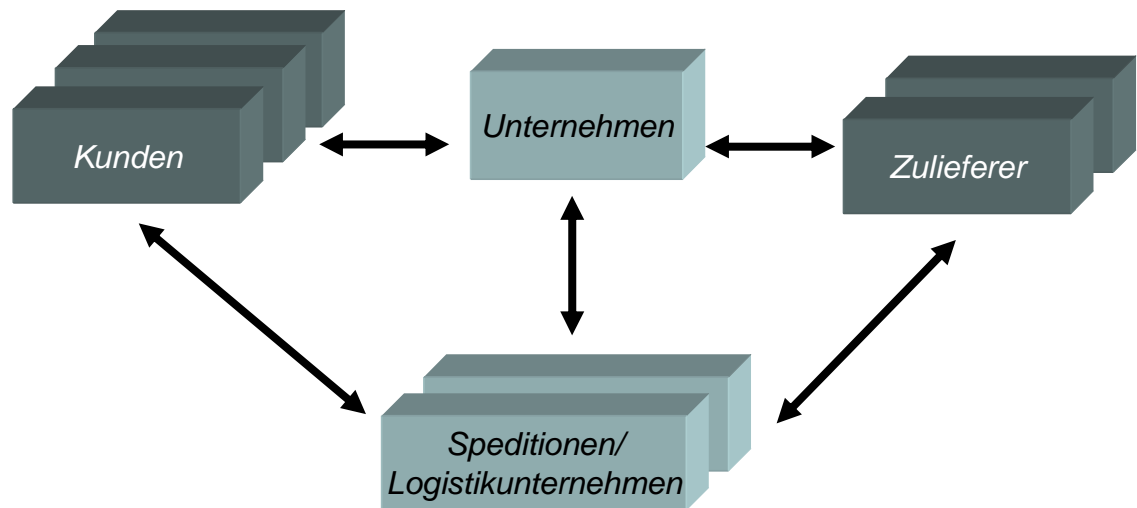


Abbildung 17: Einbindung von Partnern im Unternehmensumfeld im Rahmen unternehmensübergreifender Prozesse [Quelle: WATT (2002), S. 144]

Bei Partnern und Kunden wird für den Fall einer derartigen Nutzung üblicherweise von einer längerfristigen Geschäftsbeziehung ausgegangen. Dies stellt jedoch keine notwendige Bedingung dar, da der Aufwand für die Einbindung der Services und Nutzung der Applikation nicht so hoch ist. Die Problematik liegt hier eher im Aufbau der entsprechenden Nachrichten. Eine allgemein gültige Regelung wie beispielsweise eine Rechnung zu gestalten ist, existiert nicht oder nicht in der erforderlichen Verbreitung mit der entsprechenden breiten Akzeptanz.

Für die Integration der Kunden und Geschäftspartner sprechen beispielsweise bessere Informationsqualität durch Echtzeit-Daten, Zeitersparnis durch die Automatisierung von Vorgängen oder aber bessere Kommunikation, die sich in einer engeren Bindung der Kunden bzw. Partner widerspiegelt.<sup>163</sup>

### IV.3 Shared Services Center (SSC)

Ein Shared Services Center lässt sich nach WISSKIRCHEN/ MERTENS (1999) als wirtschaftlich und rechtlich unabhängige Organisationseinheit charak-

<sup>163</sup> Vgl. WATT (2002), S. 141

terisieren, die zur Optimierung des Ressourceneinsatzes verschiedene Organisationseinheiten eines Unternehmens auf mittel- bzw. langfristiger Basis bei bisher dezentral organisierten Geschäftsprozessen unterstützt.<sup>164</sup>

Im Unterschied zu Outsourcing wird kein eigenständiger, schon existierender externer Dienstleister mit der Erbringung der Leistung beauftragt, sondern es wird eine Organisationseinheit geschaffen, die ein bestimmtes Prozessbündel anbietet und zuerst nur in einer quasi internen Geschäftsbeziehung zu den Organisationseinheiten steht.<sup>165</sup> Die Dienstleistungsgesellschaft wird also ausgegliedert und hat bei einer rechtlichen Unabhängigkeit nur das Mutterunternehmen als Kunden – kommt die wirtschaftliche Selbstständigkeit hinzu, können sowohl die Muttergesellschaft als auch der Dienstleister Geschäftsbeziehungen zu Dritten unterhalten.<sup>166</sup> Die Leistungen werden dabei auf die Bedürfnisse der internen Kunden zugeschnitten und stehen in Konkurrenz zu externen Mitbewerbern.<sup>167</sup> Insbesondere werden Prozesse aus unterschiedlichen Organisationseinheiten integriert, was zwar eine Standardisierung voraussetzt, jedoch unter Beachtung der speziellen Anforderungen der internen Kunden auch zu speziellen Prozessausprägungen führen kann.<sup>168</sup> Die Einflussmöglichkeiten des Managements sind also vergleichsweise hoch. Selbst Anforderungen der angeschlossenen Teilbereiche des Unternehmens werden stärker berücksichtigt, als dies im Falle des reinen Outsourcing möglich wäre. In der Abbildung 18 werden beispielsweise eine Abteilung (= Department in der Grafik) und eine Business Unit an ein SSC angeschlossen, das neben einem ERP-System auch eine Legacy-Anwendung betreibt und Schnittstelleninformationen über einen UDDI-Verzeichnisdienst verwaltet und bereitstellt.

Zur Bildung eines Shared Services Centers durchlaufen die Prozesse deshalb eine Transformation über mehrere Schritte, wobei die Phasen durchaus verschieden angeordnet werden können:<sup>169</sup>

---

<sup>164</sup> Vgl. WISSKIRCHEN/ MERTENS (1999), S. 85

<sup>165</sup> Vgl. WISSKIRCHEN/ MERTENS (1999), S. 88

<sup>166</sup> Vgl. BEHME (1993), S. 291 f.

<sup>167</sup> Vgl. WIESSKIRCHEN/ MERTENS (1999), S. 89

<sup>168</sup> Vgl. WIESSKIRCHEN/ MERTENS (1999), S. 91

<sup>169</sup> Vgl. WIESSKIRCHEN/ MERTENS (1999), S. 108 f.

- Standardisierung: Die zu betrachtenden Prozesse in den verschiedenen Organisationseinheiten werden auf ihr Harmonisierungs- und Standardisierungspotenzial untersucht. Falls keine völlige Standardisierung gelingt, werden spezielle Varianten des Standardprozesses definiert.
- Konsolidierung: Die Prozesse werden dabei in das SSC übernommen. Hier erfolgt der Aufbau der technischen Infrastruktur.
- Reengineering: Hierbei werden Optimierungspotenziale bei der Prozessgestaltung ausgenutzt. Es werden insbesondere die Schnittstellen auf technischer und organisatorischer Ebene zu den angeschlossenen Geschäftseinheiten untersucht. Ziel ist dabei die optimale technologische Unterstützung.

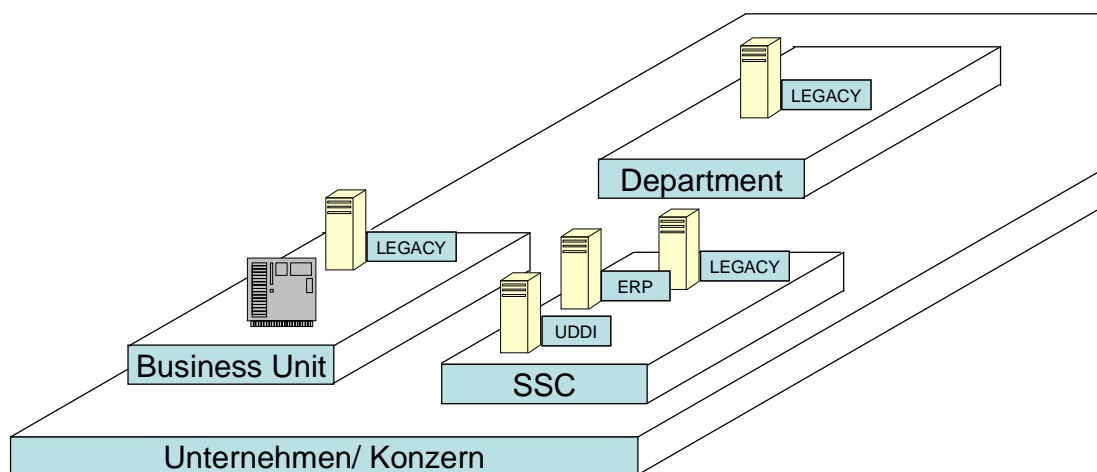


Abbildung 18: Shared Services Center

Bei der Organisation des SSC als Profitcenter findet eine interne Verrechnung der Kosten statt, dabei können Herausforderungen bei der Sicherstellung der Kostentransparenz und der marktähnlichen Führung entstehen.<sup>170</sup> Das Profitcenter ist dabei nur wirtschaftlich und nicht rechtlich selbstständig.<sup>171</sup>

<sup>170</sup> Vgl. SCHRÖDER (1995), S. 31

<sup>171</sup> Vgl. HORCHLER (1996), S. 17



### **Schnittstellen im Shared Services Center**

Die Herausforderungen, die aus der service-orientierten Gestaltung von Schnittstellen im Shared Services Center entstehen, sind ähnlich geartet, wie bei dem rein innerbetrieblichen Szenariotyp. Durch die zentrale Erbringung von Leistungen müssen Dienste standardisiert werden, was unter Umständen Veränderungen in der Struktur der Daten für die Nutzung des Dienstes mit sich bringen kann. Dies verursacht Aufwendungen für das Mapping der Dienste mit den Datenstrukturen der nutzenden Anwendungen. Die Vermeidung von rein technischen Schnittstellen wird umso wichtiger, da hierdurch sonst die Anzahl an notwendigen Diensten für die Einbindung einer Funktionalität steigen würde und damit die Komplexität der Integration zunehmen würde.

Die Organisationseinheiten, die Kunden des SSC sind, verlieren die administrative Kontrolle über die früher dezentralen Anwendungen bzw. Dienste, die jetzt vom SSC bereitgestellt werden. Um diesen Kontrollverlust abzumildern und die Nutzung des Dienstes bzw. der Anwendung bei einer Verlagerung weiterhin gewährleisten zu können, wird der Aufbau eines privaten UDDI-Verzeichnisses innerhalb der Unternehmung wichtig.

Erfolgt die Anbindung der Organisationseinheiten an das Shared Services Center über das Internet, muss die Datenübertragung zu den Schnittstellen zusätzlich gesichert werden.

### ***IV.4 Out-/ In- und Backsourcing***

Beim Outsourcing von Anwendungen entsteht genauso wie bei der Wiedereingliederung von Anwendungen eine komplexe Entscheidungssituation. Neben betriebswirtschaftlichen Überlegungen kommen Fragestellungen aus dem juristischen Bereich hinzu, bevor technische Aspekte berücksichtigt werden können. Gerade die Problemstellungen aus technischer Sicht werden nach der Begriffsklärung, Charakterisierung und Erklärung der Outsourcingformen behandelt. Bei den technischen Fragestellungen ist insbesondere der Aspekt des Managements und der Gestaltung von Schnittstellen wichtig. Im Anschluss daran wird auf In- bzw. Backsourcing eingegangen. Nach der Erläuterung der

damit verbundenen Problemstellungen wird die Rolle von Web Services bei In- bzw. Backsourcing erläutert.

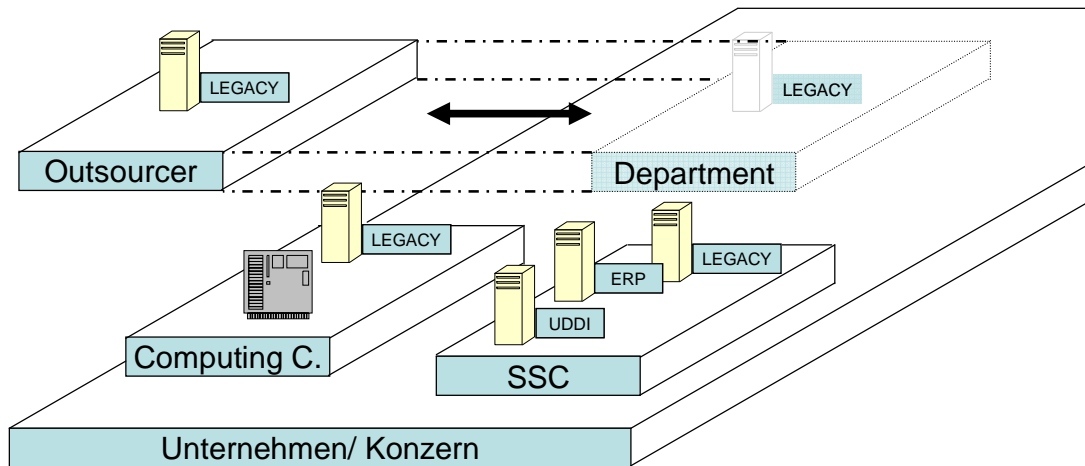


Abbildung 19: In- bzw. Outsourcing

#### IV.4.1 Der Begriff des Outsourcing

Outsourcing ist eine Wortneuschöpfung, die aus *Outside Resource Using* entstanden ist.<sup>172</sup> Damit lässt sich das Outsourcing von IT-Dienstleistungen als Adaption des Make-or-buy-Prinzips auf die Verringerung der Leistungserstellungstiefe bei IT-Dienstleistungen in Form langfristiger Beziehungen zu einem Dienstleister charakterisieren.<sup>173</sup> BEHME (1993) definiert vor diesem Hintergrund Outsourcing als „(...) die Externalisierung bestimmter Teilleistungen oder Funktionen eines Unternehmens und deren Übernahme durch einen externen Anbieter“.<sup>174</sup>

Ausgangspunkt für Outsourcing war früher die Auslagerung des Betriebs von Rechenzentren, bevor man das Konzept auf verschiedene Bereiche der IT ausgedehnt hat.<sup>175</sup> Dabei war die Überlegung aus der Management Literatur

<sup>172</sup> Vgl. BÜHNER/ TUSCHKE (1997), S. 21 und KÖHLER-FROST (1995), S. 13

<sup>173</sup> Vgl. SCHRÖDER (1995), S. 25

<sup>174</sup> BEHME (1993), S. 291

<sup>175</sup> Vgl. BITKOM (2005), S. 8

der 90er Jahre des letzten Jahrhunderts ausschlaggebend, die zur Aufrechterhaltung der Konkurrenzfähigkeit betriebliche Teilfunktionen in Kern- und Nicht-Kernkompetenzen des Unternehmens aufteilt und im Fall der Nicht-Kernfunktion die Vergabe an einen auf diese Funktion spezialisierten Dienstleister empfiehlt.<sup>176</sup> Zur Diskussion von auslagerungsfähigen Bereichen des Unternehmens nach dem Transaktionskostenansatz bzw. nach der Resource-Based Theory kann beispielsweise BÜHNER/ TUSCHKE (1997) herangezogen werden.

#### IV.4.2 Charakterisierung und Abgrenzung des Outsourcing

Outsourcing lässt sich insbesondere durch folgende Aspekte charakterisieren:<sup>177</sup>

- Die Leistung wird durch einen externen Dienstleister unter dessen Verantwortung erbracht.
- Die zwischen dem Dienstleister und dem beauftragenden Unternehmen geschlossenen Verträge sind langfristig angelegt.
- Die Verträge schließen meist den Übergang von Vermögenswerten und Personal an den Dienstleister mit ein.

Vor dem Hintergrund der begrifflichen Vielfalt kann die Charakterisierung der Stellung des Dienstleisters als einen Externen als grundlegendes Abgrenzungsmerkmal herangezogen werden.<sup>178</sup> Insbesondere verwendet man in der Beziehung zum Dienstleister Verträge als Koordinationsinstrument bei der Übertragung der (Teil-)Funktionen des Unternehmens, ohne Kapitalverflechtung einzugehen.<sup>179</sup> Outsourcing lässt sich somit als Auslagerung bezeichnen.<sup>180</sup>

Als Mindestlaufzeit des Outsourcing-Vertrages kann ein Jahr angesehen

---

<sup>176</sup> Vgl. HEYWOOD (2001), S. 29

<sup>177</sup> Vgl. DITTRICH/ BRAUN (2004), S. 2

<sup>178</sup> Vgl. hierzu u. a. MYLOTT (1995), S. 10

<sup>179</sup> Vgl. BEHME (1993), S. 292

<sup>180</sup> Vgl. BEHME (1993), S. 292

werden.<sup>181</sup> Würde man diese Untergrenze deutlich unterschreiten, wäre ein Missverhältnis zwischen beispielsweise den Kosten für die Suche des Dienstleisters oder den Kosten für die Vertragsgestaltung und den in kurzer Zeit zu erwartenden Einsparungen im Vergleich zur unternehmensinternen Erbringung der Leistung zu beobachten. Auf der anderen Seite ergibt sich auch eine Einschränkung der Vertragslaufzeit nach oben. So müsste aus Sicht des Kunden bei sehr lang laufenden Verträgen ein sehr hohes Maß an Flexibilität, bedingt durch die Unsicherheit der Entwicklung innerhalb und im Umfeld der Unternehmung, vertraglich vereinbart und entsprechend monetär abgegolten werden.<sup>182</sup> Kürzere Vertragslaufzeiten können dabei nicht als Indiz für kurze Dauern der Outsourcingpartnerschaften gesehen werden, da nach dem Ende eines Kontraktes neben der Vergabe an einen anderen Dienstleister, oder der Wiedereingliederung, auch die Neuverhandlung eines Anschlussvertrages stehen kann.

Einflussmöglichkeiten des Managements werden über Verträge geregelt. Im Allgemeinen bedeutet dies eine Einschränkung der Flexibilität bei technischen Lösungsmöglichkeiten, falls hierfür keine speziellen Regelungen in den vertraglichen Vereinbarungen enthalten sind. Ein hoher Grad an Flexibilität und damit ein hoher Anpassungs- und Fortentwicklungsaufwand für den Dienstleister würde hohe Kosten für die Outsourcingpartnerschaft mit sich bringen. Kürzere Vertragslaufzeiten erleichtern hier jedoch über mögliche Neuverhandlungen mit dem „alten“ Dienstleister, meist auch unter dem Druck des Wechsels des Dienstleisters, Anpassungen in die Outsourcingpartnerschaft einzubringen. Die Einflussmöglichkeit des Managements ist jedoch deutlich geringer als bei der Schaffung eines Shared Services Centers.

### IV.4.3 Outsourcingformen

Wie sich schon bei den IT-Sourcing-Formen nach JOUANNE-DIEDRICH (2005) in Tabelle 7 zeigt, lassen sich verschiedene Ausprägungen von Outsourcing am

---

<sup>181</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 32

<sup>182</sup> Vgl. RIPIN/ SAYLES (1999), S. 105 f.

Markt beobachten. Dabei kann eine erste, sehr grobe Einteilung in komplettes und selektives Outsourcing getroffen werden.<sup>183</sup> Komplettes Outsourcing bedeutet dabei nicht die vollständige Auslagerung des gesamten IT-Personals, da neben dem Chief Information Officer unbedingt weiteres Personal an der Schnittstelle zum Outsourcing-Dienstleister und für die Beurteilung der Einhaltung der Service Level nötig ist.<sup>184</sup>

Für die Arbeit von besonderer Bedeutung ist der Aspekt des selektivem Outsourcing, das sich vom Outsourcing der IT-Infrastruktur über das Application Management und Processing Services bis hin zu Business Process Outsourcing erstreckt, wobei man die letzten 3 Formen als anwendungsnahe Outsourcing bezeichnen kann.<sup>185</sup>

---

<sup>183</sup> Vgl. DITTRICH/ BRAUN (2004), S. 3

<sup>184</sup> Vgl. WILLIAMS (1998), S. 15

<sup>185</sup> Vgl. DITTRICH/ BRAUN (2004), S. 3

<b>Business Process Outsourcing i. w. S.</b>		<p><b>Transactional Outsourcing:</b></p> <p>Hochvolumiges Abwickeln von Standardtransaktionen. Die Abrechnung erfolgt proportional zu der Menge an Transaktionen. Unter Umständen müssen hier kein Personal und keine Vermögenswerte an den Dienstleister übergehen.</p>
		<p><b>Application Outsourcing:</b></p> <p>Der Dienstleister übernimmt den Betrieb einer komplexen Applikation inklusive der Plattform zu deren Betrieb in seine Verantwortung.</p>
	<b>Business Process Outsourcing i. e. S.</b>	<p><b>Business Processing Outsourcing/ Business Process Services/ Business Transformation Outsourcing/ Transformational Outsourcing:</b></p> <p>Der Dienstleister übernimmt einen Prozess, der durchaus verschiedene Applikationen auf der technischen Seite erfordern kann. Schnittstellen aufseiten der Technik und der Mitarbeiter stehen bei den mit unterschiedlicher IT-Unterstützung abzuwickelnden Prozessen im Vordergrund.</p>

*Tabelle 8: Gliederung des anwendungsnahen Outsourcing*

*[vgl. DITTRICH/ BRAUN (2004), S. 4]*

Das in Tabelle 8 aufgezeigte Business Process Outsourcing i. w. S. bildet den Referenzpunkt für die weiteren Betrachtungen. Hierbei treten sowohl hochvolumige Standardservices wie auch einzelne Applikationen und ganze Geschäftsprozesse als Objekte des Outsourcingprozesses auf.

Ziel von BPO ist vor allem die Nutzung von *economies of scale* und *economies of scope*, die sich durch die Nutzung eines externen Dienstleisters erschließen lassen. Während sich der Nutzen bei den *economies of scale* aus Kostenersparnissen durch ein hohes Volumen ergibt, bringt bei den *economies of scope* das vom Dienstleister eingebrachte Spezialwissen den Vorteil mit sich.<sup>186</sup>

<sup>186</sup>

Vgl. für diesen Absatz DITTRICH/ BRAUN (2004), S. 10

#### **IV.4.4 Bedeutung und Gestaltung von Schnittstellen bei Outsourcingprojekten**

Bei der Betrachtung von Schnittstellen muss in Outsourcingprojekten zwischen organisatorischen Schnittstellen und technischen Schnittstellen von Auftraggeber zu Dienstleister unterschieden werden.

Kompetenzen, die sich aus dem Wissen über Schnittstellen im Rahmen des Schnittstellenmanagements ergeben, machen es dem auftraggebenden Unternehmen erst möglich, weiterhin strategische Entscheidungen über die zukünftige Ansiedelung der ausgelagerten Geschäftsprozesse und Anwendungen zu treffen.<sup>187</sup> Die reine Anwendung kann beispielsweise ersetzt werden, falls alle nötigen Schnittstellen zu anderen Systemen im Unternehmen weiterhin in Betrieb bleiben. Diese Entscheidung fußt jedoch auf der genauen Kenntnis der Integrationspunkte durch Mitarbeiter des Unternehmens bzw. durch spezielle Schnittstellenmanager. Eben diese Wissensträger spielen über den Verlauf der Outsourcingpartnerschaft hinweg eine wichtige Rolle als Mittler zwischen Fachabteilung und Dienstleister.

Die hohe Bedeutung der Schnittstellen ist einerseits die Folge ihrer großen Anzahl und andererseits jedoch aus der Verschiedenartigkeit der technischen Ausgestaltung zu erklären. Beides kann die Folge eines ohne durchgängige IT-Strategie verfolgten Best-of-Breed-Ansatzes bei der Auswahl der Anwendungssysteme sein, was zu sehr hoher Heterogenität der Anwendungsplattformen und damit auch der Schnittstellen führt.<sup>188</sup>

Positive Wirkungen durch die Fremdvergabe eines Prozesses in Form geringer Kosten lassen sich teilweise auf die Konsolidierung im Bereich der Schnittstellen zurückführen. DITTRICH/ BRAUN (2004) nennen beispielsweise eine hohe Anzahl unterschiedlicher Technologieplattformen und niedrige Standardisierung bei der Software als fördernde Faktoren bei der Entscheidung für BPO.<sup>189</sup> Beide genannten Punkte führen durch die Vergabe an einen

---

<sup>187</sup> Vgl. DITTRICH/ BRAUN (2004), S. 16

<sup>188</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 136

<sup>189</sup> Vgl. DITTRICH/ BRAUN (2004), S. 29

Dienstleister zu einer Reduzierung an Schnittstellen, da der Übernahme meist eine Konsolidierung auf System- und Technologieebene vorausgehen wird. Wurden die Applikationen des auszulagernden Prozesses in Bezug auf die Technologieplattform und die Software standardisiert, entschärft sich das Problem zwar, die Schnittstellen zu im Unternehmen verbleibenden Systemen müssen jedoch immer noch umgesetzt und damit gewartet werden. Je geringer jedoch die Zahl der bestehen bleibenden Schnittstellen wird, desto leichter kann ein Wechsel zu einem anderen Dienstleister vollzogen werden bzw. desto eher besteht die Möglichkeit weitere Dienstleister zum Abfangen einer wachsenden Prozessmenge einzusetzen.<sup>190</sup> Bei weiteren Geschäftsprozessen und Anwendungen, die eine hohe Verbindung zu einem schon ausgelagerten Prozess oder zu einer ausgelagerten Anwendung haben, kann die Schnittstellenproblematik sogar auch zu deren Auslagerung führen.<sup>191</sup>

In einem sich über die Zeit ändernden Unternehmensumfeld entstehen neue Anforderungen an bestehende Schnittstellen bzw. für die Einrichtung weiterer Schnittstellen. Aus diesem Grund sollte für die Dauer der Geschäftsbeziehung zum Dienstleister eine Verpflichtung zur Fortentwicklung der Anwendungsschnittstellen vereinbart werden.<sup>192</sup>

Insbesondere während der operativen Implementierung des Outsourcing können Schnittstellen durch folgende Gesichtspunkte das Projektbudget negativ beeinflussen:<sup>193</sup>

- hohe Schnittstellenkomplexität zu im Unternehmen verbleibenden Prozessen
- großer Umfang an Schnittstellen, die weiterhin zu bedienen sind.
- große Datenmengen
- hohe Interaktionsfrequenz über die Schnittstellen.
- Schnittstellen, für deren Funktionieren manuelle Eingriffe oder zusätz-

---

<sup>190</sup> Vgl. DITTRICH/ BRAUN (2004), S. 50

<sup>191</sup> Vgl. DITTRICH/ BRAUN (2004), S. 34

<sup>192</sup> Vgl. DITTRICH/ BRAUN (2004), S. 55

<sup>193</sup> Vgl. DITTRICH/ BRAUN (2004), S. 90



liche Kommunikation innerhalb des Unternehmens nötig sind

Werden einzelne dieser Faktoren identifiziert, kann hier im Rahmen des Transformationsprozesses auf Web Services gezielt angesetzt werden. Insbesondere die Standardisierung kann senkend auf die Schnittstellenkosten wirken. Erfolgt die Fremdvergabe einer Anwendung im Zuge des Transformational Outsourcing, wird also parallel zu dem externen Betrieb einer alten Anwendung eine neue entwickelt, kann deutlich stärker an der Integration von Web Services gearbeitet werden.

Die Rolle von UDDI-Verzeichnisdiensten im Outsourcingprozess ist abhängig vom Grad der Serviceorientierung im Unternehmen und der Anzahl der ausgelagerten Anwendungen bzw. der damit verbundenen Anzahl an Schnittstellen zu diesen Anwendungen. Existiert in einem Unternehmen schon eine SOA wird sicherlich ein interner UDDI-Server genutzt werden. Der gleiche Fall tritt ein, falls eine Vielzahl an unternehmensinternen Anwendungen an die ausgelagerte Anwendung bzw. an den ausgelagerten Geschäftsprozess angebunden werden müssen. Hier bedeutet die Nutzung von UDDI neben einer gesteigerten Sicherheit im Falle der Verlagerung der externen Anwendungen gleichzeitig eine Senkung des Konfigurationsaufwandes. Ohne den Zugriff müsste jede Anwendung einzeln angepasst werden, falls sich der Standort der externen Anwendung ändert.

#### **IV.4.5 In- bzw. Backsourcing**

Werden Anwendungen, die von externen Dienstleistern für ein Unternehmen betrieben wurden, in das Unternehmen integriert, so gilt es in Bezug auf die Bezeichnung dieses Vorgangs zwischen dem Fall einer neuen Anwendung und dem Fall einer Anwendung zu unterscheiden, die ursprünglich schon einmal durch das Unternehmen betrieben wurde.

Insourcing kann als „Rückübernahme eines ehemals ausgelagerten (IT-) Betriebes in die eigene Organisation nach Ablauf des Outsourcing-Vertrages

oder bei dessen vorzeitiger Beendigung“<sup>194</sup> bezeichnet werden. Der Begriff Insourcing impliziert dabei, dass die IT-Funktionen oder Prozesse noch nie im Unternehmen angesiedelt waren, während man bei einem Zurückholen ehemals interner IT-Funktionen und Prozesse genauer von Reinsourcing spricht.<sup>195</sup> Statt Reinsourcing kann dieser Prozess auch als Backsourcing bezeichnet werden.

#### **IV.4.6 Problemstellung bei der Reintegration von ausgelagerten Anwendungen und der Beendigung von Outsourcingprojekten**

Bei der Betrachtung von Insourcing-Projekten sind grundsätzlich zwei verschiedene Situationen zu unterscheiden. Im ersten Fall wird eine Anwendung, die schon innerhalb des Unternehmens betrieben wurde, wieder eingegliedert. Im zweiten Fall wird eine Anwendung, die noch nie innerhalb der Unternehmung betrieben wurde, in die Anwendungslandschaft integriert. Während im ersten Fall zumindest noch ein rudimentäres Wissen über die Anwendung im Unternehmen vorhanden ist, muss im zweiten Fall neben der Schnittstellenproblematik auch noch Know-how zum Betrieb der Anwendung im Unternehmen erworben werden.

Probleme können sich in beiden Fällen also aus dem Wissensverlust bezüglich der Einbettung der Anwendung in die Anwendungslandschaft und dem Betrieb der Anwendung über die Dauer der Outsourcing-Partnerschaft ergeben.<sup>196</sup> Dies sollte idealerweise schon bei der Vorbereitung der Outsourcing-Entscheidung mit in die Überlegungen einbezogen werden.<sup>197</sup> Insbesondere sollte darauf geachtet werden, dass so genannte Schnittstellenmanager im Unternehmen vorhanden sind, die nicht nur den Kontakt zum Outsourcer pflegen. Sie sollten vielmehr genau über die Einbettung der ausgelagerten Anwendung in die

---

<sup>194</sup> BITKOM (2005), S. 54

<sup>195</sup> Vgl. DITTRICH/ BRAUN (2004), S. 87 f.

<sup>196</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 177

<sup>197</sup> Vgl. CUNNINGHAM/ FRÖSCHL (1995), S. 177

Geschäftsprozesse Bescheid wissen und insbesondere im Falle der Zurückführung der Anwendung in das Unternehmen diesen Prozess unterstützen. Hierfür sollten sie einen gezielten Wissenserwerb durch Mitarbeiter über die Dauer des Outsourcingvorhabens fördern.

#### **IV.4.7 Die Rolle von Web Services bei Insourcing-Projekten**

Zur Untersuchung der Rolle von Web Services bei Insourcing-Projekten wird die vorher gemachte feine begriffliche Unterscheidung aufgeben, da die Folgen des Einsatzes von Web Services in beiden Fällen gleich zu beurteilen sind.

Wird eine Anwendung, die schon innerhalb des Unternehmens betrieben wurde, an einen externen Dienstleister übergeben, so kommen im Fall der Anbindung über Web Services zu den normalen Kommunikationsprotokollen noch Anforderungen aus dem Bereich der Sicherheit hinzu. Die Kommunikation muss also mit komplexeren Mitteln durchgeführt werden. Wird eine solche Anwendung wieder zurückgeholt, so können die Schnittstellenausgestaltungen zu den anderen Anwendungen beibehalten werden. Die Kommunikation wird dann zwar aufwendiger durchgeführt, aber so lässt sich der Anpassungsaufwand minimieren. Nebenbei kann es auch innerhalb der Unternehmung zwischen räumlich getrennten Niederlassungen zur Nutzung von Verbindungen über das Internet kommen, sodass hier das zusätzliche Maß an Sicherheit gut gebraucht werden kann.

Begibt man sich auf die rein technische Ebene, so kommt es bei der Verlagerung zu einer Umstellung in den UDDI-Verzeichnissen des Unternehmens. Wurde bei der Konzeption der angrenzenden Anwendung auf die Implementierung der losen Kopplung geachtet, so ist dies der einzige Umstellungsschritt. Bei der direkten Anbindung in die Anwendung mit Datenstrukturen für den Datenaustausch und URI können zwei Fälle auftreten. Im Extremfall sind die Daten in den Quellcode eingebunden, sodass es hier einer Änderung bedarf. Der häufigere Fall ist wahrscheinlich, dass die Daten durch die Änderung an Konfigurationsdateien bzw. durch die Anpassung der Werte über grafische Oberflächen abänderbar sind. Sind Anwendungen also service-

orientiert gestaltet, reduziert sich der technische Aufwand bei Insourcing.

## **IV.5 Provisioning**

Bei der Betrachtung von Application Service Providern (ASP) soll im Rahmen der Arbeit zuerst zwischen dem klassischen ASP der ersten und zweiten Generation, dem Provider von Software as a Service und dem Next Generation Service Provider unterschieden werden.

Die ersten ASPs, die während des .com-Hypephase auf dem Markt getreten sind, stammten hauptsächlich aus dem Bereich der Value-added Reseller, Telekommunikationsunternehmen, Internet Service Provider, Systemintegratoren, Outsourcing-Dienstleister und Softwarehersteller, die einen neuen Vertriebskanal für ihre Produkte suchten.<sup>198</sup> Dabei musste mehr und mehr Wert auf die Anpassung der angebotenen Standardlösung auf die Bedürfnisse der Kunden im Rahmen des so genannten Customizings gelegt werden.<sup>199</sup> Trotzdem sollte im Unterschied zum klassischen Outsourcing keine vollständig auf Kundenbedürfnisse individualisierte Lösung entstehen, sondern weiterhin der Standardcharakter der Applikation erhalten bleiben.<sup>200</sup> Aus diesem Widerspruch heraus lässt sich schließen, dass nur eine sehr begrenzte Anzahl an Applikationen diese Anforderungen erfüllen kann, wie etwa Textverarbeitungen.

Um die gewünschte Leistung in entsprechender Qualität und Quantität dem Kundenkreis anbieten zu können, kam es im ASP-Markt zu einer ganzen Zahl an vertikalen strategischen Partnerschaften – ein Softwareanbieter kooperiert dabei mit Hardwarepartnern und Anbieter eher technischer Lösungen arbeiten mit Unternehmen mit guten Vertriebskanälen zusammen.<sup>201</sup>

Aus dem klassischen ASP-Modell kommend, hat sich Software as a Service (SaaS) als neues Vertriebsmodell für Softwarelösungen etabliert, das sich

---

<sup>198</sup> Vgl. COLUMBUS (2000), S. 23 und KNOLMAYER (2000), S. 443

<sup>199</sup> Vgl. COLUMBUS (2000), S. 23

<sup>200</sup> Vgl. KNOLMAYER (2000), S. 443

<sup>201</sup> Vgl. COLUMBUS (2000), S. 171

stärker als es bei ASP-Anwendungen der Fall war auf die Konfigurierbarkeit konzentriert – die Softwarebereitstellung erfolgt direkt vom Hersteller.<sup>202</sup> Das hierbei angestrebte Provisioning-Modell kann als Zwischenstufe auf dem Weg von den ASPs der ersten und zweiten Generation zu den Next Generation ASPs gesehen werden.

Vor allem durch die SOA ergeben sich bei SaaS sowohl durch die Ergänzung der Applikation durch zusätzliche Funktionen aufseiten der Hersteller (z. B. Quality of Service Tools, weitere Funktionalitäten oder Abrechnungssysteme) bzw. auf der Seite der Nutzer (z. B. Integration mit Daten anderer Anwendungen im Unternehmen oder die direkte Integration der SaaS-Anwendung in die Systemlandschaft des Unternehmens) weitreichende Möglichkeiten.<sup>203</sup> Nach einer Marktanalyse von IDC betragen die weltweiten Ausgaben für SaaS 2005 4,2 Milliarden und für 2009 werden sogar 10,7 Milliarden US-Dollar prognostiziert.<sup>204</sup>

Beim Next Generation Service Provider kommen zu den Eigenschaften des SaaS-Providers noch die Nutzung von Grid-Technologien hinzu. Hier werden die Grenzen zwischen dem reinen Bereitstellen von Anwendungen aus eigener Entwicklung, der Bereitstellung kleiner Anwendungen als Services und der Bereitstellung von Infrastruktur in Form von Hosting-Umgebungen für Web Services oder Grids aufgebrochen. Es entsteht ein flexibler Dienstleister.

## IV.5.1 Definitionen

BITKOM (2005) definiert Application Service Providing als eine „ (...) Spezialform des Application Outsourcing bei der (...) [in der Regel] browser-/ internetfähige Applikationen auf Basis eines nutzungsabhängigen Miet-Modells über ein Wide-Area-Network (WAN) – (...) [in der Regel] das World-Wide-Web – vom Rechenzentrum des Application Service Providers aus zur Verfügung gestellt

---

<sup>202</sup> Vgl. BLOZAN (2005)

<sup>203</sup> Vgl. BLOZAN (2005)

<sup>204</sup> Vgl. IDC (2005)

werden.<sup>205</sup> Diese Definition entspricht dem Verständnis des klassischen ASPs, der komplexe Applikationen über das Internet anbietet. Hierin ist auch die Einordnung in das Application Outsourcing begründet. Wenn man der Begriffsbildung von Outsourcing folgt, soll ja nur eine Ressource (die Applikation) außerhalb des Unternehmens genutzt werden, die Forderung nach dem vorherigen selbstständigen Betrieb durch das Unternehmen wird nicht erhoben.

Software as a Service kann als Vertriebsmodell für Softwarelösungen bezeichnet werden, in dem Geschäftsfunktionen unter Einhaltung eines Service Level Agreements auf Basis nutzungsabhängiger Entgelte direkt vom Softwareanbieter zur Verfügung gestellt werden.<sup>206</sup> Die Abgrenzung zum klassischen ASP liegt hierbei in der Rolle des Anbieters des Softwareservices, der gleichzeitig die Infrastruktur für den Service bereitstellt.<sup>207</sup>

Ein Next Generation Service Provider ist ein Anbieter von Services verschiedener Granularität und/ oder eines Infrastrukturdienstes in Form eines Grids bzw. einer Web Service-Hostingplattform. Es wird ein nutzungsabhängiges Entgelt erhoben. Der Zugriff erfolgt über die Infrastruktur des Internets nach Maßgabe von Service Level Agreements. Er tritt gegenüber Kunden als Full-Service-Provider auf.

## IV.5.2 Diensteeinteilung

Die Services des Next Generation Service Providers lassen sich grob in zwei Teilbereiche einteilen. Auf der einen Seite steht das Angebot von fertigen Lösungen. Dem gegenüber bietet er auch Infrastrukturlösungen an.

Die fertigen Anwendungen umfassen Dienste aus allen in Abschnitt I.1 vorgestellten Servicearten (z. B. Fire-and-Forget, Request-Response, Dienste als Teile von Geschäftsprozessen, Rechendienste). Es erfolgt jedoch kein Hosting von Applikationen im konventionellen Sinne des Hosting z. B. eines ERP-

---

<sup>205</sup> BITKOM (2005), S. 49

<sup>206</sup> Vgl. BLOZAN (2005)

<sup>207</sup> Vgl. BLOZAN (2005)

Systems. Alle angebotenen Anwendungen weisen ein dediziertes serviceorientiertes Design auf und lassen sich deshalb vergleichsweise einfach in eine unternehmensinterne SOA in Abhängigkeit von deren Reifegrad integrieren. Je höher der Reifegrad der SOA ist, desto mehr höherwertige Services (z. B. Rechendienste) werden einbindbar.

Die bereitgestellte Infrastruktur umfasst das Hosting von Web Services, die individuell gestaltet wurden und fertige Services aus der Umgebung des Next Generation Service Providers enthalten können. Daneben stehen Grid-Umgebungen für individuelle rechenaufwendige Problemlösungen durch Kunden zur Verfügung.

Die Kombination aus beiden Angeboten ermöglicht eine optimierte Auslastung der Hardware des Providers unter intensiver Nutzung der Web Service-Technologie. Des Weiteren gestatten die komplementären Technologien den effektiven Einsatz von Virtualisierungslösungen zur Bereitstellung von Rechenleistung für unterschiedliche Zwecke bzw. Kunden und gleichermaßen zur Abfederung von Lastspitzen eventuell unter Einsatz von weiteren Providern, die als Subunternehmer agieren. Wie Erfahrungen aus dem SAP Hochschulkompetenzzentrum in Passau als ASP für Hochschulen<sup>208</sup> mit etwa 84 Servern für unterschiedliche SAP-Systeme gezeigt haben, ist die durchschnittliche Systemlast unter Umständen eher gering und die Last steigt nur zu gewissen Zeiten stark an. Der Einsatz von Virtualisierungslösungen kann hier zur Reduktion der Server bei gleichbleibender Servicequalität im Provisioning der Systeme beitragen bzw. es können mit der gleichen Hardware mehr Kunden bedient werden.

Durch den Einsatz von Diensten eines ASP, deren Schwerpunkt auf der Umsetzung einer technischen Lösung und weniger auf der Bedeutung von in dem Dienst implementierten Prozesswissens liegt, kann zusätzlich die Realisierungsgeschwindigkeit steigen.<sup>209</sup>

---

<sup>208</sup> Vgl. KLEINSCHMIDT/ EDER (2000)

<sup>209</sup> Vgl. DITTRICH/ BRAUN (2004), S. 64 f.

### IV.5.3 Provisioningszenariotypen

Bei den Provisioningszenarien wird zwischen der Nutzung einzelner Anwendungsteile, der Bereitstellung ganzer Applikationen und dem Angebot von Rechendiensten unterschieden. Abbildung 20 zeigt den entstehenden Szenariotyp analog zum Dreieck der Servicenutzung innerhalb der SOA bzw. bei Web Services.

#### IV.5.3.1 Provisioning von kleinen Anwendungen

Der Einstieg in die Nutzung von Diensten eines Service Providers ist sicherlich die Nutzung einzelner, kleinerer Services. Unter einem kleinen Service kann man dabei Dienste verstehen, die weniger geschäftskritisch sind. Es sind beispielsweise als *Fire-and-forget* zu klassifizierende Services.

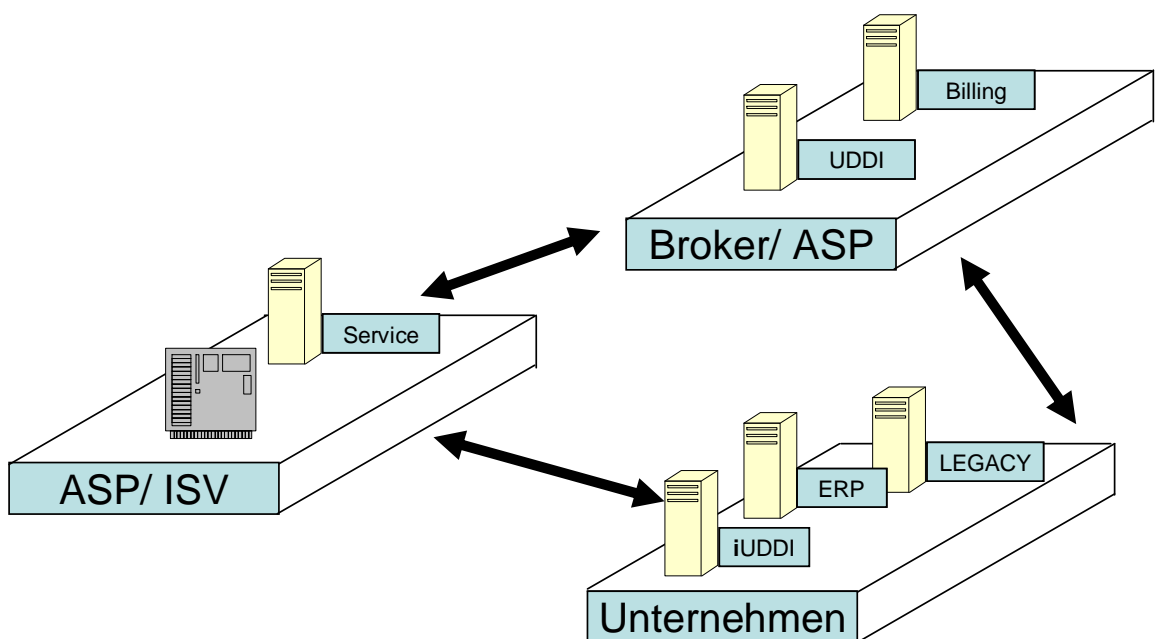


Abbildung 20: Traditionelles ASP-Szenario

Die Einfachheit wird an zwei Eigenschaften deutlich. Zum einen brauchen die Dienste wenige Daten, um die Leistung zu erbringen. Zum anderen werden



auch eher einfach strukturierte Daten zurückgeliefert. Für diese Dienste ist es nicht untypisch, dass kein Nutzungsentgelt anfällt, da sie im Rahmen einer Zusatzleistung für eine andere Leistung angeboten werden. Das Nutzungsentgelt kann alternativ sehr gering sein. Es wäre hier auch denkbar, dass das Nutzungsentgelt nur einmalig in Form einer Registrierungsgebühr erhoben wird. Eine relativ einfach strukturierte und gut standardisierbare Dienstleistung wäre beispielsweise eine Kreditwürdigkeitsprüfung.

#### ***IV.5.3.2 Anwendungsbereitstellung***

Unter einer Anwendungsbereitstellung wird allgemein das entgeltliche Angebot der Nutzung einer Applikation über das Internet verstanden. Die Anwendung erreicht dabei nicht die komplexe Vielfalt einer ERP-Lösung, sondern ist für spezielle Anforderungen (z. B. Personalabrechnung) konzipiert und auf das Angebot der Lösung durch einen Service Provider hin optimiert. Im Idealfall ist die Anwendung ohne zusätzliche Konfiguration nutzbar und setzt sich selbst aus Web Services zusammen.

In dem Kontext der Arbeit findet keine direkte Benutzerinteraktion mit der Applikation über ein grafisches Benutzerinterface in Form eines Fat Client oder durch eine Browser-Oberfläche statt. Die Applikation weist eine höhere Komplexität auf und ermöglicht die Nutzung über Schnittstellen, die einen service-orientierten Charakter aufweisen. Die Komplexität der Anwendung lässt sich einerseits durch die Anzahl der angebotenen Web Services oder andererseits durch die an die Anwendung zu liefernden Daten ablesen.

#### ***IV.5.3.3 Provisioning von Rechendiensten***

Beim Provisioning von Rechendiensten lassen sich mehrere Varianten unterscheiden. Agiert ein Dienstleister als Provider des Rechendienstes, so kann ein von ihm erstellter Dienst mit unterschiedlichen Mengen an Ressourcen (z. B. Rechenleistung, Speicherplatz, Spitzenlast, etc.) angeboten werden. Alternativ dazu kann ein Service des Requestors innerhalb des Grid des Dienstleisters

betrieben werden. Auf der nächsten Stufe lassen sich auch die Ressourcen anderer Rechendienstleister in das Grid des Hauptrechendienstleisters integrieren. Die Dienstleister geben dabei die Kontrolle über die Nutzung der Ressourcen gegen Entgeltzahlung auf. Diese Situation wird in Abbildung 21 dargestellt. Für den Nutzer des Dienstes gestaltet sich dieser Vorgang transparent.

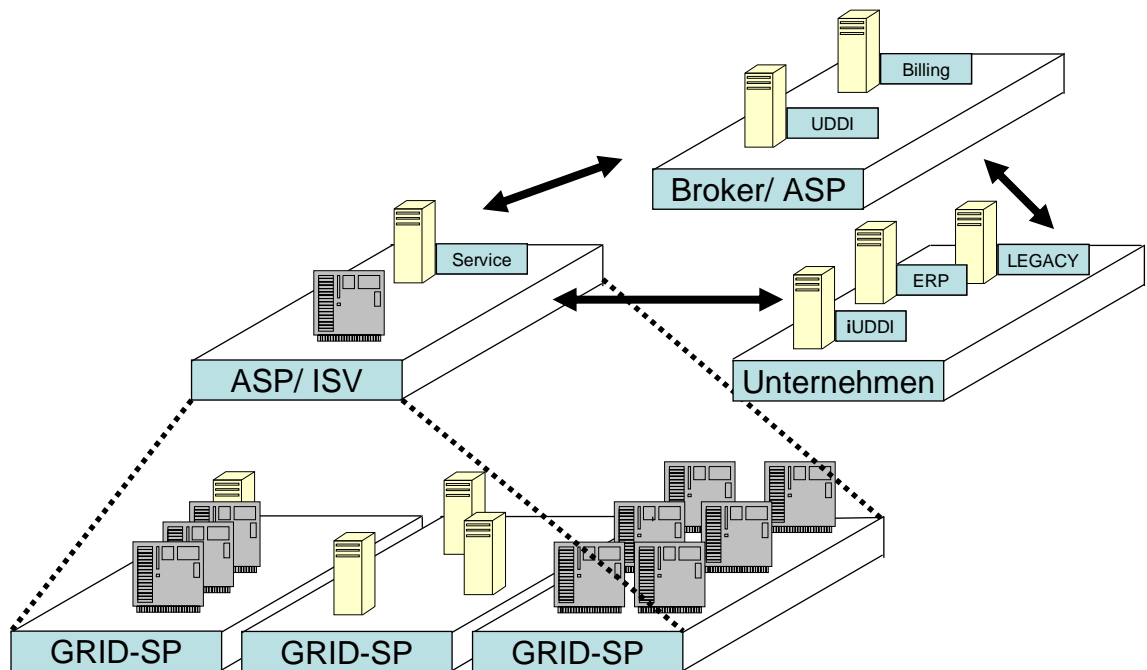


Abbildung 21: Next Generation Service Provider

#### IV.5.4 Abrechnungsvarianten

Im Provisioningfall lassen sich die entgeltliche von der unentgeltlichen zur Verfügungsstellung von Diensten unterscheiden. Die Frage, ob eine Abrechnung nötig wird, entscheidet sich zum einen durch die Abrechenbarkeit des Dienstes und zum anderen dadurch, ob die Abrechnung des Dienstes vom Anbieter beabsichtigt wird bzw. möglich ist. Die Frage der Abrechnung ist unabhängig von der Durchführung ausschlaggebend für das Angebot des Dienstes.

Die Abrechenbarkeit ist zum einen aus technischer und zum anderen aus wirtschaftlicher Sicht zu beurteilen. Während es sich bei der technischen Sicht

um die technische Komplexität der Ermittlung abrechnungsrelevanter Produktmerkmale handelt, ist für die betriebswirtschaftliche Sicht das Verhältnis von Preis der Leistung zu den Kosten für die Durchführung der Abrechnung des Dienstes entscheidend.

Selbst wenn sich ein entgeltliches Angebot aufgrund der Abrechenbarkeit rechtfertigen lassen würde, können andere Gründe, wie z. B. ein kostenloses Anfangsangebot im Rahmen einer Markteintrittsstrategie oder das Angebot des Dienstes als Zusatzleistung einer anderen Leistung, gegen eine Abrechnung sprechen (vergleiche hierzu die Ausführungen in Abschnitt I.1). Die Möglichkeit einer Einnahmenerzielung durch die gleichzeitige Schaltung von Werbeanzeigen im Rahmen der Dienstnutzung ist durch die hier vorliegende Maschine zu Maschine Kommunikation nicht gegeben.

#### ***IV.5.4.1 Kriterien für die Auswahl der Abrechnungsart***

Die Auswahl der Abrechnungsart durch den Anbieter eines Dienstes hängt von mehreren Faktoren ab. Zum einen ist die Art des Dienstes entscheidend. Je umfangreicher bzw. je spezieller die erbrachte Leistungsart und damit je höher die Zahlungsbereitschaft anzusetzen ist, desto höher dürfen demgemäß die mit der Bezahlung des Dienstes verbundenen Transaktionskosten sein. Andererseits spielt die Nutzungsintensität eine wichtige Rolle. Wird ein Dienst nur gelegentlich, oder sogar nur einmalig von einem Nutzer aufgerufen, bieten sich andere Abwicklungswege an, als bei einem häufig genutzten Dienst. Daneben ist auch der Grad der Kundenbindung an den Web Service-Provider entscheidend. Die Nutzungsintensität durch die einzelnen Dienstinhaber kann als Maßgröße dienen, deren Steigerung als ein Indiz für eine höhere Kundenbindung gelten kann. Weiterhin sind die auf dem Markt üblichen Bezahlungssysteme durch den Anbieter zu berücksichtigen und hinsichtlich ihrer Eignung zu evaluieren. Hierbei ist die Frage der Verbreitung der einzelnen Bezahlvarianten bei den potenziellen Nachfragern des Dienstes zu klären.

Zur Auswahl stehen neben „konventionellen“ Abrechnungsvarianten, die schon in anderen Bereichen eingesetzt werden, Abrechnungsvarianten, die erst durch

die besonderen Eigenschaften von elektronischen Dienstleistungen (z. B. Web Services) und Gütern (z. B. elektronischen Publikationen) über das Internet Bedeutung gewonnen haben und für das erfolgreiche und nachhaltige Angebot eine wichtige Rolle spielen.

#### ***IV.5.4.2 Konventionelle Abrechnungsvarianten***

Hervorzuheben ist die Eigenschaft von konventionellen Abrechnungsvarianten sich bei der Bestimmung der zu zahlenden Entgelte nicht direkt an die Nutzung einer Leistung und damit am Ressourcenverbrauch zu orientieren. Es werden Messgrößen gesucht, die sich an der Bereitstellung der Leistung orientieren. So wird die Nutzung nach dem Zeitraum der tatsächlichen Inanspruchnahme eines Dienstes abgerechnet. Alternativ dazu kann auch eine über einen bestimmten Zeitraum unbegrenzte Nutzung vereinbart werden.

Die vorgestellten Abrechnungsvarianten nach der Nutzungszeit haben den Vorteil der Einfachheit gemeinsam. Die Problematik der Einmalnutzung und die betriebswirtschaftlich sinnvolle Abwicklung eines Bezahlvorgangs stellen sich nicht.

Daneben bietet sich auch die Abrechnung nach der Anzahl der einzelnen Nutzungen eines Dienstes an. Hierbei muss jedoch der Preis für die Leistung die Transaktionskosten für die Abwicklung der Abrechnung deutlich übersteigen. Unter Umständen kann die Abrechnung erst am Ende eines Zeitraumes erfolgen. Es wird bei der Nutzung eine über einen bestimmten Zeitraum abonnierte Leistung impliziert, zumindest findet aber eine längere Nutzung statt. Bei komplexen Diensten, oder Diensten mit hohen Volumina bei den einzelnen Nutzern bieten diese Varianten eine einfache Abwicklung, etwa durch die direkte Registrierung des Requestors beim Provider des Dienstes.

Bei näherer Beschäftigung wird jedoch deutlich, dass dies nur bei Diensten mit identischen und konstanten, oder nicht nennenswerten Ressourcenverbrauch bei jeder Nutzung, eine sinnvolle Art der Abrechnung für den Serviceprovider sein kann. Aber selbst dann besteht die Gefahr einer ungleichmäßigen Anfragestruktur. Damit verbunden ist die Notwendigkeit der Abfederung von Last-

spitzen, die für den Provider zusätzliche Kosten bedeutet. Diese Kosten werden an die Kunden in Form von höheren Nutzungsentgelten weitergegeben. Die Varianten sind demnach eher für den Provider und weniger für den Requestor von Vorteil.

#### **IV.5.4.3 Web Service-spezifische Abrechnung**

Die im vorangegangenen Abschnitt behandelten konventionellen Abrechnungsvarianten werden den Nutzungsszenarien der Web Services eher unzureichend gerecht. Im Falle von Web Services kommt es unter Umständen zu der einmaligen Nutzung eines geringwertigen Dienstes. Daneben ist für den Requestor eine nutzungsabhängige und an dem tatsächlichen Ressourcenverbrauch orientierte Abrechnung von Vorteil. Dies ist aus Sicht des Providers neben dem Erfassungsproblem, mit der Notwendigkeit der Identifizierung und Authentifizierung von Dienstenachfragern verbunden. Bei der Web Service-Nutzung entfallen außerdem unter Umständen Entgelte in Höhe eines Bruchteils von Cents. Daneben tritt das Problem auf, dass ein Anbieter mehrere Zahlungssysteme aufgrund der Präferenzen der Nachfrager unterstützen muss und auf der anderen Seite die Nachfrager der Dienste auf Anbieter treffen, die unterschiedlichste Bezahlungsfunktionen nutzen.<sup>210</sup>

Dies wirft zuerst einmal die Frage nach einer geeigneten Handhabung der geringen Geldbeträge auf. Eine Lösung liegt in einem Micropayment-Modell. Dabei werden unter einem Micropayment Beträge verstanden, die unterhalb der minimalen Gebühr anderer gebräuchlicher Bezahlverfahren wie z. B. einer Kreditkartenzahlung liegen, und damit aus Sicht des Anbieters als Träger dieser Gebühren nicht über diese Bezahlwege abgewickelt werden können.<sup>211</sup> Andererseits kann es sich bei Micropayments um Zahlungen handeln, die einen

---

<sup>210</sup> Vgl. hierzu PARHONYI/ PRAS/ QUARTEL (2004), die diese Tatsache aus einer Studie des Verbandes der Deutschen Zeitungsverleger e.V. und Sapient über die Zahlungsbereitschaft der Kunden für digitalen Content ableiten.

<sup>211</sup> Vgl. HERZBERG (2003), S. 245

Bruchteil der kleinsten Einheit einer Währung ausmachen.<sup>212</sup>

Um dennoch eine sinnvolle Abwicklung gewährleisten zu können, werden Systeme eingeführt, die Transaktionen entweder auf Basis einer Vorauszahlung abwickeln, oder die entstandenen Forderungen dem Nutzer des Micropayment-Systems später in Rechnung stellen.<sup>213</sup> Während bei der ersten, wenig erfolgreichen Generation von Micropayment-Systemen der Prepaid-Fall in Form von elektronischem Geld dominierte, handelt es sich bei der zweiten Generation um Systeme, die auf Konten aufgelaufene Kosten nach einer gewissen Zeitperiode abrechnen.<sup>214</sup> Abrechnungssysteme der ersten Generation sind nicht zuletzt wegen komplexer Verwaltungslösungen für das digitale Geld vom Markt verschwunden.<sup>215</sup> Weitere Unterscheidungsmerkmale ergeben sich aus dem Ablauf der Transaktion, der Höhe der Zahlungen, der Art des Mechanismus zur Gewährleistung der Durchführung der Zahlungen, Sicherheitsmechanismen, Transaktionskosten, der Währungsproblematik und der Interaktion zwischen verschiedenen Payment-Service-Providern.<sup>216</sup>

#### **IV.5.4.4 Abrechnung als Dienst**

Die Übernahme der Abrechnung kann selbst wieder als Dienst angeboten werden. Bei der Abrechnung handelt es sich um eine standardisierte Leistung, die auf eine Vielzahl von Services angewendet werden kann. Insbesondere sind verschiedene Abrechnungsvarianten bei dem Dienst implementierbar. Dabei müssen sich sowohl Provider als auch Requestor bei dem Abrechnungsprovider registrieren. Diese Tatsache birgt den Vorteil in sich, dass sich bei einer genügend großen Zahl an Kunden des Bezahlproviders eine Lösung für das Problem der eindeutigen Identifikation des Kunden bei dem Diensteanbieter ergibt. Der Kunde erhält durch die Registrierung genauso wie der Provider eine

---

<sup>212</sup> Vgl. KOU (2003), S. 305

<sup>213</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

<sup>214</sup> Vgl. PARHONYI/ NIEUWENHUIS/ PRAS (2005)

<sup>215</sup> Vgl. PARHONYI/ NIEUWENHUIS/ PRAS (2005)

<sup>216</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

Identität, die es unter anderem möglich macht, ein Ranking der Zuverlässigkeit der angebotenen Dienste bzw. auch der Zahlungsmoral des Kunden zu erstellen.

Insbesondere das erfolgreiche Beispiel des Unternehmens firstgate, das die Abrechnung von im Internet angebotenen Publikationen automatisiert hat, zeigt, dass sich einfachere Lösungen unter Umständen sogar deutlich besser am Markt durchsetzen als zu komplexe Lösungen. Firstgate erzeugt einen Link zu einer Publikation des Anbieters und gewährt den Kunden erst dann Zugriff, falls dieser als Kunde bei Firstgate identifiziert wurde oder er sich vor dem Zugriff registriert.

Sowohl bei Kunden als auch aufseiten des Providers kommt es jedoch zu erheblichen Mehraufwendungen, falls mehrere Bezahlssysteme gleichzeitig bedient werden müssen.<sup>217</sup> Diese Problematik tritt selbst dann auf, falls zwar ein verbreiteter lokaler Bezahlendienst existiert, aber die Dienstleistung des Providers grenzüberschreitend angeboten werden soll, was bei elektronischen Dienstleistungen nicht ungewöhnlich ist.<sup>218</sup> Abhilfe kann hier beispielsweise ein von PARHONYI/ PRAS/ QUARTEL (2004) vorgeschlagener Payment-Gateway-Provider ermöglichen, durch den sowohl auf Kunden- als auch auf Providerseite mehrere von bei diesem Gateway unterstützte Bezahlssystemprovider für die Abrechnung des gewünschten Dienstes angesprochen werden können.<sup>219</sup> Der Vorschlag senkt die Kosten für die Implementierung bei Provider und Requestor und macht sich die Vorteile aus der Tatsache, dass die einzelnen Payment-Provider jeweils die lokalen Rechtsvorschriften beachten, und einer größeren Menge an unterstützten Bezahlssystemen zunutze.<sup>220</sup>

---

<sup>217</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

<sup>218</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

<sup>219</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

<sup>220</sup> Vgl. PARHONYI/ PRAS/ QUARTEL (2004)

## V. Prototypische Umgebungen

Die prototypischen Umgebungen dienen der Verdeutlichung einzelner Lösungsmöglichkeiten aus dem Bereich der stateful und stateless Web Services, die für die IT-Sourcing-Szenariotypen wichtig sind. Insbesondere werden Integrationspunkte zwischen den genutzten Web Services und Grid-Plattformen gezeigt, die eine integrierte Umgebung für einen Next Generation Service Provider möglich machen. Den Ausgangspunkt bildet eine ERP-Umgebung ohne Web Service-Einsatz in Verbindung mit einem Integrationsserver. Anschließend wird je eine Web Service- und eine Grid-Umgebung vorgestellt, bevor diese in eine integrierte Umgebung eingehen.

Bei der Implementierung wurde bis auf das innerbetriebliche Enterprise Resource Planning-Szenario auf Open Source-Komponenten und -Betriebssystemumgebungen gesetzt. Neben der sich daraus ergebenden Kostenersparnis kommt bei Anwendungen aus Gebieten mit intensiver Forschungstätigkeit und aktiven Communities der schnellere Zugriff auf Software hinzu. Auf der anderen Seite werden Teile der Open Source-Entwicklungen später in kommerzielle Produkte integriert, sodass sich auch Erkenntnisse über zukünftige Entwicklungen erzielen lassen.

### V.1 *Enterprise Resource Planning*

Die innerbetriebliche Umgestaltung der Anwendungslandschaft soll am Beispiel eines ERP-Systems verdeutlicht werden. Im Speziellen wird hier ein SAP R/3-System Release 4.6c verwendet. Das Release erlaubt noch keine direkte Kommunikation über Web Services, ist jedoch ein aktuelles Release, das in Unternehmen eingesetzt wird. Eine im März 2005 erschienene Umfrage unter 300 Mitgliedsunternehmen der Deutschsprachigen SAP Anwendergruppe (DSAG) ergab unter anderem, dass noch 57,5 % der befragten Unternehmen



mit dem SAP R/3 Release 4.6c produktiv arbeiteten.<sup>221</sup>

### V.1.1 Rahmenbedingungen

Wird in einem Unternehmen ein ERP-System betrieben, so existieren meistens noch Systeme für spezielle Einsatzgebiete nach dem Best-of-Breed Prinzip. Es werden also beispielsweise Speziallösungen für die Produktionsplanung mit einer hohen Bedarfsabdeckung neben einem ERP-System betrieben, das auch Module für die Produktionsplanung mit nicht ausreichender Funktionalität bereitstellen würde. Des Weiteren gibt es unter Umständen noch Altsysteme, die kritische Teile des Geschäfts abbilden und in absehbarer Zeit nicht ersetzt werden sollen. Daneben kommen noch verschiedene Unternehmensstandorte hinzu, die an das ERP-System anzubinden sind. Kunden und externe Geschäftspartner sollen an die Systeme des Unternehmens angebunden werden. Ein weiteres Beispiel wurde schon im Rahmen des motivierenden Szenarios in Abschnitt I.3 gegeben.

Diese kurzen Szenarien verdeutlichen, dass selbst bei Konzentration auf den engeren Bereich des eigenen Unternehmens unterschiedlichste Anforderungen an die Anwendungsintegration zu stellen sind. Durch die nahezu überall verfügbare und kostengünstige Internetinfrastruktur ergeben sich schon im Falle der Anbindung geographisch getrennter Standorte, Anforderungen aus unterschiedlichen Schnittstellenformaten oder Sicherheitsstandards, wie man sie erst bei einer stärkeren Öffnung der Unternehmens-IT für Externe erwarten würde.

Das realisierte Szenario soll folgenden Kriterien genügen, die sich aus dem vorher erläuterten Szenarien ableiten:

- Verwendung eines Middleware-gestützten Ansatzes zum Datenaustausch: Durch den Middleware-Ansatz soll auf eine Umgestaltung existenter Applikationen verzichtet werden.
- Verwendung von XML-Dialekten zum Datenaustausch zwischen SAP- und nicht SAP-Systemen über das Internet. Diese Vorgehensweise soll

---

<sup>221</sup> Vgl. DSAG (2005), S. 2

den Übergang zu Web Services erleichtern.

- Nutzung verschiedener internetbasierter Übertragungsprotokolle wie http. Diese Protokolle werden auch bei Web Services eingesetzt und verwendet und gewährleisten die Existenz bzw. den Aufbau einer zukunfts-trächtigen Infrastruktur. Beispielsweise wird schon eine geeignete Firewall-Technologie aufgebaut.
- Flexibilität bei der Bestimmung der Kommunikationspartner. Der Grad an Flexibilität muss jedoch noch nicht so hoch wie bei Web Services sein.
- Möglichkeiten zur Transformation der Nachrichten, um die Nachrichten an die Erfordernisse des Kommunikationspartners anzupassen. Der hier angesprochene Mapping- und Transformationsprozess muss selbst bei Web Services noch erfolgen.

### V.1.2 Datenaustausch mit einem Integrationsserver

Zur Durchführung der Anbindung des SAP-Systems wurde der SAP Business Connector in den Versionen 4.0 bis 4.7 eingesetzt.

Der Business Connector verfügt über einen SAP-Adapter, der den ursprünglich von WebMethods stammenden Business to Business Integration Server die Kommunikation mit einem SAP-System über Remote Function Call (RFC)<sup>222</sup> bzw. transaktionalen RFC, über eine native Unterstützung von Business Application Programming Interface (BAPI)<sup>223</sup> und Intermediate Document (IDoc) erlaubt (siehe Abbildung 23).<sup>224</sup> Ein IDoc ist ein SAP-eigener Standard für Nachrichten, die zwischen SAP Systemen ausgetauscht werden, und enthält

---

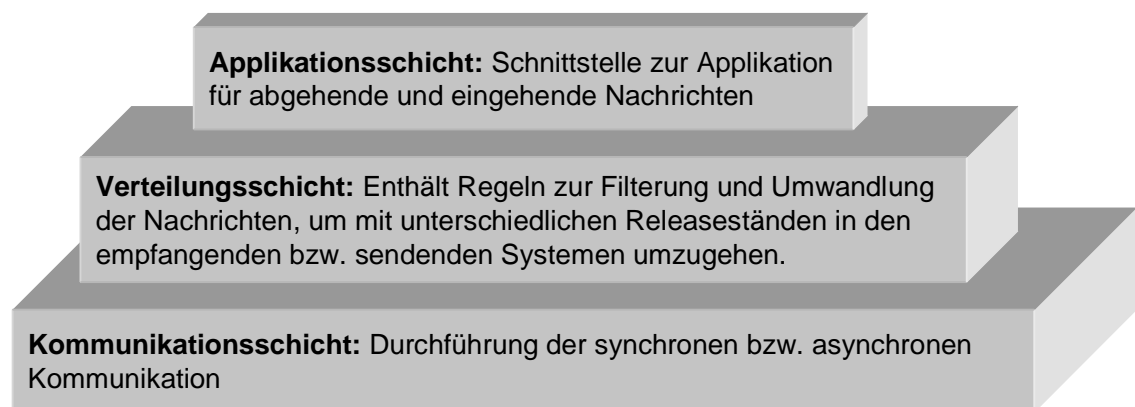
<sup>222</sup> Ein Remote Function Call dient in einer SAP Systemumgebung zum synchronen oder asynchronen Aufruf eines Funktionsbausteins in einem entfernten SAP System oder zum Aufruf eines externen Programmes. (Vgl. HOMBERGER/ SCHNEIDER (2000), S. 127)

<sup>223</sup> Ein Business Application Programming Interface stellt verschiedene Zugriffstechniken zu den SAP Business Objekten, wie z.B. RFC, zur Verfügung. (Vgl. BUCK-EMDEN (1999), S. 245)

<sup>224</sup> Vgl. SAP (2001), S. 2-9 und 2-6

Informationen aus einem Business Objekt.<sup>225</sup>

RFCs und IDocs sind die Grundlage für Application Link Enabling (ALE), dessen Ziele die lose Kopplung von Anwendungen auf autonomen SAP-Systemen und die Anbindung von nicht SAP-Systemen zur Realisierung verteilter Geschäftsprozesse sind.<sup>226</sup> ALE lässt sich durch den schichtenartigen Aufbau, der in der Abbildung 22 dargestellt ist, in die Anwendungs-, Kommunikations- und Verteilungsschicht zerlegen.<sup>227</sup>



*Abbildung 22: Schichtenaufteilung bei Application Link Enabling  
[eigene Darstellung nach LINTHICUM (2001), S. 358 f.]*

Im SAP-System wurde die Kommunikation im Rahmen der Stammdatenverteilung über ALE und bei remote-fähigen Funktionsbausteinen, also bei Funktionen ohne Benutzerdialog, durchgeführt. Im Speziellen wurde die Stammdatenverteilung von Materialstammsätzen durchgeführt, etwa um in verschiedenen SAP-Systemen den gleichen Stammdatenbestand innerhalb der Logistik zu gewährleisten. Das Customizing aufseiten des SAP-Systems unterscheidet sich dabei nicht von der direkten Verbindung zweier Systeme innerhalb eines Administrationsbereichs. Logisches System und RFC-Destination sind die einzigen in Bezug zum Business Connector unterschiedlich zu

<sup>225</sup> Vgl. LINTHICUM (2001), S. 359

<sup>226</sup> Vgl. HOMBERGER/ SCHNEIDER (2000), S. 127 und BUCK-EMDEN (1999), S. 240 und LINTHICUM (2001), S. 357

<sup>227</sup> Vgl. LINTHICUM (2001), S. 358

pflegenden Parameter.<sup>228</sup> Auf der anderen Seite können innerhalb des Business Connector neben einer Namenstransformation, um Parameternamen zu ändern, Werttransformationen (beispielsweise am Datumsformat) und Strukturtransformationen an der Nachricht vorgenommen werden.<sup>229</sup> Durch die Transformationen, die im Falle komplexer Veränderungen auch über JAVA-Services oder innerhalb einer komplexen Kette verschiedenartiger Transformationen als so genannter Flow-Service (Service in Abbildung 23) geschehen können, lassen sich nicht nur verschiedenartige Fremdsysteme anbinden.<sup>230</sup> Es ist auch möglich, Versionsgrenzen zwischen verschiedenen SAP-Systemreleases zu überwinden.

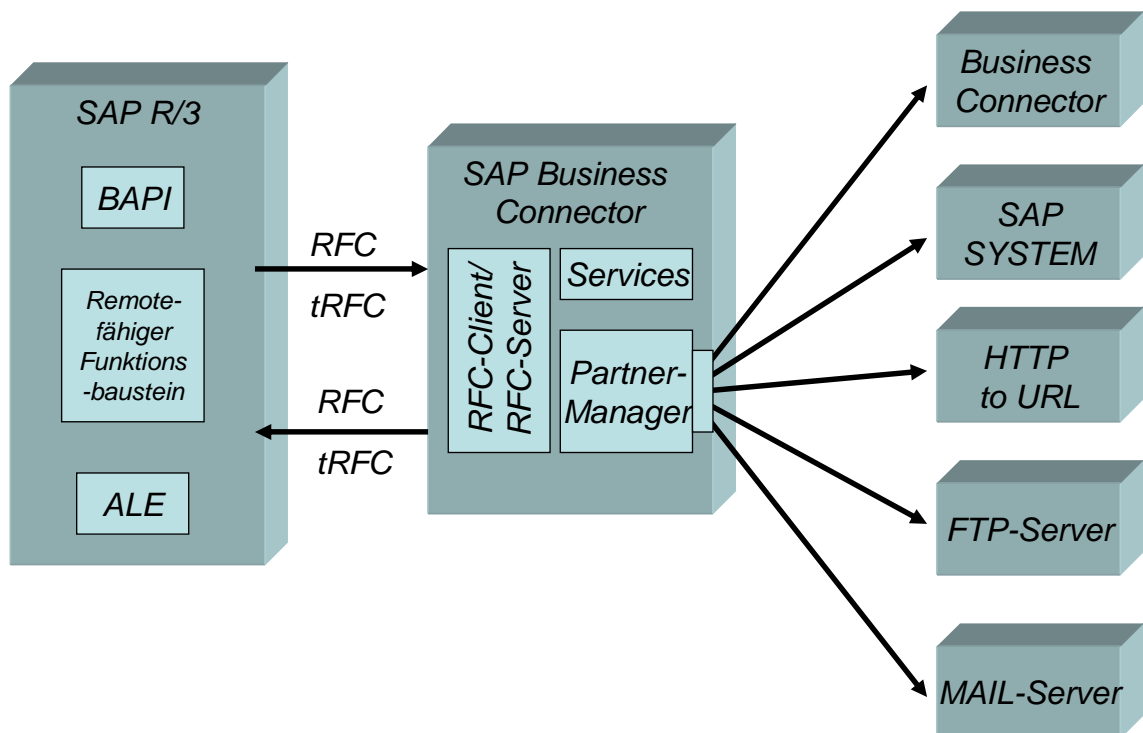


Abbildung 23: Architektur und Einsatzbereiche des SAP Business Connectors  
[Quellen: SAP 2005a und SAP 2005b]

Bei der Durchführung der Kommunikation unter Verwendung von Funktions-

<sup>228</sup> Vgl. SAP (2001), S. 3-20 und 3-24

<sup>229</sup> Vgl. SAP (2001), S. 4-23

<sup>230</sup> Vgl. SAP (2001), S. 4-37

bausteinen zeigt sich ein weiterer Aspekt der Kommunikation über einen Integrationsserver. Durch gezielte Erweiterung eines RFC-Aufrufs über empfangerspezifische Parameter lässt sich im Business Connector eine dynamische Partnerfindung nach im Partnermanager hinterlegten Regeln durchführen.<sup>231</sup>

Vor allem bei der Kommunikation mit Fremdsystemen und abhängig von der Art der Öffnung der Systemlandschaften zum Internet können unterschiedliche Transportarten für die Nachrichten realisiert werden. Neben der http-Kommunikation stehen FTP und SMTP aus dem Business Connector auf dem empfangenden und auf dem sendenden Nachrichtenweg zur Verfügung (siehe Abbildung 23).<sup>232</sup>

### V.1.3 Auswertung

Die durchgeführten Untersuchungen wurden sowohl auf der eher technischen Ebene der Nutzung von RFCs zum Aufruf kleinerer Funktionen auf entfernten Systemen, als auch im Rahmen verteilter Geschäftsprozesse mit ALE-Technik verwendet.

Durch den SAP Business Connector lassen sich einige Eigenschaften der Serviceorientierung umsetzen. Obwohl der Business Connector die Möglichkeit zum Aufbau von Web Services bietet, wurde hiervon kein Gebrauch gemacht. Web Services bilden zwar eine sehr gute Umsetzungsmöglichkeit für eine service-orientierte Architektur, sie sind aber keine direkte Anforderung, die sich aus der Architektur ergibt.<sup>233</sup> Insbesondere konnten Mechanismen zur automatischen Generierung von Schnittstellen und Möglichkeiten zur dynamischen Partnerfindung auch ohne Web Services realisiert werden.

So können für die Generierung der recordähnlichen Datenstrukturen innerhalb des Modellierungswerkzeuges SAP Business Connector Developer XML-

---

<sup>231</sup> Vgl. SAP (2001), S. 3-27

<sup>232</sup> Vgl. SAP (2001), S. 2-9

<sup>233</sup> Vgl. ERL (2005), S. 37

Schema Dokumente verwendet werden, die sich entweder in einem SAP-System über die Transaktion WE60 generieren lassen bzw. von Partnern zur Verfügung gestellt, oder vom Hersteller des Systems, mit dem kommuniziert werden soll, bezogen werden können. Im Falle von eingehenden Nachrichten in ein SAP-System kann auf das SAP Repository direkt zugegriffen werden, um die Schnittstellenstruktur für den Aufruf eines Funktionsbausteins bzw. zum Aufruf eines BAPIs zu ermitteln. Die dynamische Partnerfindung lässt sich über die Konfiguration des Partner Managers im SAP Business Connector erreichen, ohne im SAP System selbst Anpassungen vornehmen zu müssen.

## **V.2 Web Service-Umgebung**

Die Umgebung für Web Services wird auf Basis des Apache AXIS Framework auf einem Tomcat Applikationsserver aufgebaut. Die UDDI-Registry basiert ebenfalls auf AXIS und wird als unternehmensinterne Registry aufgebaut. Zur Demonstration der Interoperabilität wurde auch mit der Umsetzung von Web Service-Clients und -Servern unter PHP gearbeitet.

Im folgenden Abschnitt werden nach der technischen Architektur einer Web Service-Umgebung die Programmiersprachenunabhängigkeit und eine Möglichkeit zur Integration von Web Services in eine Portallösung präsentiert.

### **V.2.1 Technische Architektur und Technologie**

Apache AXIS kann als Plattform für Web Services dienen, die entweder in der Programmiersprache C oder, wie im Rahmen der Arbeit geschehen, in Java implementiert werden. Bei der Java-Variante wird eine vom Transport Listener empfangene SOAP-Nachricht durch den AXIS-Server in ein MessageContext Object transferiert, das als Repräsentanz für die XML-Nachricht dient.<sup>234</sup> Wie die Abbildung 24 zeigt, durchläuft jedes MessageContext Object sequenziell, in

---

<sup>234</sup> Vgl. DEITEIL u. a. (2003), S. 110

Abhängigkeit von der Konfiguration des Servers und bestimmter Parameter im repräsentierenden Objekt, eine Folge sogenannter Handler bzw. von Chains als Zusammenfassung von Handlern mit dem Ziel, Transformationen an dem Request bzw. der Response vorzunehmen.<sup>235</sup>

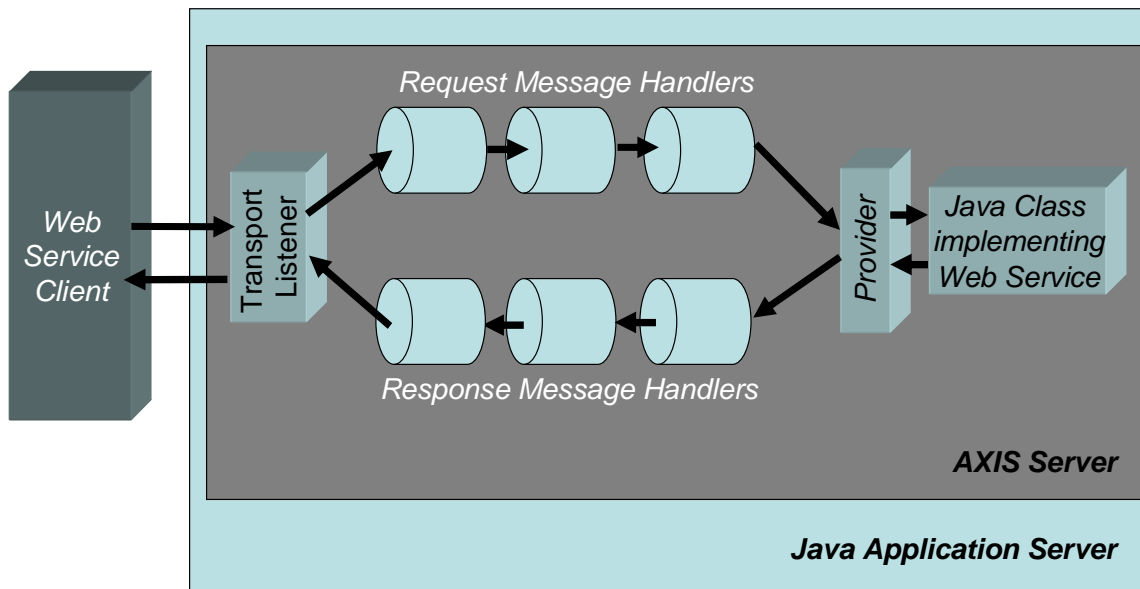


Abbildung 24: Schematischer Aufbau des AXIS Servers auf einem Applikationsserver [Quelle: DEITEL u. a. (2003), S. 111]

Die Handler lassen sich auf der eingehenden und ausgehenden Seite gleichermaßen in transportspezifische, globale bzw. servicespezifische Handler-Ketten einteilen, die zum Provider als letzten Handler hinführen bzw. von ihm wegführen.<sup>236</sup> Die verschiedenen Handler(-Ketten) lassen sich dabei wie folgt charakterisieren:<sup>237</sup>

- Abhängig vom Transportprotokoll der eingehenden bzw. ausgehenden Nachricht erlauben transportspezifische Handler-Ketten Verarbeitungsschritte mit Handlern auszuführen, die für das verwendete Protokoll geeignet sind. Dabei durchlaufen die Handler auf der eingehenden Seite

<sup>235</sup> Vgl. WANG u. a. (2004), S. 278

<sup>236</sup> Vgl. WANG u. a. (2004), S. 280

<sup>237</sup> Vgl. WANG u. a. (2004), S. 280 f.

Objekte des Typs `MessageContext` und auf der ausgehenden Seite Objekte des Typs `Message`, die vom Provider erzeugt wurden.

- Die globale Handler-Kette nimmt transport- und serviceunspezifische Aktionen vor.
- Die servicespezifische Handler-Kette ermöglicht individuell auf den Service zugeschnittene Verarbeitungsschritte.
- Für die Kommunikation mit dem Java Web Service ist der Provider zuständig.

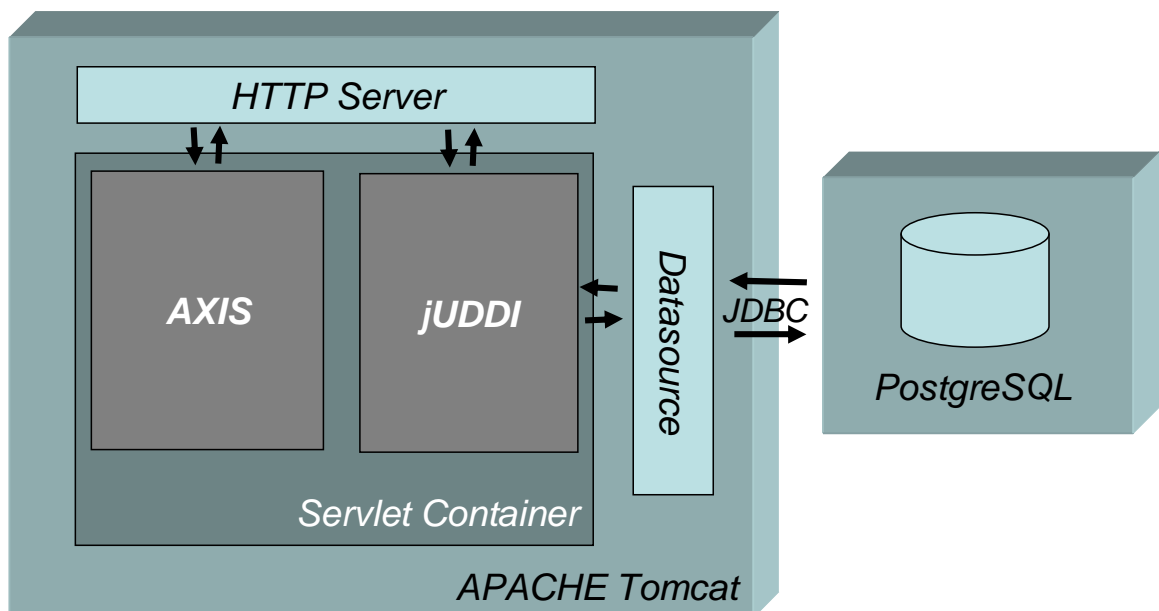
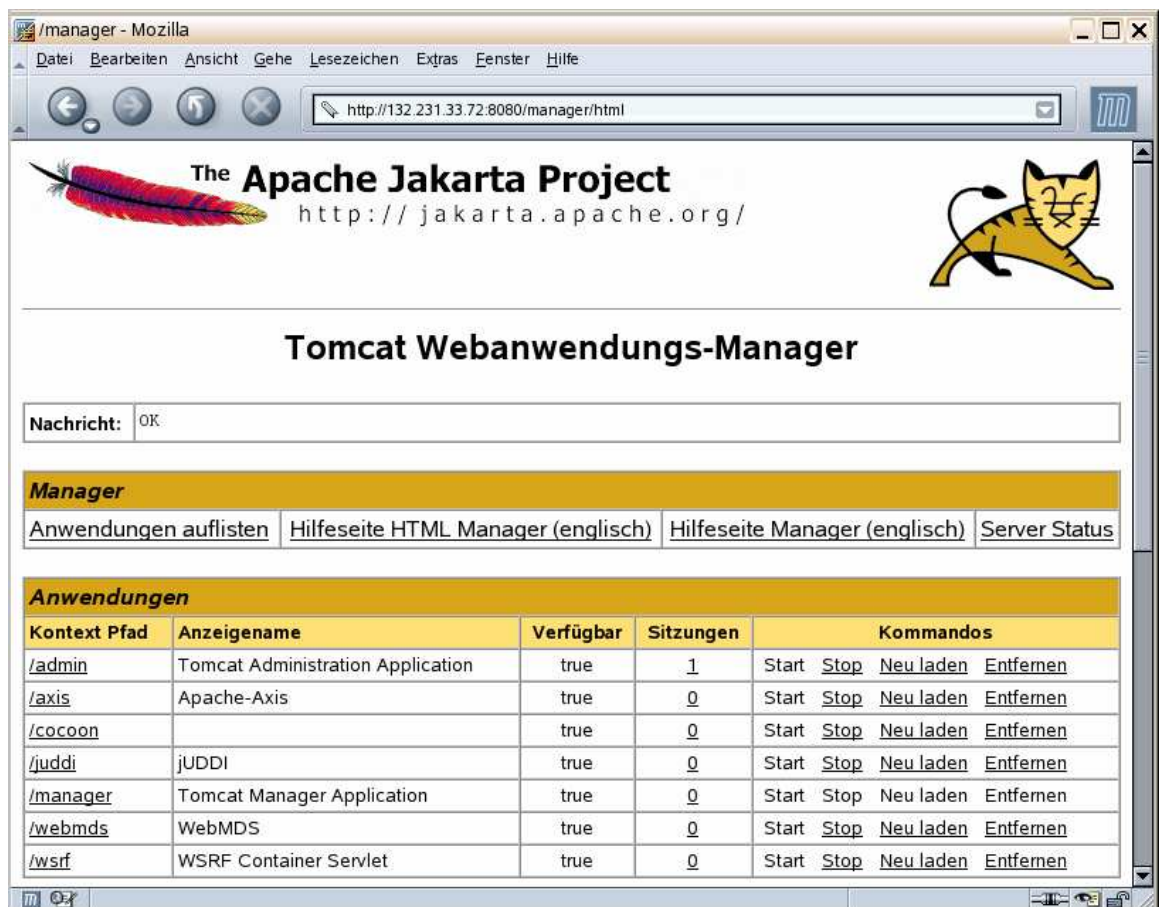


Abbildung 25: Konfiguration der Web Service-Umgebung aus Apache AXIS und jUDDI auf einem Apache Tomcat Server mit Anbindung an eine PostgreSQL-Datenbank

Sowohl AXIS als auch jUDDI eine Open Source-Implementierung einer UDDI Registry in der Version 2 des UDDI-Standards werden auf einem Apache Tomcat-Server betrieben (vgl. Abbildung 25). Sie laufen innerhalb verschiedener Kontexte im Servlet-Container (vgl. Abbildung 26). Zur Gewährleistung der persistenten Speicherung der Einträge in der Registry wird ein PostgreSQL Datenbankserver über eine JDBC-Schnittstelle als Datenquelle für die jUDDI Web Applikation zur Verfügung gestellt. Der Zugriff auf das UDDI-Verzeichnis



erfolgte mithilfe der UDDI4J Java-Klassenbibliothek.



*Abbildung 26: Verwaltungswerkzeug für Webapplikationen auf dem Apache Tomcat Server*

Daneben wird ein Apache http-Server in der Version 2.x eingesetzt, der die Skriptsprache PHP 5 als Modul anbietet und für PHP-Web Service-Server und -Client benötigt wird.

Für die Integration der Web Services in ein Portal ist das Cocoon 2.x Framework auf dem Tomcat-Applikationsserver installiert (vgl. Abbildung 26). Cocoon bietet neben Funktionalitäten zur medienneutralen Publikation von Inhalten auch die Möglichkeit zum direkten Zugriff auf Web Services und eine Portal-Umgebung. Das Portal wird für die Integration des PHP-Web Service-Clients verwendet.

## V.2.2 Programmiersprachenunabhängigkeit

Eine wichtige Eigenschaft von Web Services ist die Tatsache, dass die Implementierung des Services weitestgehend als Blackbox betrachtet wird. Für den Nutzer des Services sind nur die zu versendenden und empfangenen Nachrichten wichtig. Damit liegt es nahe, dass die Implementierung des Services in einer beliebigen Programmiersprache erfolgen kann, die den standardkonformen Nachrichtenaustausch ermöglicht und die benötigten Web Service Protokollstandards unterstützt, die für die Aufgabe benötigt werden. Die Wahl der Programmiersprache wird weiterhin maßgeblich durch das im Unternehmen bzw. das für ein Projekt verfügbare Know-how bestimmt. Daneben sind die im Unternehmen eingesetzten Applikationen in den Auswahlprozess einzubeziehen, da hierdurch unter Umständen nur spezielle Schnittstellen zu Programmiersprachen zur Verfügung stehen.

Um diese Tatsache im Rahmen der prototypischen Umgebung zu dokumentieren, wird ein einfach strukturierter Service zum Auslesen von Bauteilen eines Produktes aus einer Datenbank zunächst in PHP umgesetzt. Anschließend wird die Implementierung in der für Web Services gebräuchlichen Programmiersprache JAVA dargestellt.

Bei der Implementierung fällt auf, dass bis auf die Einbindung der PHP-Datei mit den Klassen von NuSOAP nur wenige Zeilen zur Programmierung eines einfachen Services nötig sind. Im Vergleich zum Funktionsumfang eines mit Apache AXIS entwickelten Services sind zwar nur eingeschränkte Funktionalitäten realisierbar. Dafür können aber die Schnelligkeit der Umsetzung und die Möglichkeit für andere Web Seiten Funktionalität anbieten zu können, ins Feld geführt werden.

```
<?PHP
require_once "nusoap.php";
$server = new soap_server;
$server->register('get_bom');
$server->service($HTTP_RAW_POST_DATA);

function get_bom($material)
{
    $connect = pg_connect("host=132.231.33.72 port=5432
                        dbname=material user=username
                        password=password");
    If (!$connect) return Error;

    $query="select unterteil from bom where oberteil='"
        .$material."'";
    $res = pg_query($connect,$query);

    $result=pg_fetch_all($res);
    return $result;
}
?>
```

*Abbildung 27: Quellcode-Beispiel für einen mit NuSOAP  
implementierten PHP Web Service*

Im obigen Quellcode-Abschnitt (vgl. Abbildung 27) wird eine Instanz der `soap_server`-Klasse erzeugt, eine Methode `get_bom` registriert und die Transportart instanziiert. Anschließend wird die Methode als `function` programmiert. Es wird eine Datenbankverbindung hergestellt. Falls die Verbindung besteht, wird eine SQL-Abfrage zur Ermittlung der Bestandteile einer einstufigen Baukastenstückliste durchgeführt. Das Ergebnis der Abfrage wird als `Array` durch den Web Service zurückgeliefert.

Ähnlich gestaltet sich die Umsetzung des äquivalenten Dienstes unter Apache AXIS. Im Unterschied zu der PHP-Variante muss hier keinerlei Modifikation für die Serverumsetzung gemacht werden. Die JAVA-Klasse enthält die Methode `getBOM`, die über eine JDBC-Anbindung eine Verbindung zu der `mySQL`-Datenbank herstellt und anschließend die Anfrage absetzt.

```
import java.util.Vector;
import java.sql.*;

public class BOM {
    public Vector getBOM(String material) throws Exception {

        Statement statement = null;
        ResultSet result = null;
        Connection connect = null;
        Vector results = new Vector();

        try {
            Class.forName("com.mysql.jdbc.Driver").newInstance();

            connect = DriverManager.getConnection(
                "jdbc:mysql://132.231.33.72:3306/material",
                "user", "password");
            String query = "select upp from bom where up =" + material;
            statement = connect.createStatement();
            result = statement.executeQuery(query);

            do {
                results.add(result.getString("ut"));
            } while (result.next());

            result.close();
            statement.close();
            connect.close();

        } catch (Exception e)
        {
            results.add(e.getMessage());
        }
        return results;
    }
}
```

*Abbildung 28: Quellcode-Beispiel für einen JAVA-Web Service*

Das Ergebnis der Abfrage wird als Objekt vom Typ Vector zur Verfügung gestellt und anschließend von AXIS automatisch in einer SOAP-Nachricht verpackt und an den Service-Nutzer geschickt.

### V.2.3 Integration von Web Services in ein Portal

Zur Integration von Web Services in ein Portal bieten sich verschiedene Technologien an. Welche Technologie letztlich Anwendung findet, hängt von den Rahmenbedingungen ab. Komplexe Anforderungen zum Design von wiederverwendbaren Oberflächen für Web Services können mithilfe des Standards *Web Services for Remote Portlets* (=WSRP) realisiert werden.<sup>238</sup>

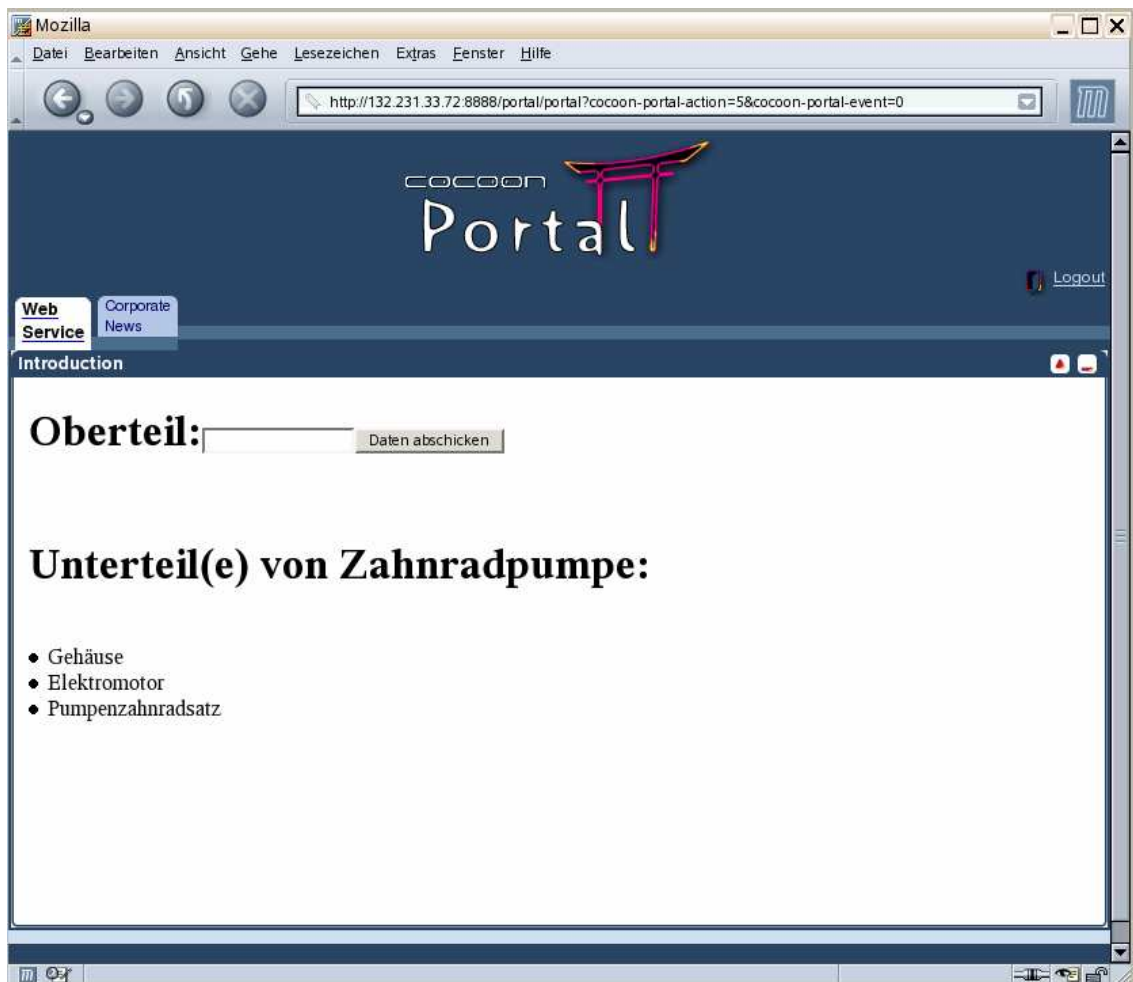


Abbildung 29: Integration eines PHP Web Service Clients in das Apache Cocoon-Portal

Werden einfachere Anforderungen gestellt, so kann man den Aufruf eines Web Services beispielsweise direkt aus einer mit PHP generierten Web Seite starten und das Ergebnis mit den Mitteln der Skriptsprache verarbeiten. Auf der anderen Seite kann bei geringen Anforderungen an die Funktionalität des Web Services, auch der gesamte Service in PHP implementiert werden. Die erwähnten Funktionalitäten sind mittlerweile in den Sprachumfang von PHP eingegangen. Hier wird jedoch exemplarisch die Implementierung mit Hilfe von NuSOAP auf Basis von SOAP Version 1.1 demonstriert.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<HTML>
<HEAD>
<TITLE>PHP-Web Serviceaufruf</TITLE>
</HEAD>
<?PHP
echo '<FORM ACTION=http://132.231.33.72/nusoap/client.php
      method="POST">';
echo '<H1>Oberteil:<input type="text" name="material">';
echo '<input type="submit" name="Abschicken"></H1>';
echo '<br>';

if (!empty($_POST['material']))
{
    $parameter[]=$_POST['material'];
    require_once "nusoap.php";
    $client =
        new soapclientp("http://132.231.33.72/nusoap/server.php");
    $result = $client->call('get_bom',$parameter);

    echo '<H1>Unterteil(e) von '.$_POST['material'].':</H1><BR>';
    foreach ($result as $value)
    {
        echo '<LI>'.$value['unterteil'].'</LI>';
    }
}
?>
</HTML>
```

*Abbildung 30: Quellcode-Beispiel zu einem  
Web Service-Aufruf mit PHP*

Der eigentliche Aufruf des Web Services erfolgt nachdem eine Instanz des Objekttyps `soap_client` erzeugt wurde. Beim Aufruf der Methode `call` wird der aufzurufende Service angegeben und eventuell vorhandene Parameter in Form eines Arrays (hier: *\$parameter*) übergeben. Der Rest der PHP-Datei ist konventionell aufgebaut und dient neben der Ausgabe eines Formularfeldes zur Ausgabe der Ergebnisse des Web Services.

## V.2.4 Zusammenfassung der Ergebnisse

Im Rahmen der Neuerstellung von Web Services bieten sich verschiedene Möglichkeiten an, die abhängig von der Komplexität der Aufgabe unterschiedlich hohen Implementierungsaufwand bedeuten.

Einfache Services lassen sich sogar mit den Mitteln einer Skriptsprache wie PHP umsetzen. Auf der anderen Seite bietet PHP die Möglichkeit, externe Web Services zu konsumieren und die Ergebnisse in eine Portalumgebung einzubinden. Komplexere Dienste lassen sich beispielsweise mit JAVA umsetzen. Bei der Implementierung mit Apache AXIS fällt auf, dass keinerlei Implementierungsaufwand auf die Integration des Services in die Umgebung auf dem Applikationsserver entfällt. Falls jedoch Anforderungen in Form von Logging von Daten und der Nutzung einer permanenten Datenbankverbindung in Form der Datenquelle des Applikationsservers entstehen, können diese durch den Web Service genutzt werden. Die direkte Anbindung der Datenbank in dem Beispiel-Web Service wurde zu Zwecken der Anschaulichkeit und der Vergleichbarkeit der Implementierung mit dem PHP Web Service gewählt. Der Nachteil dieses Vorgehens ist jedoch, dass bei jedem Aufruf des Web Services erneut ein Verbindungsaufbau mit der Datenbank erfolgen muss. Falls demgegenüber die Datenquelle des Applikationsservers genutzt wird, kann über komfortable Verwaltungswerkzeuge beispielsweise ein anderes Datenbankmanagementsystem eingestellt werden, ohne den Quellcode des Web Services ändern zu müssen.

Gerade die einfache Umsetzung von Web Services ruft unter Umständen eine Vielzahl an Diensten hervor, die entweder nur temporär angeboten werden

sollen oder nach einer gewissen Zeit nicht mehr gepflegt werden. Neben der Nutzung des UDDI-Verzeichnisses zur Registrierung von Diensten muss deshalb immer auch die fortlaufende Pflege des Verzeichnisses betrieben werden. Mit anderen Worten bedeutet dies auch das Löschen von Diensten, die nicht mehr zur Verfügung stehen. Die Möglichkeit der Deregistrierung steht zwar jedem Publisher (also jedem Provider eines Services) zur Verfügung, es ist jedoch nicht automatisch davon auszugehen, dass hiervon auch Gebrauch gemacht wird. Es muss deshalb die Aufgabe des Brokers als Organisation sein, die Dienste regelmäßig einer Kontrolle zu unterziehen. Ein weiterer Grund zur Löschung eines Dienstes kann durch Informationen gegeben sein, an die der Broker gelangt. Dies sind vor allem Nachrichten, die gegen die Zuverlässigkeit des Providers sprechen oder sogar auf strafrechtlich relevante Delikte hindeuten. Ein weiterer wichtiger Punkt ist die Verhinderung von Datenmüll im Verzeichnis, der beispielsweise durch unvollständige Einträge entsteht, oder durch Tests der registrierten Publisher verursacht wird.

### **V.3 Grid-Umgebung**

Die Grid-Umgebung wurde mit Hilfe des Globus Toolkits der Globus Alliance aufgebaut. Ziel der Globus Toolkits ist die Verwaltung von verteilten Ressourcen in wissenschaftlichen und betrieblichen Anwendungsbereichen, wobei es sich bei einer Ressource sowohl um Rechner, Speicherplatz, Daten, Dienste, Netze oder auch um Sensoren handeln kann.<sup>239</sup>

Im Folgenden werden zunächst praxisrelevante Anforderungen an die Grid-Umgebung definiert, bevor auf die konkrete Technologie des Globus Toolkit eingegangen wird und die Lösungsmöglichkeiten für die formulierten Anforderungen präsentiert werden. Anschließend wird das Web Services Resource Framework vorgestellt.

---

<sup>239</sup> Vgl. FOSTER (2005), S. 3



### V.3.1 Praxisrelevante Anforderungen

Im Rahmen des beschriebenen Szenariotyps für einen Grid Service Provider spielt die automatische Bereitstellung von Diensten auf einem Grid eine zentrale Rolle. Die von Qi u. a. (2006) vorgeschlagene High Available Dynamic Deployment Infrastructure for Globus Toolkit 4 orientiert sich an Anforderungen, die aus dem Betrieb von Grids stammen und unter anderem folgende Ziele verfolgen:<sup>240</sup>

- Die erneute Konfiguration und Bereitstellung sollte genauso wie das Entfernen von Diensten ohne erneutes Starten der Hosting-Umgebung möglich sein.
- Die Dienste müssen dynamisch auf Veränderungen der Nutzerzahlen reagieren.
- Die Verwaltungs- und Entwicklungskosten sollen durch die Einbindung der neuen Fähigkeiten sinken.

Es werden zwei Arten des dynamischen Deployment<sup>241</sup> implementiert, die über das schon im Apache Tomcat-Applikationsserver eingebaute Einspielen einer gesamten Webapplikation ohne andere Applikation beenden zu müssen, hinaus gehen:<sup>242</sup>

- *Service-level deployment* bezeichnet das Beenden eines Dienstes innerhalb des Containers und das anschließende Austauschen und Reaktivieren des Dienstes.
- Beim *Container-level deployment* wird der gesamte Container nach dem Installieren des neuen Dienstes reinitialisiert und neu geladen, ohne den gesamten Applikationsserver neu zu starten.

Beide Ansätze führen abhängig von der Dateigröße der Archivdatei, die für das Deployment verwendet wird, und von der Anzahl der schon laufenden Dienste

---

<sup>240</sup> Vgl. Qi u. a. (2006), S.1

<sup>241</sup> Deployment bezeichnet das Bereitstellen einer Applikation in einer Ablaufumgebung innerhalb eines Applikationsservers.

<sup>242</sup> Vgl. Qi u. a. (2006), S. 2

---

im Container, zu guten Ergebnissen.<sup>243</sup>

### V.3.2 Technologie

Die Architektur des Globus Toolkit Version 4 (GT4) lässt sich grob in Infrastruktur-Services, Hosting-Umgebungen für individuelle Dienste (in den Programmiersprachen Java, C oder Python) und Bibliotheken für Clientprogramme zum Zugriff auf die Infrastrukturdienste und eigenentwickelte Dienste unterteilen.<sup>244</sup> Die folgende Grafik vermittelt einen differenzierteren Blick des Aufbaus und gliedert die Infrastruktur-Dienste nach Sicherheit, Daten Management, Informationsdiensten und Diensten, die für das Management der Durchführung von Berechnungen zuständig sind.

---

<sup>243</sup> Vgl. QI u. a. (2006), S. 10

<sup>244</sup> Vgl. FOSTER (2005), S. 4 f.

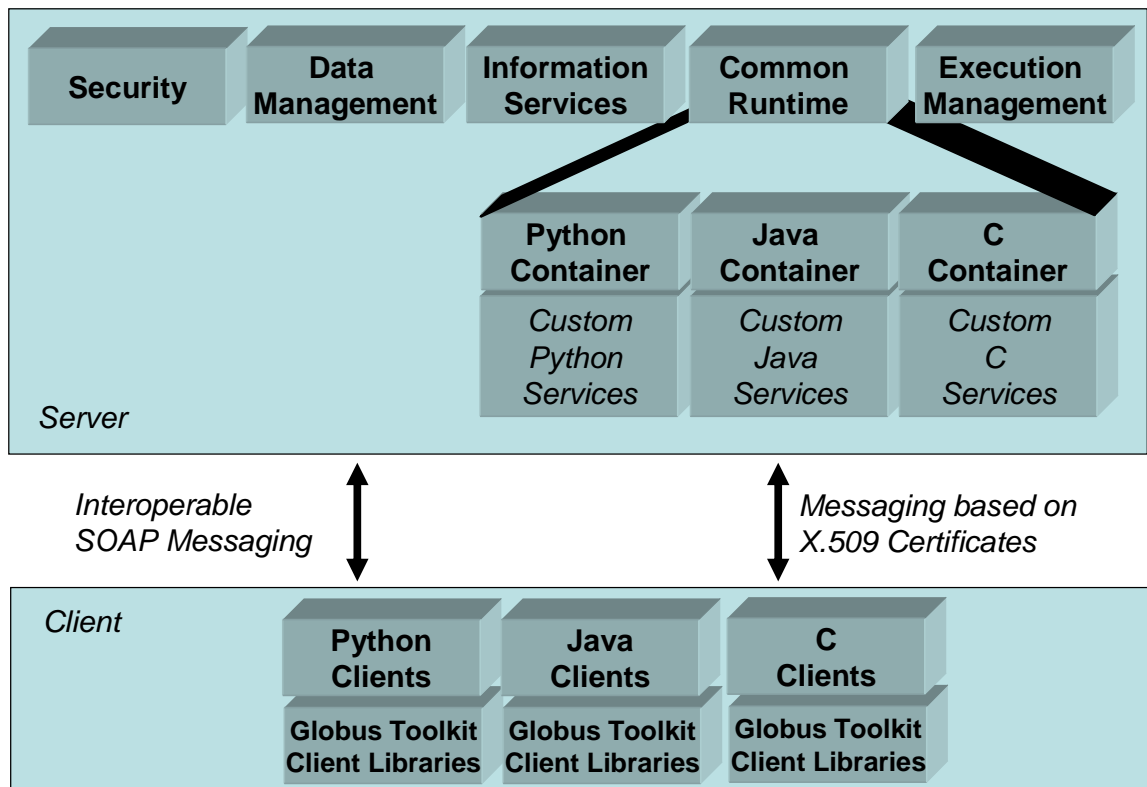


Abbildung 31: Schematischer Aufbau des Globus Toolkit (Version 4)

[Quellen: FOSTER (2005), S. 4 und 6]

Zentraler Dienst im Execution Management ist der *Grid Resource Allocation and Management (GRAM)*-Dienst, der neben der Zuweisung einer beliebigen Rechenoperation an eine an das Grid angeschlossene Ressource, auch das Monitoring und das Management der Ausführung übernimmt.<sup>245</sup> Für das Data Management stehen Dienste für unterschiedliche Anforderungen zur Verfügung:<sup>246</sup>

- *GridFTP* ermöglicht nicht nur die Interaktion mit konventionellen FTP-Servern und Clients, sondern garantiert auch die gesicherte, verlässliche und performante Übertragung von Daten.
- Der *Reliable File Transfer*-Dienst koordiniert den Datentransfer bei einer Übertragung, die auf mehrere GridFTP-Vorgänge verteilt wurde.
- Der *Replica Location Service* ist für die Verwaltung und den Zugriff auf

<sup>245</sup> Vgl. FOSTER (2005), S. 5

<sup>246</sup> Vgl. FOSTER (2005), S. 7

Daten verantwortlich, die durch Replikation redundant an verschiedenen Orten im Grid zur Verfügung stehen.

- Die Replikation von Daten wird vom *Data Replication Service* bewerkstelligt, der sich auf den Replica Location Service und den GridFTP-Dienst stützt.
- Weiterhin kann per Dienst auf relationale Datenbanken und XML-Dateien zugegriffen werden.

Information Services dienen zur Überwachung und zum Auffinden von Diensten, die von GT4 zur Verfügung gestellt werden bzw. innerhalb der Container ablaufen.<sup>247</sup> Es ist möglich, Dienste zur selbstständigen Registrierung bei dem Container, der als Ablaufumgebung dient, zu konfigurieren und einen Container im Zuge einer Hierarchiebildung bei anderen, übergeordneten Containern anzumelden.<sup>248</sup> Aggregationsdienste werden dabei für die Sammlung von Informationen über den Zustand der registrierten Ressourcen verwendet und stellen diese Informationen entweder in Form eines Verzeichnisses bzw. eines ereignisgesteuerten Datenfilters zur Verfügung.<sup>249</sup> Für die Analyse der Daten werden neben kommandozeilenorientierten Tools auch Web Frontends, wie das WebMDS angeboten (siehe Abbildung 36).<sup>250</sup>

Die Sicherheit der Nachrichten, die Authentifikation bzw. die Überwachung der Autorisierung wird entweder über X.509-Zertifikate im Rahmen einer Implementierung von WS-Security auf Nachrichtenebene, durch Benutzernamen und Passwörter oder über die Sicherung der Datenübertragung mit X.509-Zertifikaten gewährleistet.<sup>251</sup>

---

<sup>247</sup> Vgl. FOSTER (2005), S. 7

<sup>248</sup> Vgl. FOSTER (2005), S. 7

<sup>249</sup> Vgl. FOSTER (2005), S. 7

<sup>250</sup> Vgl. FOSTER (2005), S. 7

<sup>251</sup> Vgl. FOSTER (2005), S. 8

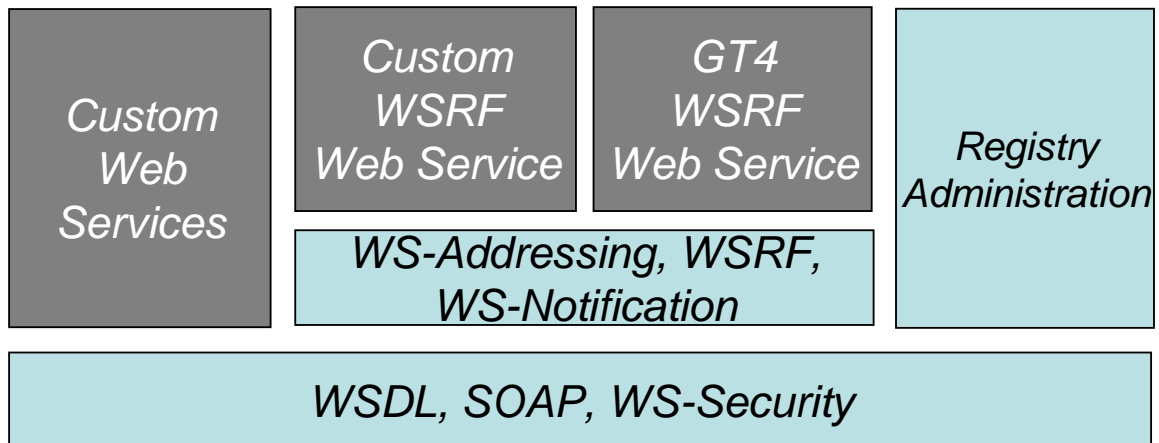


Abbildung 32: Nutzungsmöglichkeiten eines Containers im Globus Toolkit

[Quelle: FOSTER (2005), S. 10]

Die Abbildung 32 zeigt den schematischen Aufbau der Ablaufumgebung für C-, Java- und Python-Web Services innerhalb eines Containers. Benutzerdefinierte, zustandslose Web Services können hierbei genauso gehostet werden wie Web Services, die auf das WSRF aufsetzen.<sup>252</sup> Dies macht die Eignung des Globus Toolkits insbesondere als Hosting-Umgebung für den Next Generation Service Provider aus. Zur Infrastruktur eines Containers gehören Verzeichnis- und Verwaltungsfunktionen für benutzerdefinierte Dienste und für GT4 Services.<sup>253</sup>

### V.3.3 Anwendung des Web Services Resource Frameworks<sup>254</sup>

Bei Verwendung des Web Services Resource Frameworks für Web Services, wird es möglich, *stateful web services* zu erstellen. Dabei werden so genannte Resource Factory Web Services eingesetzt, die jeweils eine Instanz einer *Resource* erzeugen können. Diese Resource befindet sich in einem bestimmten Zustand. In der Abbildung 33 wird zusätzlich noch ein Resource Home Web Service eingeführt. Somit ist es nicht nur möglich mit einer Factory einen einzigen Typ von Resource zu erzeugen, sondern es können mehrere

<sup>252</sup> Vgl. FOSTER (2005), S. 9

<sup>253</sup> Vgl. FOSTER (2005), S. 9 f.

<sup>254</sup> Vgl. für diesen Abschnitt SOTOMAYOR (2005)

Resources unterschiedlichen Typs erzeugt werden (Aktivität 1 und 2 in Abbildung 33). Dazu ist jedoch jeweils ein Resource Home Web Service nötig.

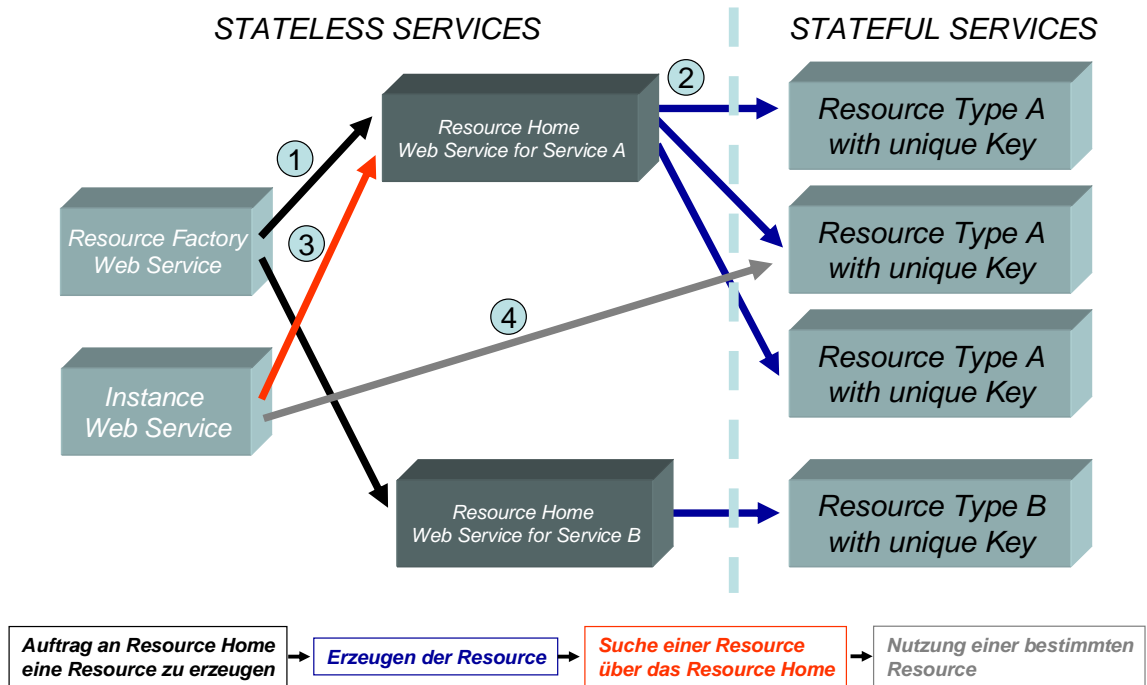


Abbildung 33: Nutzung des WSRF für stateful resources

[Quelle: SOTOMAYOR (2005)]

Zum Zugriff auf eine Resource und eine damit einhergehende Veränderung der Resource Properties wird der Instance Web Service benötigt. Zuerst muss die benötigte Instance unter Verwendung der Resource Home gefunden werden (Aktivität 3 in Abbildung 33), bevor eine Veränderung der Properties durch direkten Zugriff des Instance Web Service auf die Resource durchgeführt werden kann (Aktivität 4 in Abbildung 33).

Die Abbildung 33 zeigt außerdem den Übergang von *stateless* zu *stateful web services*. Nur die *Resource* ist mit einem Zustandsdokument versehen. Die restlichen Web Services inklusive des Web Service-Clients bleiben zustandslos und ermöglichen nur die Generierung, Nutzung und das Management der *stateful resources*.

### **V.3.4 Analyse der Ergebnisse**

Die Implementierung eines Web Services mit dem Web Services Resource Framework ist aufwendiger als bei der Entwicklung eines zustandslosen Web Services. Der Zugriff auf die eigenen Services des Globus Toolkit erfolgt jedoch mit demselben Instrumentarium, sodass hier keine zusätzliche Umstellung erforderlich ist. Bei der Programmierung stehen dem Entwickler einige Java-Klassenbibliotheken zur Verfügung. Analog zur konventionellen Web Service-Entwicklung können Client-Stubs generiert werden auf die eine Client-Applikation aufgebaut wird.

Positiv ist die Tatsache zu beurteilen, dass auf derselben Plattform auf der das Grid betrieben wird, gleichzeitig auch konventionelle Web Services ohne Zustandsinformation betrieben werden können. Der Unterschied zwischen diesen Web Services und den Web Services mit Zustandsinformationen in Form von Resource Properties lässt sich jedoch nicht in dem Grad an Komplexität festmachen. Die Frage, welche Form für einen bestimmten Einsatzzweck geeignet ist, wird durch die Anforderungen an Rechenleistung bzw. sonstigen Ressourcen, wie Speicherplatz und Zugang zu Spezialgeräten, bestimmt.

## **V.4 Integrierte Umgebung**

Innerhalb der integrierten Umgebung sollen das Hosting von Web Services und der Betrieb eines Grid demonstriert werden. Es sollen folgende Anforderungen erfüllt werden:

- Virtualisierung von Mehrprozessorsystemen, um die vorhandenen Ressourcen besser auf die einzelnen Anforderungen der Hosting-Umgebungen anpassen zu können.
- Partitionierung von Systemen entsprechend der Anforderungen an die Hosting-Umgebung
- Abgrenzung/ Abschottung der Ablaufumgebung

- Zentralisierung von gemeinsam nutzbaren Serviceleistungen für alle Ablaufumgebungen

Die Anforderungen leiten sich aus einem potenziellen Einsatzszenario für einen Next Generation Service Provider ab. Neben der Konsolidierung von Rechnern auf leistungsfähige Hardware mit mehreren Prozessoren oder Blade-Server Umgebungen wird ein möglichst einfaches Verlagern von Betriebssystemplattformen inklusive der darauf betriebenen Anwendungssoftware immer wichtiger. Durch die Möglichkeit der Abschottung der verschiedenen virtuellen Rechner ist ein isolierter Betrieb von Anwendungen für verschiedene Kunden möglich. Dabei ist auf der einen Seite der Bedarf an Rechenleistung im Grid nicht immer gleich und kann durch Anhalten einer virtuellen Ressource und Starten einer anderen Ressource besser auf die Anforderungen verteilt werden. Auf der anderen Seite muss jedoch auch die Zuteilung von Rechenleistung genau dosierbar sein. Es muss also möglich sein, zumindest die Anzahl der Prozessoren und den Speicher relativ flexibel auf die virtuellen Rechner verteilen zu können. Darüber hinaus soll auch die Nutzung von Ressourcen anderer Provider ermöglicht werden. Um hierbei den Konfigurationsaufwand so klein wie möglich zu halten, kann eine komplette Umgebung weitergegeben und bei einem anderen Provider weiterbetrieben werden.

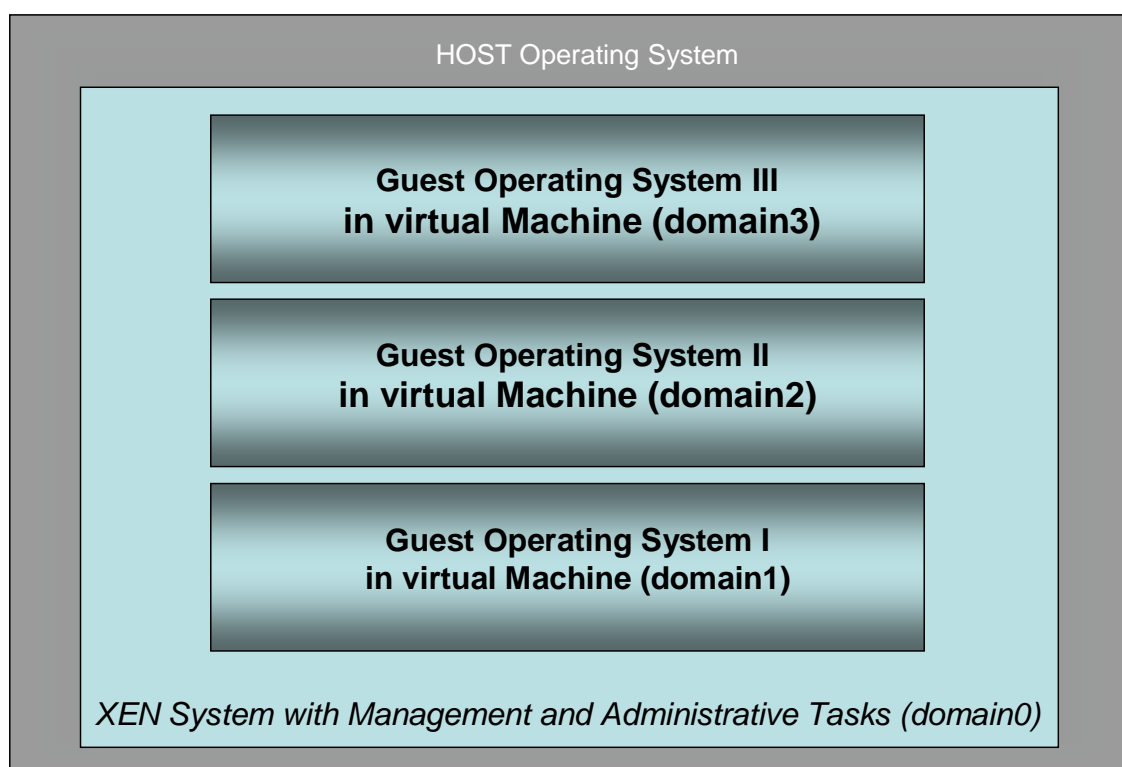
#### **V.4.1 Auswahl geeigneter Systemumgebungen**

Für den Server mit der integrierten Umgebung sollen ausschließlich Open Source-Komponenten zum Einsatz kommen. Linux steht somit als Betriebssystem fest. Durch die Verwendung von Apache Tomcat als Applikationsserver stehen auf der einen Seite Möglichkeiten zum Hosting von Web Services offen, die in der Apache AXIS-Umgebung ausgeführt werden. Auf der anderen Seite kann auf dem Applikationsserver das Grid der Globus Alliance betrieben werden. Beide Umgebungen wurden schon in vorigen Kapiteln vorgestellt.

Zur Unterstützung der Anforderungen des Service Providers wurde eine geeignete Virtualisierungssoftware gesucht. Die Kriterien hierfür waren neben der Verfügbarkeit als Open Source Lösung möglichst hohe Performanz und die



Unterstützung der Migration von virtuellen Systemumgebungen. Eine reine Abschottung einer Applikation in Bezug auf Dateisystem und Prozesse eines Linux Systems war für die Anforderungen nicht ausreichend, da ein zu enger Bezug zum Betriebssystem und ein beträchtlich höherer Konfigurationsaufwand dem entgegen standen. Bei dieser Lösung müssten alle Softwarekomponenten, die für die Applikation wichtig sind in einem Verzeichnis zusammengefasst werden, da die sonst gewohnte Umgebung eines Linux-Systems eben nicht als Ganzes zur Verfügung stehen soll.



*Abbildung 34: Schematischer Aufbau eines XEN-basierten Systems zur Virtualisierung von Betriebssystemressourcen  
[eigene Darstellung nach XEN (2006), S. 3]*

Die Wahl fiel auf die von der Universität Cambridge entwickelte Virtualisierungssoftware XEN. Die Software erfordert eine Modifikation am Betriebssystemkern des Gast- und des Host-Betriebssystems, kann aber beliebige Anwendungssoftware ohne Modifikationen ausführen und dem Gast-Betriebssystem über die Physical Addressing Extensions von Intel bis zu 64 GB an Haupt-

speicher zur Verfügung stellen und profitiert sowohl von Mehrprozessormaschinen als auch von Prozessoren mit Hyperthreading Technologie.<sup>255</sup> Durch die von AMD und INTEL entwickelte Hardware-Virtualisierung wird der Betrieb der Gastbetriebssysteme sogar ohne Modifikationen möglich.<sup>256</sup>

Wie Abbildung 34 zeigt, ist das XEN-System in mehrere Ebenen oder Domains aufgeteilt, wobei die unterste Schicht mit dem Management und der Verwaltung der Gastbetriebssysteme betraut ist und diese unter anderen startet und beendet.<sup>257</sup> Dabei wird auch die Netzwerkanbindung über die Domain0 abgewickelt, indem Werkzeuge des Host-Betriebssystems für Routing und Bridging zum Einsatz kommen.<sup>258</sup>

## V.4.2 Technische Architektur

Die Installation der Softwarekomponenten auf dem Mehrprozessorsystems ist durch 3 Hauptkomponenten gekennzeichnet:

1. Zentrale Dienstleistungen für alle virtuellen Server (Central Services)
2. Administrative Serverinstanz (XEN Server 1)
3. Grid und Web Service Hosting Umgebungen (XEN Server 2 - 4)

---

<sup>255</sup> Vgl. XEN (2006), S. 2

<sup>256</sup> Vgl. XEN (2006), S. 2

<sup>257</sup> Vgl. XEN (2006), S. 3

<sup>258</sup> Vgl. XEN (2006), S. 26

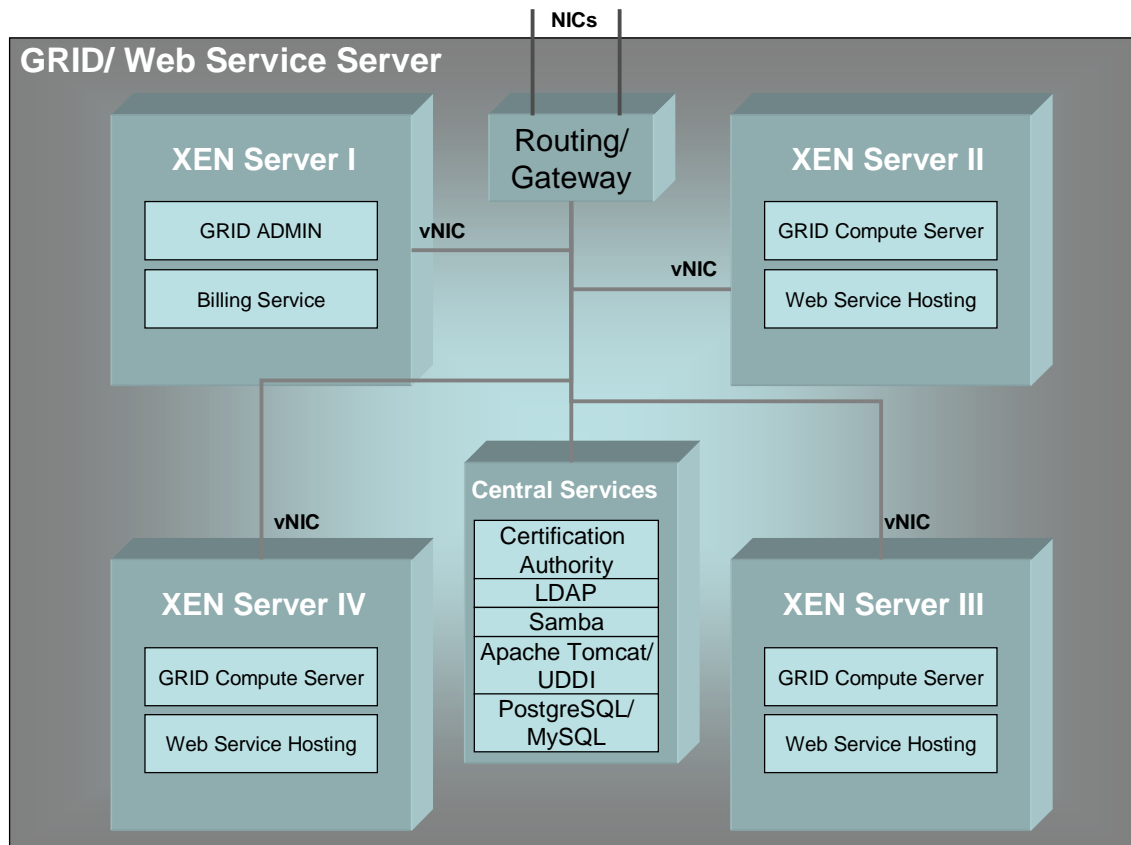


Abbildung 35: Integrierte Web Service-/ Grid-Umgebung

Bei den zentralen Dienstleistungen lassen sich 2 Gruppen von Anwendungen unterscheiden. Die erste Gruppe sind allgemeine Anwendungen, die unterstützende Dienste leisten. Hier ist zuerst die Certification Authority zu nennen, die die Zertifikate für die Grid-Umgebungen und deren Nutzer signiert. Hierbei handelt es sich um eine lokale Zertifizierungsstelle, die auf der von der Globus Alliance bereitgestellten SimpleCA basiert und sich auf die openssl-Anwendung der Betriebssystemumgebung stützt. Daneben steht ein SAMBA-Server für die Bereitstellung von Fileserver-Diensten im Netz zur Verfügung. Dieser Dienst ist für das Migrieren von XEN-Servern auf unterschiedliche Server nötig, da eine Live-Migration, also ohne den virtuellen Server zu beenden, nur dann möglich ist, falls der Zugriff auf ein gemeinsames Dateisystem etwa in Form eines Storage Area Networks oder wie es hier der Fall ist, in Form einer Emulation eines Microsoft Windows NT-Fileservers besteht.<sup>259</sup> SAMBA wurde gewählt, da

<sup>259</sup> Vgl. XEN (2006), S. 35 f.

so sowohl unter Linux/ Unix-Umgebungen ein Einbinden des Dateisystems möglich ist, als auch bei Betreiben einer Grid-Instanz unter Windows Betriebssystemen auf die gemeinsamen Dateien zugegriffen werden kann. Zur Unterstützung des Identity Managements ist noch ein LDAP-Verzeichnisserver installiert, der auf dem openLDAP-Projekt basiert. Hiermit können sowohl die Kennungen für die Betriebssystembenutzer, als auch die Benutzer für die SAMBA-Umgebung verwaltet werden. Für Anwendungen, die die Unterstützung von Datenbankmanagementsystemen benötigen, stehen MySQL und PostgreSQL zur Verfügung. MySQL wird zusätzlich von dem UDDI-Server auf jUDDI-Basis benötigt und in den Apache Tomcat-Applikationsserver als Datenquelle eingebunden.

Die zweite Gruppe von unterstützenden Anwendungen wird direkt von der Grid-Umgebung benötigt. Hier spielt der schon erwähnte UDDI-Server jUDDI, der auf dem unter Apache Tomcat installierten AXIS-Server betrieben wird, eine zentrale Rolle. Daneben ist ein Apache http-Server installiert, der Informationen zum Grid und den angebotenen Leistungen des Providers zur Verfügung stellt.

Die erste direkt produktiv nutzbare XEN-Instanz bietet Raum für die zentrale Installation des Globus Alliance Toolkit in der Version 4 und eine auf Web Service Basis entwickelte Billing-Applikation. Die Toolkit Installation dient als Ausgangspunkt für die Installation der restlichen untergeordneten Grid-Instanzen. Unter dem Billing-Service kann sich beispielsweise der schon vorher erwähnte Billing-Gateway verbergen. Hier kann aber auch jede beliebige andere Anwendung zur Durchführung der Abrechnung der Ressourcennutzung betrieben werden.

Die weiteren XEN-Instanzen beherbergen die eigentlichen Provisioning-Instanzen. Hier wird jeweils eine Grid-Umgebung und eine Umgebung zum Hosten von Web Services zur Verfügung gestellt. Es wird jeweils beides angeboten, weil der Applikationsserver Tomcat in Verbindung mit dem Apache AXIS Framework nicht nur als Basis für das Globus Toolkit dient, sondern gleichzeitig auch für die Bereitstellung von Web Services verwendet werden kann. Diese Ausstattung ermöglicht es bei Bedarf einfach zusätzliche XEN-Instanzen zu generieren. Dabei ist es durch die Verwaltungswerkzeuge von Apache Tomcat einfach über ein Web Frontend möglich, entweder die Hosting-Umgebung für

Web Services- oder aber die Grid-Umgebung abzuschalten und so die Ressourcen der jeweils anderen Anwendung exklusiv zur Verfügung zu stellen.

Die XEN-Umgebung ermöglicht die Zuteilung von Speicher und Prozessorressourcen für die einzelnen XEN-Instanzen. So lassen sich die Ressourcen entsprechend der Anforderungen der Kunden verteilen. Gleichzeitig erfolgt eine weitgehende Abschirmung der Umgebungen. Die einzelnen virtuellen Server können beispielsweise eine Firewall betreiben. Durch die Möglichkeit der Migration wird zusätzlich eine hohe Verfügbarkeit der Applikation etwa bei Wartungsarbeiten an der physischen Hardware erreicht.

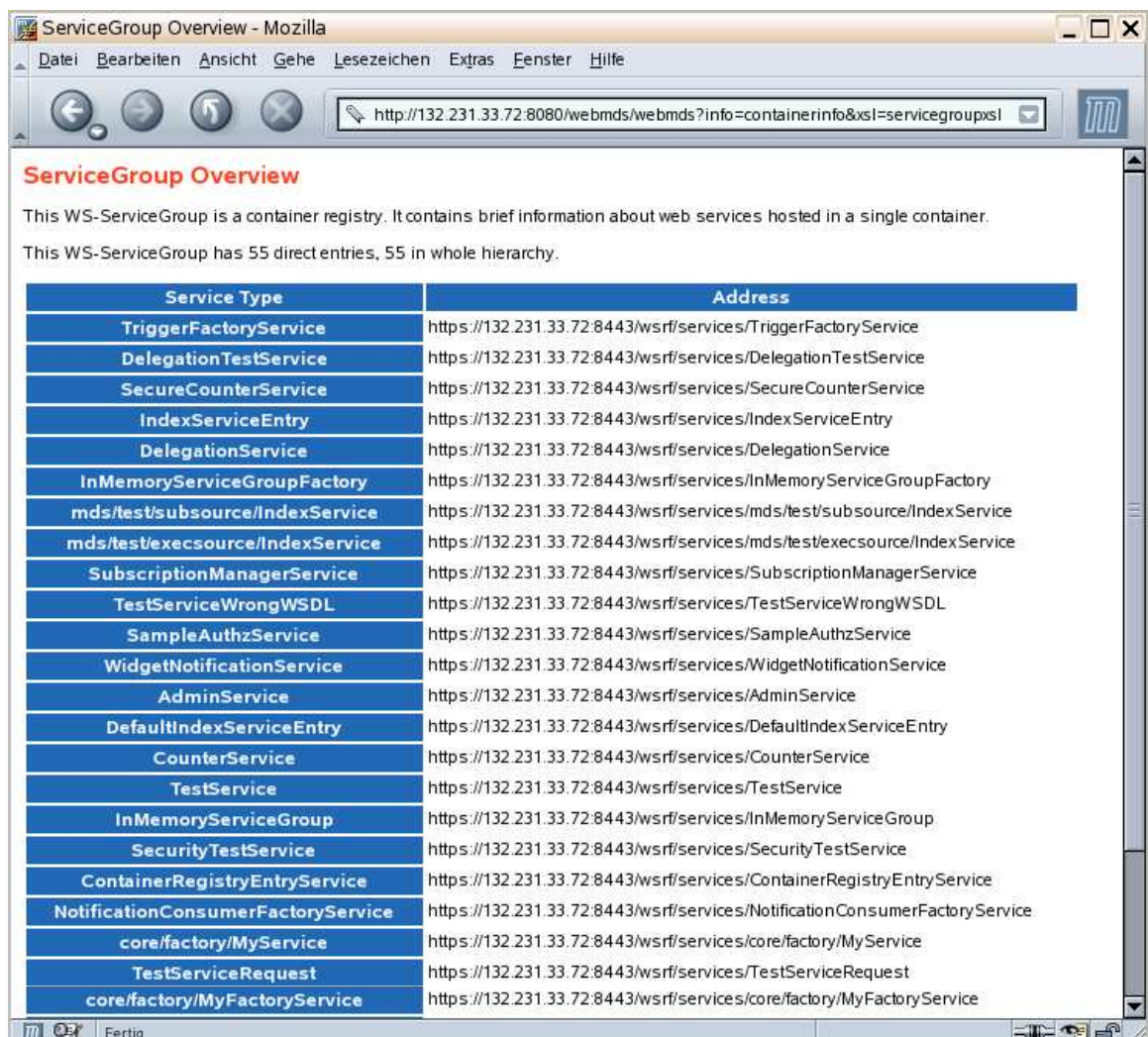
### **V.4.3 Metadaten in der integrierten Provisioning-Umgebung**

Metadaten treten in unterschiedlicher Form auf. Die augenfälligste Form sind Daten, die für die Abrechnung der erbrachten Leistung anfallen. Im einfachsten Fall ist dies die Information, dass ein bestimmter User auf einen bestimmten Dienst zugegriffen hat. Mit diesen Daten kann die Abrechnung sofort erfolgen, falls der User bekannt ist und der Dienst mit einer konstanten Nutzungsgebühr belegt ist.

Im Umfeld der direkten abrechnungsrelevanten Daten sind Daten angesiedelt, die entweder Auskunft über den Ressourcenverbrauch der individuellen Dienstenutzung bieten, oder Informationen über den Gesamtverbrauch bzw. die Auslastung von Ressourcen liefern. Der individuelle Ressourcenverbrauch eines Dienstes kann zur Berechnung der Kosten verwendet werden, die als Grundlage für die Ermittlung der Höhe der Preise für den Kunden dienen kann. Wichtig ist hierbei, die Balance zwischen Komplexität der Datenerhebung und der Höhe des zu erwartenden Entgeltes zu finden. Eine einfache Variante der Abrechnung ist die Abrechnung pro Prozessor und pro Stunde, wie sie im Augenblick auch schon in der Praxis betrieben wird. Bei Anwendung der Virtualisierung über XEN ist dies relativ unproblematisch, wenn ein virtueller Server mit den zugeteilten virtuellen Prozessoren für genau einen Kunden exklusiv zur Verfügung steht.

Neben der Auslastung von individuellen Services lässt sich in einem Grid auch

die Auslastung eines für alle Dienste zur Verfügung stehenden Dienstes, wie beispielsweise des Reliable File Transfers, messen. Hierbei wird die Menge der in einer Queue wartenden Aufträge angegeben. Eine zu starke Belastung eines wichtigen Servicedienstes kann als Maß für die Servicequalität verwendet werden. Im Kontext der vorgeschlagenen integrierten Hosting-Umgebung würde dies ein Indiz für die Adjustierung der an einen virtuellen Server zugewiesenen Prozessor- und Speicherressourcen darstellen. Sollte ein virtueller Server nicht ausreichen, kann der Service auf mehrere aufgeteilt oder auf eine Hardwareplattform mit leistungsfähigeren Ressourcen migriert werden.



Service Type	Address
TriggerFactoryService	https://132.231.33.72:8443/wsrf/services/TriggerFactoryService
DelegationTestService	https://132.231.33.72:8443/wsrf/services/DelegationTestService
SecureCounterService	https://132.231.33.72:8443/wsrf/services/SecureCounterService
IndexServiceEntry	https://132.231.33.72:8443/wsrf/services/IndexServiceEntry
DelegationService	https://132.231.33.72:8443/wsrf/services/DelegationService
InMemoryServiceGroupFactory	https://132.231.33.72:8443/wsrf/services/InMemoryServiceGroupFactory
mds/test/subsource/IndexService	https://132.231.33.72:8443/wsrf/services/mds/test/subsource/IndexService
mds/test/execsource/IndexService	https://132.231.33.72:8443/wsrf/services/mds/test/execsource/IndexService
SubscriptionManagerService	https://132.231.33.72:8443/wsrf/services/SubscriptionManagerService
TestServiceWrongWSDL	https://132.231.33.72:8443/wsrf/services/TestServiceWrongWSDL
SampleAuthzService	https://132.231.33.72:8443/wsrf/services/SampleAuthzService
WidgetNotificationService	https://132.231.33.72:8443/wsrf/services/WidgetNotificationService
AdminService	https://132.231.33.72:8443/wsrf/services/AdminService
DefaultIndexServiceEntry	https://132.231.33.72:8443/wsrf/services/DefaultIndexServiceEntry
CounterService	https://132.231.33.72:8443/wsrf/services/CounterService
TestService	https://132.231.33.72:8443/wsrf/services/TestService
InMemoryServiceGroup	https://132.231.33.72:8443/wsrf/services/InMemoryServiceGroup
SecurityTestService	https://132.231.33.72:8443/wsrf/services/SecurityTestService
ContainerRegistryEntryService	https://132.231.33.72:8443/wsrf/services/ContainerRegistryEntryService
NotificationConsumerFactoryService	https://132.231.33.72:8443/wsrf/services/NotificationConsumerFactoryService
core/factory/MyService	https://132.231.33.72:8443/wsrf/services/core/factory/MyService
TestServiceRequest	https://132.231.33.72:8443/wsrf/services/TestServiceRequest
core/factory/MyFactoryService	https://132.231.33.72:8443/wsrf/services/core/factory/MyFactoryService

Abbildung 36: Metainformationen zu den Diensten innerhalb einer Hosting-Umgebung des Globus Toolkit

Neben leistungsbezogenen Informationen stehen Verwaltungsdaten zur Verfügung. Auf der Ebene der Grid-Instanz bzw. eines Containers kann die Art der darauf laufenden Services ermittelt werden (vgl. Abbildung 36). Hiermit stehen Informationen zur Verfügung, die für die geeignete Aufteilung der Dienste auf mehrere Grid-Knoten oder die Konzentration aller Dienste für einen Kunden auf einer Grid-Instanz nötig sein können.

Auf der Ebene des Apache Tomcat-Applikationsservers stehen Informationen über den Ressourcenverbrauch von Webapplikationen und über alle installierten Webapplikationen bereit (vgl. Abbildung 26). Die hier vorgestellten jUDDI, Apache AXIS und Globus Toolkit stellen solche Webapplikationen dar. Hiermit besteht die Möglichkeit, einzelne Dienste bzw. Laufzeitumgebungen anzuhalten, zu löschen oder zu starten.

#### **V.4.4 Migrationsszenarien**

Für eine Migration eines virtuellen XEN-Servers kann es mehrere Gründe geben. Zum einen wird ein Umzug des Servers nötig, falls eine Hardwareplattform gewartet werden muss oder aufgrund mangelnder Performanz in Zukunft nicht mehr zur Verfügung steht. Zum anderen ist eine Weitergabe des Servers an einen Dienstleister eine mögliche Ursache für die Notwendigkeit zur Relokation eines virtuellen Servers.

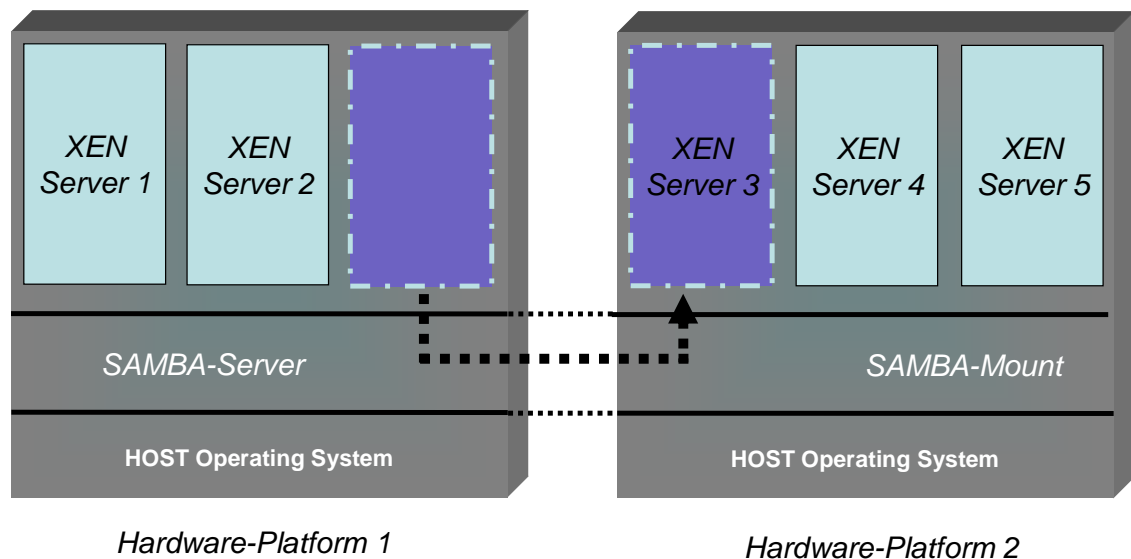


Abbildung 37: Migration eines XEN-Servers über eine gemeinsame SAMBA-Freigabe

Für die Durchführung der Migration stehen bei XEN zwei Varianten zur Verfügung – die Offline-Migration und die Live-Migration. Die Offline-Migration erfordert das Herunterfahren bzw. Beenden des virtuellen Servers und kann anschließend mittels eines konventionellen Dateitransfers erfolgen. Befinden sich beide Systeme innerhalb einer administrativen Domäne so kann der Transfer einfach über ein Storage Area Network oder eine Netzwerkfreigabe erfolgen. In unserem Fall übernimmt dies der SAMBA Server.<sup>260</sup>

Die Live-Migration erfordert kein Beenden des virtuellen Servers im konventionellen Sinne. Unter der Voraussetzung, dass sich das Quell- und das Zielsystem innerhalb desselben Subnetzes befinden und von beiden Systemen der Zugriff auf eine Netzwerkfreigabe besteht, kann der Umzug des Systems mit nur einer Unterbrechung des Betriebes im Bereich von Millisekunden erfolgen.<sup>261</sup>

<sup>260</sup> Vgl. für diesen Absatz XEN (2006), S. 35 f.

<sup>261</sup> Vgl. für diesen Absatz XEN (2006), S. 35 f.



## V.4.5 Ergebnisauswertung

Die integrierte Umgebung baut auf die Web Service- und Grid-Umgebung auf und kann damit in vollen Umfang auf die dort gesammelten Erkenntnisse zurückgreifen.

Durch die Kombination der beiden Hosting-Umgebungen kann ein flexibles Leistungsportfolio von einem Next Generation Service Provider realisiert werden. Die Flexibilität innerhalb der vorgestellten virtuellen Hosting-Umgebungen und die anforderungsgerechte Nutzung der zur Verfügung stehenden Hardwareplattformen durch den Start von mit ausreichend Ressourcen konfigurierten XEN-Servern, ermöglicht nicht nur die individuelle Unterstützung eines Kunden, sondern auch die Unterstützung mehrerer Kunden auf einer Hardwareplattform. Dabei wird nicht nur eine optimale Abschirmung der einzelnen Instanz der Hosting-Umgebung eines Kunden gewährleistet, sondern es werden auch kostengünstige Angebote individuell konfigurierter Umgebungen für Kunden mit besonderen Anforderungen möglich.

Metainformationen über die integrierte Hosting-Umgebung gewährleisten optimale Informationen für die Verwaltung und Optimierung der Umgebung.

Diese Informationen bilden den Ausgangspunkt für das Erkennen eines Migrationsbedarfs, der im Rahmen der integrierten Umgebung durch den Einsatz der Virtualisierungssoftware mit einem verträglichen Aufwand abgewickelt werden kann. Die vorgestellten Migrationsvarianten sind für den Next Generation Service Provider wichtig zur einfachen Nutzung zusätzlicher Provider, die als Subunternehmer die Leistung für den Kunden erbringen. Hierbei soll der Konfigurationsaufwand beim zusätzlichen Provider so gering wie möglich gestaltet werden.

Die Vorteilhaftigkeit der Nutzung der Virtualisierungslösung in kommerziell betriebene ASP-Umgebungen kann anhand des Beispiels des SAP Hochschulkompetenzzentrums Passau gezeigt werden. Das Hochschulkompetenzzentrum war in Größe und Professionalität des Systembetriebs trotz des non-

profit Charakters mit einem kommerziellen ASP vergleichbar.<sup>262</sup> Durch Beobachtungen und Analysen des Lastverhaltens der Systeme wurden Lastprofile erkannt, die durch einen hohen Anteil an geringer Systemlast gekennzeichnet waren. MOHR/ SIMON/ KRCCMAR (2005) haben im SAP Hochschulkompetenzzentrum München die Lastprofile näher untersucht und charakterisieren das Lastverhalten als Whiplash-Effekt, der durch atypische Workload (z. B. durch Mandantenkopien), kurzfristige ungleichmäßige Lastverteilung über den Tag hinweg (z.B. durch Übungsbetrieb) und langfristige ungleichmäßige Lastprofile (z. B. durch Semesterferien) hervorgerufen wird.<sup>263</sup> Das Potenzial durch Virtualisierung mit der selben Anzahl von Servern deutlich mehr Kunden bedienen zu können und ausgleichend auf das Lastprofil der Server einzuwirken, wurde schon im Hochschulkompetenzzentrum Passau erkannt. Einen Lösungsansatz zur Abfederung der Spitzenlast, zur Homogenisierung der Last auf den Servern und zum Schutz vor Systemausfällen mit Blade-Server-Technologie unterstützt durch die selbsterstellte Software-Lösung Blade-Runner präsentieren MOHR/ SIMON/ KRCCMAR (2005).<sup>264</sup> Ein anderer Lösungsansatz ist die in dieser Arbeit verwendete Virtualisierungssoftware XEN.

## ***V.5 Die prototypischen Umgebungen im Kontext des IT-Sourcing***

In den prototypischen Umgebungen wurden verschiedene Aspekte der Web Service-Nutzung gezeigt.

Die erste Umgebung soll innerhalb der Arbeit den Ausgangspunkt für Sourcing-Überlegungen im Unternehmen darstellen. Diese Umgebung zeigt Ansätze der Serviceorientierung ohne den Einsatz von Web Services. Sie kann als Beleg

---

<sup>262</sup> Vgl. KLEINSCHMIDT/ EDER (2000)

<sup>263</sup> Vgl. MOHR/ SIMON/ KRCCMAR (2005), S. 4 f. (Die Seitenzahl wurde der Online-Publikation entnommen.)

<sup>264</sup> Vgl. MOHR/ SIMON/ KRCCMAR (2005), S. 9 und S. 11 (Die Seitenzahl wurde der Online-Publikation entnommen.)

dafür dienen, dass eine SOA nicht notwendigerweise mit dem vollständigen Umbau der vorhandenen Systeme verbunden sein muss. Soll der evolutorische Weg beschritten werden, so können wie hier bei einem betrieblichen Standardsoftwareprodukt, service-orientierte Bestandteile identifiziert werden und als Basis für das weitere Vorgehen verwendet werden. Die vorliegende Umgebung zeigt, dass durch die Umsetzung einer einfachen Web Service-Schnittstelle die Möglichkeit für eine Altanwendung besteht, unternehmensexterne Dienste zu nutzen.

Die Web Service-Umgebung ermöglicht das Angebot und die Nutzung von Web Services mit unterschiedlichen Komplexitätsgraden. Zur Integration von Ressourcen in Portale können einfache PHP-Skripte genutzt werden. Diese relativ kleinen Anwendungen eines Application Service Providers können als Einstieg in die Nutzung externer Ressourcen bzw. in die Öffnung der Unternehmensgrenzen gesehen werden. Wie das PHP-Beispiel verdeutlicht, lässt sich eine nahtlose Integration in eine Portalumgebung erreichen und andere Web Applikationen mit neuen Web Service Schnittstellen versehen, um diese mit unterschiedlichen Anwendungen zu verknüpfen.

Steigen die Anforderungen an die Lösung, kann auf leistungsfähigere Umgebungen und Programmiersprachen umgestiegen werden. Wie der einfache JAVA-Service in Abbildung 28 zeigt, kann die Geschäftslogik dabei unter Umständen aus bestehenden Anwendungen einfach übernommen werden, falls die dazu nötige Geschäftslogik schon identifiziert wurde und gekapselt zur Verfügung steht. In dem Beispiel wurde die in der SQL-Anweisung steckende Logik weiterverwendet.

Die Funktionalität steht bei internen genauso wie bei externen Diensten im Vordergrund. Bei der Nutzung eines Dienstes eines Service Providers müssen also die Leistungsfähigkeit und die Anforderungen an die zu übermittelnden Daten passen, die Implementierung kann als Blackbox betrachtet werden. Insofern sind also Angebote zweier Provider einfacher zu vergleichen. Komplexere Service-Implementierungen sind jedoch auch für das Outsourcing von Anwendungen wichtig. Die konkrete Integration mit den im Unternehmen befindlichen Systemen lässt sich analog zu kleineren Diensten realisieren.

Die Grid-Umgebung zeigt den Zusammenhang zwischen zustandslosen und zustandsbehafteten Web Services. Durch die Annäherung der Standards wird es möglich, zusätzlich zu den sonstigen Eigenschaften von Web Services auch Rechenleistung in größerem Umfang zu nutzen. Reichen die Ressourcen innerhalb eines Unternehmens für die beabsichtigten Aufgabenstellungen nicht aus, so kann über die schon für interne Grid-Anwendungen aufgebaute Infrastruktur zusätzlich externe Rechenleistung bezogen werden.

Die integrierte Umgebung verlässt die Perspektive des Unternehmens, das externe Dienste nutzt, und stellt den Next Generation Service Provider in das Blickfeld der Betrachtung. Anhand der vorgeschlagenen Hosting-Umgebung werden derzeit realisierbare Konzepte des effizienten Managements von Ressourcen vorgestellt, die eine effektive Unterstützung des Geschäftsablaufes des Providers bieten. Wie es für Anwendungen mit höherem Informationsbedarf für die Verarbeitung wichtig ist, werden Aspekte der Abschottung der Ablaufumgebungen gezeigt, die eine Sicherung der kundenspezifischen Daten erlauben. Die für die Nutzer der Dienste transparente Einbindung weiterer Rechenleistungsprovider rundet die Darstellung des Potenzials für Service Provider der neuen Generation ab. Im Unterschied zu den Providern der ersten und zweiten Generation wurde bei dem Konzept auf die Realisierung effizienter und im Zweifelsfall eher pragmatischer Lösungen geachtet. Aspekte, die das Service Level Management betreffen wurden an geeigneten Stellen angesprochen. Insbesondere wurde auf Maßnahmen im Rahmen der Migration der Hosting-Umgebung eingegangen, die die Anpassung der Leistungsfähigkeit eines Dienstangebotes betreffen.

Auf die Implementierung einer vollständigen Abrechnungsinfrastruktur wurde bewusst verzichtet, da ein Abrechnungsdienst in einer Hosting-Umgebung keinen anderen technischen Charakter als jeder andere Service hat.

## VI. Schlussbetrachtung

### VI.1 Resümee

Die Arbeit legt den Schwerpunkt auf die gemeinsame Betrachtung der betriebswirtschaftlichen und der technischen Dimension der SOA. Ziel war es zudem prototypische Anwendungen vorzuschlagen.

Ein Szenario aus dem branchenunabhängigen Funktionsbereich des Personalwesens zeigt zu Beginn der Untersuchung die Bedeutung von Schnittstellen und verdeutlicht die Schwierigkeiten durch die Verlagerung einer in eine gewachsene Systemlandschaft eingebetteten Applikation. In Abhängigkeit von der Art der Kommunikation des Systems mit den umliegenden Systemen (ständige Kommunikation, periodische Kommunikation, sporadische Kommunikation) gestaltet sich schon die Erfassung der Systemlandschaft aufwendig. Selbst bei einer überschaubaren Systemlandschaft mit wenigen Systemen können dennoch eine Vielzahl von technischen Schnittstellen vorhanden sein, die eine betriebswirtschaftlich motivierte Herauslösung des Systems aus dem Systemkontext erschweren.

Die veranschaulichten Problematiken wurden unter dem Blickwinkel der service-orientierten Architektur betrachtet und auf Szenarien mit externen Geschäftspartnern erweitert. Die genaue Charakterisierung der SOA in Bezug auf Rollen bzw. Akteure und der Interaktion zwischen ihnen bilden die zentrale Grundlage für die Analyse. Hierbei wurde eine Trennung zwischen dem Softwaresystem, das eine Rolle (Requestor, Broker, Provider) übernimmt, und der Organisationseinheit, in deren Hoheit das Softwaresystem steht, vorgenommen. In Verbindung mit der Tatsache, dass zwischen einem innerbetrieblichen Einsatz der SOA und der Einbeziehung des Internets aus Sicht der Architektur kein Unterschied besteht, können verschiedene Ebenen der Interaktion zwischen den Rollen einer SOA identifiziert werden. Die Ebenen I - III weisen einen steigenden Komplexitätsgrad durch die immer stärkere Einbindung der

externen Geschäftspartner auf und stellen unterschiedliche Reifegrade einer SOA dar. Daneben wird die Dynamik der Kommunikation eingeführt, die von der engen Kopplung bei bekannten Providern bis zur dynamischen Entdeckung von Providern durch den Einsatz von Verzeichnisdiensten reicht.

Die hier gemachten Ausführungen sind nicht nur für die Gestaltung von Services im Rahmen von Entwicklungsprojekten wichtig, sondern geben auch Hinweise in Bezug auf die Verwendung von Diensten innerhalb von Business Process Management-Tools bzw. sonstigen service-orientierten Plattformen für den Aufbau, die Koordination und das Management von Diensten. Das Service-Design spielt hierbei eine entscheidende Rolle, obgleich nur grobe Leitlinien herausgearbeitet werden können, die neben Erfahrungen aus konkreten Projekten eine Implementierung unterstützen können. Auf einer höheren Abstraktionsebene existieren grundlegende Prinzipien, die die Serviceorientierung als solche betreffen.

Die Technologie der Web Services wurde als technische Möglichkeit zur Umsetzung einer SOA vorgestellt. Web Services sind jedoch nur eine, wenn auch geeignete, Umsetzungsvariante einer SOA. Es besteht keine Forderung nach Web Services aus der Architektur selbst heraus. Dies wurde bei der Darstellung verwandter Technologien betont und zeigt sich auch im Rahmen der prototypischen innerbetrieblichen ERP-Umgebung. Hier werden service-orientierte Ansätze identifiziert, ohne dass das verwendete SAP-Release und der SAP Business Connector als Integrationsserver direkt Web Service-fähig sind.

Web Services sind auf die Übertragung von Nachrichten mittels standardisierter Protokolle angewiesen und zeigen sich sonst eher als Blackbox, was beispielsweise die Unabhängigkeit von einer bestimmten Implementierungssprache unterstützt. Die Einteilung dieser Protokolle innerhalb des Web Service Protocol Stack kann sich an der Gliederung des W3C orientieren und reicht von den grundlegenden Protokollen zur Übertragung von Nachrichten, zur Beschreibung von Schnittstellen und dem für die dynamische Suche wichtigen Verzeichnisdienst bis hin zu komplexeren bzw. zukunftsgerichteten Protokollen. Die Vielfalt der Standards und Protokolle hat Konsequenzen auf den Entwicklungsprozess. Web Service-Entwicklungsprojekte erfordern eine ständige Betrachtung der Entwicklung auf dem Gebiet der Web Service-Standards und -Protokolle. Es

stehen dabei sowohl Fragen bezüglich der Eignung von Protokollen für den konkreten Einsatzzweck, als auch Fragen nach der geeigneten Version eines speziellen Standards an. Zentrale Träger des Standardisierungsprozesses sind Organisationen, die zumeist durch Firmen unterstützt, eigene Klassifikationen bezüglich des Reifegrades eines Standards herausgeben.

Angrenzende Technologien verschmelzen entweder wegen der technologischen Ähnlichkeit mit Web Services bzw. liefern durch ihre Eigenschaften einen Beitrag zur Vervollständigung der Leistungsfähigkeit. Es wurden P2P-Computing und Grid-Computing untersucht. Grids werden verstärkt im betriebswirtschaftlichen Umfeld eingesetzt und ermöglichen über eine standardisierte Web Service-Schnittstelle mit Hilfe des Web Services Resource Framework den Zugriff auf Rechenleistung, große Datenbestände und kollaborative Anwendungen. P2P wird auf dem Gebiet der Selbstorganisation des Netzes und der Anpassung der Peers an die Nachfrage nach spezifischen Diensten wichtig und liefert zusammen mit den Grids Impulse auf dem Gebiet des Data Sharing.

Aufbauend auf den Grundlagen aus der Architektur und der Technologie, wurde eine Auswahl von IT-Sourcing Szenariotypen getroffen, die auf die Auswirkung des Einsatzes der service-orientierten Architektur und der technischen Umsetzung durch Web Services untersucht wurden. Entscheidende Auswahlkriterien für die Szenariotypen waren die Kunden-Dienstleister-Beziehung, die Notwendigkeit zur Standardisierung beim Sourcing und das Ausmaß des Einflusses durch das Management.

Als Referenzszenario diente die Betrachtung der innerbetrieblichen Serviceorientierung. Vorhandene und neue Anwendungen im Kontext von Web Services und die Einbindung von Kunden und externen Geschäftspartnern bilden die Erfahrungsbasis für die Entwicklung des Shared Services Center auf Basis einer SOA. Hier werden unter starker Einflussmöglichkeit des Management Anwendungen an zentraler Stelle für verschiedene Organisationseinheiten im Unternehmen betrieben. Die service-orientierte Gestaltung von Schnittstellen im Rahmen eines Standardisierungsprozesses ermöglicht zusammen mit dem Aufbau eines innerbetrieblichen UDDI-Verzeichnisses den Kontrollverlust der angeschlossenen Organisationseinheiten abzufedern und eine Anbindung an die zentralen Applikationen selbst über das Internet zu

gewährleisten.

In dem Szenariotyp Outsourcing zeigt sich der geringer werdende Einfluss der Unternehmensführung und dessen Konsequenzen auf die Schnittstellen der ausgelagerten Anwendung. Während gezeigt werden konnte, dass genauso wie im SSC durch Web Services die technischen Hürden für die Fremdvergabe des Betriebes einer Anwendung und der Wiedereingliederung im Rahmen des Insourcing gesenkt bzw. beseitigt werden können, bleibt die Managementaufgabe im Erhalten von Wissen über die Anwendung weiter erfolgskritisch. Der letzte Szenariotyp beschäftigte sich mit dem Provisioning. Ausgehend von klassischen ASP wurden nach der Betrachtung einzelner Dienstarten im Provisioning verschiedene Provisioningszenarien im Blickwinkel des Next Generation Service Provider geschildert. Unterstützt durch die Web Service- und Grid-Technologie bietet dieser neuartige Providertyp mehr als die Dienste an, die früher im Rahmen des dem Application Outsourcing zuzuordnenden ASP möglich waren. Es werden Szenarien der einmaligen Nutzung eines Dienstes genauso denkbar, wie die Inanspruchnahme externer Rechenleistung in Form des Grid als Umsetzung der Vision der „Rechenleistung aus der Steckdose“. Wichtig ist hierbei der Standardcharakter der Dienste. Eine direkte Einflussnahme der Unternehmensleitung auf die Gestaltung der Leistung und auf das Aussehen der Schnittstellen ist nicht mehr gewährleistet.

Vor diesem Hintergrund wurden Abrechnungsvarianten thematisiert, die einerseits den neuen Nutzungsszenarien gerecht werden und andererseits die wirtschaftlich sinnvolle Unterstützung der Abrechnung einer einmaligen Service-nutzung gewährleisten. Insbesondere die Abrechnung bildet neben der technischen Ausgereiftheit der Infrastruktur ein wichtiges Erfolgskriterium für das Provisioning der Zukunft.

Die prototypischen Umgebungen konzentrieren sich auf einzelne Aspekte, die für die IT-Sourcingszenarien wichtig sind. Durch sie wird das flexible Sourcing erst ermöglicht bzw. die Grundlage für den Next Generation Service Provider gelegt. Auch hier steht eine für die innerbetriebliche Service-Ausrichtung stehende ERP-Umgebung am Anfang. Durch den Einsatz von Business Objekten im Rahmen der ALE-Integration mit SAP und Fremdsystemen und dem Einsatz des SAP Business Connectors konnten service-orientierte Ansätze



wie das Generieren von Schnittstellen und eine Art der dynamischen Partnerfindung gezeigt werden. Web Services bieten die für Altanwendungen und Anwendungssysteme mit speziellen Programmierschnittstellen wichtige Programmiersprachenunabhängigkeit bei der Implementierung. Hier wurde dies im Rahmen der Integration von Web Services in ein Portal mit Hilfe der Skriptsprache PHP gezeigt, wobei die Server-Komponenten sowohl in PHP als auch in Java umgesetzt wurden. Die mit dem Globus Toolkit vorgestellte Grid-Umgebung verdeutlicht die Verwendung des Web Services Resource Frameworks als Bindeglied zwischen der Web Service-Welt mit zustandslosen Services und den stateful services, die bei Grids nötig sind. Die beiden letzten Umgebungen bilden die Grundlage einer integrierten Hosting-Umgebung wie sie für den Next Generation Service Provider wichtig wird. Während bis jetzt das Angebot und die Nutzung bestimmter Services bedeutsam war, werden nun vor allem Aspekte der geeigneten Infrastruktur in den Vordergrund gestellt. Insbesondere werden die Rolle von Metadaten für die Abrechnung bzw. für das Management der Umgebung und die Möglichkeit der Migration von Hosting-Umgebungen betrachtet. Durch den Einsatz von Virtualisierungstechniken entsteht eine flexible Infrastruktur, die auch die Einbindung von weiteren Providern von Rechenleistung transparent ermöglicht.

Zusammenfassend zeigt sich also die Eignung von Web Services zur Abmilderung der Folgen der abnehmenden Einflussnahme durch das Management im Rahmen des IT-Sourcing. Der Aufbau einer serviceorientierten Architektur unterstützt sowohl innerbetriebliche Zentralisierungsprozesse als auch die Integration von Diensten, die durch unternehmensexterne Partner erbracht werden. Es konnte gezeigt werden, dass Einmalnutzungsszenarien von Services auch unter dem Aspekt der Abrechnung als kritischem Erfolgsfaktor auch wirtschaftlich sinnvoll realisiert werden können. Zusätzlich wird durch die Kombination von Grid-Computing und Web Services ein neuartiger Typ von Service Provider möglich, der neben konkreten Anwendungsdiensten auch als Infrastrukturprovider im Rahmen des Web Service-Hosting und des Hostings von Grid-Umgebungen auftreten kann.

## **VI.2 Ausblick**

Weiterer Forschungsbedarf liegt nun in der Validierung der Ergebnisse anhand konkreter Sourcing-Projekte in der betrieblichen Praxis. Aufgrund der Neuartigkeit der vorgestellten Technologien musste dies im Rahmen der Arbeit unterbleiben. Impulse der verwendeten Open Source-Produkte werden im Augenblick in kommerzielle Produkte integriert bzw. es wird eine Verbindung zwischen klassischen Grid-Softwareprodukten und den im Globus Toolkit implementierten WSRF angeboten. Sun Microsystems bietet beispielsweise seit Mitte März in den USA ihr Grid Compute Utility, ein Grid das auf Solaris 10 und der Sun N1 Grid Engine basiert, an und verrechnet für die genutzte Rechenleistung 1 US-\$ pro Stunde und Prozessor.<sup>265</sup>

Provider mit Angeboten im Bereich Web Services und Grids sollten analysiert werden. Somit könnten Informationen über die Chancen eines integrierten Grid- und Web Service-Angebots evaluiert werden und weiteres Wissen über die Gestaltung eines marktgerechten Produktportfolios gesammelt werden. Das Angebot von Sun bietet einen Hinweis darauf, dass Bedarf an den im Rahmen der Arbeit vorgestellten Next Generation Service Provider zumindestens schon im Bereich der zur Verfügungsstellung von Rechenleistung besteht. SUN möchte das bisher nur in den USA bestehende Angebot auch auf Europa ausdehnen.<sup>266</sup>

Zusätzlich könnten Semantic Web Services betrachtet werden, um Hindernisse bei der Suche geeigneter Dienste aus dem Weg zu schaffen. Semantische Informationen sind dort hilfreich, wo der Standardisierungsprozess keine einheitlichen Datenformate hervorbringt.

---

<sup>265</sup> Vgl. SUN MICROSYSTEMS (2006), S. 7 f. und HEISE ONLINE (2006)

<sup>266</sup> Vgl. HEISE ONLINE (2006)

## Literaturverzeichnis

### **BASS/ CLEMENTS/ KAZMAN (1998)**

Bass, Len; Clements, Paul; Kazman, Rick: Software Architecture in Practice, Reading, Massachusetts (1998)

### **BAYER (2005)**

Bayer, Martin: SAP bringt den ESA-Zug nicht ins Rollen; in: Computerwoche Nr. 48 (2005), S. 6 f.

### **BEHME (1993)**

Behme, Wolfgang: ZP-Stichwort: Outsourcing; in: Zeitschrift für Planung (1993) 3, S. 291 – 294

### **BECKERLE u. a. (2005)**

Beckerle, Mike; Buddhikot, Mukund; Chatterjee, Debu; Clark, Alan; Coulter, Tim; Enescu, Michael; Goyal, Brajesh; Kataoka, Masato; Kumar, Raj; Pearson, David; Reddy, Surendra; Rodriguez, Tony; Saiyed, Junaid; Schibler, Ross; Schleimer, Stephen; Schmitz, Michael; Sheen, Ron; Skardal, Harald; Souder, Benny, Strong, Paul (Hrsg.); Sudo, Kazunori; Thome, Bob; Viswanathan, Vinod: Enterprise Grid Alliance – Reference Model v1.0 13<sup>th</sup> April 2005; URL: [http://www.gridalliance.org/en/documents/TWGDocs/05198r01EGA\\_RefMod-EGA%20Reference%20Model/%20v1.0%20\\_English.pdf](http://www.gridalliance.org/en/documents/TWGDocs/05198r01EGA_RefMod-EGA%20Reference%20Model/%20v1.0%20_English.pdf) [Stand: 13.04.2005, letzter Zugriff am 26.01.2006]

### **BITKOM (2005)**

Bundesverband Informationswirtschaft, Telekommunikation und neue Medien e.V. (BITKOM): Business Process Outsourcing Leitfaden – BPO als Chance für den Standort Deutschland, Stand 20. September 2005, URL: [http://www.bitkom.org/files/documents/BITKOM\\_Leitfaden\\_BPO\\_Stand\\_20.09.05.pdf](http://www.bitkom.org/files/documents/BITKOM_Leitfaden_BPO_Stand_20.09.05.pdf) [letzter Zugriff am 17.11.2005]

**BLOZAN (2005)**

Blozan, Nick: Service-Oriented Architectures and Software as a Service; URL: [http://www.ebizq.net/topics/biz\\_opt/features/5899.html](http://www.ebizq.net/topics/biz_opt/features/5899.html) [Stand: 08.05.2005, letzter Zugriff: 03.01.2006]

**BOOTH u. a. (2004)**

Booth, David; Haas, Hugo; McCabe, Francis; Newcomer, Eric; Champion, Michael; Ferris, Chris; Orchard, David: Web Services Architecture - W3C Working Group Note 11 February 2004, URL: <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/> [Stand: 11.02.2004, letzter Zugriff am 30.03.2005]

**BUCK-EMDEN (1999)**

Buck-Emden, Rüdiger: Die Technologie des SAP R/3 Systems; Bonn (1999)

**BÜHNER/ TUSCHKE (1997)**

Bühner, Rolf; Tuschke, Anja: Outsourcing; in: Die Betriebswirtschaft 57 (1997), S. 20 - 30

**BURBECK (2000)**

Burbeck, Steve: The Tao of e-business services, URL: <http://www-106.ibm.com/developerworks/library/ws-tao/index.html> [letztes Update am 31.10.2000, letzter Zugriff am 27.08.2003]

**CARDOSO/ SHETH (2005)**

Cardoso, Jorge; Sheth, Amit: Introduction to Semantic Web Services and Web Process Composition; in: Cardoso, Jorge; Sheth, Amit (Hrsg.): Semantic Web Services and Web Process Composition 2004, San Diego, California, July 2004, Lecture Notes in Computer Science Volume 3387 (2005), S. 1-13

**CARR (2005)**

Carr, Nicolas G.: The End of Corporate Computing; in: MIT Sloan Management Review, Vol. 46 No. 3, (Spring 2005), S. 67 - 73

**CASTLE (2005)**

Castle, Bryan: Introduction to Web Services for Remote Portlets; URL: <http://www-128.ibm.com/developerworks/webservices/library/ws-wsrp/> [Stand: 15.04.2005, letzter Zugriff am 01.03.2006]

**CHAPPEL/ TYLER (2003)**

Chappell, David A.; Jewell, Tyler: Java Web Services, 1. Auflage, (deutsche Ausgabe), Beijing u.a. (2003)

**COLUMBUS (2000)**

Columbus. Louis: Realizing e-Business with Application Service Providers – The Authoritative Solution, Indianapolis (2000)

**CUNNINGHAM/ FRÖSCHL (1995)**

Cunningham, Peter A.; Fröschl, Friedrich: Outsourcing – Strategische Bewertung einer Informationsdienstleistung, Frankfurt am Main (1995)

**CZAJKOWSKI u. a. (2004)**

Czajkowski, Karl; Ferguson, Donald F.; Foster, Ian; Frey, Jeffrey; Graham, Steve; Sedukhin, Igor; Snelling, David; Tuecke, Steve; Vambenepe, William: The WS-Resource Framework; URL: <http://www.globus.org/wsrf/specs/ws-wsrf.pdf> [Stand: 05.03.2004, letzter Zugriff am 14.02.2006]

**DADAM (1996)**

Dadam, Peter: Verteilte Datenbanken und Client/Server-Systeme - Grundlagen, Konzepte, Realisierungsformen; Berlin u. a. (1996)

**DEITEL u. a. (2003)**

Deitel, Harvey; Deitel, Paul; Gadzik, Jon; Lomeli, Kyle; Santry, Sean; Zhuang, Su: JAVA Web Services for Experienced Programmers; Upper Saddle River (2003)

**DIBBERN (2004)**

Dibbern, Jens: The Sourcing of Application Software Services – Empirical Evidence of Cultural, Industry and Functional Differences; zugleich Dissertation, Universität Bayreuth, 2003; Heidelberg (2004)

**DITTRICH/ BRAUN (2004)**

Dittrich, Jörg; Braun, Marc: Business Process Outsourcing – Ein Entscheidungsleitfaden für das Out- und Insourcing von Geschäftsprozessen, Stuttgart (2004)

**DSAG (2005)**

Deutschsprachige SAP Anwendergruppe e. V. (DSAG): Newsletter – Ausgabe Februar 05/ März 05, URL: [http://www.dsag.de/pdf/dsag\\_newsletter\\_2005-03-1719.pdf](http://www.dsag.de/pdf/dsag_newsletter_2005-03-1719.pdf) [letzter Zugriff am 24.11.2005]

**EBERHART/ FISCHER (2003)**

Eberhart, Andreas; Fischer, Stefan: Web Services – Grundlagen und praktische Umsetzung mit J2EE und .NET, München, Wien (2003)

**ERL (2005)**

Erl, Thomas: Service-Oriented Architecture – Concepts, Technology, and Design; New York u.a. (2005)

**FOEGEN (2003)**

Foegen, Malte: Architektur und Architekturmanagement, in: Brenner, Walter; Meier, Andreas; Zarnekow, Rüdiger (Hrsg.): Strategisches IT-Management; HMD – Praxis der Wirtschaftsinformatik Heft 232 (August 2003); S. 57 - 65

**FORSTER/ DE MEER (2004)**

Forster, Florian; De Meer, Hermann: Discovery of Web Services with a P2P Network; in: Bubak, Marian; van Albada, Geert Dick; Sloot, Peter; Dongarra, Jack (Hrsg.): ICCS 2004, Lecture Notes in Computer Science Volume 3038, 2004, S. 90–97

**FOSTER u. a. (2004)**

Foster (Hrsg.), Ian; Frey (Hrsg.), Jeffrey; Graham (Hrsg.), Steve; Tuecke (Hrsg.), Steve; Czajkowski, Karl; Ferguson, Don; Leymann, Frank; Nally, Martin; Sedukhin, Igor; Snelling, David; Storey, Tony; Vambenepe, William; Weerawarana, Sanjiva: Modelling Stateful Resources with Web Services - Version 1.1; URL: <http://www-128.ibm.com/developerworks/library/ws-resource/ws-modelingresources.pdf> [Stand: 05.03.2004, letzter Zugriff am 08.02.2006]

**FOSTER (2005)**

Foster, Ian: Globus Toolkit Version 4: Software for Service-Oriented Systems; in: Jin, Hai; Reed, Daniel; Jiang, Wenbin (Hrsg.): Network and Parallel Computing 2005, IFIP International Conference, NPC 2005, Beijing, China, Lecture Notes in Computer Science Volume 3779 (2005), S. 2 - 13

**FOSTER/ IAMNITCHI (2003)**

Foster, Ian; Iamnitchi, Adriana: On Death, Taxes, and the Convergence of Peer-to-Peer and Grid Computing; URL: [http://people.cs.uchicago.edu/~anda/papaers/foster\\_grid\\_vs\\_p2p.pdf](http://people.cs.uchicago.edu/~anda/papaers/foster_grid_vs_p2p.pdf) [Stand: Februar 2003, letzter Zugriff am: 08.01.06]

**FOSTER/ KESSELMANN (2004)**

Foster, Ian; Kesselmann, Carl: Concepts and Architecture; in: Foster, Ian; Kesselmann, Carl (Hrsg.): The GRID - Blueprint for a New Computing Infrastructure; 2. Auflage ; Amsterdam u.a. (2004); S. 37 – 63

**FRIEDMANN (2005)**

Friedmann, Katharina: Auf Tuchfühlung mit SOA, in: Computerwoche Nr. 50 (2005) vom 16.12.2005, S. 39

**GLOBUS ALLIANCE (o. J.)**

Globus Alliance: An „Ecosystem“ of Grid Components; URL: [http://www.globus.org/grid\\_software/ecology.php](http://www.globus.org/grid_software/ecology.php) [letzter Zugriff am 25.01.2006]

**GRASMANN (2005)**

Grasman, Stefan: SOA-Risiken werden gern verschwiegen; in: Computerwoche Nr. 34 (2005) vom 26.08.2005, S. 16 f.

**GRAUPNER u. a. (2001)**

Graupner, Sven; Kim, Wooyoung; Sahai; Akhil; Lenkov, Dmitry: E-Speak – An XML Document Interchange Engine, in: Bauknecht, K.; Madria, S. K.; Pernul, G. (Hrsg.): EC-Web 2001, Lecture Notes in Computer Science Volume 2115 (2001), S. 270 - 279

**GRUBER (1993)**

Gruber, Thomas R.: Toward Principles for the Design of Ontologies Used for Knowledge Sharing; URL: <http://citeseer.ist.psu.edu/gruber93toward.pdf> [letzte Überarbeitung am 23.08.1993, letzter Zugriff am 03.02.2006]

**GUDGIN u. a. (2003)**

Gudgin, Martin; Hadley, Marc; Mendelsohn, Noah; Moreau, Jean-Jacques (Hrsg.): SOAP Version 1.2 Part 1: Messaging Framework – W3C Recommendation 24 June 2003; URL.: <http://www.w3.org/TR/soap12-part1/> [Stand: 24.06.2003, letzter Zugriff am 11.11.2005]

**GUGEL (2006)**

Gugel, Tim: Auf das Design der Services kommt es an; in: Computerwoche Nr. 2 (2006) vom 13.01.2006, S. 26



**HAUSER/ LÖWER (2004)**

Hauser, Tobias; Löwer, Ulrich: Web Services – Die Standards, Bonn (2004)

**HAUSWIRTH/ DUSTDAR (2005)**

Hauswirth, Manfred; Dustdar, Schahram: Peer-to-Peer: Grundlagen und Architektur; in: Datenbank Spektrum, Heft 13 (Mai 2005), S. 5 - 13

**HE (2003)**

He, Hao: What is Service-Oriented Architecture?, URL:

<http://www.xml.com/pub/a/ws/2003/09/30/soa.html> [Stand: 30.09.2003, letzter Zugriff am 14.10.2003]

**HE (2004)**

He, Hao; Haas, Hugo; Orchard, David (Hrsg.): Web Service Architecture Usage Szenarios – Working Group Note 11 February 2004, URL:

<http://www.w3.org/TR/2004/NOTE-ws-arch-scenarios-20040211/> [Stand: 11.02.2004, letzter Zugriff am 22.08.2005]

**HEISE ONLINE (2006)**

Heise Online (2006): SUN-Grid ab Montag für US-Kunden verfügbar, URL:

<http://www.heise.de/newsticker/result.xhtml?url=/newsticker/meldung/70918&words=Sun%20Grid> [Stand: 16.03.2006, letzter Zugriff am 08.04.2006]

**HEINRICH/ LEHNER (2005)**

Heinrich, Lutz J.; Lehner, Franz: Informationsmanagement - Planung, Überwachung und Steuerung der Informationsinfrastruktur; 8., vollständig überarbeitete und ergänzte Auflage; München/ Wien (2005)

**HERRMANN (2004)**

Herrmann, Wolfgang: Welche Technologien werden wichtig, in Computerwoche Nr. 33 (2004) vom 13.08.2004, S. 5.

**HERRMANN (2005)**

Herrmann, Wolfgang: In zehn Schritten zur SOA; in: Computerwoche Nr. 49 (2005), S. 8 f.

**HERZBERG (2003)**

Herzberg, Amir: Micropayments; in: Kou, Weidong (Hrsg.): Payment Technologies for E-Commerce, Berlin u.a. (2003), S. 245 - 282

**HEYWOOD (2001)**

Heywood, Brian J.: The Outsourcing Dilemma – The Search for Competitiveness; London u.a. (2001)

**HOMBERGER/ SCHNEIDER (2000)**

Homberger, Werner; Schneider, Jürgen: Sicherheit und Datenschutz mit SAP-Systemen; Bonn (2000)

**HORCHLER (1996)**

Horchler, Hartmut: Outsourcing – Eine Analyse der Nutzung und ein Handbuch der Umsetzung, Köln (1996)

**HUBER (2006)**

Huber, Walter: Blaupausen für die IT-Landschaften; in: is report, 10. Jahrgang, Heft 1+ 2 (2006), S. 63 - 65

**IBM (2005)**

IBM: Web Services Transactions Specifications; URL: <http://www-128.ibm.com/developerworks/library/specification/ws-tx/> [Erstellt: 01.11.2004, letztes Update am 16.08.2005, letzter Zugriff am 10.04.2006]

**IDC (2005)**

IDC: Spending on Software as a Service Exceeds Expectations and Will Continue to Grow Through 2009, IDC finds; URL: <http://www.idc.com/getdoc.jsp?containerId=prUS00137905> [Stand: 04.05.2005, letzter Zugriff am 03.01.2006]

**JOUANNE-DIEDRICH (2005)**

Jouanne-Diedrich von, Holger: Die ephorie.de IT-Sourcing Map; URL: <http://www.ephorie.de/it-sourcing-map.htm> [letzter Zugriff am 18.11.2005]

**JOUANNE-DIEDRICH/ ZARNEKOW/ BRENNER (2005)**

Jouanne-Diedrich von, Holger; Zarnekow, Rüdiger; Brenner, Walter: Industrialisierung des IT-Sourcings; in: Strahinger, Susanne (Hrsg.): Outsourcing; HMD – Praxis der Wirtschaftsinformatik Heft 245 (Oktober 2005); S. 18 - 27

**KLEINSCHMIDT/ EDER (2000)**

Kleinschmidt, Peter; Eder, Max: An ASP for Education; in: SAP Info Nr. 72 (2000), URL: <http://www.sap-info.de/public/INT/int/printout/article/PrintEdition-78313c678e1e5494e-int/-1/comvArticleContainer-198983c63c4f0d194e/> [Stand: 14.07.2000, letzter Zugriff am 31.03.2006]

**KNOLMAYER (2000)**

Knolmayer, Gerhard F.: Application Service Providing (ASP); in: Wirtschaftsinformatik Nr. 42 (2000), S. 443 – 446

**KOU (2003)**

Kou, Weidong: Challenges and Opportunities in E-Payment; in: Kou, Weidong (Hrsg.): Payment Technologies for E-Commerce, Berlin u.a. (2003), S. 301 - 307

**KÖHLER-FROST (1995)**

Köhler-Frost, Wilfried: Outsourcing – sich besinnen auf das Kerngeschäft; in Köhler-Frost, Wilfried (Hrsg.): Outsourcing – Eine strategische Allianz besonderen Typs; 2., völlig neu bearbeitete und wesentlich erweiterte Auflage, Berlin (1995), S. 13 - 24

**KRAFZIG/ BANKE/ SLAMA (2005)**

Krafzig, Dirk; Banke, Karl; Slama, Dirk: Enterprise SOA – Service-Oriented Architecture Best Practices; Upper Saddle River 2005

**KÜSTER (2003)**

Küster, Marc Wilhelm: Web-Services – Versprechen und Realität; in: Fröschle, Hans Peter (Hrsg.): Web-Services, HMD – Praxis der Wirtschaftsinformatik Heft 234 (Dezember 2003), S. 5 - 15

**LANGNER (2003)**

Langner, Thomas: Web Services mit Java, München (2003)

**LINTHICUM (2001)**

Linthicum, David S.: B2B Application Integration - e-Business-Enable Your Enterprise; Boston u.a. (2001)

**MAY/ SAVUR (2006)**

May, Bob; Savur, Praveen: Business Process Execution Language, Part 1: An Introduction; URL:  
<http://developers.sun.com/prodtech/javatools/jsenterprise/tpr/reference/techart/bpel.html> [Stand: 10.03.2006, letzter Zugriff am 31.03.2006]

**MITRA (2003)**

Mitra, Nilo (Hrsg.): SOAP Version 1.2 Part 0: Primer – W3C Recommendation 24 June 2003; URL: <http://www.w3.org/TR/soap12-part0/> [Stand: 24.06.2003, letzter Zugriff am 11.11.2005]

**MOHR/ SIMON/ KRCMAR (2005)**

Mohr, M.; Simon, T.; Krcmar, H.: Building an Adaptive Infrastructure for Education Service Providing; in: Ferstl, O.K.; Sinz, E.J.; Eckert, S.; Isselhorst, T. (Hrsg.): Proceedings der Tagung Wirtschaftsinformatik 2005, Bamberg (2005), S. 847 - 859 zitiert nach URL:

[http://www.krcmar.in.tum.de/lehrstuhl/publikat.nsf/intern01/CDA73DEA5ACB3469C125703E0029FAFF/\\$FILE/05-09.pdf](http://www.krcmar.in.tum.de/lehrstuhl/publikat.nsf/intern01/CDA73DEA5ACB3469C125703E0029FAFF/$FILE/05-09.pdf) [letzter Zugriff am 12.04.2006]

**MURRAY/ WILSON/ ELLISON (2006)**

Murray, Bryan; Wilson, Kirk; Ellison, Mark (Hrsg.): Web Services Distributed Management: MOWS Primer - OASIS Committee Draft, February 24, 2006;

URL: <http://docs.oasis-open.org/wsdm/wsdm-1.0-mows-primer-cd-01.pdf>

[Stand: 24.02.2006, letzter Zugriff am 15.04.2006]

**MYERSON (2004)**

Myerson, Judith M.: Guarantee second-generation Web service applications with a SLA,

URL: <http://www-106.ibm.com/developerworks/webservices/library/ws-wssla/>

[letzter Zugriff am 19.08.2004]

**MYLOTT (1995)**

Mylott III, Thomas R.: Computer Outsourcing – Managing the Transfer of Information Systems; Englewood Cliffs (1995)

**NIEMANN (2005)**

Niemann, Frank: R/3 Migration: Die Qual der Wahl; in: Computerwoche Nr. 43 (2005) vom 28.10.2005, S. 16 f.

**OASIS (2004)**

Organization for the Advancement of Structured Information Standards

(OASIS): Introduction to UDDI: Important Features and Functional Concepts;

Oktober 2004; URL: <http://uddi.org/pubs/uddi-tech-wp.pdf>

**Object Management Group (2004)**

Object Management Group: History of CORBA, URL:

[http://www.omg.org/gettingstarted/history\\_of\\_corba.htm](http://www.omg.org/gettingstarted/history_of_corba.htm) [letztes Update am 19.08.2004, letzter Zugriff am 13.12.2004]

**OEHLER (2005)**

Oehler; Karsten: Business Engineering bei Einführung betriebswirtschaftlicher Standardsoftware – Auswirkung einer serviceorientierten Architektur; in: Strahlringer, Susanne (Hrsg.): Business Engineering, HMD – Praxis der Wirtschaftsinformatik Heft 241 (Februar 2005), S. 35 – 44

**PARHONYI/ NIEUWENHUIS/ PRAS (2005)**

Parhonyi, Robert; Nieuwenhuis, Lambert J. M.; Pras, Aiko: Second generation micropayment systems: lessons learned; URL:

[http://www.finextra.com/Finextra\\_downloads/featuredocs/parhonyi\\_2ndgeneration@i3e2005.pdf](http://www.finextra.com/Finextra_downloads/featuredocs/parhonyi_2ndgeneration@i3e2005.pdf) [Stand: 2005, letzter Zugriff am: 09.01.2006]

**PARHONYI/ PRAS/ QUARTEL (2004)**

Parhonyi, Robert; Pras, Aiko; Quartel, Dick: Collaborative Micropayment Systems; URL:

<http://www.simpleweb.org/nm/research/results/publications/pras/2004-WTC-Micropayment.pdf> [Stand 2004, letzter Zugriff am 05.01.2006]

**QI u. a. (2006)**

Qi, Li; Jin, Hai; Foster, Ian; Gawor, Jarek: HAND: High Available Dynamic Deployment Infrastructure for Globus Toolkit 4; URL:

[http://www.globus.org/alliance/publications/papers/HAND\\_Submitted.pdf](http://www.globus.org/alliance/publications/papers/HAND_Submitted.pdf) [Stand: 2006; letzter Zugriff am 18.01.2006]

**REITER (2005)**

Reiter, Michael: Serviceorientierung kostet Arbeitsplätze; in Computerzeitung 35. Jahrgang Nr. 48 (2005) vom 28.10.2005, S. 1

**RIPIN/ SAYLES (1999)**

Ripin, Kathy M.; Sayles, Leonard R.: Insider Strategies for Outsourcing Information Systems – Building Productive Partnerships, Avoiding Seductive Traps; Oxford u.a. 1999

**SAP (2001)**

SAP: BIT 530 SAP Business Connector; Kursordner; Mat.-Nr. 5004 9320; Juli 2001

**SAP (2005a)**

SAP: SAP BC Architecture; URL:

[http://help.sap.com/saphelp\\_nw04s/helpdata/en/8c/c25d41a9d311d6b28f00508b6b8a93/frameset.htm](http://help.sap.com/saphelp_nw04s/helpdata/en/8c/c25d41a9d311d6b28f00508b6b8a93/frameset.htm) [letzter Zugriff am 20.12.2005]

**SAP (2005b)**

SAP: SAP BC Routing; URL:

[http://help.sap.com/saphelp\\_nw04s/helpdata/en/8c/c25d41a9d311d6b28f00508b6b8a93/frameset.htm](http://help.sap.com/saphelp_nw04s/helpdata/en/8c/c25d41a9d311d6b28f00508b6b8a93/frameset.htm) [letzter Zugriff am 20.12.2005]

**SESTER (2004)**

Sester, Peter: Vertragsschluss und Verbraucherschutz beim Einsatz von Software-Agenten; in: Informatik-Spektrum, Band 27 Heft 4 vom 04.08.2004, S. 311-322

**SCHRÖDER (1995)**

Schröder, Jürgen: Outsourcing: Entsorgungsmodell oder Innovationspartnerschaft?; in: Köhler-Frost, Wilfried (Hrsg): Outsourcing – Eine strategische Allianz besonderen Typs; 2., völlig neu bearbeitete und wesentlich erweiterte Auflage, Berlin 1995, S. 25-42

**SLAMA (2006)**

Slama, Dirk: Von der Legacy- in die SOA-Welt; in: Computerwoche Nr. 2 (2006), S. 24f.

**SNEED (2006)**

Sneed, Harry M.: Wrapping Legacy Software for Reuse in a SOA; in: Lehner, Franz; Nösekabel, Holger; Kleinschmidt, Peter (Hrsg.): Multikonferenz Wirtschaftsinformatik 2006 (Tagungsband 2), Berlin (2006), S. 345 - 360

**SOTOMAYOR (2005)**

Sotomayor, Borja: The Globus Toolkit 4 Programmer's Tutorial; URL: [http://www.casa-sotomayor.net/gt4-tutorial/singlehtml/progtutorial\\_0.2.1.html](http://www.casa-sotomayor.net/gt4-tutorial/singlehtml/progtutorial_0.2.1.html) [Stand: 2005, letzter Zugriff am: 08.03.2006]

**STEINBACH/ SEITTER (2004)**

Steinbach, Torsten; Seitter Jörg: Data Grids – eine Bestandsaufnahme; in: Datenbank-Spektrum Nr. 9 (2004), S. 34 - 39

**SUN MICROSYSTEMS (1995)**

SUN Microsystems Inc.: ONC+ Developers Guide, URL: <http://docs-pdf.sun.com/802-1997/802-1997.pdf> [letzter Zugriff am 17.01.2005]

**SUN MICROSYSTEMS (2006)**

SUN Microsystems Inc.: SUN Grid Compute Utility - Rerence Guide, URL: <http://www.sun.com/service/sungrid/SunGridUG.pdf> [Stand: Januar 2006, letzter Zugriff am 07.04.2006]

**TILKOV (2004)**

Tilkov, Stefan: UDDI Revisited – Mit der Version 3 wird UDDI erwachsen; in XML & Web Service Magazin 2/2004, S. 37 - 40

**WATT (2002)**

Watt, Douglas: e-business implementation - A guide to web services, EAI, BPI, e-Commerce, content management, portals and supporting technologies; Woburn (MA) (2002)



**WALDO u. a. (1994)**

Waldo, Jim; Wyatt, Geoff; Wollrath, Ann; Kendall, Sam: A Note on Distributed Computing, URL: <http://research.sun.com/techrep/1994/abstract-29.html> bzw. [http://research.sun.com/techrep/1994/smil\\_tr-94-29.pdf](http://research.sun.com/techrep/1994/smil_tr-94-29.pdf) [letzter Zugriff am 26.06.2004]

**WANG u. a. (2004)**

Wang, Dapeng (Hrsg.); Bayer, Thomas; Frotscher, Thilo; Teufel, Marc: Java Web Services mit Apache Axis, Frankfurt am Main 2004.

**WILKES (2005)**

Wilkes, Lawrence: The Web Service Protocol Stack, URL: <http://roadmap.cbdiforum.com/reports/protocols/> [Stand: Februar 2005, letzter Zugriff am 02.11.2005]

**WISSKIRCHEN/ MERTENS (1999)**

Wißkirchen, Frank; Mertens, Helga: Der Shared Services Ansatz als neue Organisationsform von Geschäftsbereichsorganisationen; in: Wisskirchen, Frank (Hrsg.): Outsourcing-Projekte erfolgreich realisieren; Stuttgart 1999, S. 79 - 111

**WILLIAMS (1998)**

Williams, Oakie: Outsourcing: A CIO's Perspective, Boca Raton (1998)

**WOJCIECHOWSKI/ WEINHARDT (2002)**

Wojciechowski, Remigiusz; Weinhardt, Christof: Web Services und Peer-to-Peer-Netzwerke; in: Schoder, Detlef; Fischbach, Kai; Teichmann, René (Hrsg.): Peer-to-Peer, Berlin u.a. (2002), S. 99 - 152

**WOODS (2004)**

Woods, Dan: Enterprise Service Architecture; 1. deutsche Auflage, Bonn (2004)

**XEN (2006)**

o. A. : XEN Users' Manual - XEN v. 3.0; URL:

<http://www.cl.cam.ac.uk/Research/SRG/netos/readmes/user.pdf> [Stand: 2005.

letzter Zugriff am 05.03.2006]