



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung im Sommersemester 2009

Serviceorientiertes eGovernment

Grundlegende Prinzipien

Technische Komponenten

Marktübersicht

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Terminplan (vorläufig)

Nr.	Datum	Thema
1	21.4.2009	Neue Herausforderungen für die öffentliche Verwaltung
2	28.4.2009	Die Bayerische Staatsverwaltung
3	5.5.2009	Zentrale Probleme in komplexen IT-Landschaften
4	12.5.2009	Grundlagen verteilter Systeme
5	19.5.2009	Zielsetzung der Serviceorientierung
6	26.5.2009	Geschäftsarchitektur, Domänen, Anwendungen
7	9.6.2009	Web Services und Verzeichnisdienste
8	16.6.2009	Business Process Management (BPM), Enterprise Service Bus (ESB), Identity Management (IDM)
9	23.6.2009	Die E-Government-Ausschreibung des Freistaats Bayern
10	30.6.2009	SOA Governance, Software Lifecycle mit SOA
11	7.7.2009	Marktgängige Produkte und Ansätze
12	14.7.2009	Open Source Produkte
13	21.7.2009	Warum scheitern SOA-Projekte? Beispiele aus der Praxis.

Kurze Vorstellung

Name: Frank Sarre

Beruf: Berater und IT-Sachverständiger
Geschäftsführer bei der Projective Expert Group, München

Ausbildung: Diplom und Promotion in Informatik (TU München)

Tätigkeitsschwerpunkte:

- Projektmanagement
- Sanierung von IT-Projekten
- Gerichts- und Parteigutachten
- Ausschreibungen
- Fachliche Konzeption
- Test und Abnahme

Branchen: Telko, Automobil, Finanzbranche, Entertainment,
Health Care, Öffentliche Verwaltung

Dr. Frank Sarre

**Anschrift: Ludwig-Maximilians-Universität München (LMU)
Institut für Informatik
Lehr- und Forschungseinheit für
Programmierung und Softwaretechnik (PST)
c/o Fr. M. Diem (Sekretariat von Hrn. Prof. Dr. M. Wirsing)
Oettingenstr. 67
80538 München**

**Telefon: Tel. 089 / 2180 -9151 (Fr. Diem) oder
direkt unter Tel. 089 / 18 92 37 -01**

Email: frank.sarre@pst.ifi.lmu.de

Vorlesungszeiten

Vorlesungen jeweils

dienstags,

Raum 0.33, Oettingenstr. 67

Stets aktuelle Terminpläne unter

www.pst.ifi.lmu.de/Lehre/sose-2009/soa/termine

Prüfungsrelevanz und Schein

Die Vorlesung kann als **Prüfungsfach** angegeben werden
(Details bitte mit Herrn Professor Wirsing klären).

Ein Schein kann für diese Vorlesung leider **nicht** ausgestellt werden.



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 21. April 2009

Serviceorientiertes E-Government

Neue Herausforderungen in der öffentlichen Verwaltung

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Die IT-Landschaft der öffentlichen Verwaltung ist in Deutschland u.a. durch folgende Merkmale gekennzeichnet:

- Große Anzahl von IT-Systemen
- Sehr heterogene IT-Landschaft
- Weitgehend unabhängige Einzelsysteme
- Datenaustausch zwischen den Systemen nur stellenweise möglich
- Behördliche Prozesse nicht durchgängig durch IT-Systeme unterstützt
- Online-Services (E-Government) nur in geringem Umfang verfügbar

- Richtlinien der Europäischen Union
- Vorschriften auf verschiedenen Ebenen in Deutschland:
 - Bund
 - Länder
 - Kommunen
- Notwendige Effizienzsteigerungen für verschiedene Nutzerkreise:
 - Bürger
 - Wirtschaft
 - Andere öffentliche Stellen

Die neuen Herausforderungen führen zu:

- Umstrukturierung von behördlichen Prozessen
- Änderungen der Organisation
- Änderungen, Erweiterungen und Neuentwicklungen von IT-Anwendungen
- Konzeption eines neuen Vorgehens, um die Komplexität in den Griff zu bekommen
- Anspruchsvolle Beschaffungsvorhaben (Vergaberecht)

Was versteht man unter „E-Government“?

E-Government =

Nutzung des Internets und anderer elektronischer Medien zur Einbindung der Bürger und Unternehmen in das Verwaltungshandeln sowie zur verwaltungsinternen Zusammenarbeit

E-Government betrifft also die Beziehungen:

- G2C - Government to Citizen
- G2B - Government to Business
- G2G - Government to Government

Initiativen und Rahmenbedingungen

- Europäische Ebene
- Deutschland
- Bundesländer und Kommunen
- Bayern

- EU-Länder haben unterschiedliche Entwicklungsstände
- Unterschiedliche IT-Systeme
- Unterschiedliche E-Government-Angebote

Initiativen und Rahmenbedingungen

- I2010
(strategische Ziele, E-Government-Aktionsplan)
- IDABC
(grenzüberschreitender Informationsaustausch,
Empfehlungen für die Technologie)
- Europäische Dienstleistungsrichtlinie
(„Single Point of Contact“, elektronische Abwicklung der
Verfahrensschritte für Gewerbeanmeldung und -betreuung)

- Zahlreiche Fachverfahren in den Ländern, weniger im Bund
- Überwiegend formularorientierte Anwendungen
- Zersplitterte Verantwortlichkeiten
- Realisierung behördenübergreifender Prozesse schwierig
- Oberflächen der Anwendungen größtenteils noch nicht für Portale geeignet
- Verlässliche Identifikation der Bürger und der Unternehmen höchstens ansatzweise gelöst
- Gesetzliche Regelungen beschränken den Datenaustausch

Initiativen (Deutschland) [1]

- iD2010
(Leitlinien für die IuK-Technologie, Ziele für E-Government)
- BundOnline 2005
(Bereitstellung aller internetfähigen Dienstleistungen bis Ende 2005)
- Aktionsplan „Deutschland Online“
(Schaffung einer E-Government-Plattform für Bund, Länder, Kreise und Kommunen,
Einzelvorhaben:
 - Infrastruktur
 - Standardisierung
 - Meldewesen
 - Personenstandswesen
 - KFZ-Wesen
 - Dienstleistungsrichtlinie)

Besonderheit:
„Einer-für-alle-Lösungen“ !

- E-Government 2.0
(Grundlagen für andere Vorhaben,
Ergänzung des Aktionsplans „Deutschland Online“,
des Weiteren:
 - Portfolio – Ausbau des E-Government-Angebots des Bundes
 - Prozessketten – Zusammenarbeit zwischen Wirtschaft und Verwaltung
 - Identifikation – Elektronischer Personalausweis und E-Identity-Konzepte
 - Kommunikation – Sichere Kommunikationsinfrastruktur)

- D115 – einheitliche Behördenrufnummer

- Land und Kommunen haben unterschiedliche Aufgaben und daher auch getrennte IT-Systeme
- Getrennte Verantwortungsbereiche zwischen Ländern und Kommunen
- Kein zwingend abgestimmtes Vorgehen zwischen den Ländern

Internet-Portale des Freistaats Bayern:

- www.bayern.de
- www.verwaltung.bayern.de

→ Statische Inhalte, Verzweigung auf andere Portale



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 28. April 2009

Serviceorientiertes E-Government

Die Bayerische Staatsverwaltung

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Ministerien:

- Staatsministerium des Innern (StMI)
- Staatsministerium der Justiz und für Verbraucherschutz (StMJV)
- Staatsministerium für Wissenschaft, Forschung und Kunst (StMWFK)
- Staatsministerium für Unterricht und Kultus (StMUK)
- Staatsministerium der Finanzen (StMF)
- Staatsministerium für Wirtschaft, Infrastruktur, Verkehr und Technologie (StMWIVT)
- Staatsministerium für Umwelt und Gesundheit (StMUG)
- Staatsministerium für Ernährung, Landwirtschaft und Forsten (StMLF)
- Staatsministerium für Arbeit und Sozialordnung, Familie und Frauen (StMAS)

Art. 51 der Bayerischen Verfassung:

- Jeder Staatsminister führt seinen Geschäftsbereich gemäß der politischen Vorgaben des Ministerpräsidenten unter eigener Verantwortung.
- Beschaffungsentscheidungen müssen nicht mit anderen Ministerien abgestimmt werden.

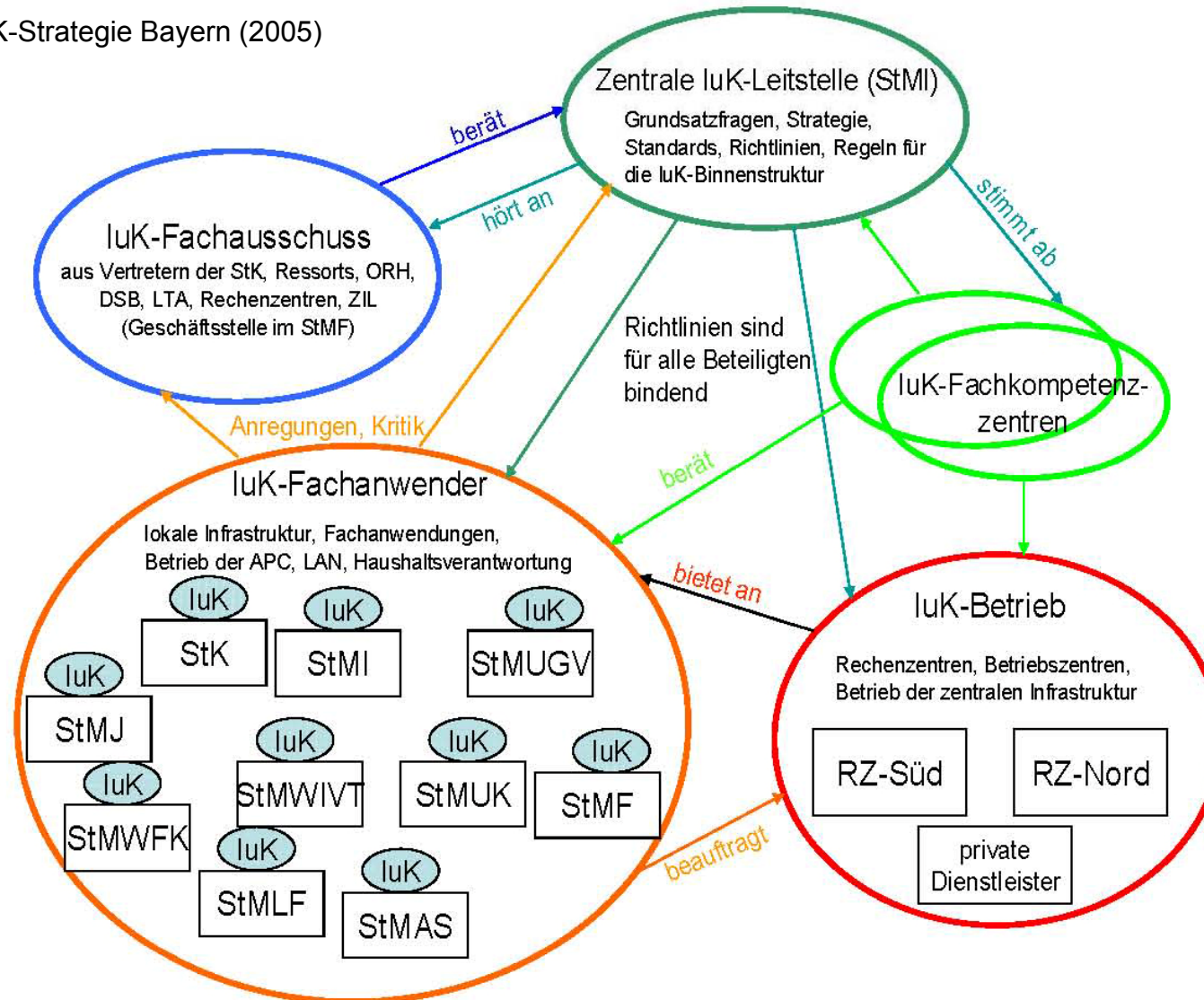
Ausnahme:

Es gibt eine bestimmte Vereinbarung zwischen allen Ministerien, bestimmte Standards einzuhalten.

- Unkoordiniert gewachsene IT-Landschaft

luK-Organisation des Freistaats Bayerns

Quelle: luK-Strategie Bayern (2005)



Aufgaben:

- Entwicklung und Fortschreibung der IuK-Strategie
- Festlegung von Richtlinien, Standards und Schnittstellen sowie die Überwachung ihrer Einhaltung
- Erlass von Richtlinien für den Betrieb der Rechenzentren
- Abstimmung mit den kommunalen Spitzenverbänden
- Vertretung des Freistaats Bayern in Bund-/Ländergremien
- Pflege von Kontakten zu Wissenschaft, Forschung und Entwicklung
- Beobachtung von Marktentwicklungen
- Erlass von Richtlinien für Wirtschaftlichkeitsberechnungen

Zentrale Behörde für amtliche Statistiken in Bayern

- IuK-Dienstleistungen
- Rechenzentrum Süd (RZ-Süd)

STK (Bayerische Staatskanzlei)

- Koordiniert die Tätigkeiten der Ministerien
- Koordiniert bei Bedarf übergreifende IT-Vorhaben

- Richtlinie für den koordinierten Einsatz der IuK in der bayerischen Staatsverwaltung,
konkrete Maßnahmen:
 - Aufbau von Fachkompetenzzentren für Verwaltungsabläufe
 - Herstellung der Interoperabilität von Fachanwendungen
 - „Ganzheitliche Lösung“ für eine Sicherheitsinfrastruktur
 - Entwicklung eines E-Government-Portals
 - Schaffung zentraler Basiskomponenten, z.B. Dokumentenmanagementsystem
 - Priorisierung von Power-Usern mit hohen Fallzahlen
 - Aufbau einer SOA, dabei Nutzung offener Standards

- Dienstorientierte, lose Kopplung von Anwendungen
- Weitestgehende Beibehaltung existierender Anwendungen
- Identifikation von Basiskomponenten / wiederverwendbaren Komponenten
- Gleiche Priorität für kommerzielle Produkte und Open Source
- Neuentwicklungen müssen in die SOA integriert werden können
- Werkzeuge für ein BPM (Business Process Management)
- Berücksichtigung von Datenaustauschstandards
- Bereitstellung einer Portallösung
- Zentraler Verzeichnisdienst (Basis für Sicherheitsinfrastruktur + SSO)
- Sicherer Datenaustausch

Die Datenschutzgesetze sind strikt einzuhalten:

- Bundesdatenschutzgesetz (BDSG)
- Bayerisches Datenschutzgesetz (BayDSG)

→ Anforderungen an technische und organisatorische Maßnahmen

→ Auswirkungen auf die Regeln für einen Datenaustausch

→ Überwachungspflichten

Hinweis:

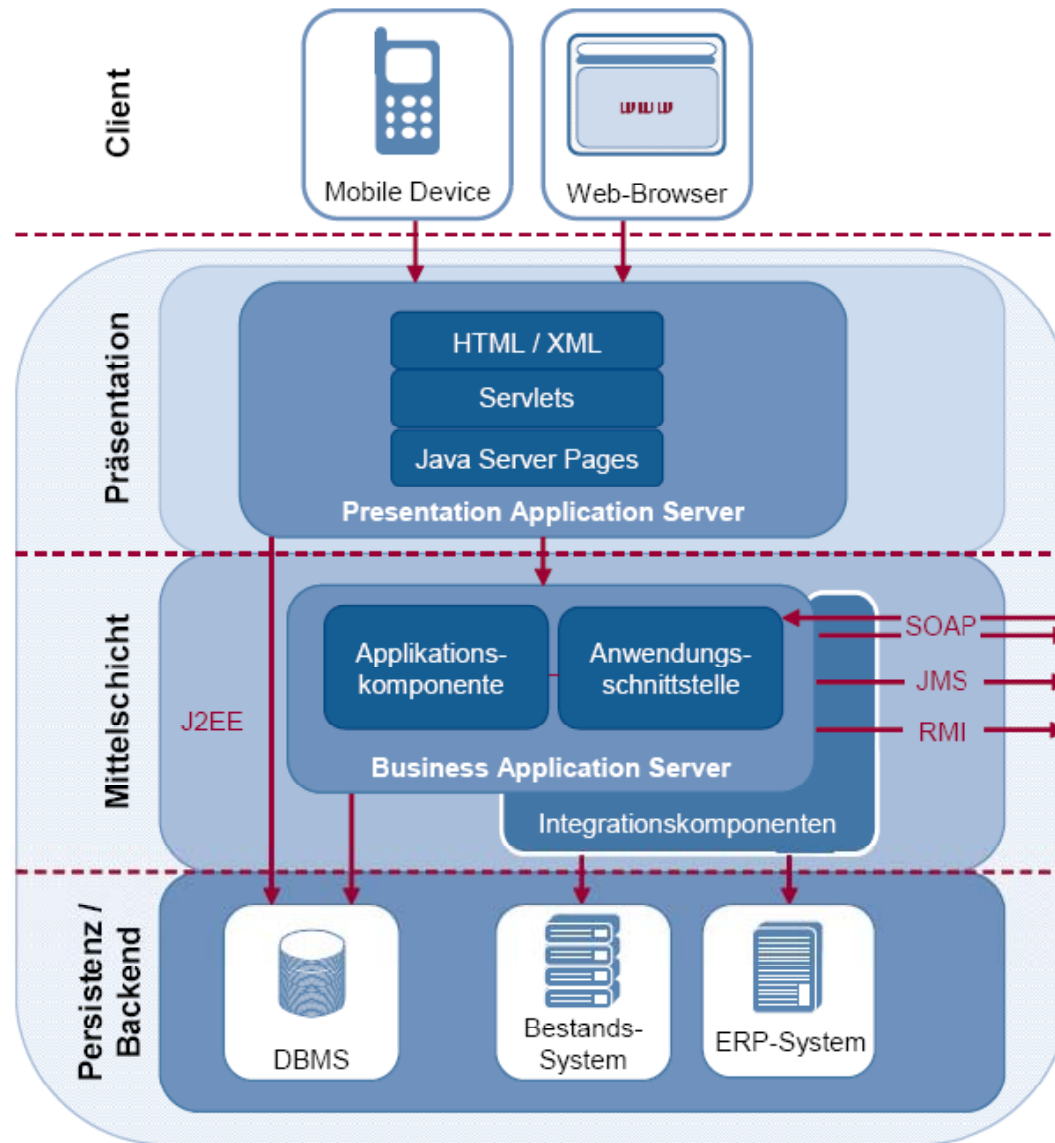
„Datenschutzgerechtes E-Government“ wird durch eine Arbeitsgruppe, die durch Mitarbeiter verschiedener öffentlicher Stellen besetzt ist, konzipiert und schriftlich zusammengefasst.

- = XML-basierter Standard für den sicheren und datenschutzkonformen elektronischen Datenaustausch
- Berücksichtigung von Standards von Web Services
- Implementierung von Sicherheitsaspekten durch enge Anlehnung an WS Security
- Erfolgreicher Einsatz im Melderegisterverfahren XMeld

- Konzeption (S.A.V.E.) wurde 2008 fertig, Umsetzung ist noch nicht erfolgt
- Aufgrund der Komplexität ist ein eigenes, umfangreiches Projekt erforderlich
- Erfolg einer SOA ohne Identitäts- und Rechtemanagement fraglich
- Eine besondere Herausforderung ist, die Identitäten und Rechte aller Bürger und Unternehmen zu erfassen

- = Vom Bund entwickelter Standard für IT-Projekte
- Überwiegend nur Empfehlungen für die Gestaltung der Softwarearchitektur von IT-Projekten (keine verbindlichen Vorgaben!)
- Jedes E-Government-Projekt muss die SAGA-Empfehlungen konkretisieren
- Teile der Empfehlungen wurden in den bayerischen Richtlinien und Standards konkretisiert (BayITR / BayITS)

SAGA – Vier-Schichtenarchitektur



- = Richtlinien und Standards hinsichtlich der Sicherheitsaspekte beim Betrieb von Informationstechnologie
- Allgemeingültiger Sicherheitsratgeber für einen mittleren Sicherheitsstandard
(also kein Ratgeber für Anwendungen mit höchstem Sicherheitsbedarf)
- Handbuch für E-Government-Anwendungen

- = Verbindliche Richtlinien des Freistaats Bayern für die Realisierung von IT-Projekten
1. Anzeige vor IuK-Vorhaben
 2. Durchführung von IuK-Projekten
 3. Planungsrichtlinien für Kommunikationsnetze
 4. Rahmenrichtlinie für die Betriebsleistungen der staatlichen Rechenzentren
 5. Nutzung von Internet und E-Mail in der bayerischen Staatsverwaltung
 6. Softwarekonfigurationsmanagement
 7. Wirtschaftlichkeitsrechnungen im IuK-Bereich
 8. Anwendung der ergänzenden Vertragsbedingungen für die Beschaffung von IT-Leistungen (EVB-IT)
 9. Organisatorische Maßnahmen für die Sicherheit der Infrastruktur

- = Architekturkonzept der ZIL, verschiedener Ministerien und IBM für die strukturelle Gestaltung von neuen E-Government-Anwendungen
 - Technische und organisatorische Anforderungen an eine SOA
 - Themen wie Sicherheitsinfrastruktur, Identitätsmanagement und Prozessmanagement bleiben vage
 - Hinweis auf die Etablierung einer SOA-Governance
- Das Architekturkonzept lässt viele Fragen offen, die für die Beschaffung einer SOA und für die Realisierung von E-Government-Anwendungen zu klären gewesen wären

Auftrag an das LfStaD, den technischen Aufbau sowie den Betrieb von Komponenten für eine E-Government-Infrastruktur auszuführen

- Das Architekturkonzept sollte anhand von folgenden Pilotanwendungen überprüft werden:
 - eNotar
 - Application & Portal Access Management (APAM)
 - ePayment
 - Anbindung des Geodaten-Servers
 - Fischerprüfung
- Anforderungsanalyse und Beschaffung von SOA-Komponenten
- Ausbau zu einem betriebsfähigen Versuchssystem
- Aufbau von eigenem Know-how im LfStaD

1. Behördenübergreifende Prozesse müssen definiert und abgestimmt werden.
2. Für den Datenaustausch zwischen den zuständigen Behörden müssen standardisierte Schnittstellen geschaffen werden.
3. Eine IuK-Infrastruktur muss den Datenaustausch in optimaler Weise unterstützen.
4. Es muss eine Sicherheitsinfrastruktur entwickelt werden, die die Identifikation und Authentisierung von Personen für den rechtsgültigen Informationsaustausch ermöglicht.
5. Für den Zugang zu allen Diensten muss eine Anwendungsoberfläche (Portal) geschaffen werden, die die Informationen aus allen Fachabteilungen integriert.
6. Die Datenschutzbestimmungen sind unbedingt einzuhalten.



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 5. Mai 2009

Serviceorientiertes E-Government

Zentrale Probleme in komplexen IT-Landschaften

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Typische Aufgabengebiete in einem Unternehmen

Einkauf Lieferanten Anfragen Bestellung Terminmanagement	Verkauf Kunden Angebote Rechnungen Terminmanagement	Lager Material Fertigwaren Chargenverwaltung Verfallsüberwachung	Logistik Flottenplanung Kapazitätsplanung Terminmanagement Lieferoptimierung	
Konstruktion Produktplanung Entwicklung Musterbau	Produktion Produktionsplanung Steuerung Einzelfertigung Serienfertigung	Dienstleistung Personalverwaltung Einsatzplanung Abrechnung	Projekt- Management Zeitplanung Ressourcenplanung Überwachung	
Buchhaltung Externes / internes Rechnungswesen Anlagenbuchhaltung G&V / Bilanz	Controlling Kostenrechnung Planung Steuerung Managementinfo	Personal Lohn/Gehalt Personalplanung Zeitmanagement	Management Planung Überwachung Steuerung	Marketing Kunden Produkte Aktionen Veranstaltungen
Archivierung Revisionssichere Langzeitspeicherung von Dokumenten	Dokumenten- Management Aufgabenbezogene Speicherung von Dokumenten	Portale Kunden Lieferanten Mitarbeiter	... Viele weitere Bereiche	

- **Automobilindustrie**
 - Serienfertigung mit Individualisierungen
 - Just-in-time Beschaffung
- **Banken**
 - Management von Finanzprodukten
 - Weltweite Transaktionen
- **Lebensmittelindustrie**
 - Chargenfertigung und Herkunftsnachweise
 - Logistik mit Kühlkette
- **Öffentliche Verwaltung**
 - Überwiegend völlig andere Aufgaben, die es sonst nirgends gibt, z.B. Melderegister, Handelsregister, Grundbuch, etc.



Selbst Unternehmen der gleichen Branche haben im Detail immer unterschiedliche Anforderungen.

Organisatorische Rahmenbedingungen (1)

- Meistens **getrennte Verantwortlichkeiten** der einzelnen Abteilungen, z.B. Vertrieb, Produktion, Logistik, etc.
 - **Häufig unkoordinierte Beschaffung von IT-Systemen**
- Einzelne Funktionsbereiche können auch bei **externen Dienstleistern** liegen, z.B. Personal, Logistik, etc.
 - **Eingeschränkter Einfluss auf externe Rahmenbedingungen**
- Es existieren häufig **keine Gremien**, um übergreifende Anforderungen zu definieren und abzustimmen.
 - **Keine Vereinheitlichung möglich**
 - **Kein zentrales IT-Architekturmanagement möglich**

- Die einzelnen Fachbereiche haben in der Regel unterschiedliche **nicht-funktionale Anforderungen**, u.a. in folgenden Bereichen:
 - Sicherheit (Integrität, Echtheit und Vertraulichkeit von Informationen)
 - Datenschutz (Verarbeitung personenbezogener Daten)
 - Verfügbarkeit (z.B. 7x24-Betrieb oder normale Bürozeiten)
 - Performance (z.B. Realtime-Maschinensteuerung oder Office-Betrieb)

→ **Einheitlicher Betrieb und Schnittstellen sind kompliziert**

- Aufgrund von einer Zusammenlegung von verschiedenen IT-Anwendungen können einzelne Funktionsbereiche auch mehrfach vorhanden sein.

→ **Schwierige Konsolidierung von Anwendungen und Daten**

- **Organisationsformen einer IT-Abteilung**
 - Zentrale Abteilung für alle Organisationseinheiten
 - Komplette oder teilweise Auslagerung (Outsourcing)

- **Hauptaufgaben einer IT-Abteilung**
 - Planung und Betrieb der technischen Infrastruktur
 - Beschaffung von IT-Komponenten
 - Benutzer-Support
 - Bereitstellung von fachlichen und technischen Diensten (Services) für die Abteilungen
 - Entwickeln und Anpassen von Anwendungen



Typischerweise gehört die Organisation von abteilungsübergreifenden Abläufen nicht zu diesen Aufgaben.

- In „gewachsenen“ IT-Landschaften muss die IT-Abteilung einen ganzen „Zoo“ von unterschiedlichen IT-Systemen beherrschen.
- Eine umfassende **Betriebsdokumentation** über die Verflechtung aller Systeme fehlt in aller Regel.
- **Wartungsfenster** sind teilweise schwer zu finden, da hierfür auch Systeme abgeschaltet werden müssten, mit denen andere Systeme kommunizieren.

→ **Was passiert, wenn man ein System herunterfährt?**

- Fehlerhafte **Updates** an einem System können zu Fehlern im Zusammenspiel mit anderen Systemen führen.

→ **Welche Systeme werden von einem Update betroffen sein?**

- **Batch-Prozesse** vertragen sich oft mit dem Online-Betrieb nicht
- Eine sachgerechte **Datensicherung** aller Systeme ist oft schwierig

- Geschäftsprozess
- Standardsoftware vs. Individualsoftware
- Schnittstellen
- Management von Benutzeridentitäten und Berechtigungen (IAM)
- ERP
- EAI

Was ist ein Geschäftsprozess?

„Ein **Geschäftsprozess** (business process) ist eine funktions- und stellenübergreifende Folge von Schritten zur Erreichung eines geplanten Arbeitsergebnisses in einem Unternehmen. Diese Schritte heißen **Geschäftsprozessaktivitäten** (business process activities), kurz **Aktivitäten**.

Der Geschäftsprozess dient direkt oder indirekt zur Erzeugung einer Leistung für einen Kunden oder den Markt.“

Quelle: „Quasar Enterprise – Anwendungslandschaften serviceorientiert gestalten“

„Ein Geschäftsprozess kann Teil eines anderen Geschäftsprozesses sein oder andere Geschäftsprozesse enthalten bzw. diese anstoßen.“

Quelle: Wikipedia

Beispiel: Verkauf von Individualreisen für Geschäftskunden

- Für fast jede Aktivität oder jeden Geschäftsprozess in einem Unternehmen gibt es eigenständige, optimierte Speziallösungen.

Typische Eigenschaften solcher „Speziallösungen“:

- In der Regel nur für einen **Teilausschnitt** aller Geschäftsprozesse eines Unternehmens entwickelt.
- Die Funktionen eines Aufgabenbereichs werden vollständig durch **eine oder mehrere eigenständige Anwendungen** abgebildet.
- Für die Durchführung der Aktivität oder den Geschäftsprozess ist kein **Datenaustausch** mit anderen Systemen erforderlich.
- Es werden **keine Funktionen anderer Systeme** genutzt.

- Standardsoftware
 - Einrichtung auf die speziellen Bedürfnisse durch Parametrisierung
 - Beispiele:
 - Finanzbuchhaltung
 - PPS-System

Anmerkung:

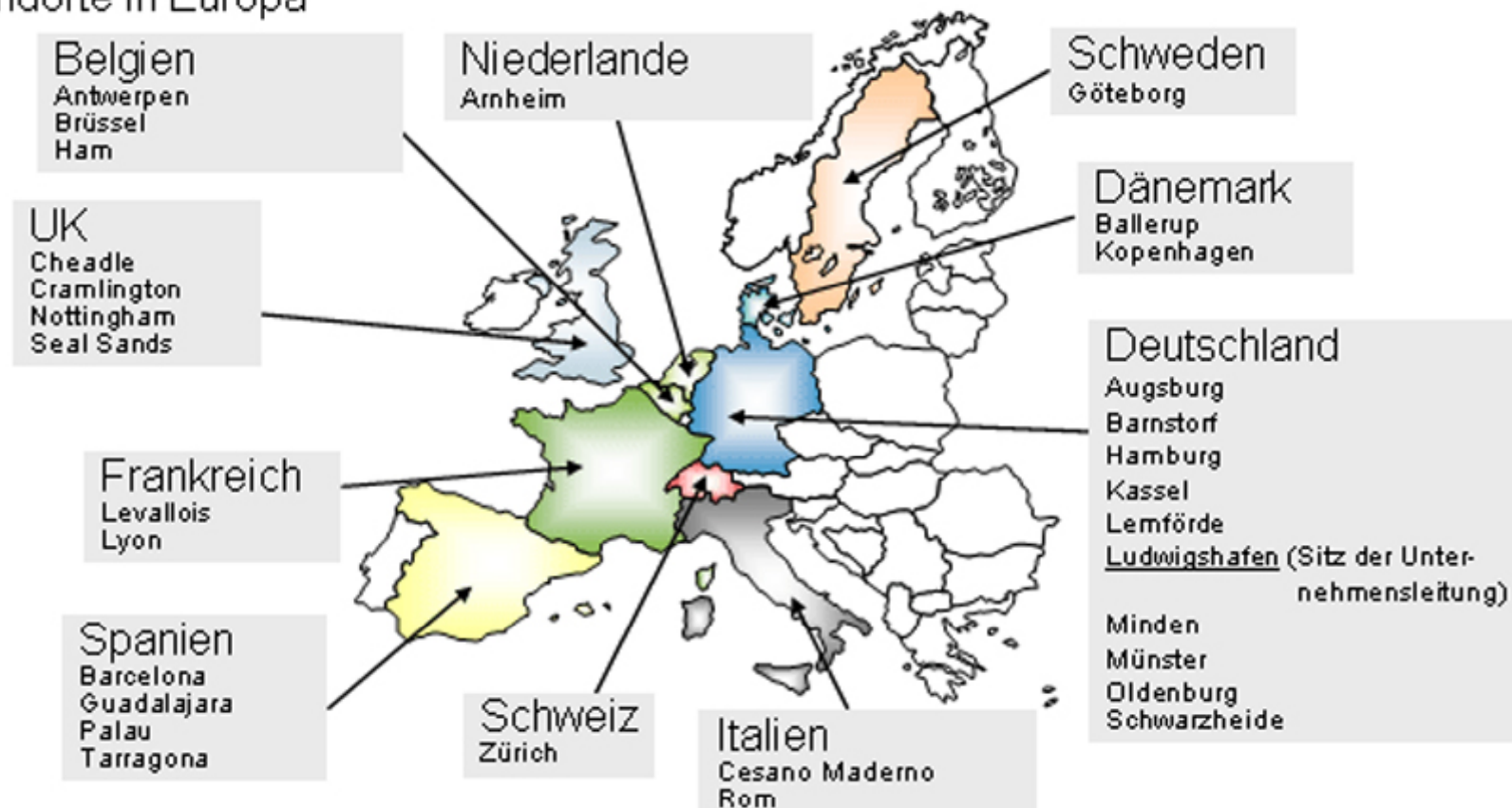
Die beste Standardsoftware für einen Anwendungsbereich wird auch als „Best of Breed“ bezeichnet

- Individualentwicklungen
 - Entwurf und Entwicklung einer Anwendung auf Basis der Anforderungen des Anwenders
- Mischformen
 - Standardsoftware mit individuell „hinzuprogrammierten“ Programmteilen

- Server
 - Großrechner, Mainframes
 - Midrange-Systeme, z.B. AS400
 - UNIX / LINUX
 - Windows
- Clients
 - Fat-Client
 - Terminal-Client
 - Web-Oberflächen
- Netzwerk
 - Ethernet, Token-Ring im LAN
 - Spezielle Maschinensteuerungen, z.B. CAN-Bus
 - Weitverkehrsnetze mit Standleitungen oder Wählverbindungen
- Softwaretechnik
 - Alte Sprachen: COBOL, BS2000-Assembler, BASIC, C, 4GL, ...
 - Moderne Sprachen: C++, Java, ...
 - Architekturen: Monolith, Client/Server- oder Multi-Tier-Architektur

Beispiel: BASF IT-Services Europa

Standorte in Europa



- Daten(austausch)schnittstellen
- Einfache Funktionsaufrufe (API)
- Komplexe Sequenzen von Funktionsaufrufen (→ Protokolle)

- In komplexen Anwendungslandschaften ist der Datenaustausch ein zentrales Thema.

Beispiele

a) Einfacher Datentransfer

- Buchungssätze
- Verkaufszahlen
- Adressdaten

b) Komplexer Datentransfer

- Behandlungsdaten (Gesundheitswesen)
- Strukturierte Objekte (Produktionsauftrag)

■ Übertragungsmethoden

- Manuell (ausdrucken und wieder eingeben)
- File-Transfer (Daten im Quellsystem in eine Datei schreiben und diese im Zielsystem wieder einlesen, manuell oder automatisiert)
- Kommunikationsprotokolle (direkte Kommunikation zwischen Anwendungen oder über eine Zwischenschicht)

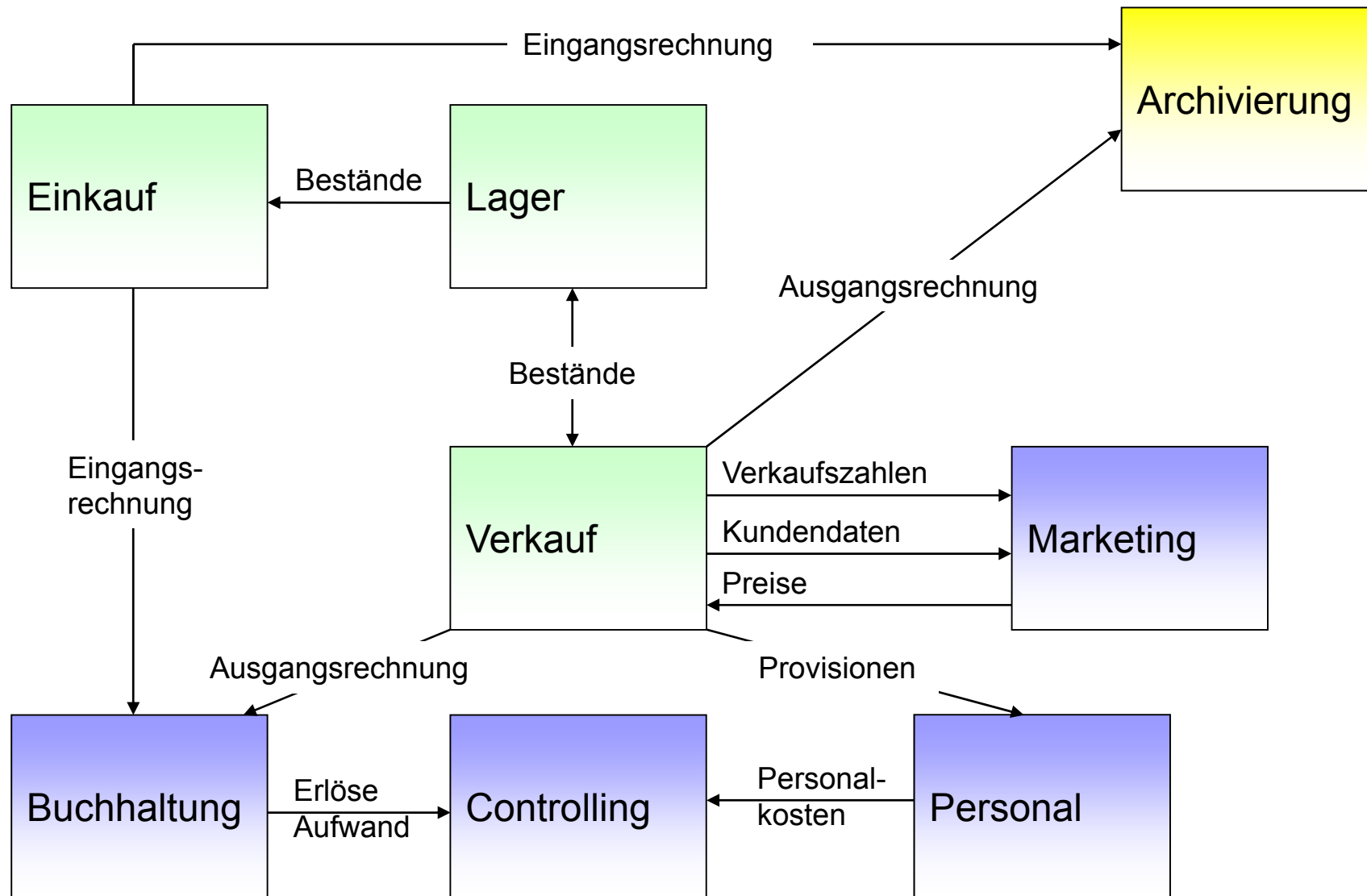
Daten(austausch)schnittstellen (2)

- Typische Eigenschaften von Daten(austausch)schnittstellen
 - Definierter Informationsinhalt
 - Definierter Kommunikationsablauf
 - Transformation von Zeichensätzen
 - Mapping von Dateninhalten

- Physische Ausprägung
 - File
 - Kommunikationsprotokoll

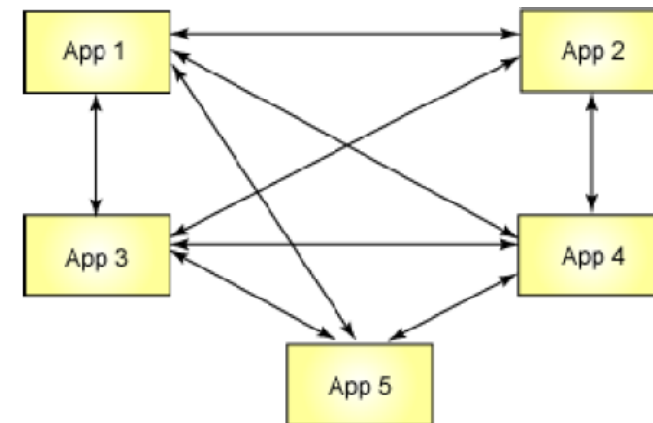
- Zeitlicher Ablauf
 - Synchron
 - Asynchron

Einfacher Datenaustausch



Typische Schnittstellenprobleme

- Punkt-zu-Punkt-Kopplung
- Häufig individuelle, nicht standardisierte Schnittstellen
- Gelegentlich Ad-hoc-Schnittstelle ohne detaillierte Analyse
- Ungenügende Kapselung
- Bei zahlreichen Anwendungen treten viele Schnittstellen auf
- Mit jedem neuen System steigt die Komplexität
- Hohe Fehleranfälligkeit
- Schwer wartbar, da bei Änderungen an einem System evtl. mehrere Systeme adaptiert werden müssen (Release-Festigkeit!)
- Bei Änderung der „Lokation“ eines Systems müssen alle verbundenen Systeme entsprechend angepasst werden
- Pufferung von Daten bei Ausfall eines Systems?



▪ **Spezielle Kommunikationsserver**

Beispiele: eGate (JCAPS), Cloverleaf im Gesundheitswesen

- Datentransformationen
- Einfache Routing-Regeln
- Verbesserte Transparenz

▪ **Komplexe Middleware-Lösungen**

Beispiele: DCE, CORBA, Websphere MQ

- Datentransformationen
- Komplexe Routing-Regeln
- Hohe Transparenz durch Verzeichnisdienst, Pufferung und Adapter
- Verteilte Transaktionen



Es handelt sich überwiegend um sehr komplexe Werkzeuge.

Die Verwendung proprietärer Standards legt den Anwender auf einen Hersteller / ein Produkt fest.

Beispiel: Cloverleaf HL7-Transformation

The screenshot shows a software interface for HL7 transformation. On the left, a tree view shows the input message format (SAP message) with segments like HEA, PAT, ANG, FAL, FKL, WF, and END. In the center, a table lists transformation rules with columns for Operation, Pre, Post, and Source/Destination. The 'COMMENT' rule is highlighted, with a note: 'VVF - Versicherungsverhältnisse'. On the right, a tree view shows the output message format (HL7 message) with segments like MSH, EVN, PID, PD1, NK1, PV1, PV2, DB1, OBX, AL1, DG1, and DRG. Callouts in orange boxes identify these elements: 'Eingabedatenstruktur SAP-Nachricht' (left), 'Transformationsregeln' (center), and 'Ausgabedatenstruktur HL7-Nachricht' (right).

- **Aufruf von Funktionen anderer Systeme**
 - Abfrage von Daten
 - Verifikation / Prüfung von Daten
 - Trigger für das Starten oder Stoppen von Anwendungen

- **Typische Umsetzungen**
 - Remote Procedure Calls (RPC)
 - Remote Method Invocation (RMI)

Anmerkung:

Auch hier ist die Nutzung von Middleware-Schichten möglich, um die (Software-) Verteilung und die technische Basis der aufrufenden und aufgerufenen Systeme zu verbergen.

Beispiel: Flugbuchung in einer Reisebüroanwendung

- Jeder Systemnutzer muss am Computer, im Netzwerk und in den Anwendungen, die er benutzt, **identifiziert** werden können.
- Betriebssysteme stellen Standardmethoden für die **Authentifizierung** von Benutzern zur Verfügung, z.B. UNIX-Berechtigungssystem, Windows Active Directory.
- Anwendungen haben meistens ihre eigene **Benutzerdatenbank**, um Benutzer zu authentisieren.
- Aufgrund der Heterogenität der Systeme ist eine **einheitliche zentrale Identität des Benutzers** nicht gegeben. Ein und derselbe Benutzer muss auf mehreren Systemen eingerichtet und administriert werden.
→ **Mehrere Benutzernamen und Passwörter für jeden Benutzer**
- Idealvorstellung wäre ein **Single Sign On (SSO)**, bei dem sich ein Benutzer nur einmal am System anmelden muss und alle Anwendungen entsprechend seiner jeweiligen Berechtigung nutzen kann.

▪ Access Management

- Die **Verwaltung der Zugriffsrechte** auf Anwendungen oder bestimmte Daten wird als „Access Management“ bezeichnet.
- Jeder authentifizierte Benutzer hat üblicherweise **definierte Rechte**.
- **Berechtigungen** müssen wie die Benutzer selbst in jeder einzelnen Anwendung administriert werden.

→ **Keine zentrale Übersicht über Berechtigungen eines Benutzers**

▪ Tools

- **Verzeichnisdienste** für zentrale Benutzerverwaltung: LDAP & Co.
- Komplette „**Identity Management Suites**“ sollen die zentrale Administration von Identitäten und Zugriffsrechten ermöglichen.



Voraussetzung ist die **Unterstützung durch die Anwendungen** bzw. die Verfügbarkeit von **Adaptern!**

→ Keine vollständig durchgängige Lösung

- Für jede Aufgabe gibt es **eigenständige Lösungen**, die entweder aus historischen Gründen noch vorhanden sind oder aufgrund herausragender Eigenschaften ausgewählt wurden („Best of Breed“).
- Manche **Funktionen und Daten** sind in einzelnen Anwendungen mehrfach und uneinheitlich vorhanden.
- Der Einsatz solcher Lösungen erfordert in der Regel einen **komplexen Datenaustausch**.
- Es gibt zahlreiche **Tools**, die den Datenaustausch unterstützen.
- Es existiert häufig **keine zentrale Steuerung** der Abläufe für einen Geschäftsprozess oder zwischen mehreren Geschäftsprozessen.

Zwischenfrage:

Wer soll für die Koordination der IT-Lösung hinsichtlich der Geschäftsprozesse und Aktivitäten verantwortlich sein?

- Klassisches, „integriertes“ Softwarepaket
- Versuch, zentrale Geschäftsprozesse mit einem einzigen Anwendungssystem abzudecken
- Anpassung auf die speziellen Bedürfnisse des Anwenders
 - a) Änderungen im Programmcode der Software
 - b) Erweiterungen durch zusätzliche Programmierungen
 - c) Einstellen von (Konfigurations-) Parametern
 - d) Eingabe / Pflege von Stammdaten
- Zentralisiertes IAM innerhalb des ERP-Systems

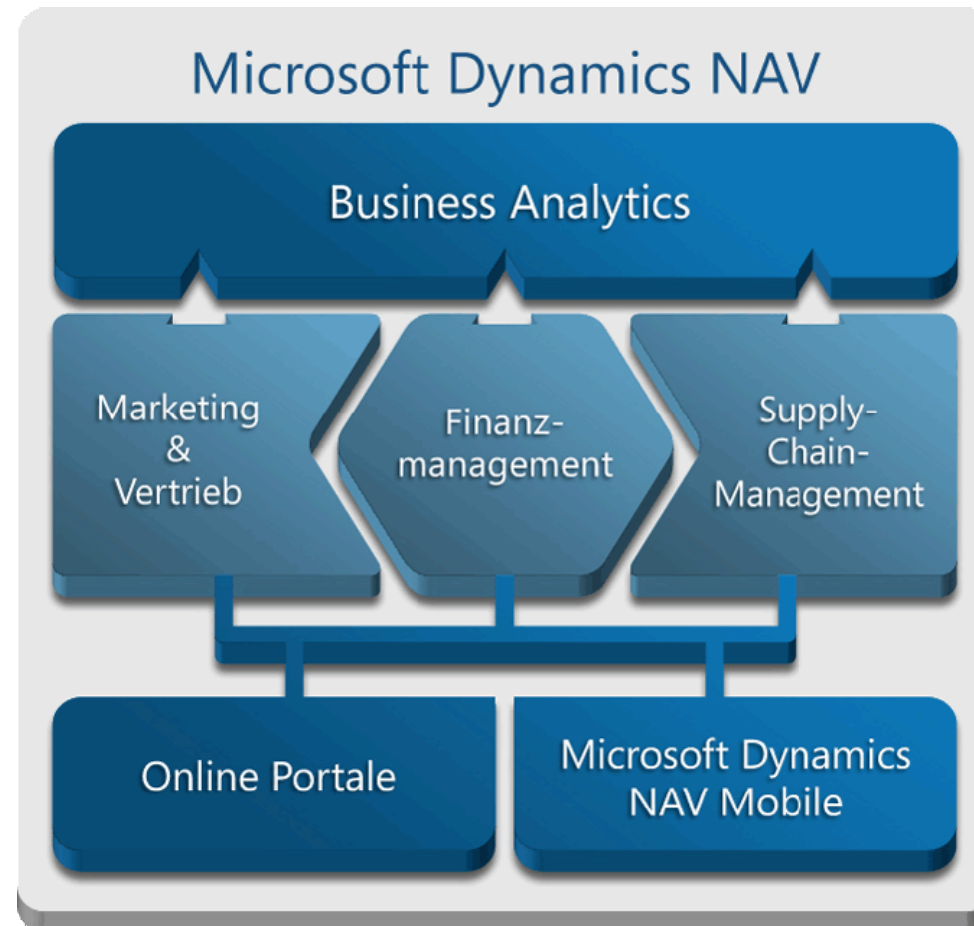
Beispiele:

SAP R/3, Microsoft Dynamics NAV (Navision), Sage, zahlreiche weitere Systeme mit teilweise spezieller Branchenausrichtung (sog. Branchensoftware oder Branchenpakete)

Beispiel: SAP-Anwendungsportfolio

Analytics	Financial Analytics		Operations Analytics		Workforce Analytics		
Financials	Financial SCM	Treasury	Financial Accounting	Management Accounting	Corporate Governance		
Human Capital	Talent Management		Workforce Process Management		Workforce Deployment		
Procurement	Procurement	Inventory Warehousemgmt.	In-/Outbound Logistics		Transportation Management		
Manufacturing	Production Planning	Manufacturing Execution	Product Development		Lifecycle Data-management		
Sales / Service	Sales Order Management		Aftermarket Sales and Service		Professional Service Delivery		
Corporate Serv.	Real Estates	Enterprise Assets	Project Portfolio	Travel Mgmt.	Health Safety	Quality Mgmt.	Global Trade

Beispiel: Microsoft Dynamics NAV



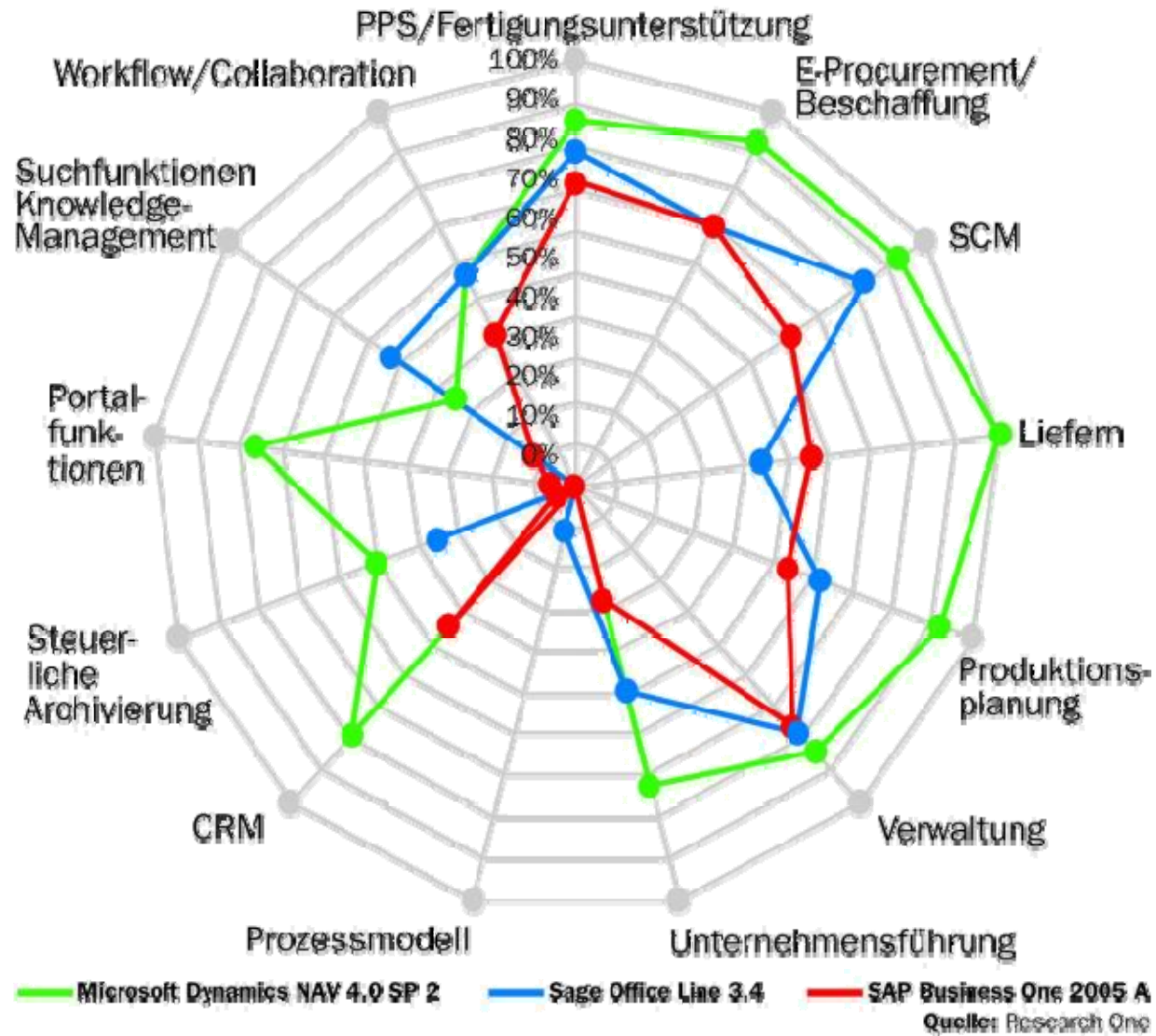
■ Vorteile

- Die Datenintegration vermindert die **Anzahl der Schnittstellen**.
- Die **Zusammenfassung von Funktionen** vereinfacht das System.
- Die Vereinheitlichung der **Benutzeroberfläche** erleichtert die Bedienung.
- Es kann eine **Kostenreduzierung** erzielt werden, wenn das System ausreichende Funktionen bietet.

■ Nachteile

- Viele Funktionen eines Standard ERP-Systems sind für **spezielle Verwendungszwecke** nicht ausreichend.
- Die Möglichkeiten für **individuelle Anpassungen** sind teilweise begrenzt oder sehr teuer.
- Es sind weiterhin Schnittstellen für die **Anbindung von Spezialem Systemen** erforderlich.

Beispiel: ERP-Funktionsabdeckung



- Prozessorientierte Integration einzelner Anwendungssysteme auf Basis standardisierter Schnittstellen (meist proprietär)

Beispiele:

- SAP Netweaver
 - Microsoft BizTalk
 - IBM Websphere Business Integration
- Anwendungsübergreifende Prozesssteuerung häufig gegeben
 - EAI ermöglicht die Integration von Einzelanwendungen und unterstützt so den „Best-of-Breed“-Ansatz sehr gut



Adapter, die für die Kopplung von Anwendungen benötigt werden, sind oft non-standard und auf Dauer schwer wartbar

- In großen IT-Systemen sind zahlreiche Aufgaben zu erledigen, die mit bisherigen Mitteln **beträchtliche Aufwände** auslösen.
- Der Überblick über das gesamte System ist unvollständig; die **Dokumentation** ist lückenhaft und schlecht gepflegt.
- Der **Betrieb** ist häufig sehr aufwendig und kostenintensiv.
- Die **Flexibilität** ist aufgrund zahlreicher Abhängigkeiten eingeschränkt.
- Eine **Wiederverwendung** von Funktionen ist aufgrund der Struktur der IT-Anwendungen praktisch nicht möglich.
- **Änderungen** an Geschäftsprozessen können zu aufwendigen Änderungen in der IT-Landschaft führen.
- Es fehlt an einer globalen **Methodik** und an den passenden **Werkzeugen** zur durchgängigen Abbildung von Geschäftsprozessen.



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 12. Mai 2009

Serviceorientiertes eGovernment

Grundlagen verteilter Systeme

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Was ist ein verteiltes System?

→ Softwaresystem, das aus mehreren Teilsystemen besteht, die mittels prozess- oder rechnerübergreifender Kommunikation zur Erfüllung einer Gesamtfunktionalität beitragen.



Ein verteiltes System kann auch auf einer einzelnen physischen Maschine laufen.

Beispiele:

- SAP R/3
- Jede Internet-Applikation, z.B. eBay
- Datenbanksysteme (z.B. Oracle 11g)
- Lufthansa-Flugbuchungssystem
-

Die Verteilung der „Einzelteile“ eines Softwaresystems kann verschiedene Gründe haben:

- Mehrere Clients nutzen eine **gemeinsame, zentrale Logik**
- Die Anwendung benötigt Funktionalitäten von **unterschiedlichen, bereits existierenden Systemen** (z.B. Nutzung von Web Services)
- Einzelne Komponenten sollen leicht **austauschbar** sein
- Hohe Anforderungen an die **Performance** erzwingen die Verwendung mehrerer Rechner (**Lastverteilung**)
- Hohe **Zuverlässigkeitsanforderungen** erfordern mehrfaches Vorhandensein von (evtl. sogar unterschiedlich implementierten) Systemen.

Beteiligte in verteilten Systemen

- **Anwendungssysteme** mit speziellen Schnittstellen
- **Standardservices**, z.B. Webserver
- **Anwendungsservices**, z.B. Payment-Services

Kopplung von Systemen

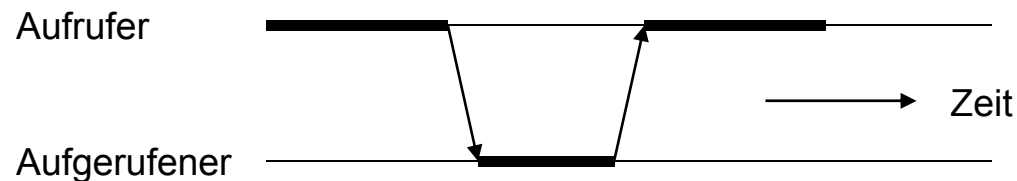
- Weisen miteinander kommunizierende Systeme starke technische und zeitliche Abhängigkeiten auf, spricht man von „**enger Kopplung**“
- Sind die technischen und zeitlichen Abhängigkeiten der Systeme gering, spricht man von „**loser Kopplung**“

Enge Kopplung versus lose Kopplung

	Enge Kopplung	Lose Kopplung
Physische Verbindung	Punkt-zu-Punkt	Über Vermittler
Kommunikationsstil	synchron	asynchron
Datenmodell	komplexe gemeinsame Typen	nur einfache gemeinsame Typen
Typsystem	streng	schwach
Bindung	statisch	dynamisch
Plattformspezifika	stark / viel	schwach / wenig
Interaktionsmuster	über komplexe Objektbäume navigieren	datenzentrierte autonome Nachrichten
Transaktionssicherheit	2PC (Two-Phase-Commit)	Kompensation
Kontrolle fachlicher Logik	zentrale Kontrolle	verteilte Kontrolle
Deployment	gleichzeitig	zu verschiedenen Zeitpunkten
Versionierung	explizite Upgrades	implizite Upgrades

Quelle: Josuttis, SOA in der Praxis, S.48

- Bei der synchronen Kommunikation zwischen zwei Systemen stellt der Aufrufer eine Anfrage und **wartet auf die Antwort**. Er ist dabei so lange **blockiert**, bis die Antwort des Kommunikationspartners eintrifft.

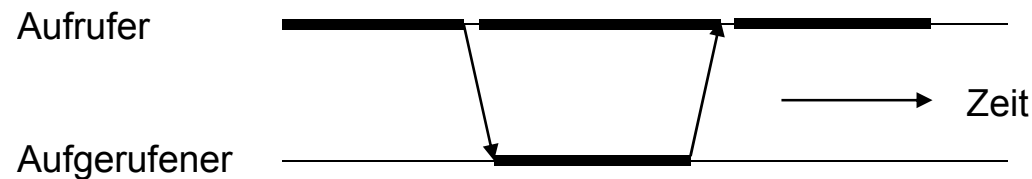


- **Vorteile / Nachteile**

- Der Aufrufer kann einfach gestaltet werden, da das Ergebnis wieder direkt zurück geliefert wird.
- Die Nachricht kann nicht unbemerkt verloren gehen.
- Der Kommunikationspartner muss verfügbar sein und in endlicher Zeit antworten.
- Wenn ein Server mehrere Kommunikationspartner gleichzeitig bedienen soll, muss der Server ausreichend schnell „dispatchen“ können.

Anwendungsbeispiel: Ein einfacher Java-Time-Server

- Bei der asynchronen Kommunikation stellt der Aufrufer eine Anfrage und **fährt mit seiner Verarbeitung fort**, ohne auf die Antwort zu warten. Wenn eine Antwort zurück kommt, muss der Aufrufer im Regelfall seine Arbeit unterbrechen, um die Antwort zu verarbeiten.



- **Vorteile / Nachteile**
 - Der Aufrufer kann mit seiner Arbeit fortfahren, ohne auf den Kommunikationspartner warten zu müssen.
 - Wird ein Ergebnis erwartet, benötigt der Aufrufer einen Event-Mechanismus, der feststellt, ob und wann eine Antwort eintrifft.
 - Die Antwort muss der entsprechenden Anfrage zugeordnet werden.
 - Anfragen / Antworten könnten verloren gehen

Anwendungsbeispiel: Initiierung von Jobs unbekannter Abarbeitungszeit

- Idealerweise verwenden miteinander kommunizierende Systeme die **gleichen Datentypen**, praktisch tun sie das aber nur selten.
- Alle Systeme müssen die gleichen **fundamentalen** Datentypen verwenden, z.B. *String*, *Integer*, *etc.*
- **Komplexe Datentypen**, z.B. „*Kunde*“ müssen nicht in allen Systemen identisch sein!
 - Mapping von Daten ist praktisch immer erforderlich

Ohne Vermittler

Es muss eine Punkt-zu-Punkt-Verbindung genutzt werden, bei der der Aufrufer die genaue physische Adresse des Aufgerufenen kennt.

→ Enge Kopplung

Mit Vermittler

Zwei alternative Vermittlungsmethoden

1. Der Aufrufer fragt anhand eines symbolischen Namens bei einem **Name Server** oder **Broker** nach, wo er den Empfänger der Nachricht finden kann (z.B. DNS). Der echte Aufruf erfolgt dann wieder „**Punkt-zu-Punkt**“ an die zurückgelieferte Adresse.
2. Der Aufrufer übergibt die Nachricht mit der symbolischen Adresse des Kommunikationspartners an eine **Kommunikationsinfrastruktur** (Netzwerk, Middleware, etc.), die die symbolische Adresse in eine physische Adresse umwandelt.

- In einem verteilten System kann es erforderlich sein, dass mehrere Aktionen auf verschiedenen Systeme **entweder alle zusammen gültig werden oder keine davon**.
- Übliche Verfahren in verteilten Systemen sind **2PC** (Two-Phase-Commit) und **Kompensation**.
- Grundlegende Anforderung ist das ACID-Prinzip (**atomicity, consistency, isolation und durability)**

Atomicity:

Die Teilschritte einer Transaktion werden ganz oder gar nicht ausgeführt („alles oder nichts“).

Consistency:

Eine Transaktion führt von einem konsistenten Zustand in einen anderen.

Isolation:

Typischerweise muss ein „Isolation Level“ angegeben werden, der angibt, welche Daten wann für andere Transaktionen sichtbar sind.

Durability:

Sobald eine Transaktion beendet ist, ist garantiert, dass auch beim Auftreten von Fehlern (System, Hardware) das Ergebnis persistent (erhalten) bleibt.

2PC (Two-Phase-Commit) [Folie 1]

1. Für die Durchführung der Transaktion erhalten alle benötigten Ressourcen zunächst ein „prepare to commit“ von einem sog. „Transaction Manager“ (TM)
2. Alle beteiligten Systeme bereiten dann die jeweilige lokale Transaktion vor und **blockieren** die Ressource.
3. Die Beteiligten antworten dem TM mit „agreed“, wenn die Vorbereitung der Transaktion gelungen ist oder mit „abort“, wenn Fehler aufgetreten sind.
4. Sind alle „agreed“, sendet der TM „commit“ an die Ressourcen für die Festschreibung der Transaktion.
5. Tritt in der query-Phase ein einziges „abort“ auf, sendet der TM ein „rollback“ an alle Ressourcen.

2PC (Two-Phase-Commit) [Folie 2]

Vorteile

- Standardisierter Automatismus
- ACID wird gewährleistet

Nachteile

- Synchrones Protokoll
 - Enge Kopplung
- Die Transaktion kann erst durchgeführt werden, wenn alle Ressourcen verfügbar sind.
 - Es kann unter Umständen zu Deadlocks kommen
- Evtl. unterstützen nicht alle beteiligten Systeme dieses Verfahren
 - 2PC kann nicht immer eingesetzt werden

Beispiel

- Die erste Einzeltransaktion gelingt, z.B. eine Flugbuchung
- Die zweite Einzeltransaktion schlägt fehl, z.B. die Hotelbuchung
→ Die erste Transaktion muss wieder rückgängig gemacht werden

Vorgehensweise

- Die Transaktion wird nicht durch einen Rollback rückgängig gemacht; stattdessen wird eine **inverse Transaktion** durchgeführt, die den Vorgang inhaltlich wieder korrigiert.

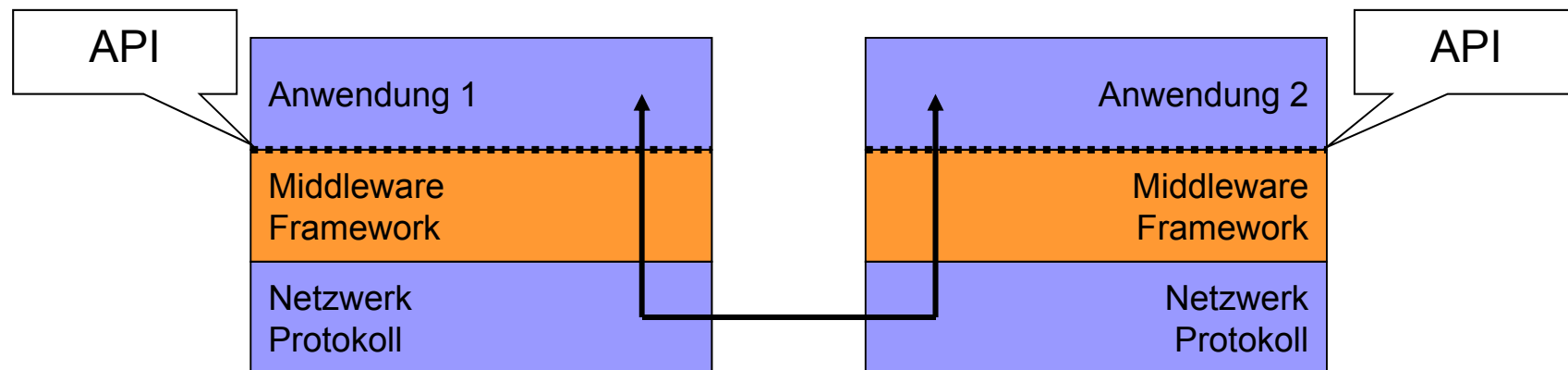
Vorteile

- Es müssen nicht alle beteiligten Systeme synchron arbeiten.
→ Lose Kopplung möglich / typisch

Nachteile

- Die inverse Transaktion muss **explizit implementiert** werden.
- Der Ablauf muss in der Anwendung ausprogrammiert werden.
- Es werden zusätzliche Daten erzeugt.

Ein Middleware-Framework bietet eine Umgebung, die es Anwendungen ermöglicht, Verbindungen aufzubauen und Daten auszutauschen.



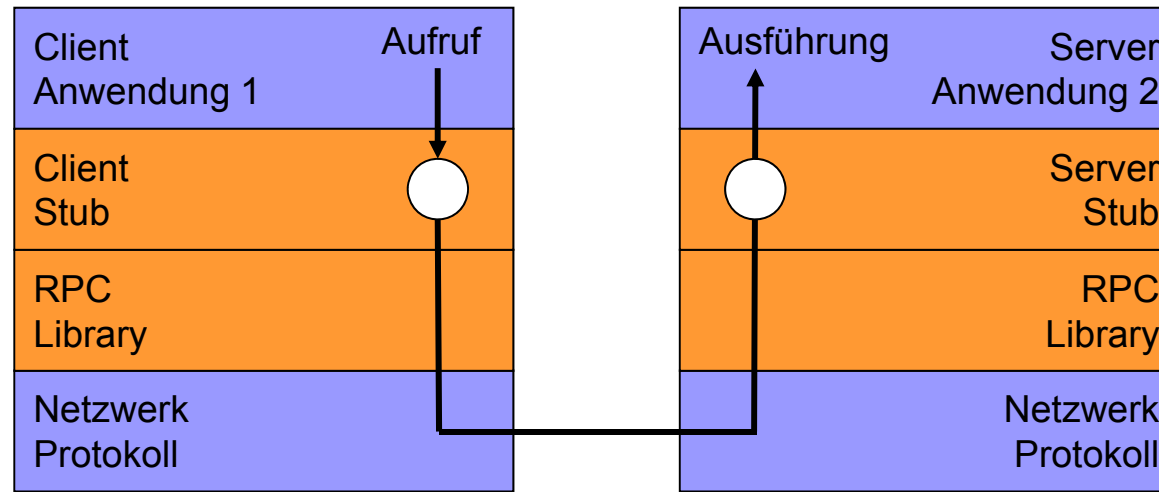
Die Middleware verbirgt technische Details:

- Betriebssystem
- Netzwerkprotokolle
- Details des Verschlüsselungsverfahrens
- ...

Remote Procedure Call (RPC)

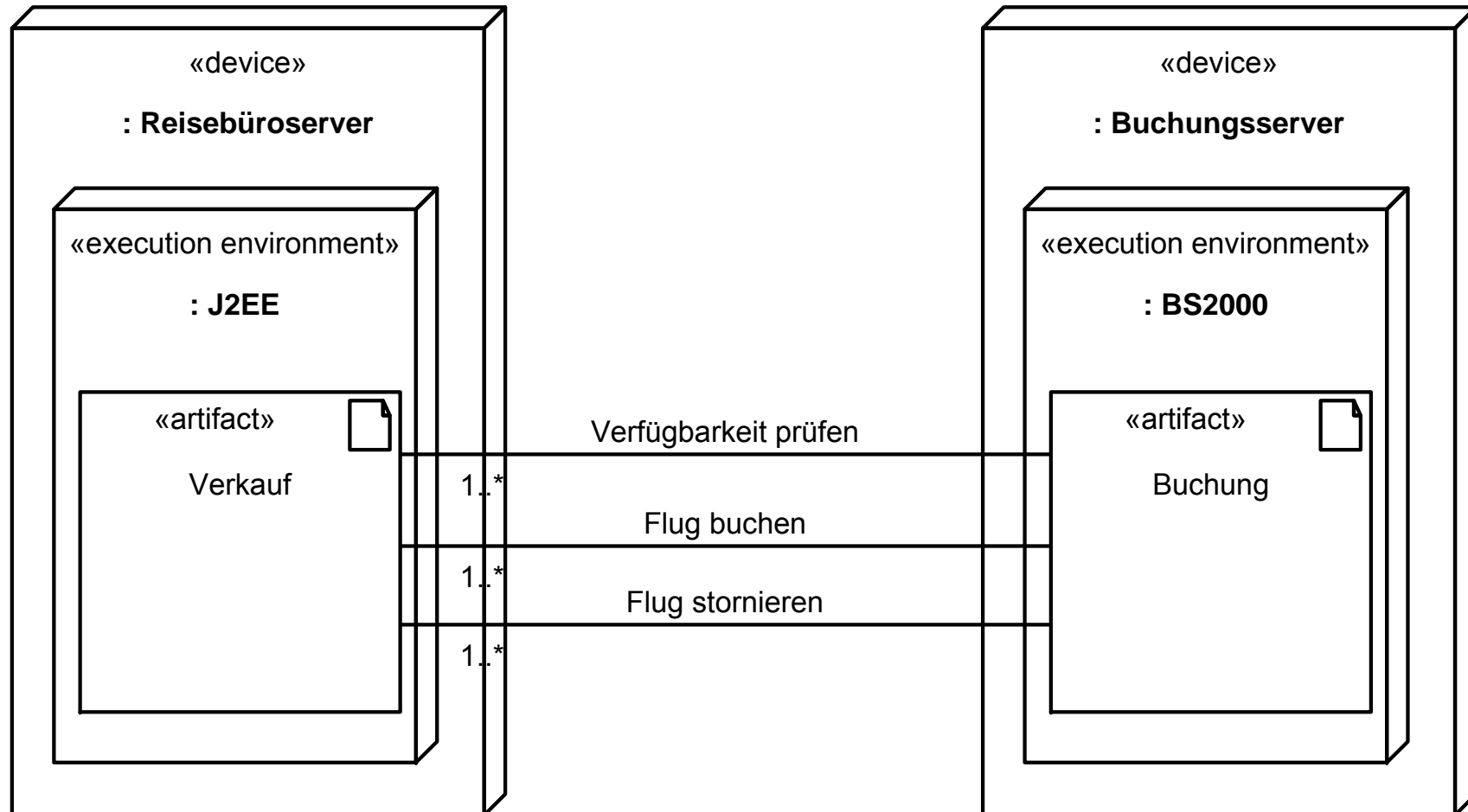
- Auf vielen **Plattformen** verfügbar
- Ursprünglich von der Fa. SUN entwickelt (SUN-RPC)
- Geringe Unterstützung für die Übertragung von **Parametern**
- Verschiedene Varianten von RPCs, z.B. DCE-RPC
- **Kein objekt-orientierter Ansatz**
- Grundsätzlich **synchrone** Kommunikation,
bei einigen Varianten aber auch asynchrone Kommunikation möglich.

RPC-Ablaufschema

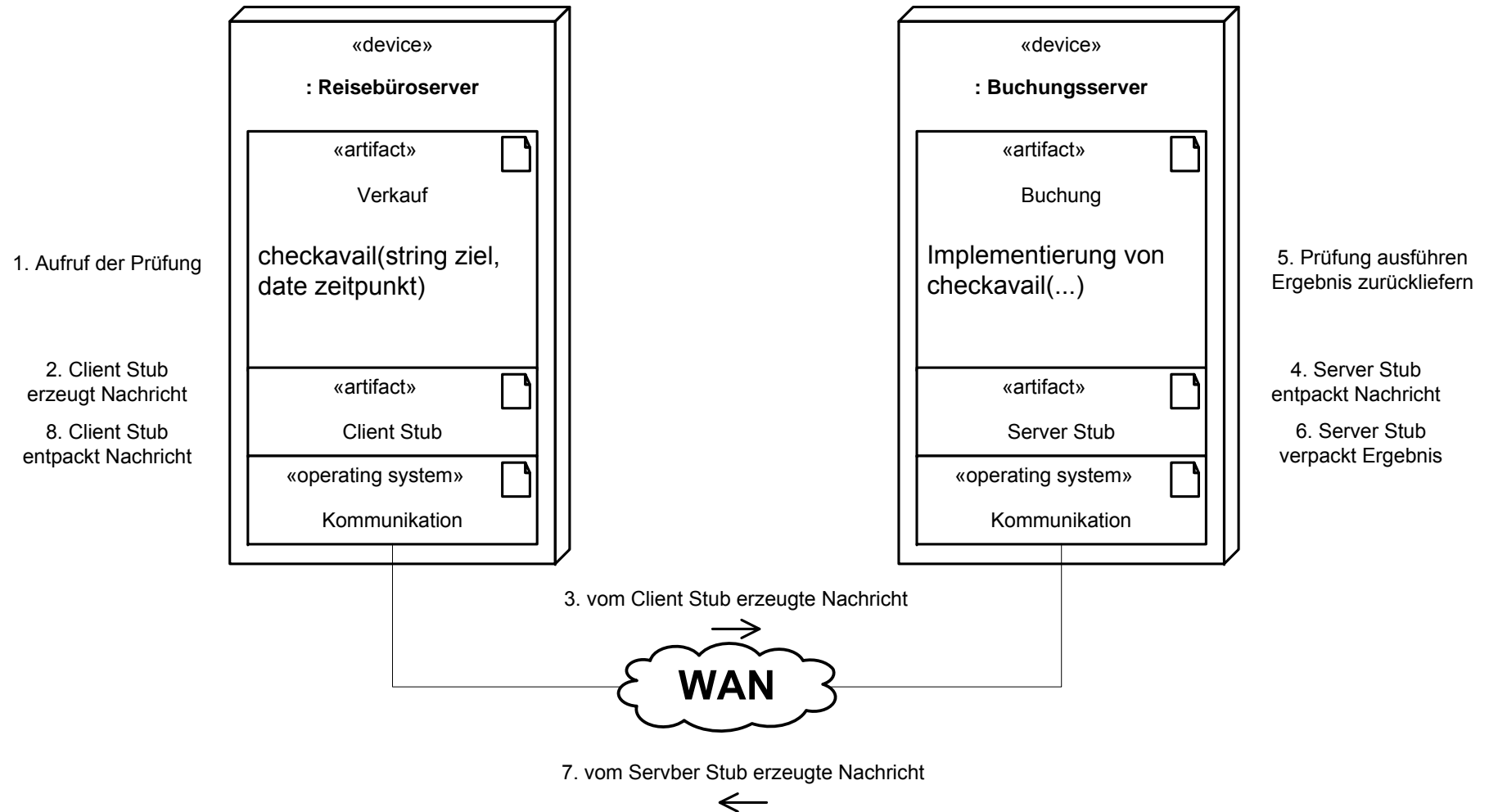


- Funktionsaufruf sieht für den Client wie ein lokaler Aufruf aus
- Der Aufruf wird zum Server geroutet, dort ausgeführt, und dann wird das Ergebnis zurück geliefert.
- Die Stubs kapseln die technischen Eigenschaften der Übertragung gegenüber den Anwendungen.
- Verwendung eines Request/Reply-Schemas:
Der Aufrufer wird so lange blockiert, bis das Ergebnis vorliegt

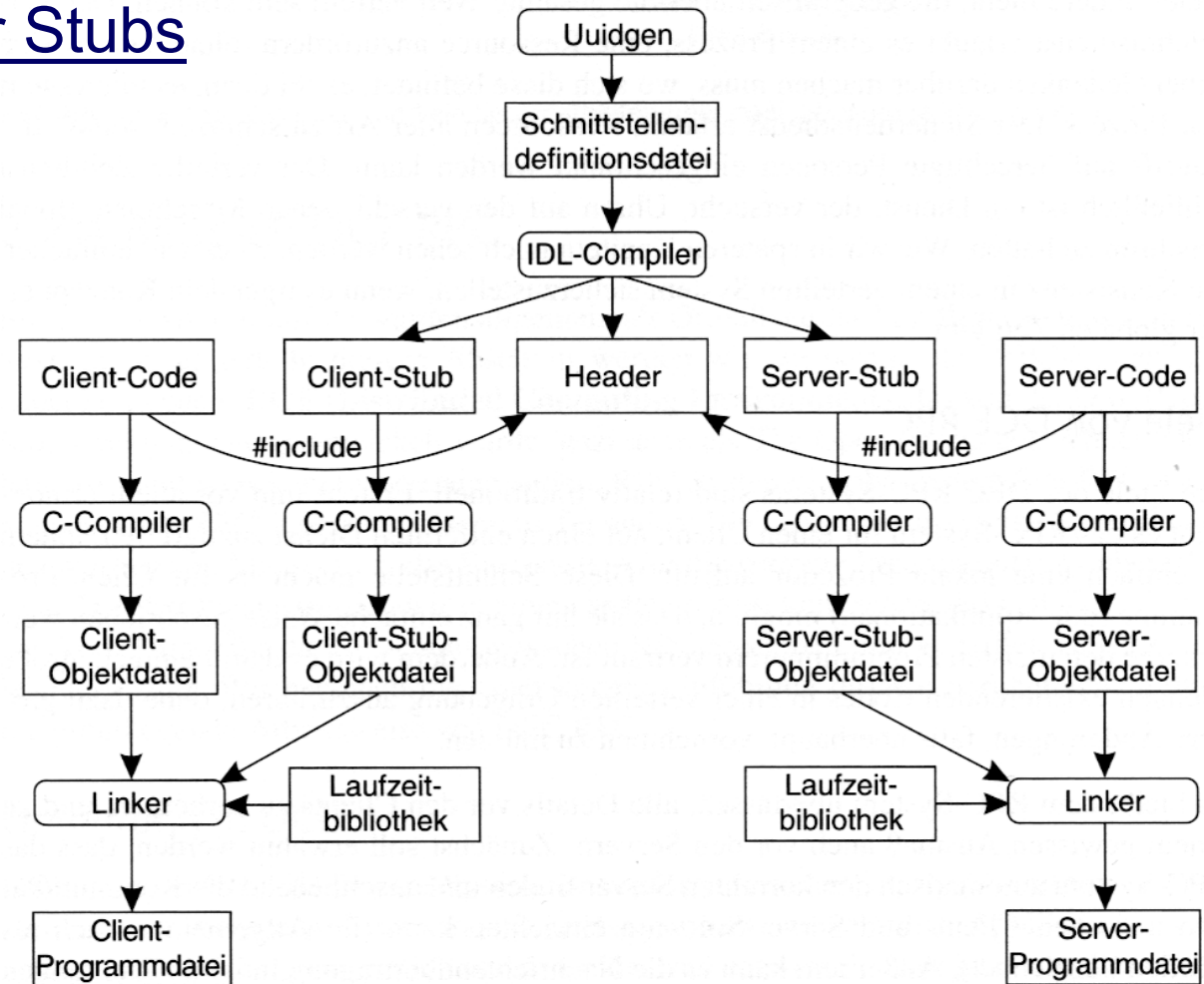
Beispiel: Flugbuchung



Beispiel: Flugbuchung mit RPC



Erzeugung der Stubs

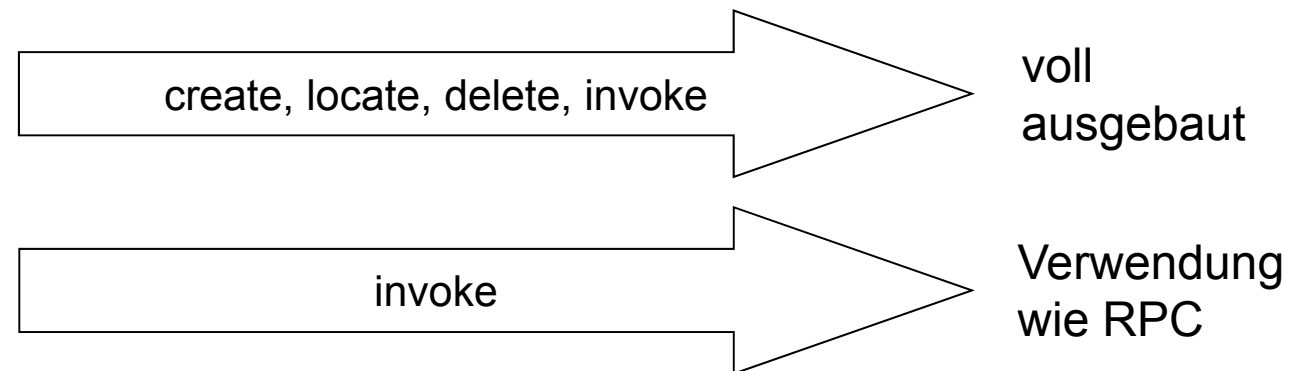
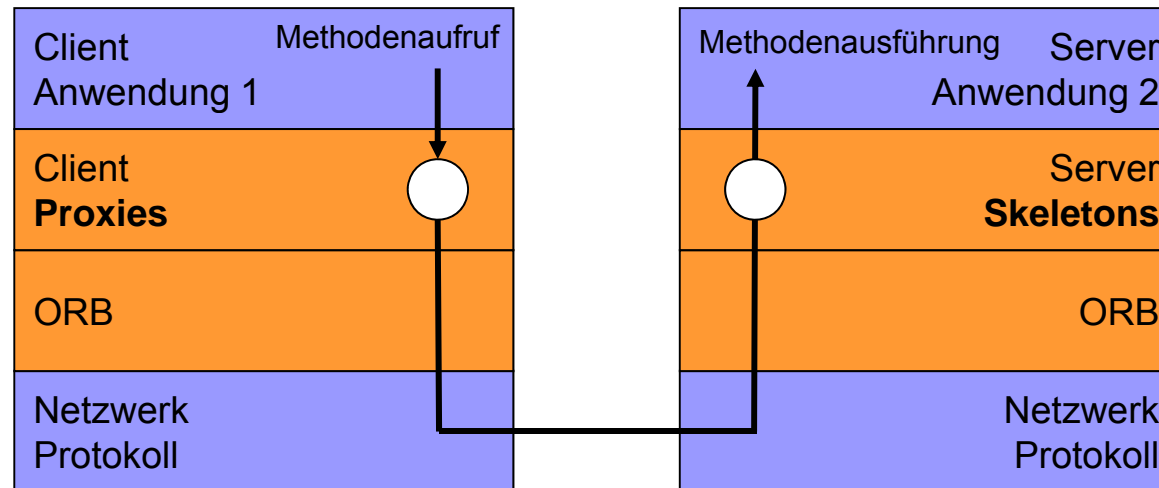


[Grafik: Tannenbaum, Verteilte Systeme]

- Seit der Verfügbarkeit objektorientierter Sprachen werden **objektorientierte Programmierparadigmen** auch auf verteilte Systeme angewendet.
- Kommunikation und Datenaustausch zwischen verteilten Objekten wird typischerweise durch einen **Object Request Broker** (ORB) gesteuert.

Der ORB unterstützt Erzeugung, Lokation, Aufruf und Löschung von verteilten Objekten.
- Weitverbreitete Implementierungen:
 - COM/DCOM (Microsoft)
 - RMI (Java)
 - CORBA (plattformunabhängig)
- Wie in objektorientierten Anwendungen üblich, werden viele Nachrichten zwischen Objekten ausgetauscht, die aber (meist) nur geringen Umfang haben.

Ablaufschema für verteilte Objekte



Schlüsselkonzepte

- Message
- Queue

Message

- Header (technische Informationen und Adressierung)
- „Payload“ (Anwendungsdaten)

Queue

- Speichert und verteilt Messages
- Entkopplung von Sender und Empfänger

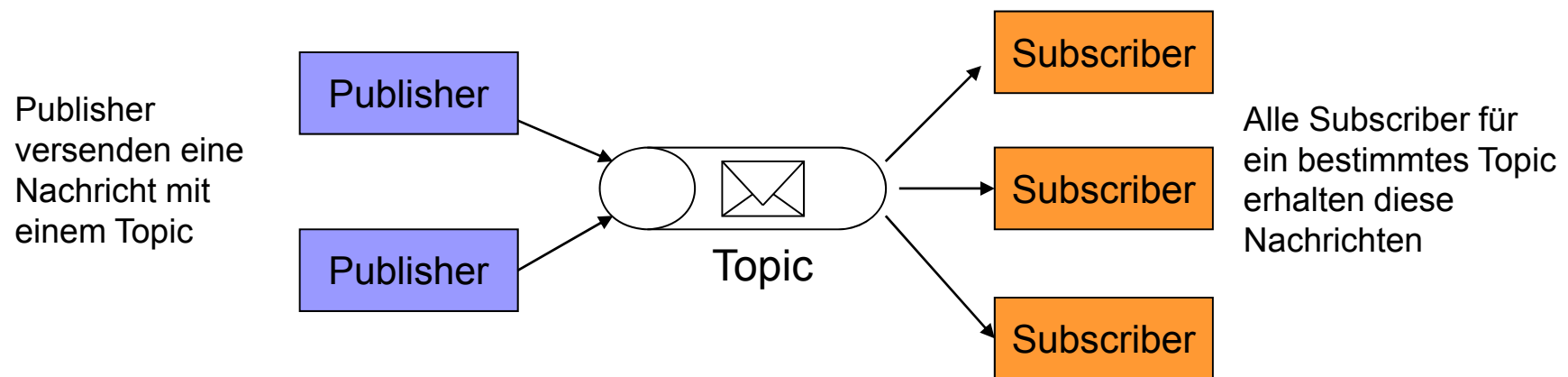
- E-Mail-Server ...
 - entkoppeln Sender und Empfänger
 - speichern die Nachricht, bis sie ausgeliefert bzw. abgeholt werden kann (Persistenz!)
- Der E-Mail-Header enthält Adressierungsinformationen (<from>, <to>, <cc>), die das Routing der Nachricht ermöglichen
- Der E-Mail-Body enthält die eigentliche Information der Nachricht.

Verbindung von Kommunikationspartnern

- one-to-one (1:1)
- one-to-many (1:n)
- many-to-many (m:n)

Technische Konzepte

- Point-to-Point (1:1)
 - Ein Sender ist mit einem Empfänger über eine Queue verbunden
- Publish / subscribe (1:n und m:n)



Queue Management

- Mehrere physikalische **Queues** können zu einer logischen Queue zusammengefasst werden.
- Queues können **vernetzt** werden.
- Intelligente **Routing-Regeln** können Nachrichten verteilen
→ Lastverteilung und Ausfallsicherheit

Service Level (QoS - Quality of Service)

Es können definierte Leistungsmerkmale einer Queue bestimmt werden, z.B.:

- **Priorität** einer Nachricht im Vergleich zu anderen Nachrichten
- **Transaktionsfähigkeit**
- Anzahl der erlaubten **Empfänger**
- **Gültigkeitsdauer** einer Nachricht
- Anzahl der wiederholten **Auslieferungsversuche**

Beispiele:

- jBoss
- IBM Websphere
- Microsoft IIS

Grundfunktionen

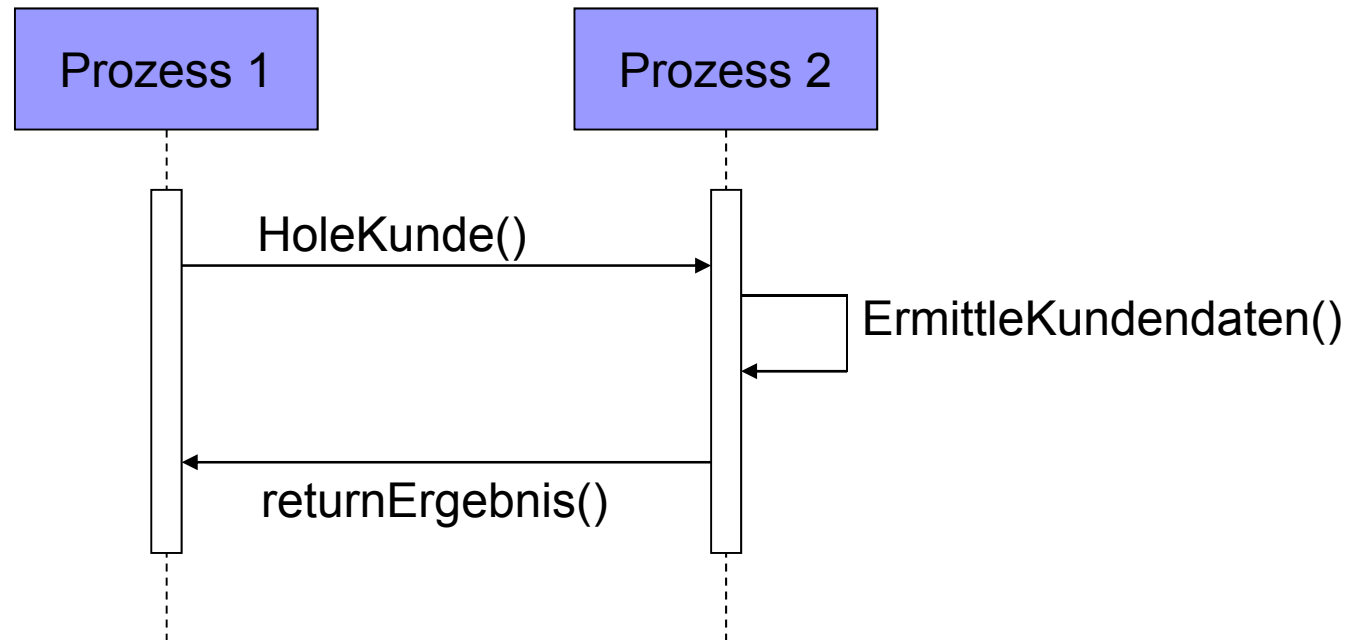
- „Hosting“ und Ausführung von Komponenten
- Datenbankverbindungen verwalten
- Unterstützung verschiedener Typen von Benutzerschnittstellen wie Web-Schnittstellen oder Fat Clients

Erweiterte Eigenschaften

- Load-Balancing
- Caching
- Transaktionsmanagement
- Sicherheitsfunktionen

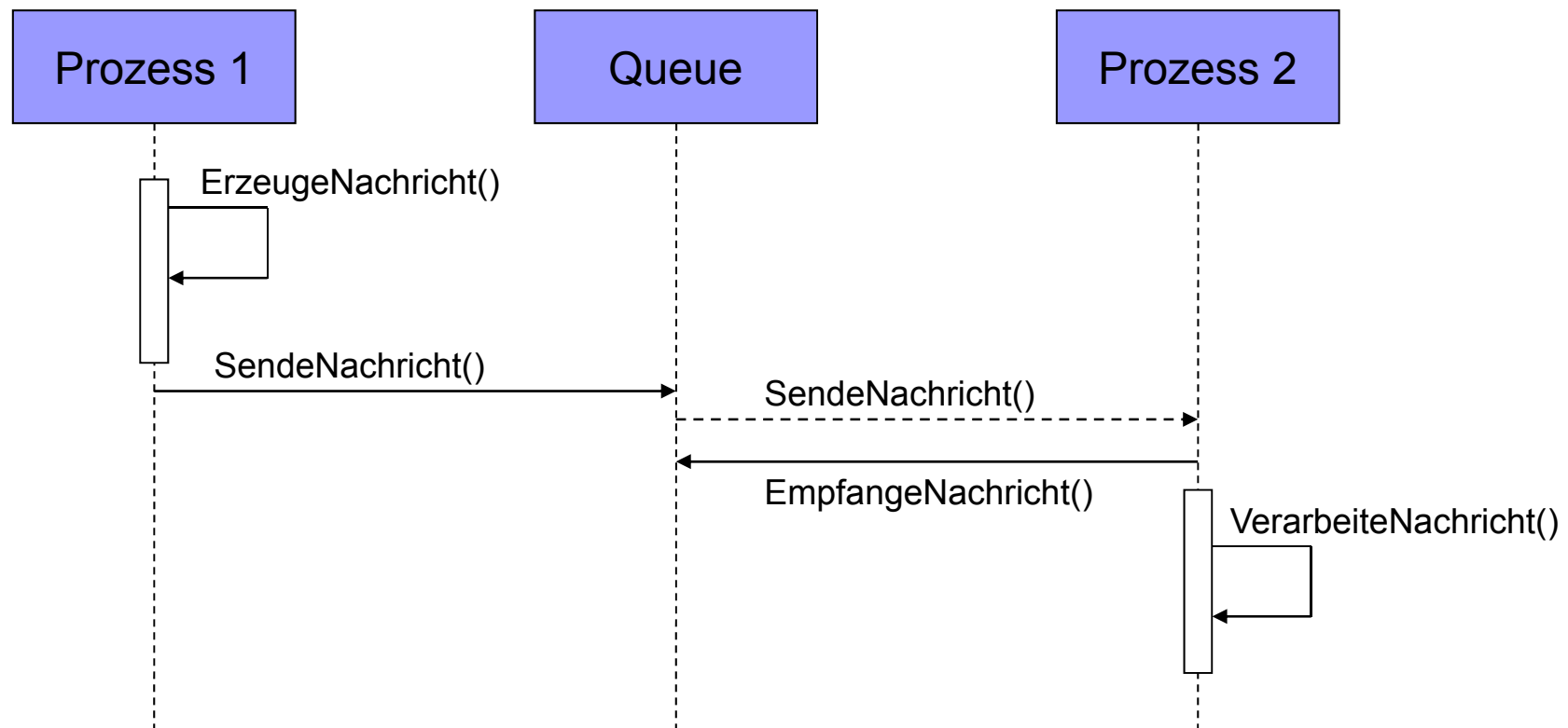
Schnittstellensemantik

→ Es werden Funktionen verwendet, deren Namen selbsterklärend sind und deren Verbindung zur server-seitigen Funktion festgelegt ist (RPC-Style).



Inhaltssemantik

- Der Funktionsaufruf erfolgt über eine neutrale Funktion.
Die Entscheidung, welche Funktion auf dem Server auszuführen ist, ist **abhängig vom Inhalt der Nachricht**.



- Schnittstellensemantik ist für den Programmierer **intuitiv verständlich**.
- Bei einer Änderung an einer derartigen Schnittstelle müssen alle Systeme **geändert** werden, die diese Schnittstelle verwenden.
→ Enge Kopplung
- Bei Verwendung von **Inhaltssemantik** ist es möglich, neue Nachrichten zu definieren, ohne die bestehenden zu beeinträchtigen.
- Die Inhaltssemantik erlaubt nur **schwache Typprüfung**.
→ Lose Kopplung



Schnittstellensemantik lässt sich auch mit MOM abbilden.
Inhaltssemantik lässt sich auch mit RPCs abbilden.

- Mit Hilfe von selbstbeschreibenden Datenstrukturen wie XML können in einer Nachricht der Name der aufgerufenen Funktion und die zugehörigen Daten strukturiert übergeben werden.
- Die Datenstrukturen können flexibel erweitert werden, ohne dass sich die Schnittstelle selbst ändert.
- Sowohl der Aufruf, als auch die Antwort werden in Form des strukturierten Dokuments übermittelt.

Beispielimplementierung: SOAP

Beispiel: Flugbuchung

```
<SOAP-ENV:Envelope xmlns:SOAP-ENC="http://schemas.xmlsoap.org/
soap/encoding/" xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Body>
    <ns1:bookFlight xmlns:ns1="http://www.openuri.org/">
      <ns1:inbound>
        <ns1:flightNumber>LH400</ns1:flightNumber>
        <ns1:flightDate>2003-11-08</ns1:flightDate>
        <ns1:isConfirmed>false</ns1:isConfirmed>
      </ns1:inbound>
      <ns1:outbound>
        <ns1:flightNumber>LH401</ns1:flightNumber>
        <ns1:flightDate>2003-11-17</ns1:flightDate>
        <ns1:isConfirmed>false</ns1:isConfirmed>
      </ns1:outbound>
      <ns1:passenger>
        <ns1:Passenger>
          <ns1:firstName>Karl</ns1:firstName>
          <ns1:lastName>Banke</ns1:lastName>
          <ns1:birthday>1970-08-05</ns1:birthday>
        </ns1:Passenger>
      </ns1:passenger>
    </ns1:bookFlight>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Beispiel aus Krafzig, Enterprise SOA



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 19.5.2009

Serviceorientiertes eGovernment

Zielsetzung der Serviceorientierung

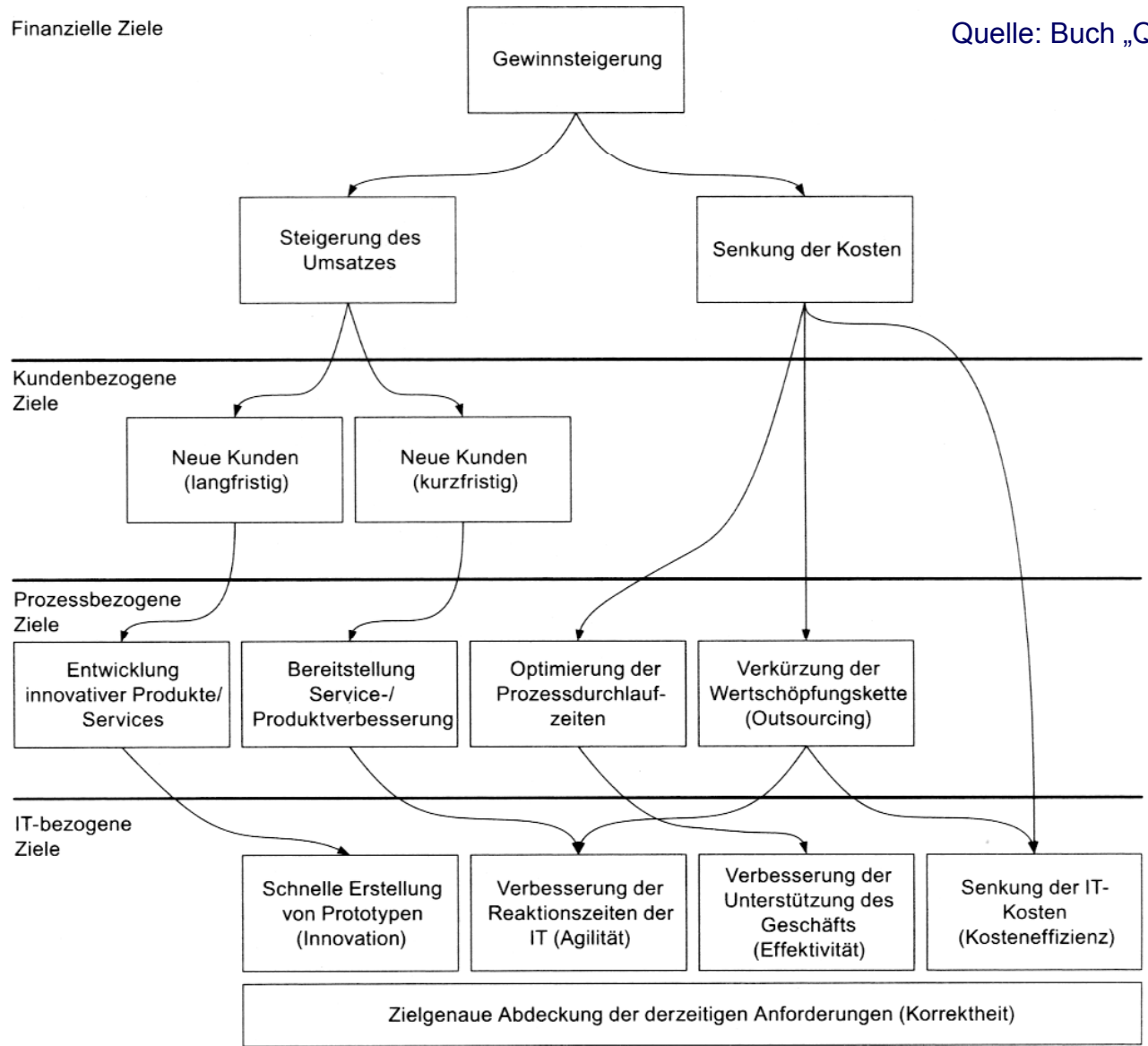
Dr. Frank Sarre

Lehrbeauftragter der LMU München

Ziele auf unterschiedlichen Ebenen

- Finanzielle Ziele
- Kundenbezogene Ziele
- Prozessbezogene Ziele
- IT-bezogene Ziele

Beispielhafte Gestaltungsziele



Korrektheit =

Die Anwendungslandschaft erfüllt die funktionalen und nicht-funktionalen Anforderungen des **gesamten** Geschäfts

Beispielhafte strategische Ansätze

- Synchronisation der in den **einzelnen** Geschäftsprozessen verwalteten Informationen
 - Vermeidung von Dateninkonsistenzen
- Verbesserung der Integration von Anwendungssystemen
 - Vermeidung von Dateninkonsistenzen, die durch manuelle Übertragung entstehen

Kosteneffizienz =

Korrektheit wird durch möglichst geringe Kosten erreicht

Beispielhafte strategische Ansätze

- Outsourcing
 - Reduzierung der Strukturkosten
- Einsatz von Standardprodukten
 - Reduzierung von Entwicklungskosten

Effektivität =

Bestmögliche Unterstützung der Geschäftsprozesse

Beispielhafte strategische Ansätze

- Höhere Automatisierung
- Modifikation von Prozessschritten
 - Reduzierung von Personalkosten
 - Reduzierung von Lagerkosten
 - Vermeidung von Fehlern

Agilität =

Bei Veränderungen im Geschäft schnell und flexibel anpassbar

Beispielhafte strategische Ansätze

- Verwendung technischer Schnittstellenstandards
 - Bessere Austauschbarkeit von Anwendungssystemen
 - Bessere Integration neuer Anwendungssysteme
- Verwendung fachlicher Schnittstellenstandards
 - Reduzierung von fachlicher Komplexität
 - Chance, einen höheren Integrationsgrad zu erreichen

Innovation =

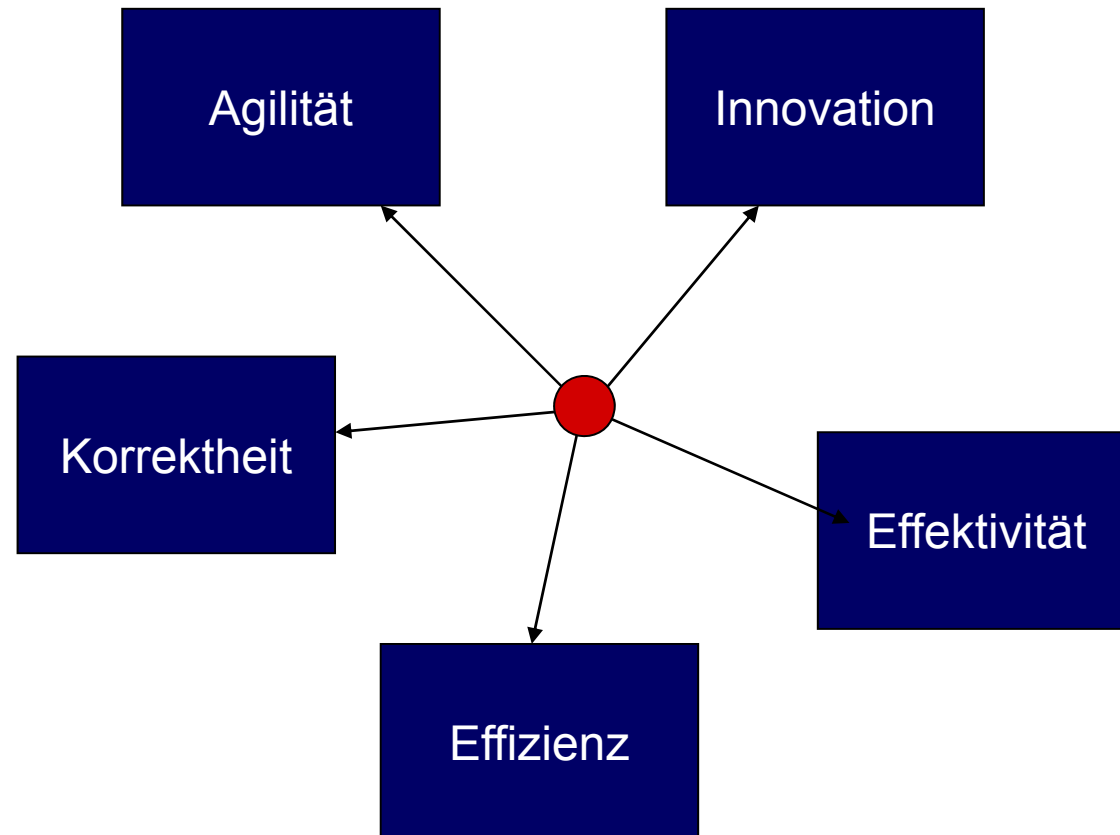
Unterstützung bei der Entwicklung neuer Geschäftsmodelle

Strategische Ansätze

- Nur individuelles Vorgehen
- Starke Abhängigkeit vom Geschäft
(Hightech, Dienstleistungen, Handel, ...)
 - Größtmögliche Freiheit bei der Erschließung neuer Geschäftsfelder

Beispiel: Internet-Shop für eine Kaufhauskette

Die Gestaltungsziele hängen untereinander ab



Unterstützungsprozesse

Beispiel: Finanz- und Personalwesen

Primärziel: Kosteneffizienz

- Standardisierung
- COTS-Produkte („Commercial Off-the-Shelf“)
- Kandidaten für Outsourcing

Abwicklung des Kerngeschäfts

Beispiel: Verkauf von Pauschalreisen

Primärziel: Effektivität und Korrektheit

- Unmittelbare Wirkung auf die Unternehmensergebnisse
- Starker Einfluss auf die Individualität des Unternehmens

Bereitstellung von strategischen Dienstleistungen und Produkten

Beispiel: Support-Hotline

Primärziel: Agilität

- Unterstützung der Kernprozesse
- Dynamische Reaktion auf aktuelle Gegebenheiten

Erarbeitung und Erprobung von Innovationen

Beispiel: Forschung- und Entwicklung

Primärziele: Innovation, Agilität

- Schaffung neuer Produkte
- Geringe Integration in die Anwendungslandschaft

Finanzielle Ziele	Kostensenkung Direkte Verrechnung von Dienstleistungen
Kundenbezogene Ziele	Mehr Dienstleistungen anbieten Dienstleistungen integrieren („die Daten sollen laufen, nicht der Bürger“) Verbesserung der Servicequalität
Prozessbezogene Ziele	Optimierung der Prozessdurchlaufzeiten Integration von Einzelprozessen
IT-bezogene Ziele	Korrektheit (primär) Effizienz (primär) Effektivität (sekundär) Agilität (sekundär)

Evolution von Anwendungslandschaften

Die Evolution vom Ist-Zustand zu einem Idealzustand erfolgt in mehreren Stufen.

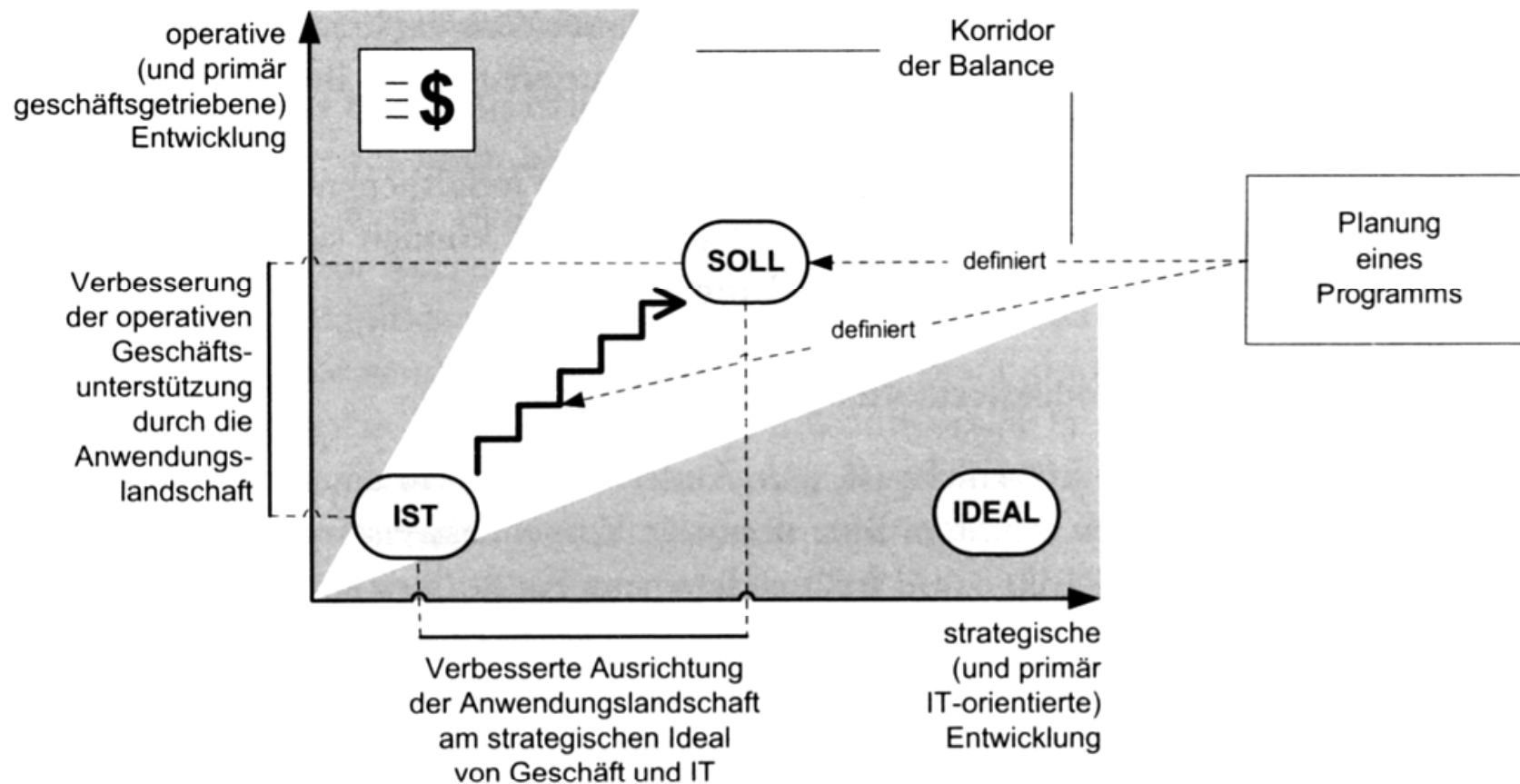


Bild aus „Quasar Enterprise“

Herstellerunabhängigkeit durch Verwendung von Standards

- Erhöhung der Lebensdauer
- Investitionsschutz
- Austauschbarkeit von Produkten
- Fördert Best-of-Breed-Lösungen



Verwendung von Standards kann kurzfristig auch höheren Aufwand als bei der Verwendung herstellerspezifischer Lösungen bedeuten.

- Abwägung gegen die langfristigen Gestaltungsziele!

- Strukturierung der Geschäftsprozesse und Abbildung von Einzelaktivitäten auf die IT
- Identifikation von gemeinsam genutzten Teilprozessen und daraus resultierender IT-Services
- Wiederverwendung von IT-Services
→ Effizienz und Agilität



Aufwand am Anfang eines SOA-Projekts höher als bei klassischen Projekten!

Der Nutzen muss durch langfristige Effizienzsteigerung gewährleistet werden.

Ziel: Mit wachsendem Geschäftsbetrieb soll auch die IT dynamisch mitwachsen können.

Grobe Definition der Skalierbarkeit:

„Skalierbarkeit“ definiert das Verhalten von Systemen bzgl. des Ressourcenbedarfs und der darauf bezogenen Performance bei wachsenden Eingabemengen

Beispiele für Performance-Parameter

- Netzwerkkapazität
- Durchsatz
- Prozessorauslastung
- Antwortzeitverhalten

Ein System skaliert gut,

wenn die mittlere Leistung bei steigender Belastung aufrecht erhalten werden kann bzw. die Leistung durch Hinzufügen verhältnismäßig geringer Ressourcen aufrecht erhalten werden kann.

Ein System skaliert schlecht,

wenn die Leistung bei linear steigender Belastung überproportional abfällt bzw. nur durch unverhältnismäßig hohen Ressourceneinsatz aufrecht erhalten werden kann.

Skalierungstransparenz

- An den Services selbst sollten idealerweise keine Änderungen erforderlich sein, wenn neue Ressourcen hinzugefügt werden.
- Änderungen sollten allenfalls auf der Infrastrukturebene erfolgen, z.B. durch die Einführung von Load-Balancing oder Server-Clustering.

Skalierbarkeit am Beispiel von Load Balancing

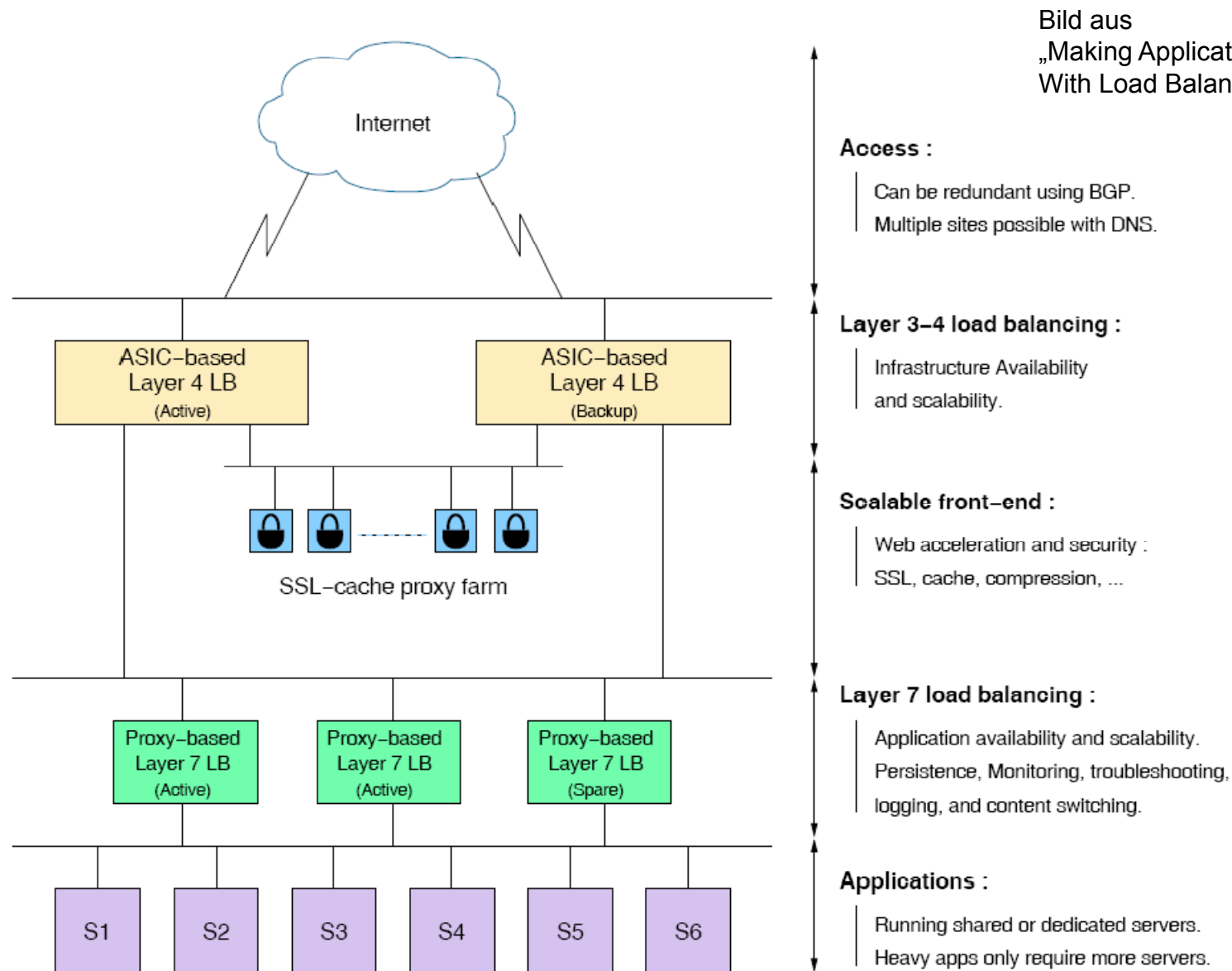


Bild aus
„Making Applications Scalable
With Load Balancing“, Willy Tareau

Detailziele einer SOA: Verteilungstransparenz

Die Lokation von Services soll für die Anwendungen transparent sein.

- Vereinfachung der Verlagerung von Services auf andere Systeme
- Nutzung mehrfach vorhandener, identischer Services möglich
 - Performance
 - Skalierbarkeit
 - Verfügbarkeit
- Vollständige Auslagerung von Services an einen Dienstleister möglich

Steigerung der Softwarequalität (nach ISO/IEC 9126)

- Funktionalität
 - Existenz von Funktionen mit festgelegten Eigenschaften
- Zuverlässigkeit
 - Fähigkeit der Software, ihr Leistungsniveau unter festgelegten Bedingungen über einen festgelegten Zeitraum zu bewahren
- Benutzbarkeit
 - Aufwand der Benutzer für den Einsatz der Software
- Effizienz
 - Verhältnis zwischen Leistungsniveau und eingesetzten Betriebsmitteln
- Änderbarkeit
 - Aufwand zur Durchführung von Änderungen
- Übertragbarkeit
 - Eignung der Software, von einer Umgebung in eine andere übertragen werden zu können

Für eine IT-Abteilung ist „SOA“ nicht nur eine Herausforderung, sondern auch eine Chance!

Herausforderungen

- Neue Methoden erlernen und anwenden
- Neue Technologien einführen
- „Vertrautes Gelände“ verlassen

Chancen

- Consulting für die Fachabteilungen hinsichtlich der Service-Strukturierung
- Strategische Aufgaben übernehmen
- Übersicht und Kontrolle über alle Services

- Die Ausrichtung der Serviceorientierung wird entscheidend von den Geschäftsprozessen bestimmt.
- Die Serviceorientierung richtet sich an allen Gestaltungszielen im Rahmen des Geschäftsmodells aus, nicht nur an den IT-bezogenen Gestaltungszielen.
- Die Serviceorientierung kann wirtschaftlich in der Regel nur in einer stufenweisen Evolution eingeführt werden.
- Die Serviceorientierung erschließt für die IT-Abteilung neue Tätigkeitsfelder, wenn die Herausforderungen angenommen werden.



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 26.5.2009

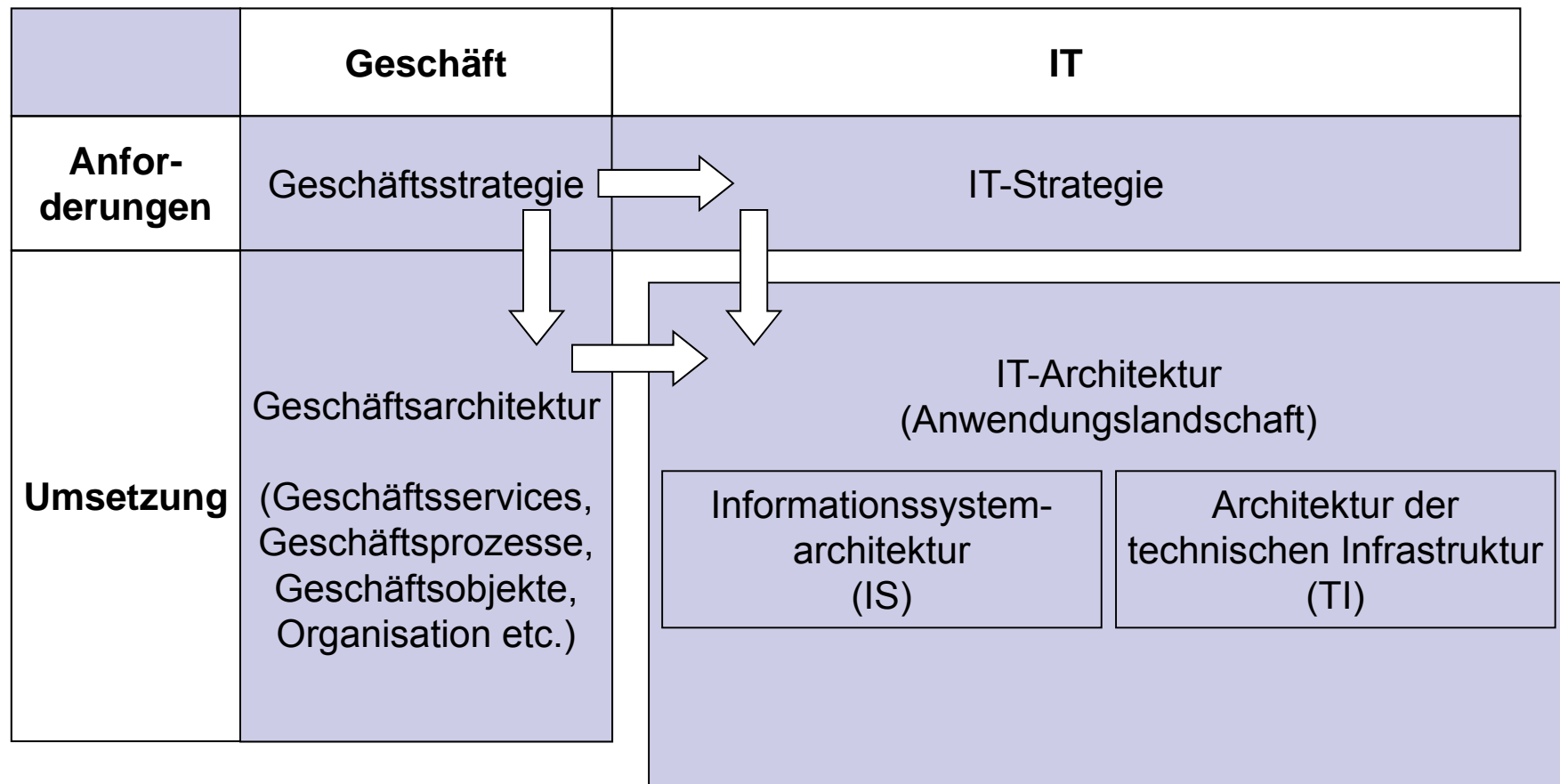
Serviceorientiertes eGovernment

Geschäftsarchitektur, Domänen, Anwendungen

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Zusammenhang zwischen Strategie und Architektur



Quelle: „Quasar Enterprise“

Die Beschreibung der Architektur von Informationssystemen geschieht typischerweise mit Hilfe von standardisierten Architektur-Frameworks.

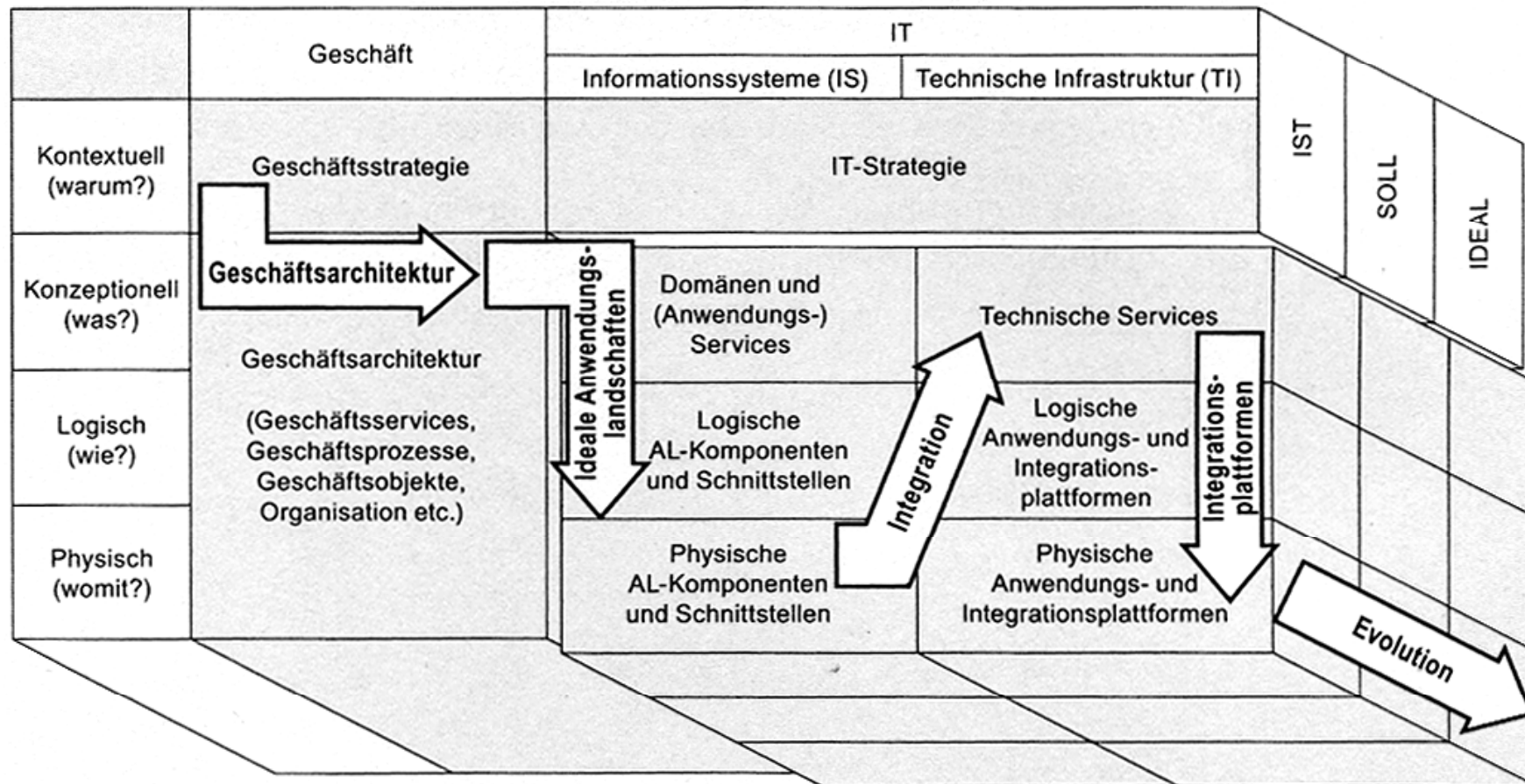
Beispiele:

- Zachman-Framework
- TOGAF (The Open Group Architecture Framework)
- Integrated Architecture Framework (IAF) von CapGemini
- Quasar Enterprise von sd&m (abgeleitet aus IAF)

Leitfaden für die Beantwortung beispielhafter Fragen:

- Was sind die aktuellen und zukünftigen Geschäftsprozesse?
- Wie werden die Geschäftsprozesse durch die IT unterstützt?
- Welche Potentiale ergeben sich für das Unternehmen aus der Umgestaltung von Geschäftsprozessen und der Anwendungslandschaft?
- ...

Landkarte mit Roadmap (Beispiel Quasar Enterprise)



Quelle: „Quasar Enterprise“

Geschäftsservice („business service“)

- Ein Geschäftsservice stellt eine **geschäftliche Leistung** dar, die ein Servicegeber gegenüber einem oder mehreren Servicenehmern erbringt.
- Jedem Geschäftsservice liegt ein **Vertrag** zugrunde:
 - Der Vertrag legt die ein- und ausgehenden **Informationen** und **Güter** fest.
 - Es werden die durchzuführenden **Schritte** und ihre **Reihenfolge** beschrieben (die einzelnen Schritte heißen Geschäftsserviceaktionen (business service actions) oder kurz „Aktionen“).

Beispiele:

- Reisebüro : Verkauf von Pauschalreisen
- Bank: Bearbeitung von Überweisungsaufträgen

Geschäftsprozess

- Ein Geschäftsprozess (business process) ist eine funktions- und stellenübergreifende **Folge von Schritten** zur Erreichung eines geplanten Arbeitsergebnisses in einem Unternehmen.

Die einzelnen Schritte heißen Geschäftsprozessaktivitäten (business process activities) oder auch kurz „**Aktivitäten**“.

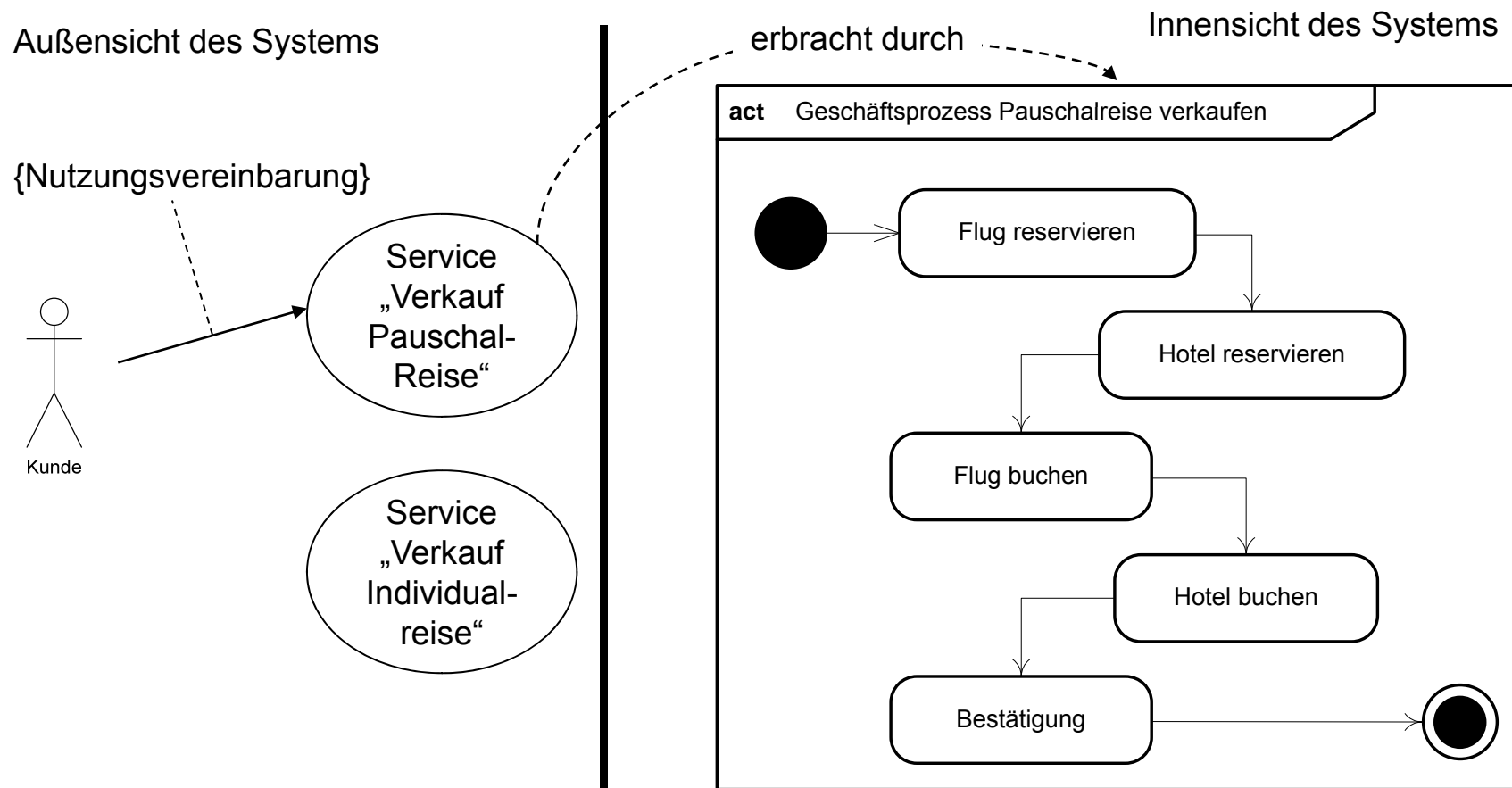
Beispiele:

- Reisebüro: Auftragsabwicklung für eine Pauschalreise
- Bank: Überweisungstransaktion

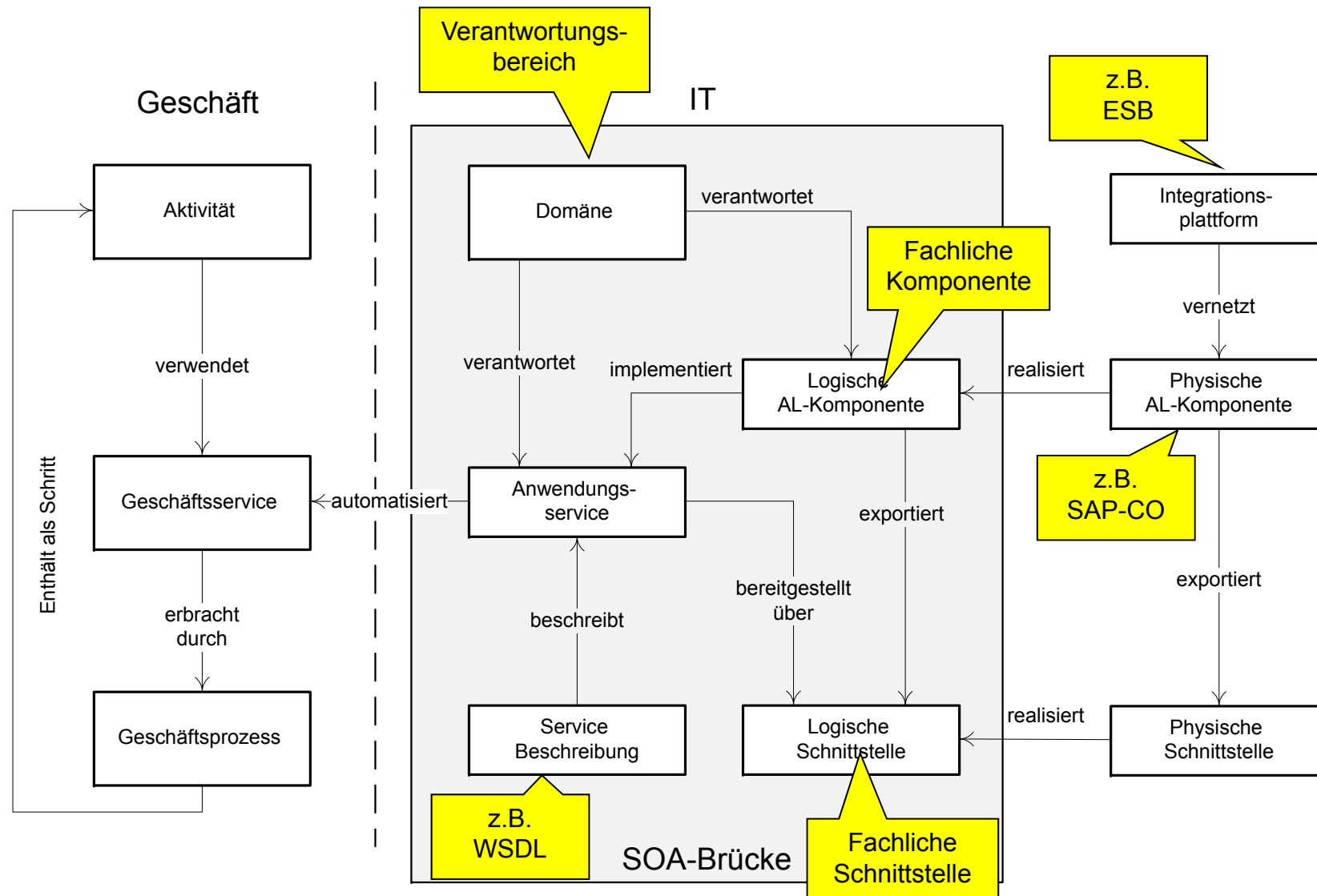
Abgrenzung **Geschäftsservice** ./ **Geschäftsprozess**

Geschäftsservices beschreiben die **Außensicht** eines Systems.

Geschäftsprozesse beschreiben die **Innensicht** eines Systems.



SOA als Brücke zwischen Geschäft und IT

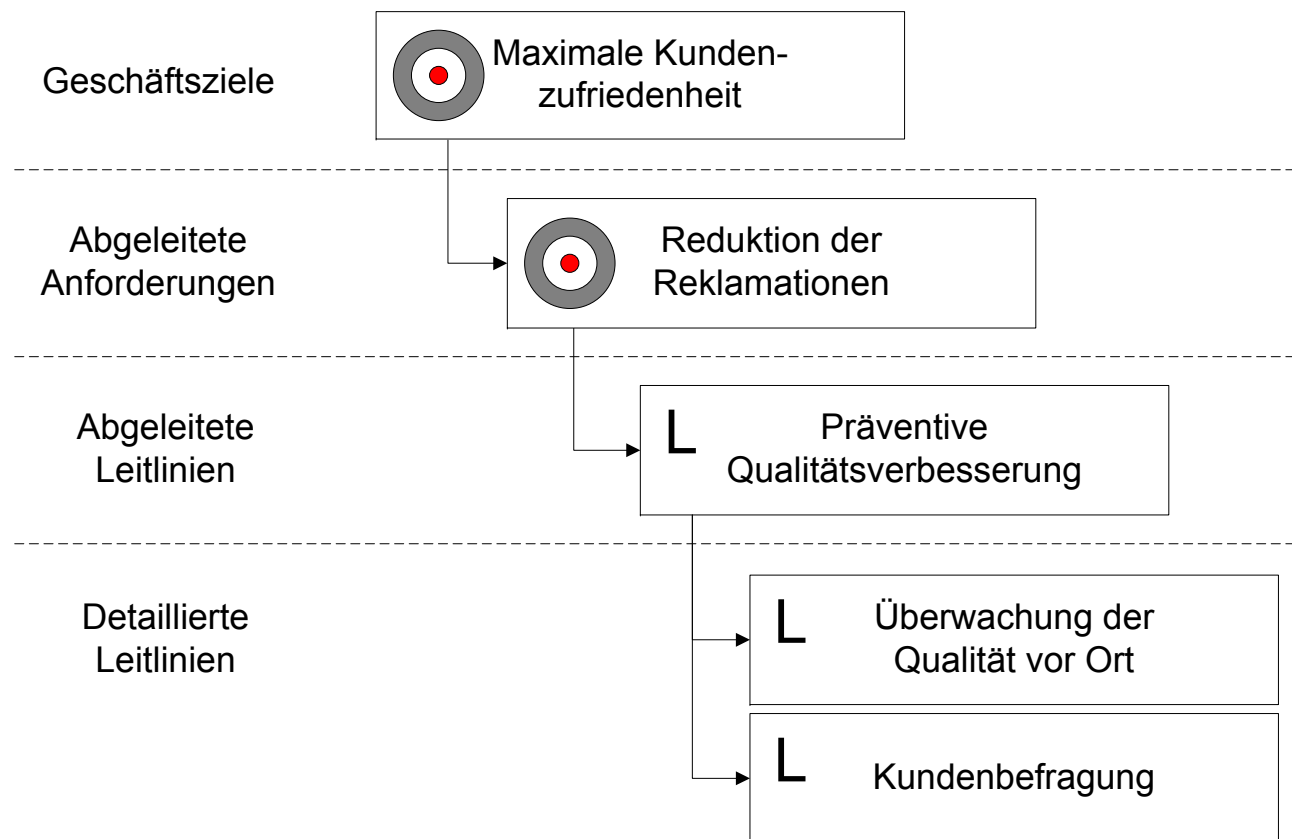


Was ist zu tun?

- Überprüfen, ob das Geschäft eines Unternehmens so beschaffen ist, dass sich **individuelle Serviceorientierung** anbietet oder auf **Standardsoftware** zurückgegriffen werden kann (in der Regel ein Mix)
- Gibt es **Gestaltungsziele** mit besonderer Priorität (z.B. Agilität)?
- Gibt es optimierte Prozesse, die beim Umbau auf Serviceorientierung **ausgespart** bleiben sollen (z.B. hochperformante Massendatenverarbeitung)?
- Ist das Unternehmen „**reif**“ für die Serviceorientierung?
- Sind **fachübergreifende Abstimmungen** effektiv möglich?
- Stehen ausreichende **Start-Budgets** zur Verfügung?

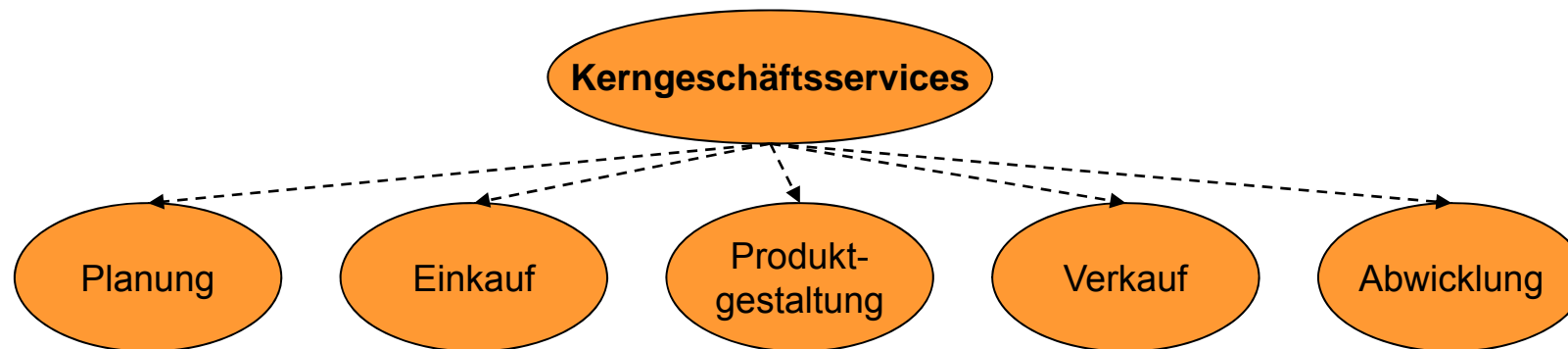
Von Geschäftszielen zu Architekturleitlinien

Ein Systemarchitekt benötigt Leitlinien, die er bei der Gestaltung der Geschäftsarchitektur berücksichtigen soll.

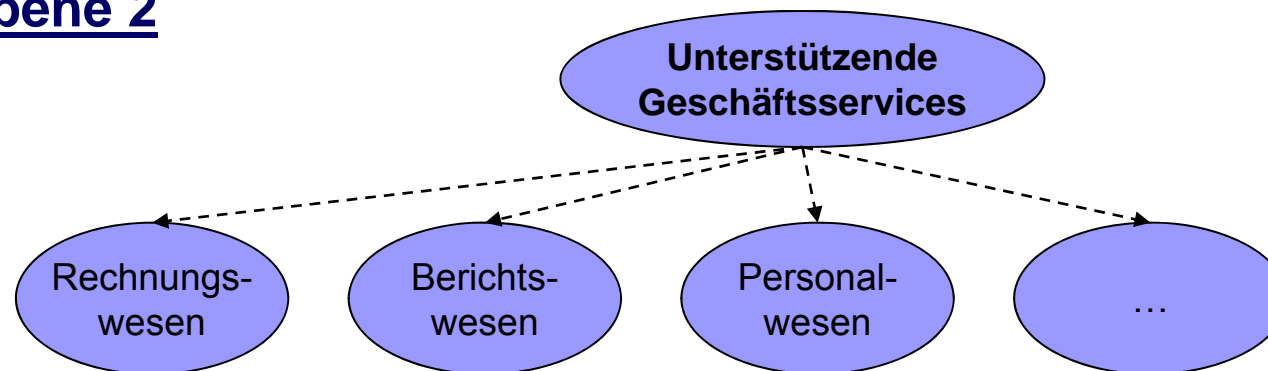


Geschäftsservices der Ebene 1

= die extern angebotenen Leistungen des Unternehmens, die es erbringt, um seinen Geschäftszweck zu erfüllen



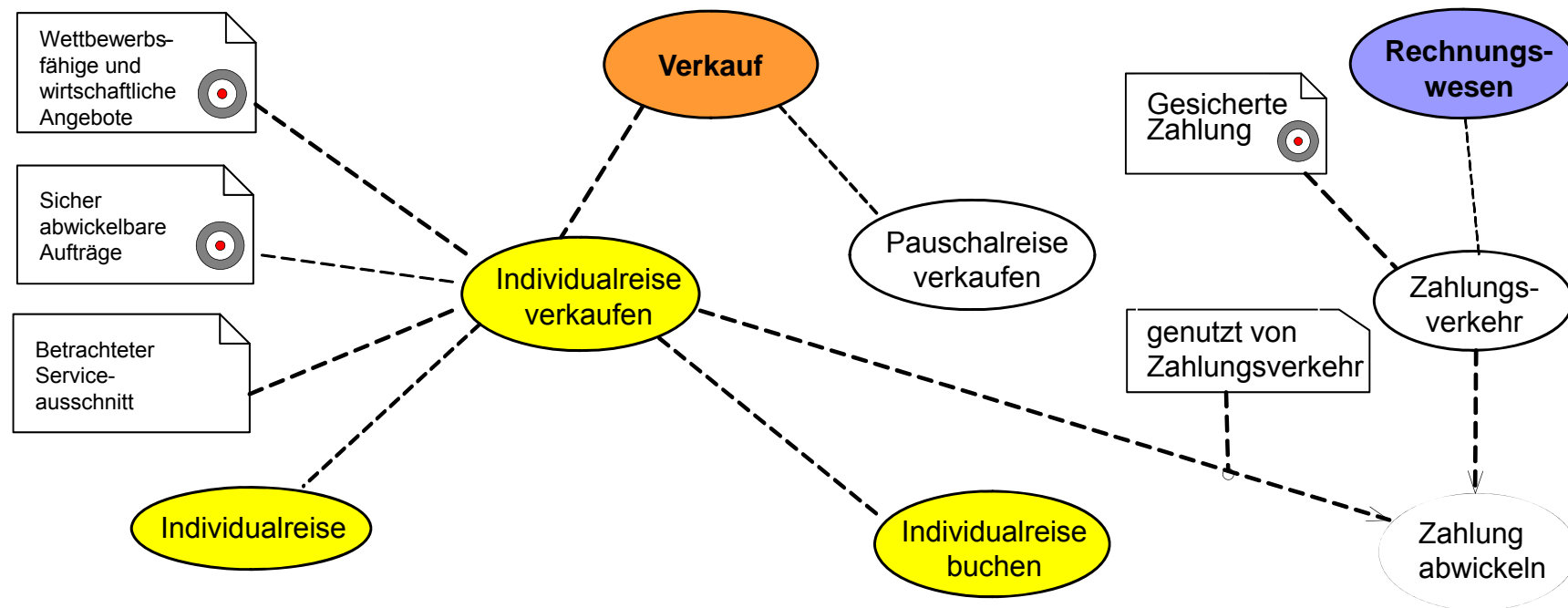
Ebene 2



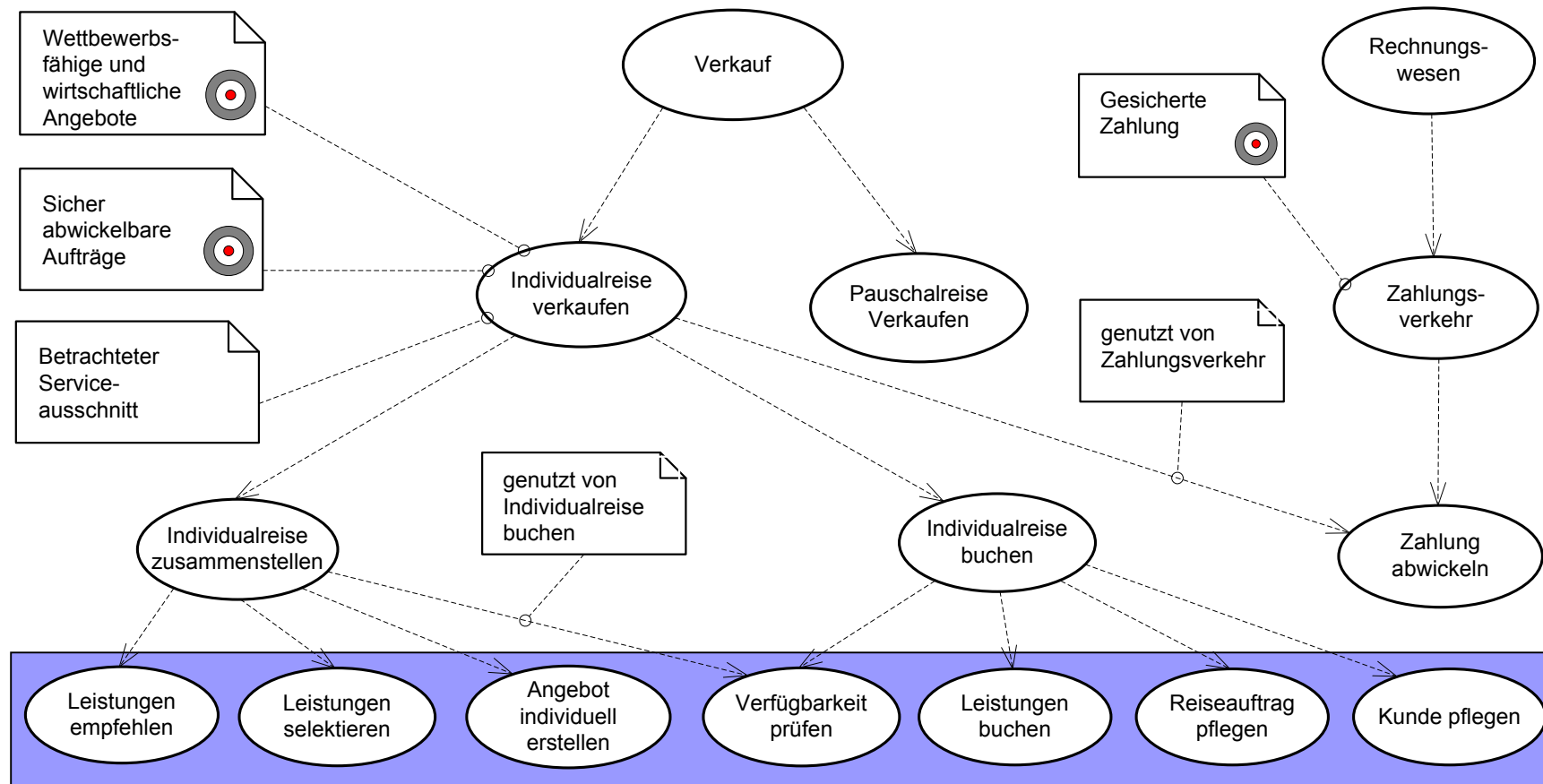
Service-Ausschnitte verfeinern

- Analyse einzelner Service-Ausschnitte
- Verfeinerung der betrachteten Ausschnitte im Hinblick auf die Ziele

Beispiel: Individualreise verkaufen

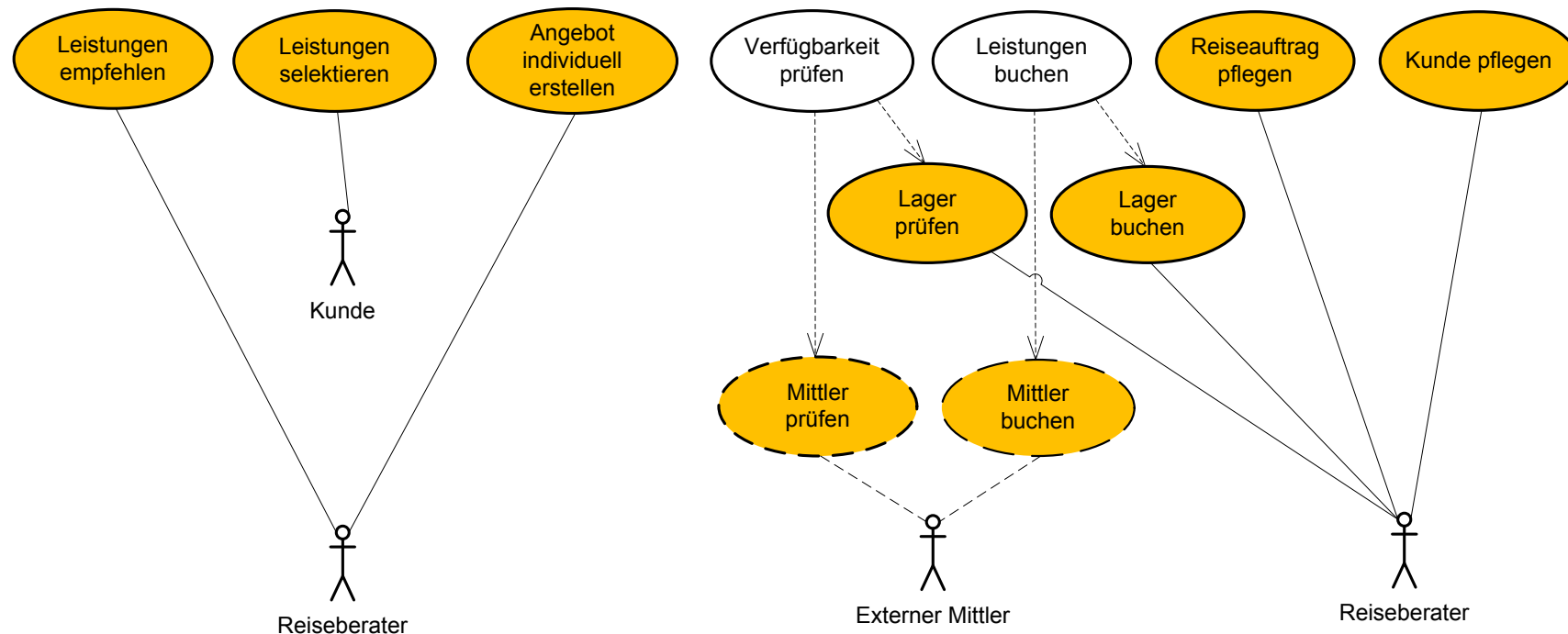


Service-Ausschnitt funktional verfeinern



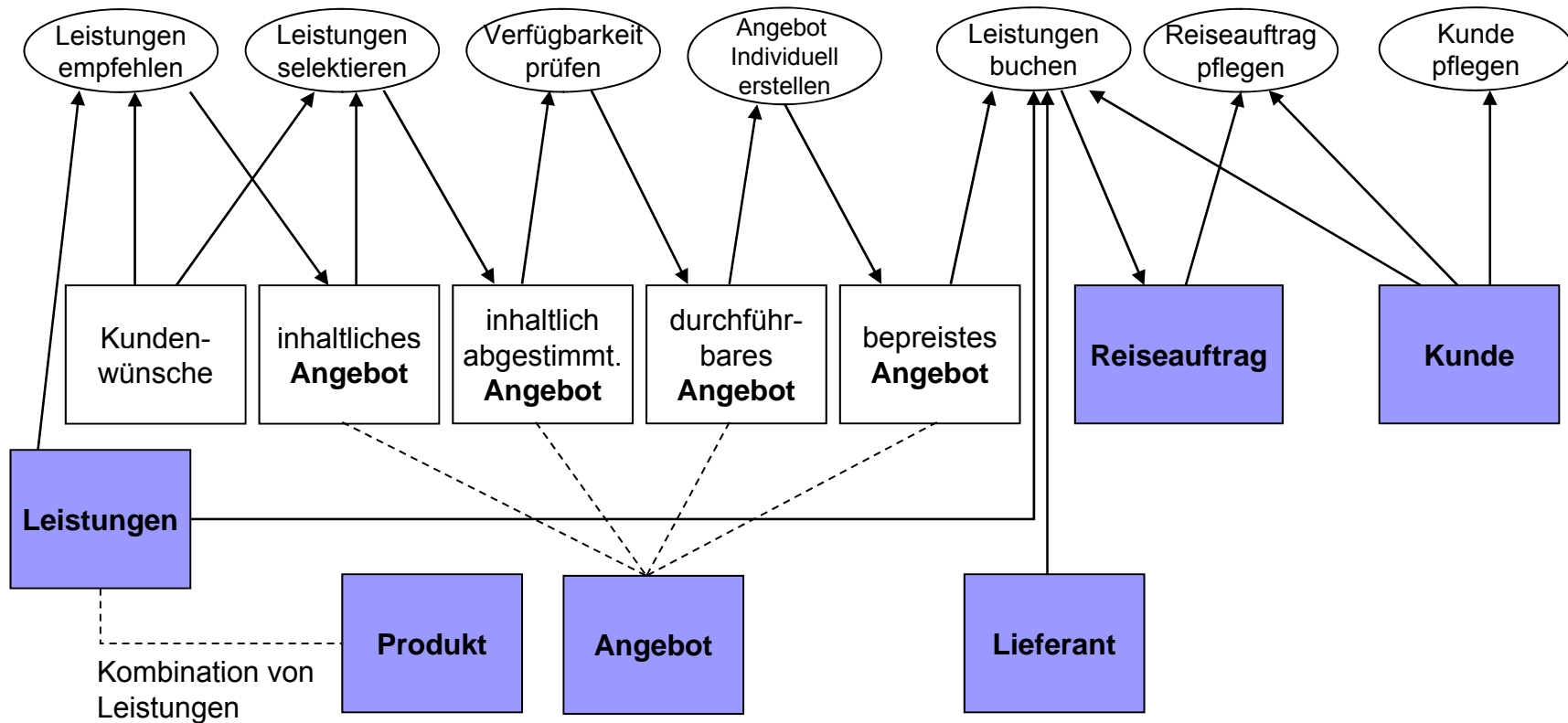
Elementare Geschäftsservices fixieren

- Elementare Geschäftsservices werden von einer **Rolle** ausgeführt. Die Rollen werden dazu genutzt, den Ablauf eines Geschäftsservices exakt zu definieren.



Geschäftsobjekte bestimmen

Anhand der **Informationen**, die bei den einzelnen **Geschäftsservices** verarbeitet werden, lassen sich die **Geschäftsobjekte** ableiten



Schrittweise Gestaltung:

- Ebene 0: Die **Anwendungslandschaft als Ganzes**
- Ebene 1..n: Strukturierung in **Domänen** und **Subdomänen**
- Ebene n+1: **Komponenten** der Anwendungslandschaft mit **Schnittstellen** und **Operationen**

Domänen =

Gruppierung von Komponenten einer Anwendungslandschaft nach fachlichen Gesichtspunkten

Aufgabe von Domänen

- Festlegung von Verantwortungsbereichen
- Abbildung des Geschäfts eines Unternehmens
- Visualisierung einer Anwendungslandschaft („Kartengrund“)

Subdomänen

ergeben sich durch eine hierarchische Gliederung von Domänen

Faustregeln

Größe der AL	Domärentiefe	Anzahl Domänen	Anzahl AL-Komponenten
klein	1	< 10	< 30
mittel	1 – 2	10 – 30	300 – 100
groß	2 – 3	30 – 100	100 – 1.000
sehr groß	≥ 3	> 100	> 1.000

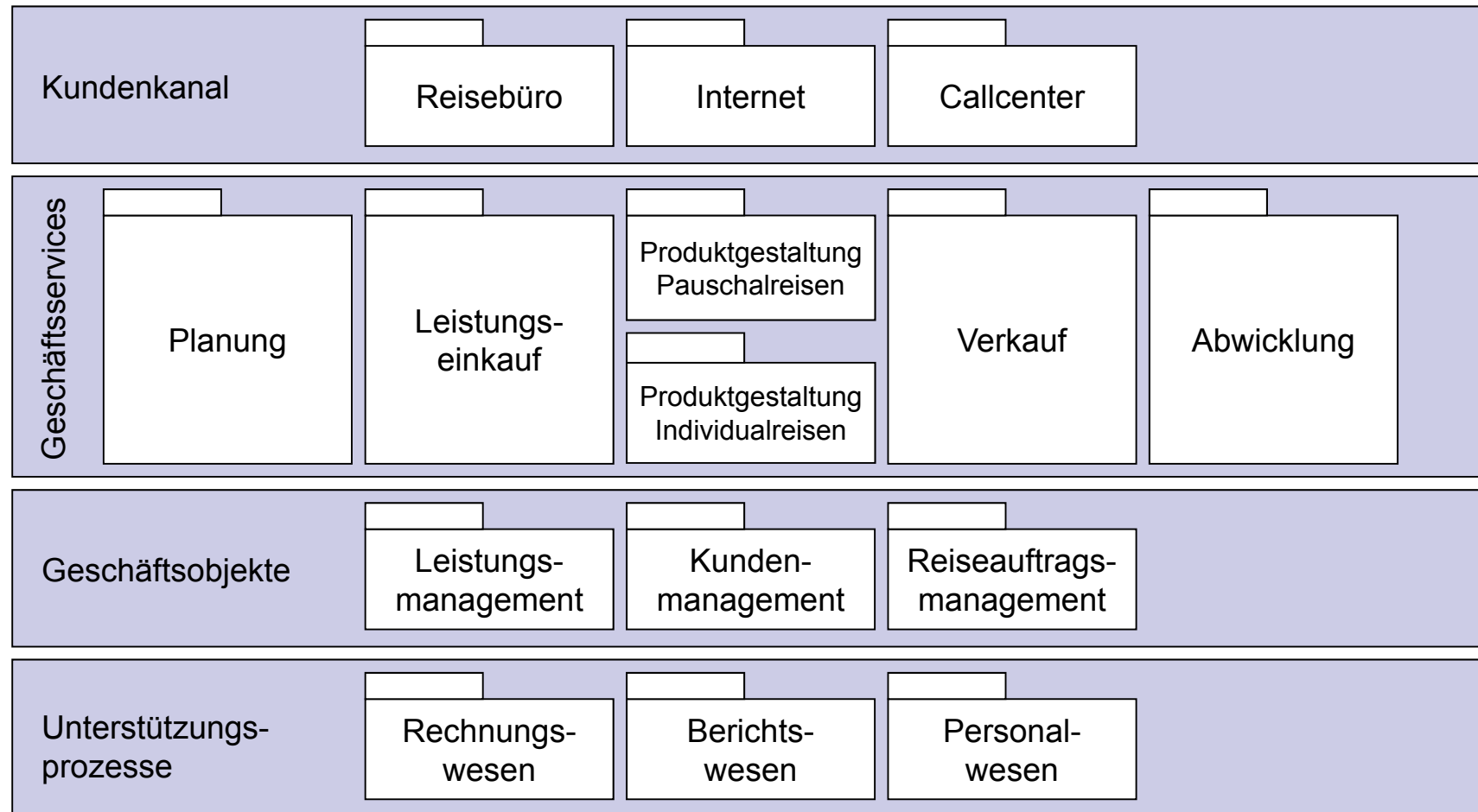
Mögliche Vorgehensweise für den Entwurf von Domänen

Verschiedene Aspekte einer Geschäftsarchitektur liefern „**Kandidaten**“:

- **Kerngeschäftsservices**
- Verfeinerung nach **Geschäftsdimensionen** oder **Teilservices**
- **Geschäftsobjekte**
- **Unterstützende Geschäftsservices** der Ebene 1

Entwurf von Domänen (4)

Beispiel:



Anwendungsservice =

Geschäftsservice oder ein Teil davon, der mittels IT erbracht wird

Identifikation von Anwendungsservices

- Geschäftsservices auswählen, die durch IT abgebildet werden bzw. werden sollen
- Analyse und Spezifikation der Kandidaten

Anmerkung:

In der Geschäftsarchitektur kann es Geschäftsservices geben, die nur manuell durchgeführt werden und nicht über IT abgebildet werden bzw. werden sollen.

Anwendungsservice spezifizieren (im Stil eines Use Cases)

- Namensgebung
- Außensicht beschreiben
- Soweit erforderlich, die Innensicht beschreiben

Beispiel:

Name	Leistungen empfehlen
Außensicht	
Servicenutzer	Reiseberater
Auslösendes Ereignis/Vorbedingungen	Kundenanfrage; Leistungskatalog liegt vor
Aktionen und Service-Protokoll	Kein Protokoll, da nur eine einzige Serviceaktion
Ergebnis/Nachbedingungen	Ausgewählte Leistungen als inhaltliches Angebot
Nichtfunktionale Anforderungen	Die Antwortzeit beträgt < 1 s
Innensicht	
Prozess	Die Leistungen werden aus den Angeboten aller Lieferanten gewählt

AL-Komponente (Anwendungslandschaftskomponente)

- **implementiert** einen Anwendungsservice
- hat explizite und wohl definierte Schnittstellen für Operationen, die sie **anbietet**
- hat explizite und wohl definierte Schnittstellen für Operationen, die sie **nutzt**
- kann mit anderen AL-Komponenten **gekoppelt** werden



AL-Komponenten sind in der Regel sehr umfangreich. Sie stellen nicht einzelne kleine Operationen, sondern eine große Anzahl von inhaltlich zusammengehörenden Operationen bereit.

Beispiel: Ein SAP-Modul (PP, HCM, etc.)

Kategorien (nach Quasar Enterprise)

- **Bestand**
→ Verwaltung von Datenbeständen
- **Funktion**
→ IT-unterstützte Geschäftsservices mit algorithmischem Charakter
- **Prozess**
→ IT-unterstützter Geschäftsprozess
- **Interaktion**
→ Interaktion mit dem Anwender oder anderen Anwendungslandschaften



In der Fachliteratur gibt es noch weitere Kategorien, die teilweise feiner sind (z.B. von Thomas Erl).

Vorgehensweise

1. Anwendungsservices Domänen zuordnen
2. Anwendungsservices kategorisieren (Bestand, Funktion, ...)
3. Kandidaten für Komponenten bilden
4. Komponentenschnitt verfeinern
5. Spezifikation finalisieren

Entwurf von AL-Komponenten (4)

Domänenzuordnung und Kategorisierung

Anwendungsservice	Domäne	Kategorie
Individualreise verkaufen	Reisebüro	Interaktion
Individualreise verkaufen	Internet	Interaktion
Individualreise verkaufen	Callcenter	Interaktion
Individualreise zusammenstellen	Reisebüro	Interaktion
Individualreise zusammenstellen	Internet	Interaktion
Individualreise zusammenstellen	Callcenter	Interaktion
Leistungen empfehlen	Produktgestaltung Individualreisen	Funktion
Leistungen selektieren	Produktgestaltung Individualreisen	Funktion
Individualreise buchen	Verkauf	Prozess
Verfügbarkeit prüfen	Leistungsmanagement	Prozess
Lager prüfen	Leistungsmanagement	Bestand
Leistung buchen	Leistungsmanagement	Prozess
Lager buchen	Leistungsmanagement	Bestand
Reiseauftrag pflegen	Reiseauftragsmanagement	Bestand
Kunde pflegen	Kundenmanagement	Bestand

Entwurf von AL-Komponenten (5)

Kandidaten für Komponenten

Zusammenfassung der Anwendungsservices einer Domäne und Kategorie

<u>Domäne</u>	<u>Kategorie</u>	<u>Anwendungsservices</u>	<u>Komponentenkandidat</u>
Reisebüro	Interaktion	Individualreise verkaufen Individualreise zusammenstellen	Reisebüro-Buchung
Internet	Interaktion	Individualreise verkaufen Individualreise zusammenstellen	Reiseportal
Callcenter	Interaktion	Individualreise verkaufen Individualreise zusammenstellen	Callcenter-Buchung
Produktgestaltung Individualreisen	Funktion	Leistungen empfehlen Leistungen selektieren	Individualreise- Konfigurator
Verkauf	Prozess	Individualreise buchen	Individualbuchungs- prozess
Leistungsmanagement	Prozess	Verfügbarkeit prüfen Leistung buchen	Virtuelles Lager
Leistungsmanagement	Bestand	Lager prüfen Lager buchen	Lagermanagement
Reiseauftragsmanagement	Bestand	Reiseauftrag pflegen	Reiseauftrags- management
Kundenmanagement	Bestand	Kunde pflegen	Kundenmanagement

Zusammenfassung der Regeln für den Entwurf von Komponenten

- Komponenten sollen **eindeutig einer Domäne** zugeordnet werden
- Komponenten sollen **nach fachlichen Kriterien** gebildet werden
- Alle Operationen **einer** Komponente sollen **von genau einer Kategorie** (Bestand, Funktion, Prozess, Interaktion) sein.
- Die Kopplungen zwischen AL-Komponenten unterschiedlicher Kategorien sollen einer **Schichtung** folgen

Reihenfolge: Interaktion → Prozess → Funktion → Bestand

Keine zyklischen Kopplungen zwischen Komponenten!

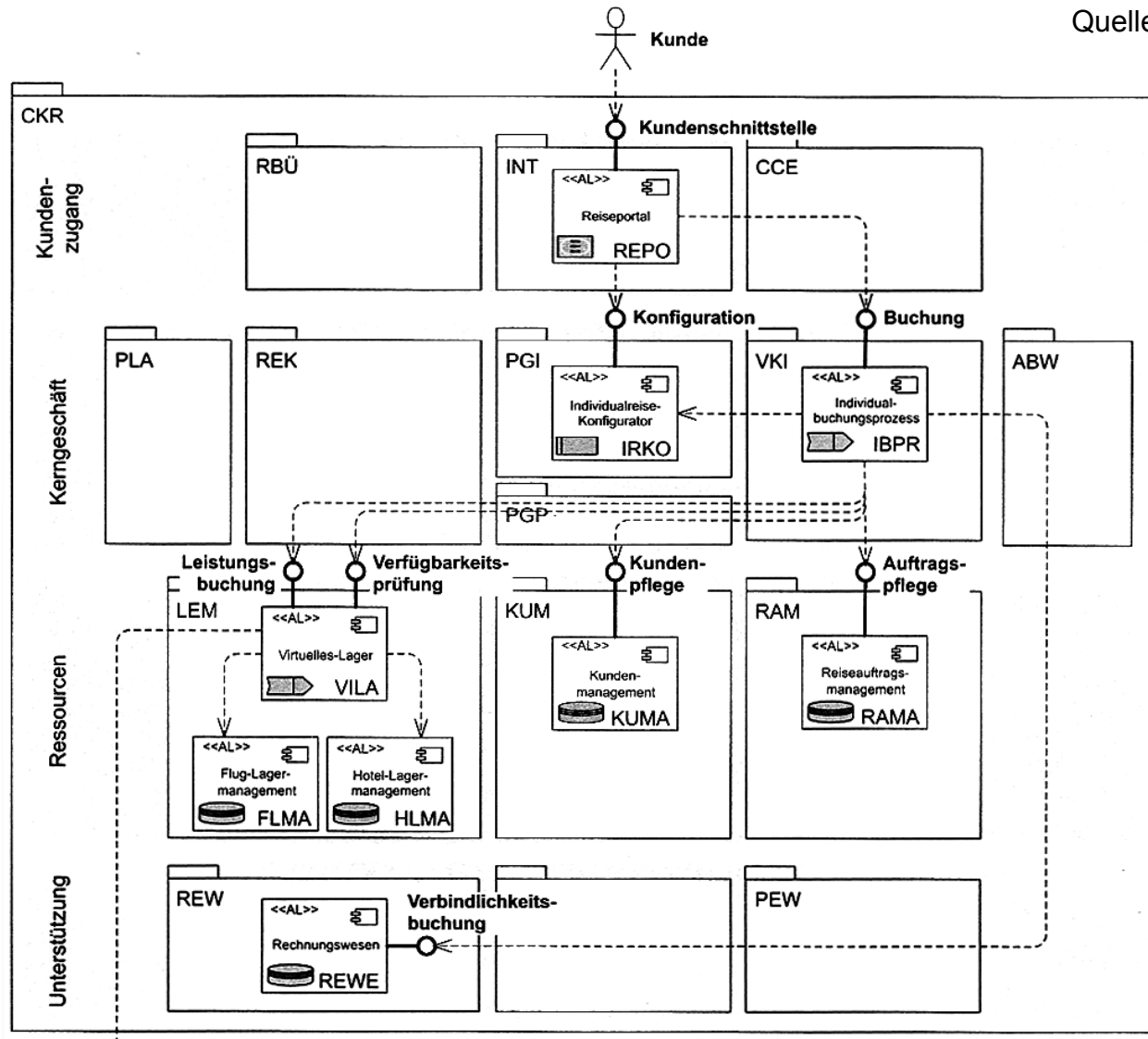
- **Komponentenintern hohe Kopplung, zwischen Komponenten geringe Kopplung**
- Bestandskomponenten sollen **Datenhoheit über die Geschäftsobjekte** haben.

Schnittstellen

- Eine Schnittstelle fasst Operationen zusammen
- Eine Schnittstelle wird spezifiziert durch
 - einen eindeutigen Namen
 - die Menge der zugehörigen Operationen
 - ein Schnittstellenprotokoll im Sinne von Reihenfolgen und Restriktionen beim Aufruf der Operationen

Beispiele für Schnittstellen

Quelle: „Quasar Enterprise“



Operationen

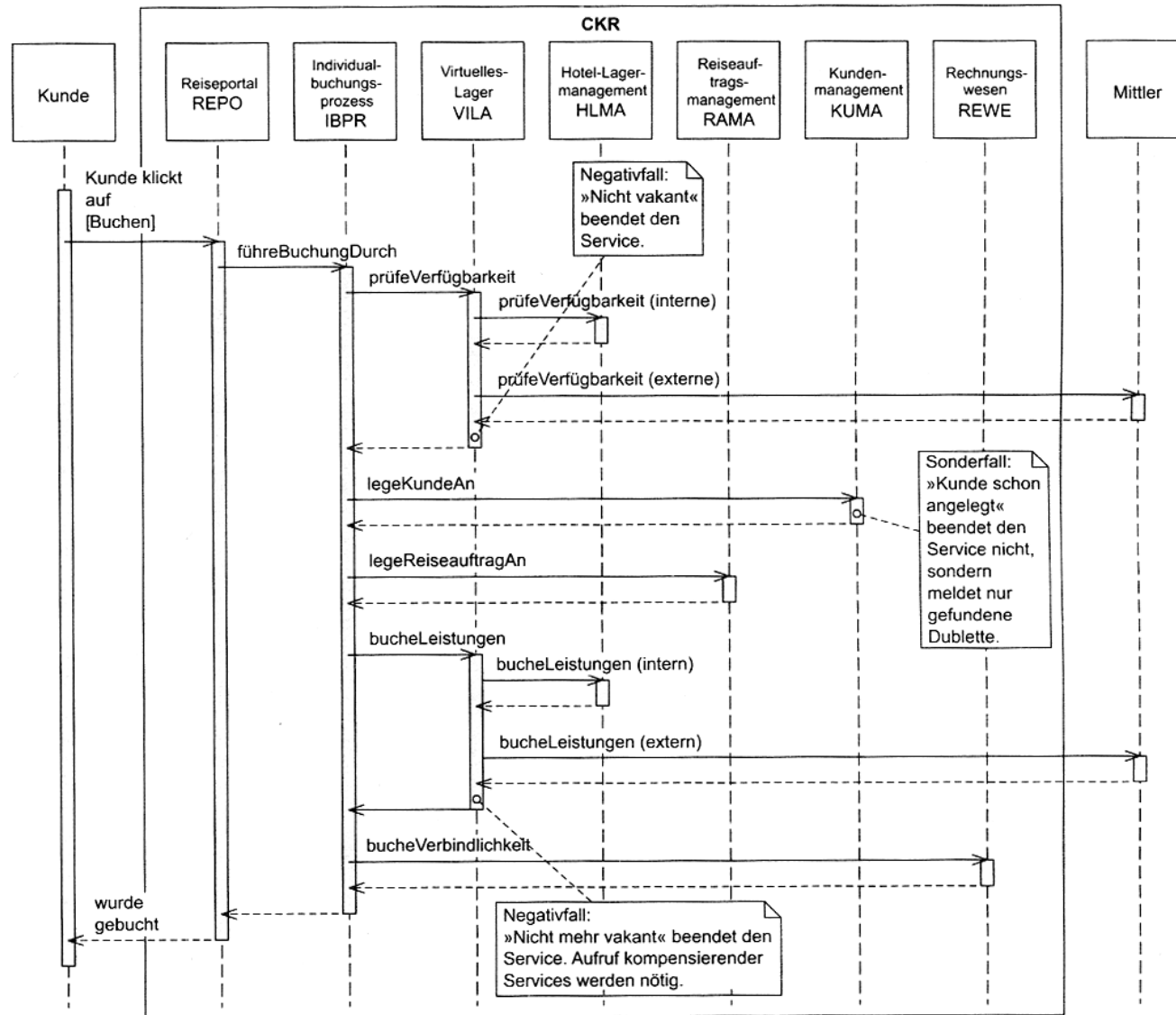
- Operationen beschreiben das Verhalten von Komponenten
- Operationen werden spezifiziert durch:
 - Signatur: Name, Parameter, Rückgabewerte und deren Typen
 - Semantik: Verhalten der Operation
 - Nicht-funktionale Eigenschaften: Performance, Verfügbarkeit, etc.

Regeln für den Entwurf von Operationen

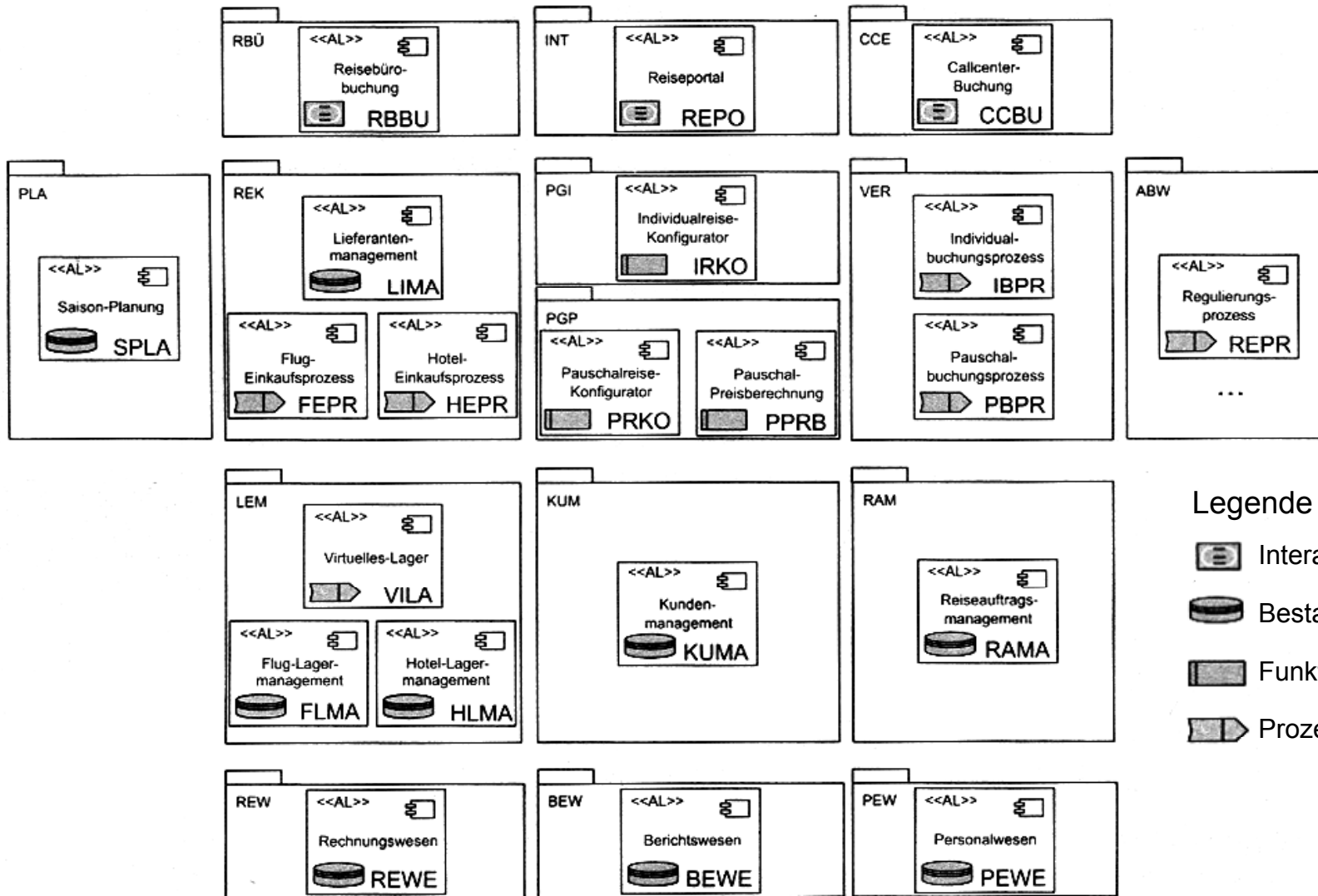
- Operationen sollen angemessen **grobgranular** sein
→ Reduktion der Komplexität
- Operationen sollen - falls fachlich sinnvoll und möglich - **idempotent** sein
→ Mehrmaliger Aufruf mit denselben Parametern hat den gleichen Effekt wie der einmalige Aufruf
- Für alle Operationen sollen **kompensierende Operationen** angeboten werden (so dass es möglich wird, Operationen rückgängig zu machen)
- Operationen sollen **minimales Wissen über den Aufrufkontext** haben (sonst nur eingeschränkte Einsetzbarkeit)

Beispiele für Operationen

Quelle: „Quasar Enterprise“



Finale Anwendungslandschaft



Quelle: „Quasar Enterprise“



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 9. Juni 2009

Serviceorientiertes E-Government

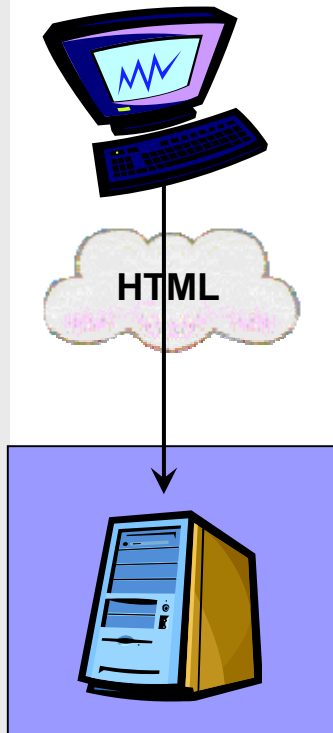
Web Services und Verzeichnisdienste

Dr. Frank Sarre

Lehrbeauftragter der LMU München

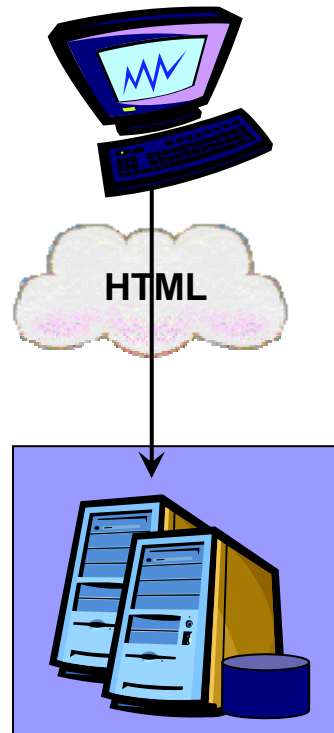
Evolution der Internet-Technologie

1



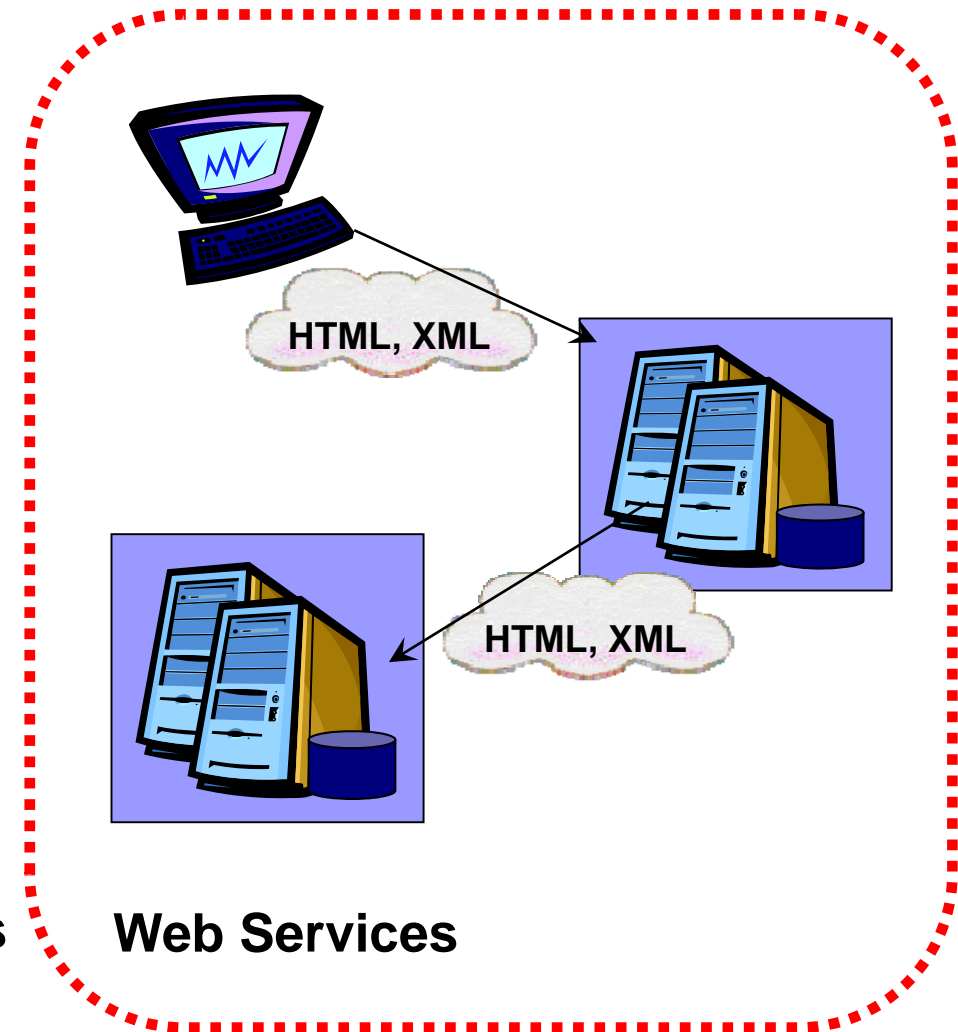
Static HTML

2



Web Applications

3



Web Services

W3C-Definition

„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.“



Hier ist nicht die Rede von Mensch-Maschine-Interaktion. Benutzer (Menschen) nutzen Web Services nur mittelbar.

Web Services

- ... bieten für andere **Programme** (innerhalb und außerhalb des Unternehmens) **plattform- und programmiersprachenunabhängige Dienste** an
- Die (sich z.T. **selbst beschreibenden**) Dienste eines Web Services können im Web **publiziert, lokalisiert** und **aufgerufen** werden
- ... basieren auf bekannten **Web-Technologie-Standards**
- ... tauschen mit ihren Aufrufern **Nachrichten** (in einem Standardformat) aus



Web Services erleichtern den **Aufbau** und die **Integration** verteilter Web-Applikationen

- Internet-Reisebuchungssysteme
- Datenaustausch zwischen eBusiness-Partnern
- Supply Chain Management
- (Informations-) Portale
- E-Procurement, elektronische Marktplätze
- E-Payment, Billing

Aufgaben

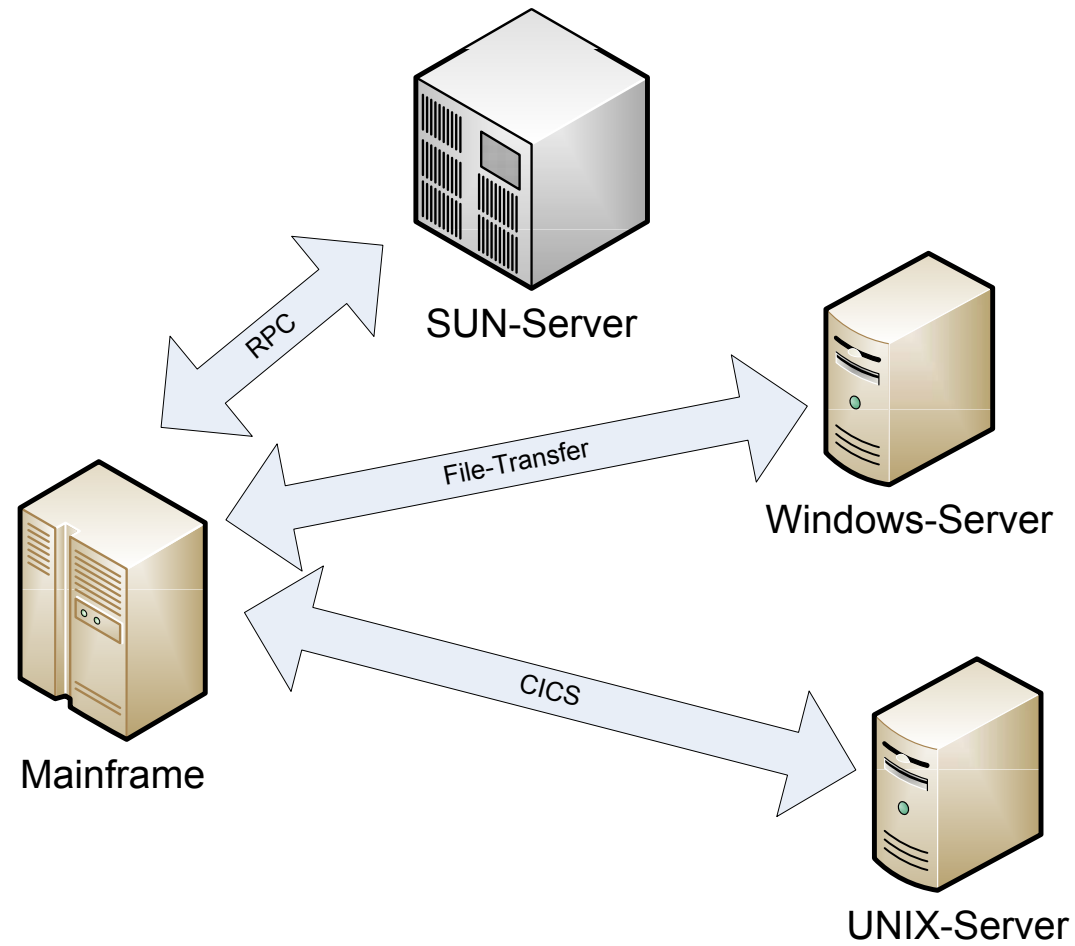
- Aufruf von entfernten Prozeduren
- Übermittlung von Daten

Aber auch:

- Beseitigung proprietärer Schnittstellen
- Beseitigung komplexer Kommunikationsprotokolle
- Herstellung einer losen Kopplung
- Reduktion der Abhängigkeit von einem oder mehreren Herstellern
- ...

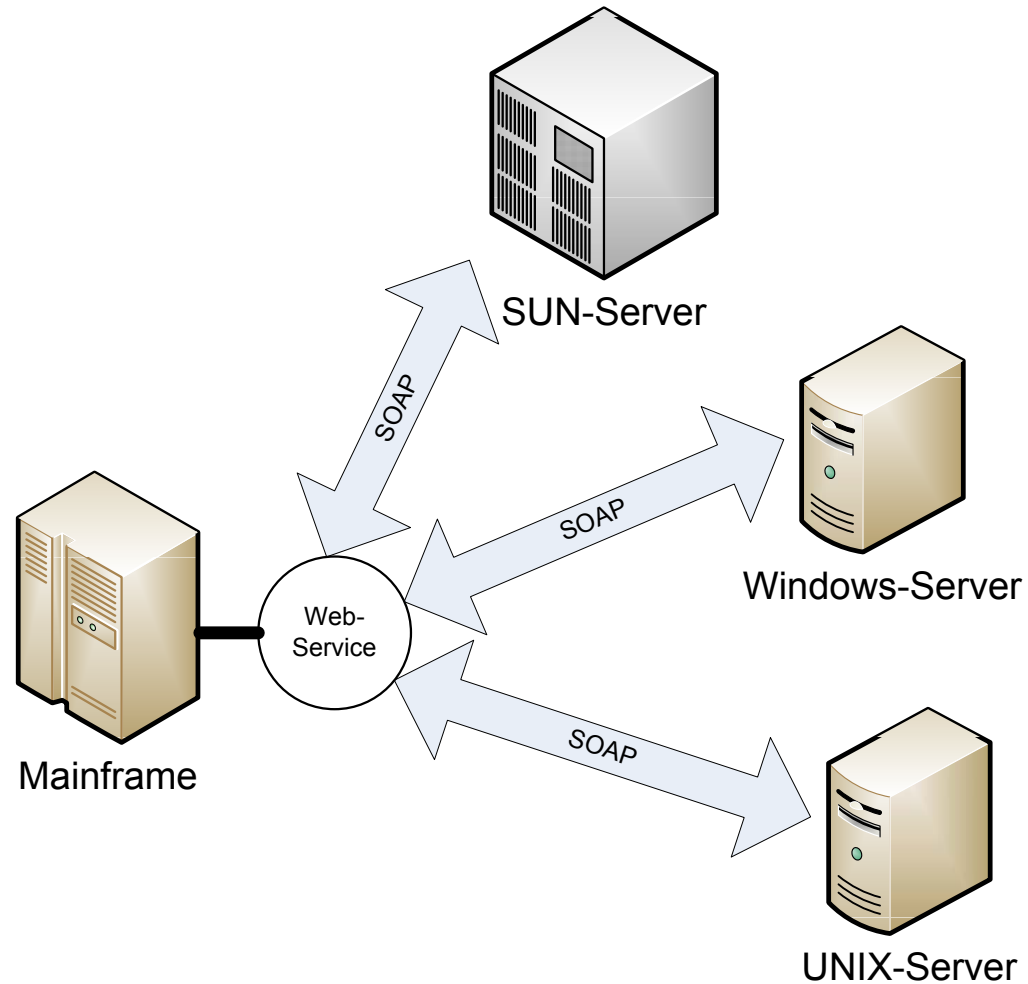
Aufgaben von Web Services (2)

Ausgangssituation



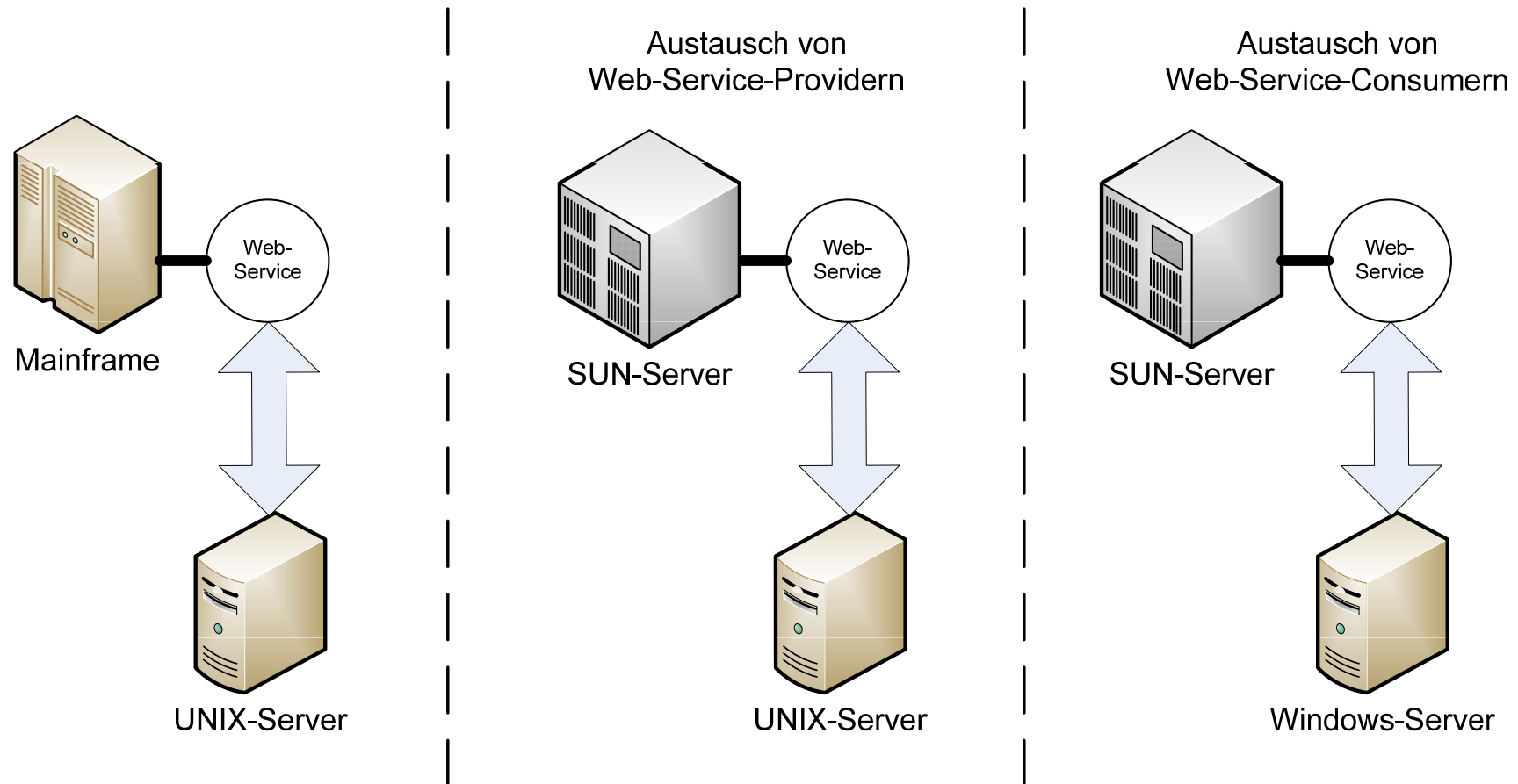
Aufgaben von Web Services (3)

Transparente Kommunikation unabhängig von Hardware, Betriebssystem und Netzwerk



Aufgaben von Web Services (4)

Verbesserte Austauschbarkeit von Systemen



Was bieten Web Services von Haus aus nicht?

- Sicherheit
- Transaktion
- Session Handling
- Autorisierung / Authentisierung
- Hohe Performance
- Hohe Verfügbarkeit
- Abrechnungsmodelle für die Nutzung
- Konzepte für die Anwendungsentwicklung im großen Stil
- ...

Standards

- SOAP (Kommunikationsprotokoll)
- WSDL (Service-Beschreibung)
- UDDI (Verzeichnisdienst)

Standardisierungsgremien

- W3C – World Wide Web Consortium (www.w3c.org)
→ Web-Standards, z.B. XML, WSDL
- OASIS - Organization for the Advancement of Structured Information Standards (www.oasis-open.org)
→ erweiterte WS-Standards, z.B. WS-Security
- WS-I – Web Services Interoperability Organization (www.ws-i.org)
→ Standardprofile, z.B. Basic Profile

- SOAP basiert auf **XML**
- Kommunikation in Form von „**SOAP-Messages**“

Es gibt zwei Message-Typen:

- a) **SOAP-Request** (Aufruf eines Service)
- b) **SOAP-Response** (Antwort eines Service)

Standardisierte Struktur für Fehlermeldungen, die in einer SOAP-Response übermittelt werden

Die aktuelle SOAP-Version ist **1.2**

Transport von SOAP-Nachrichten

- Grundsätzlich ist SOAP unabhängig vom Transportprotokoll
- In der Praxis wird jedoch fast immer HTTP verwendet

Warum HTTP?

- Weit verbreitetes Standard-Protokoll
- Auf allen Plattformen verfügbar
- Einfach zu implementieren
- Firewall-freundlich (aber: Sicherheitsrisiko!)

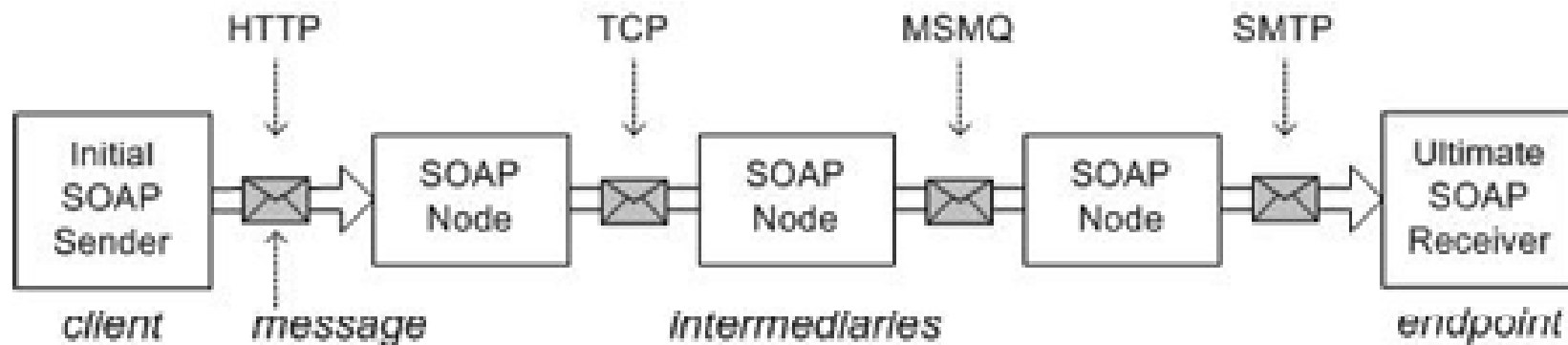
Nachteile von HTTP

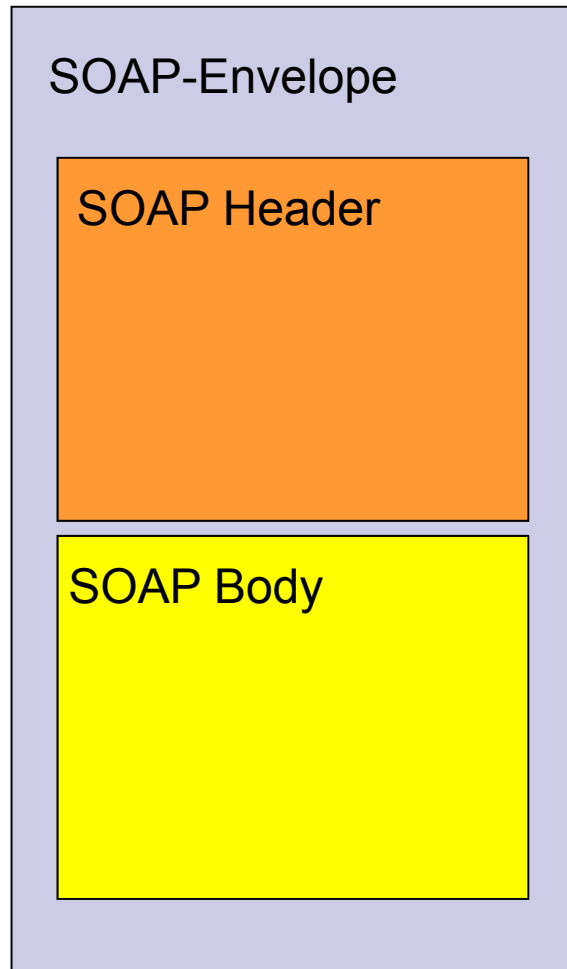
- Synchrones Protokoll

Beispiel einer SOAP-Kommunikationskette

Quelle: <http://msdn.microsoft.com/en-us/library/ms995800.aspx>

So könnte eine SOAP-Kommunikationskette **mit verschiedenen Transportprotokollen** aussehen:





<SOAP-ENV:Envelope>

<SOAP-ENV:Header>

- Kein zwingender Bestandteil der Nachricht
- Parameter, die die Kommunikation steuern
- Gegenstand vieler Erweiterungen des Standards, wie z.B. WS-Addressing

</SOAP-ENV:Header>

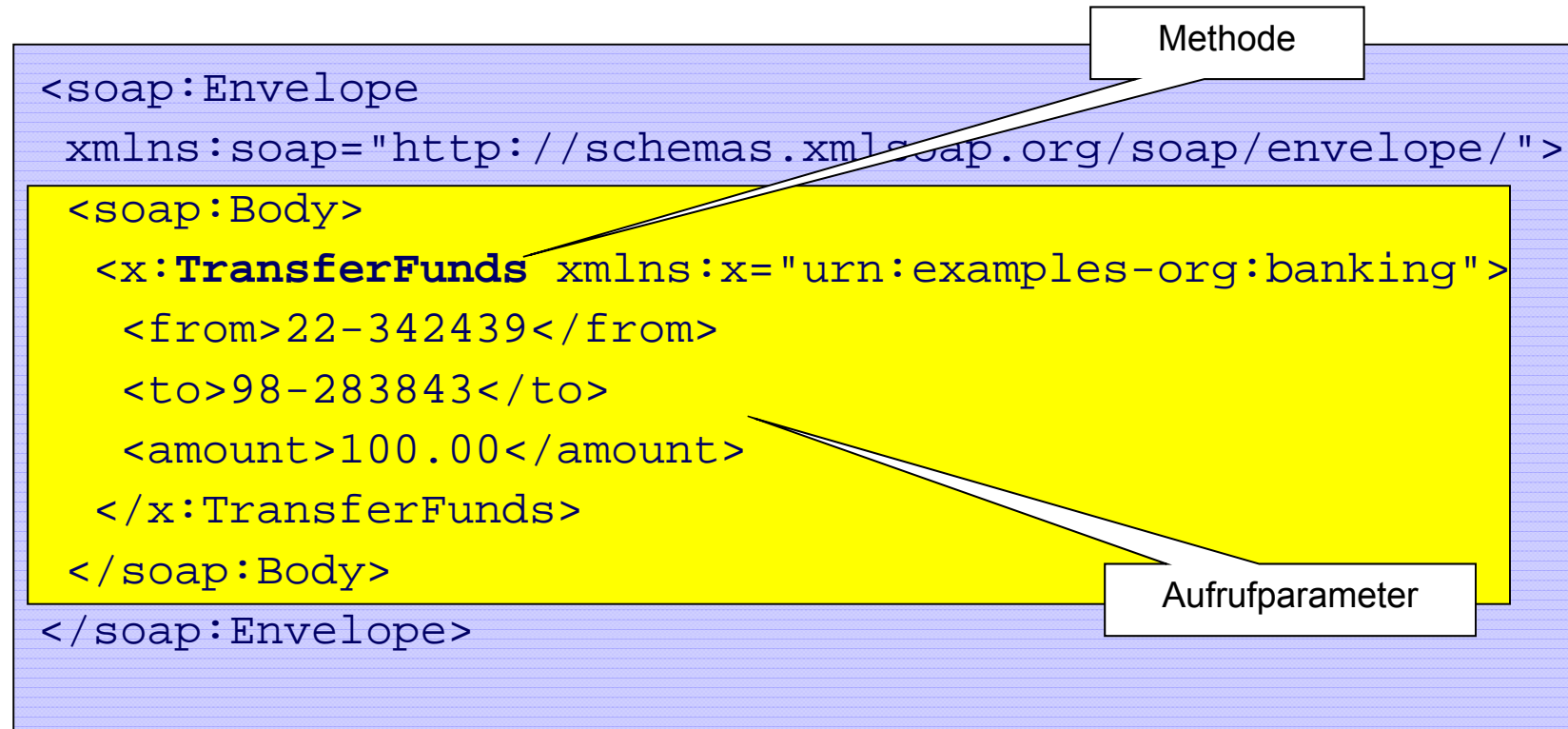
<SOAP-ENV:Body>

- Daten
- Funktionsaufrufe
- Funktionsparameter

</SOAP-ENV:Body>

</SOAP-ENV:Envelope>

Struktur eines SOAP-Requests



Es handelt sich um einen „RPC-Style“-Request, d.h. Aufruf einer Methode mit Parametern

SOAP im HTTP-Request

```
POST /soap/handlefunds HTTP/1.0
Host: localhost:80
Content-Type: text/xml; charset=utf-8
Content-Length: 265
SOAPAction: ""
```

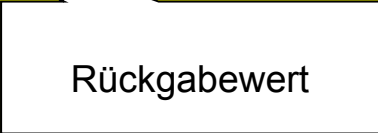
HTTP

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <x:TransferFunds xmlns:x="urn:examples-org:banking">
      <from>22-342439</from>
      <to>98-283843</to>
      <amount>100.00</amount>
    </x:TransferFunds>
  </soap:Body>
</soap:Envelope>
```

SOAP

Struktur einer SOAP-Response

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <x:TransferFundsResponse
      xmlns:x="urn:examples-org:banking">
      <balances>
        <account>
          <id>22-342439</id>
          <balance>33.45</balance>
        </account>
        <account>
          <id>98-283843</id>
          <balance>932.73</balance>
        </account>
      </balances>
    </x:TransferFundsResponse>
  </soap:Body>
</soap:Envelope>
```



SOAP-Styles (stark vereinfacht)

Document style

```
<soap:Body>  
  <m:purchaseOrder xmlns:m="someURI">  
    ...  
  </m:purchaseOrder>  
</soap:Body>
```

Inhalt der Nachricht ist frei strukturierbar und muss durch Sender und Empfänger validiert werden, z.B. über ein XML-Schema.

Verwendung eines Methodennamens

RPC style

```
<soap:Body>  
  <m:placeOrder xmlns:m="someURI">  
    <m:purchaseOrder>  
      ...  
    </m:purchaseOrder>  
  </m:placeOrder>  
</soap:Body>
```

Parameter können durch Standarddatentypen abgebildet und automatisch validiert werden.

Im Standardschema sind bereits 40 Datentypen definiert.

- Wichtige Standardattribute
 - **encodingStyle**

Informationen, wie Daten in einer SOAP-Message serialisiert werden sollen
 - **role**

Spezifiziert Empfänger bzw. Zwischenstation, die dieses Header-Element verarbeiten darf
 - **mustUnderstand**

Legt fest, ob ein Header-Element bei der Weiterleitung der SOAP-Message ausgewertet werden muss
- Individuelle Erweiterungen sind möglich
- Zahlreiche standardisierte Erweiterungen verfügbar, z.B. WS-Addressing, WS-Security

Beispiel für einen SOAP-Header

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <!-- security credentials -->
    <s:credentials xmlns:s="urn:examples-org:security"
      soap:mustUnderstand="1">
      <username>dave</username>
      <password>evad</password>
    </s:credentials>
  </soap:Header>
  ...
</soap:Envelope>
```

Das Beispiel stellt dar, dass der Empfänger dieser Nachricht die credentials `<username>` und `<password>` verstehen und auswerten **muss**.

SOAP-Fault

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Receiver</faultcode>
      <faultstring>Insufficient funds</faultstring>
      <detail>
        <x:TransferError xmlns:x="urn:examples-org:banking" >
          <sourceAccount>22-342439</sourceAccount>
          <transferAmount>100.00</transferAmount>
          <currentBalance>89.23</currentBalance>
        </x:TransferError>
      </detail>
    </x:TransferFunds>
  </soap:Body>
</soap:Envelope>
```

<faultcode> und
<faultstring> sind
Pflichtinformationen

<detail> ist optional

- **VersionMismatch**

Ein Knoten (ein Kommunikationspartner in der gesamten Kommunikationskette) erwartet eine andere SOAP-Version

- **MustUnderstand**

Ein Knoten kann eine Pflichtinformation im Header nicht auswerten

- **DataEncodingUnknown**

Es sind Datentypen verwendet worden, die nicht in eine SOAP-Nachricht übersetzt werden konnten

- **Sender**

Die Nachricht konnte vom Sender nicht verarbeitet werden

- **Receiver**

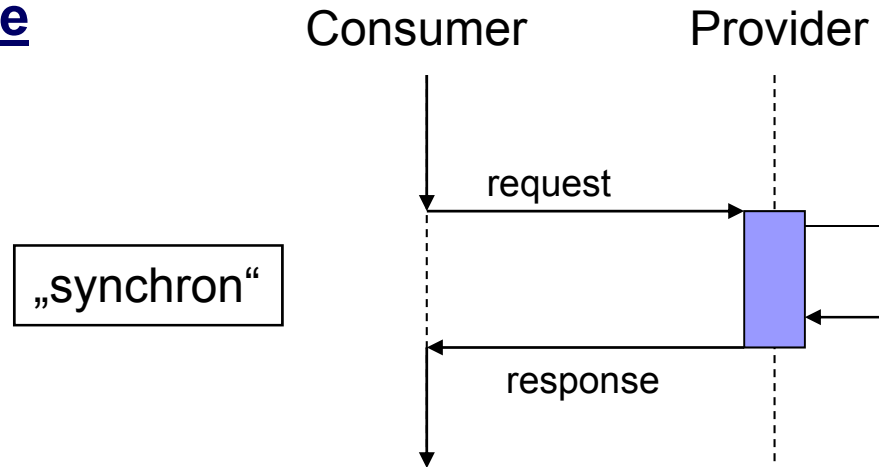
Die Nachricht konnte vom Empfänger nicht verarbeitet werden

Message Exchange Patterns (1)

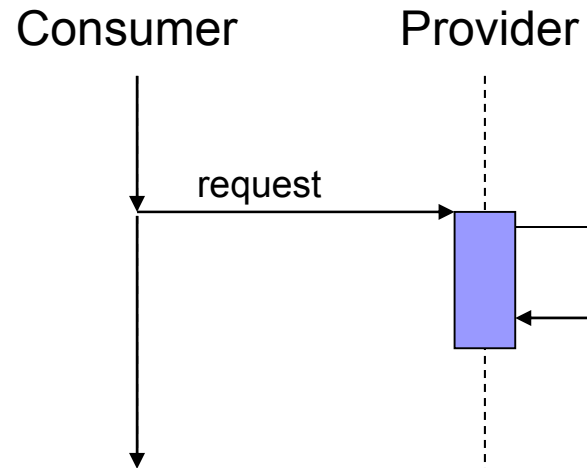
- Nachrichten werden in einer festgelegten Abfolge verschickt, sog. **Message Exchange Patterns** (MEP)
- Die wichtigsten Patterns sind:
 - Request / Response
 - One-Way
 - Request / Callback
 - Publish / Subscribe

Message Exchange Patterns (2)

Request / Response

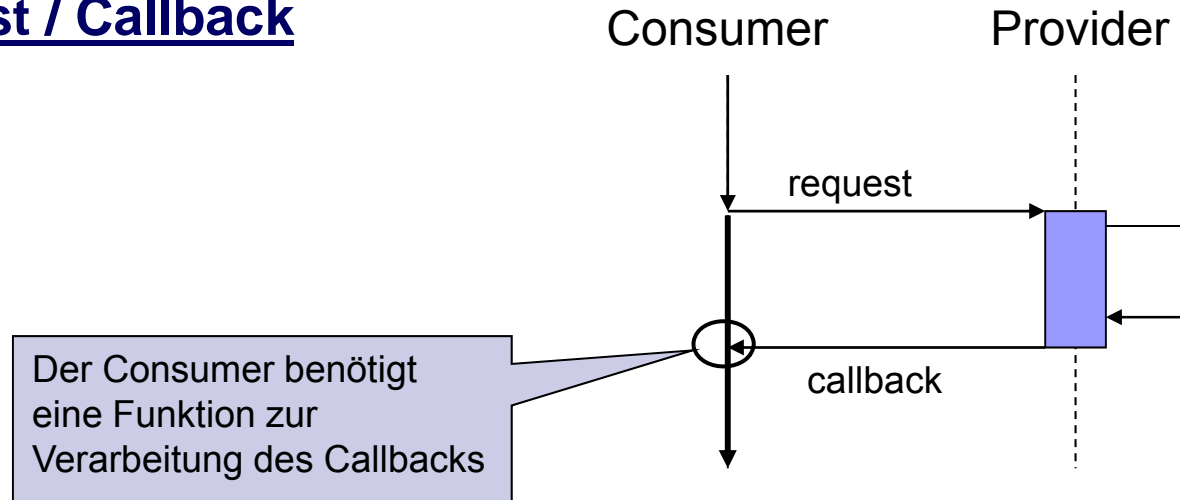


One-Way

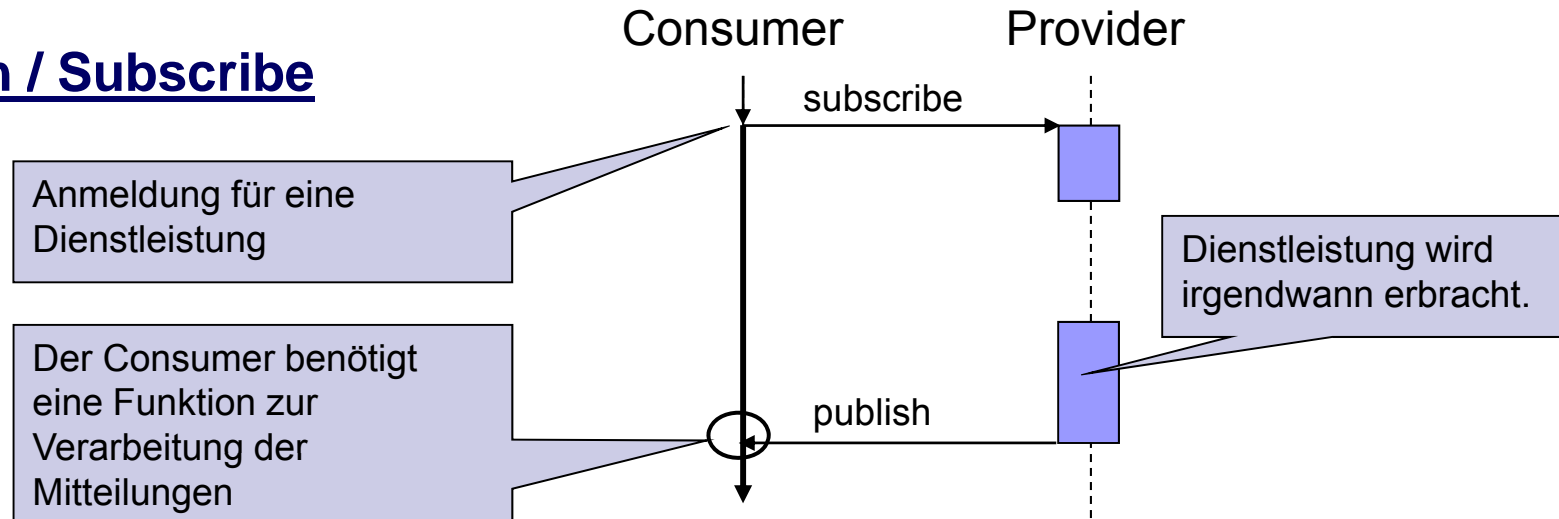


Message Exchange Patterns (3)

Request / Callback



Publish / Subscribe

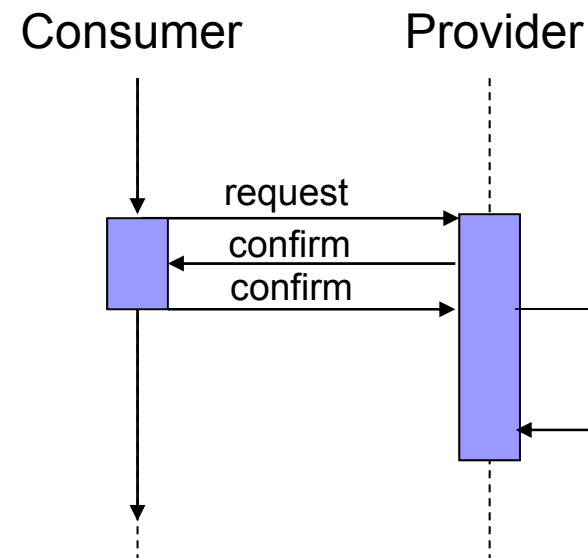


Vergleichbare MEPs in der SOAP-Spezifikation

- **In-Only:** One-Way-Nachricht vom Consumer zum Provider
- **Robust-In-Only:** One-Way-Nachricht mit Eingangsbestätigung
- **In-Out:** Request mit erwarteter Response
- **In Optional-Out:** Request mit optionaler Response

Beispiel: Robust-In-Only

- Stellt sicher, dass eine One-Way-Nachricht tatsächlich ankommt
- Doppelte Bestätigung erforderlich



Beschreibung der Schnittstelle und Funktion eines Services

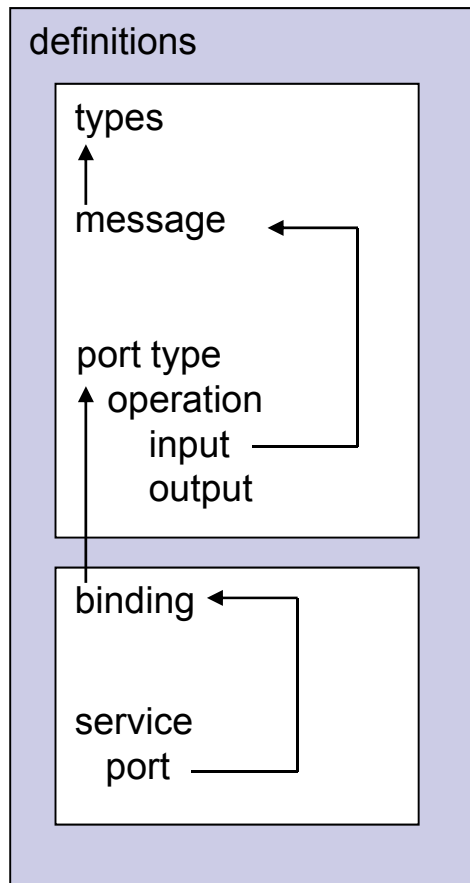
- Um einen Web Service nutzen zu können, muss eine Beschreibung der Service-Schnittstelle vorliegen
- Beschreibung geschieht mit **WSDL** (Web Service Definition Language)
- Weit verbreitet ist die Version 1.1, aktuell ist die Version 2.0
- Eine WSDL-Datei enthält:
 - Beschreibung der angebotenen **Operationen**
 - **Protokoll**, mit dem der Service aufgerufen werden kann
 - **Adresse**, über die der Service erreichbar ist
- Anhand einer WSDL-Beschreibung kann automatisch das Gerüst einer SOAP-Nachricht generiert werden

Servicebeschreibung (2)

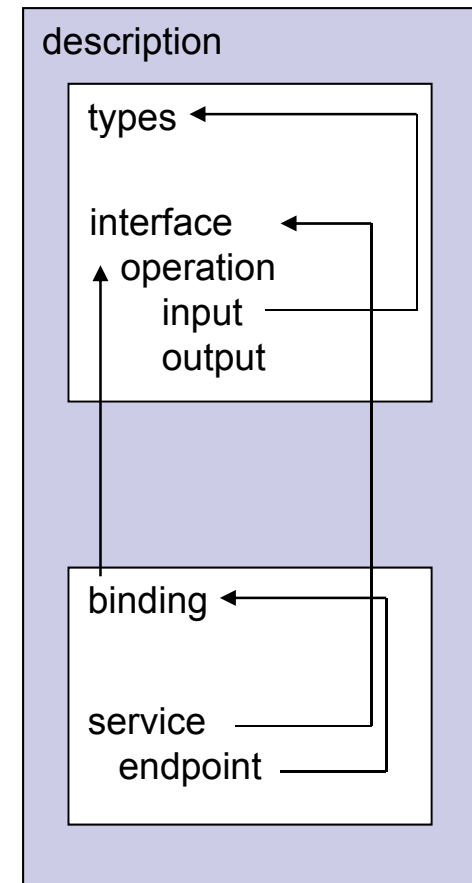
Struktur einer WSDL-Datei

verwendet →

WSDL 1.1



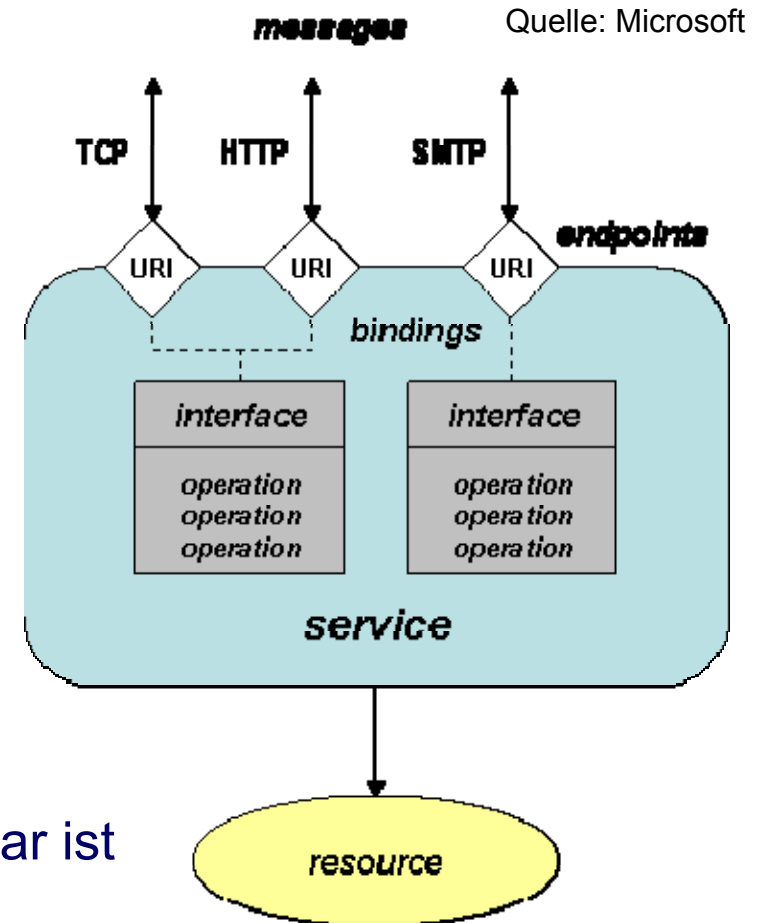
WSDL 2.0



Servicebeschreibung (3)

Bedeutung der einzelnen Elemente

- **types**
Definition verwendeter Datentypen
- **portType** bzw. **interface**
Beschreibung der Operationen einer Schnittstelle
- **binding**
Informationen über das zu verwendende Protokoll
- **service / port** bzw. **service / endpoint**
Adresse, unter der der Service erreichbar ist



Nicht alle Informationen liegen zum gleichen Zeitpunkt vor.

WSDL-“Lebenszyklus“

Teil der WSDL-Datei	Zeitpunkt der Integration
Service-Schnittstelle	Design, Entwurf
Service-Detaillierung z.B. Types	Desing, Implementierung
Protokollinformationen	Implementierung, Systemkonfiguration
Adressinformationen	Deployment

Servicebeschreibung (5)

Quelle: Josuttis, SOA in der Praxis

```
<types>
  <xsd:schema
    targetNamespace="http://soa-in-der-praxis.de/xsd"
    xmlns="http://soa-in-der-praxis.de/xsd">

    <xsd:element name="lieferAdresse">
      <xsd:complexType>
        <xsd:sequence>
          <xsd:element name="kundennummer" type="xsd:long"/>
        </xsd:sequence>
      </xsd:complexType>
    </xsd:element>

    <xsd:element name="lieferAdresseResponse" type="adresse"/>
    <xsd:complexType name="adresse">
      <xsd:sequence>
        <xsd:element name="strasse" type="xsd:string"/>
        <xsd:element name="ort" type="xsd:string"/>
        <xsd:element name="plz" type="xsd:string"/>
      </xsd:sequence>
    </xsd:complexType>

  </xsd:schema>
</types>
```

Beschreibung eines
Typs für den Aufruf

Beschreibung eines
Typs für das Ergebnis

Servicebeschreibung (6)

```
<interface name="KundeInterface">
```

Aufzurufende Methode

```
  <operation name="lieferAdresse"
    pattern="http://www.w3.org/2006/01/wsdl/in-out"
    style="http://www.w3.org/2006/01/wsdl/style/iri"
    wsdlx:safe="true">
```

```
    <input messageLabel="In"
      element="xsd1:lieferAdresse"/>
```

Aufruf-Parameter

```
    <output messageLabel="Out"
      element="xsd1:lieferAdresseResponse"/>
```

Ergebnis-Parameter

```
  </operation>
```

```
</interface>
```

Servicebeschreibung (7)

```
<binding name="KundeSOAPBinding"
  interface="tns:KundeInterface"
  type="http://www.w3.org/2006/01/wsdl/soap"
  wsoap:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP">

  <operation ref="tns:lieferAdresse"
    wsoap:mep="http://www.w3.org/2003/05/soap/mep/soap-response"/>

</binding>

<service name="KundeService"
  interface="tns:KundeInterface">

  <endpoint name="KundeEndpoint"
    binding="tns:KundeSOAPBinding"
    address="http://soa-in-der-praxis.de/kunde20"/>

</service>
```

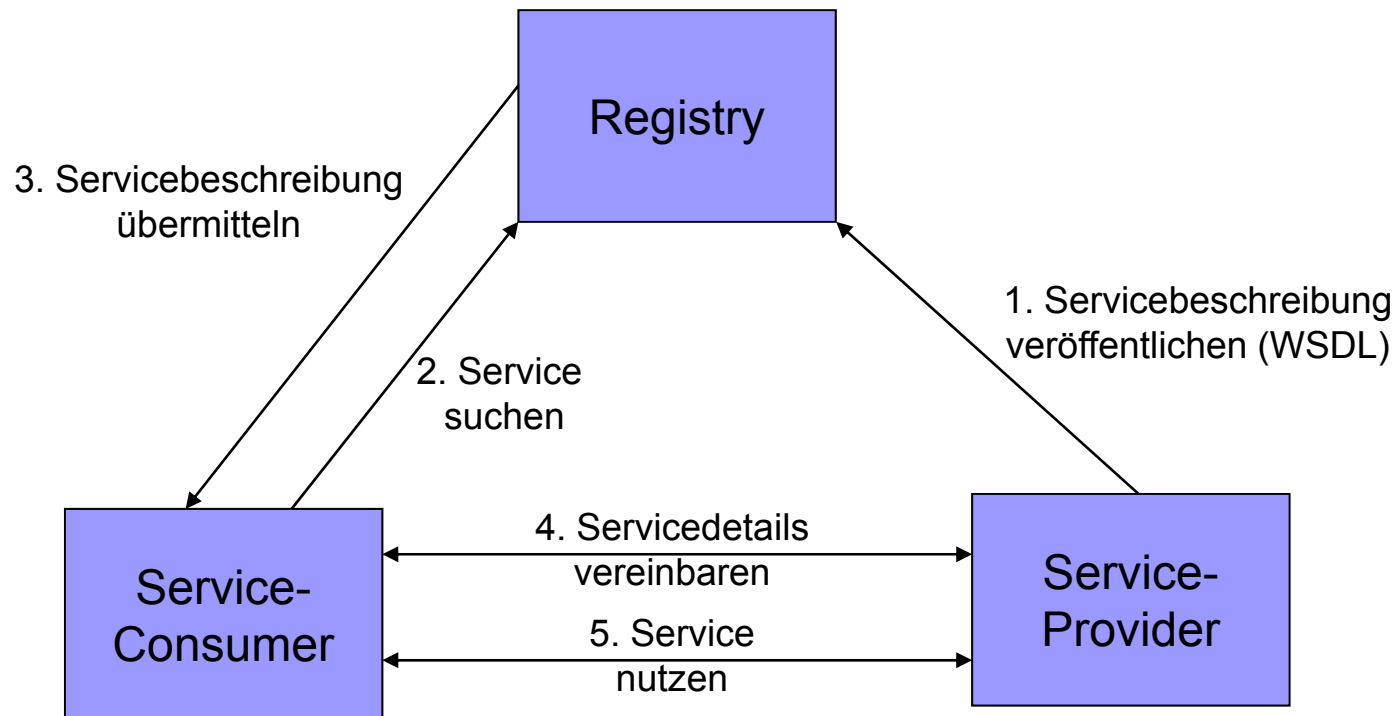
Protokoll:
SOAP mit HTTP

MEP:
Request / Response

Adresse für den Aufruf

- WSDL ist primär für die technische Beschreibung des Services geeignet (Signaturen, Parameter, Adressen).
 - Darüber hinausgehende Eigenschaften wie z.B. Verfügbarkeit, Performance, Vor- und Nachbedingungen sowie Zugriffsrechte sind nicht darstellbar.
 - Verbesserungen in der Version 2.0 ermöglichen die semantische Erweiterung von WSDL-Dateien.
 - Derzeit werden weitere Dokumente (DOC, PDF, EXCEL, ...) für die Servicebeschreibung zur Verfügung gestellt.
- Ohne Erweiterungen von WSDL-Dateien kommt man in der Praxis nicht aus.

Veröffentlichung von Service-Informationen mit Hilfe einer Registry



UDDI

- Verbreiteter Standard, derzeit Version 3.0
- Speicherung von WSDL-Informationen und zusätzlicher Service-Beschreibungen
- Größter Schwachpunkt:
Detailinformationen zu Services werden nur unzureichend repräsentiert, z.B. Service Level Agreements (SLAs) oder Sicherheitsanforderungen
→ für komplexe SOA nicht geeignet
- Stattdessen:
Verwendung von sog. **Registry / Repositorys** wie z.B. ebXML, in dem detaillierte Informationen abgelegt werden können.



Ein Verzeichnisdienst wird im Anfangsstadium einer SOA nicht zwingend benötigt, da die Services noch überschaubar sind

- **Performance**

Durch Verwendung von XML viel Kommunikationsoverhead

→ nicht für Kleinstoperationen geeignet

→ nur eingeschränkt für performance-kritische Aufgaben geeignet

- **Sicherheit**

Die Standardtechnologie beinhaltet keine Sicherheitsmechanismen. Solche Mechanismen müssen durch zusätzliche Maßnahmen realisiert werden.

- **Management**

Regelung der Nutzung eines Services noch nicht vollständig gelöst

- **Art der technischen Lösung**

Aufgrund von technischen Restriktionen können die Vorteile von Web Services nicht immer erreicht werden

XML-RPC

- Basiert ebenfalls auf XML
- Verwendet einfachere Strukturen
- Kennt nur wenige Datentypen
- Jedoch nicht standardisiert

→ Nur für einfache Aufgaben geeignet

Spezialisierte Standards

- **ebXML** (enterprise business XML)
→ als Erweiterung bzw. Ersatz für EDI im Industriebereich
- **ACORD XML**
(Association for Cooperative Operations Research and Development)
→ Speziell für die Versicherungsbranche
- **RosettaNet**
→ Speziell für Supply Chain Management

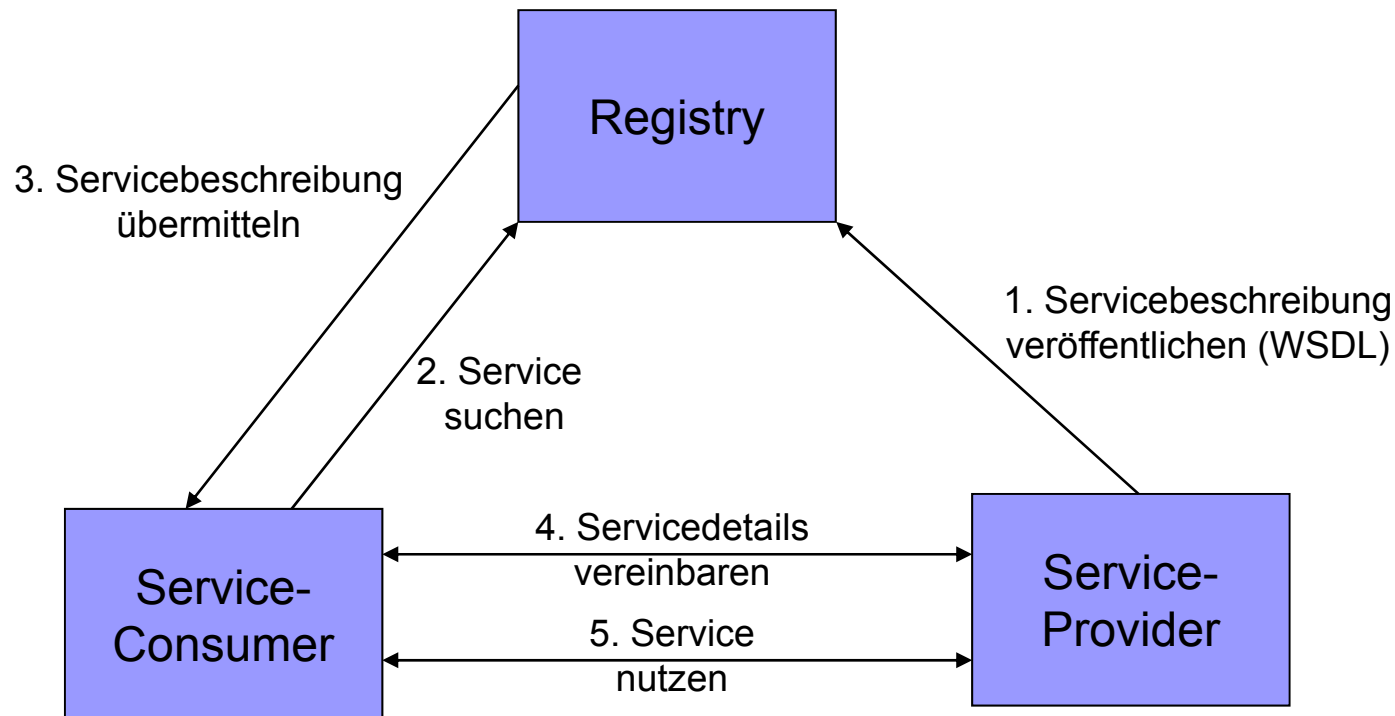
REST (REpresentational State Transfer)

- Aufruf des Services über eine URL bzw. über die HTTP-Standardbefehle GET, POST, PUT, DELETE
 - Alle Parameter sind in der URL enthalten
 - Keine Datentypisierung
 - Format der Antwort ist beliebig, in der Regel aber XML
- Sehr einfach aufzurufen und zu nutzen
- Auswertung des zurückgelieferten Ergebnisses muss vollständig manuell codiert werden

Beispiel:

Amazon Web Services (aws.amazon.com)

Service-Veröffentlichung und Service-Discovery mit Hilfe einer Registry



UDDI (Universal Description, Discovery & Integration)

- Verbreiteter Verzeichnisstandard, derzeit Version 3.0
- Vorläuferversionen sind zum Teil noch im Einsatz

Aufgaben

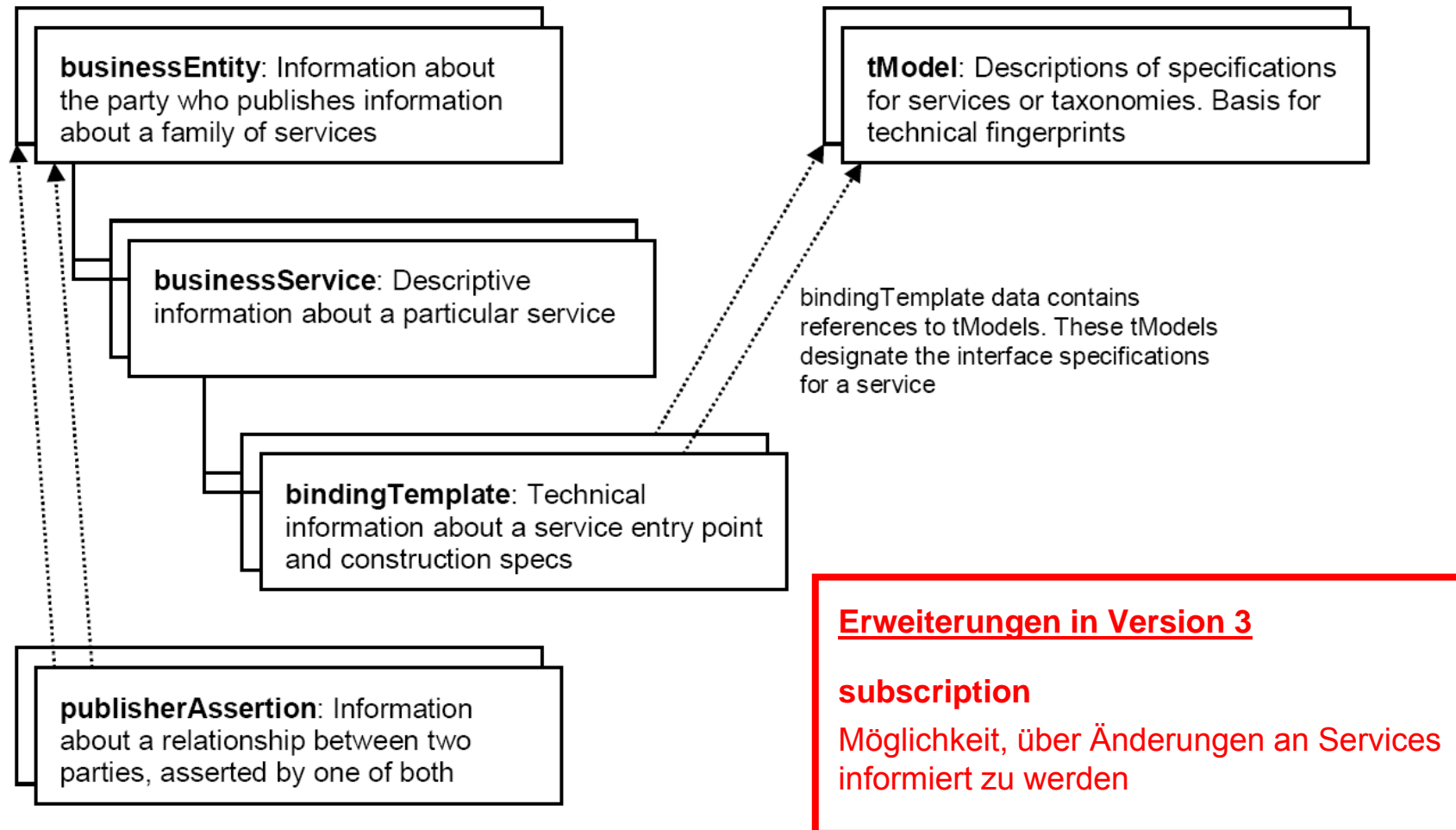
- Registrierung von WSDL-Informationen und Metadaten zu Services
- Recherchefunktionen für die Abfrage der WSDL- und Meta-Informationen

Umsetzung

- Definierte Datenstrukturen
- Definierte API-Calls und die dazugehörigen SOAP-Messages

Quelle: <http://uddi.org/pubs/DataStructure-V2.03-Published-20020719.pdf>

UDDI-Strukturen



Struktur der businessEntity

Die businessEntity liefert Informationen über den Anbieter und dessen Angebot und ist evtl. der erste Einstieg, um einen Service zu finden.

```
<element name="businessEntity" type="uddi:businessEntity" />
<complexType name="businessEntity">
  <sequence>
    <element ref="uddi:discoveryURLs" minOccurs="0" />
    <element ref="uddi:name" maxOccurs="unbounded" />
    <element ref="uddi:description" minOccurs="0" maxOccurs="unbounded" />
    <element ref="uddi:contacts" minOccurs="0" />
    <element ref="uddi:businessServices" minOccurs="0" />
    <element ref="uddi:identifierBag" minOccurs="0" />
    <element ref="uddi:categoryBag" minOccurs="0" />
  </sequence>
  <attribute name="businessKey" type="uddi:businessKey" use="required" />
  <attribute name="operator" type="string" use="optional" />
  <attribute name="authorizedName" type="string" use="optional" />
</complexType>
```

Wo gibt es
weitere
Informationen?

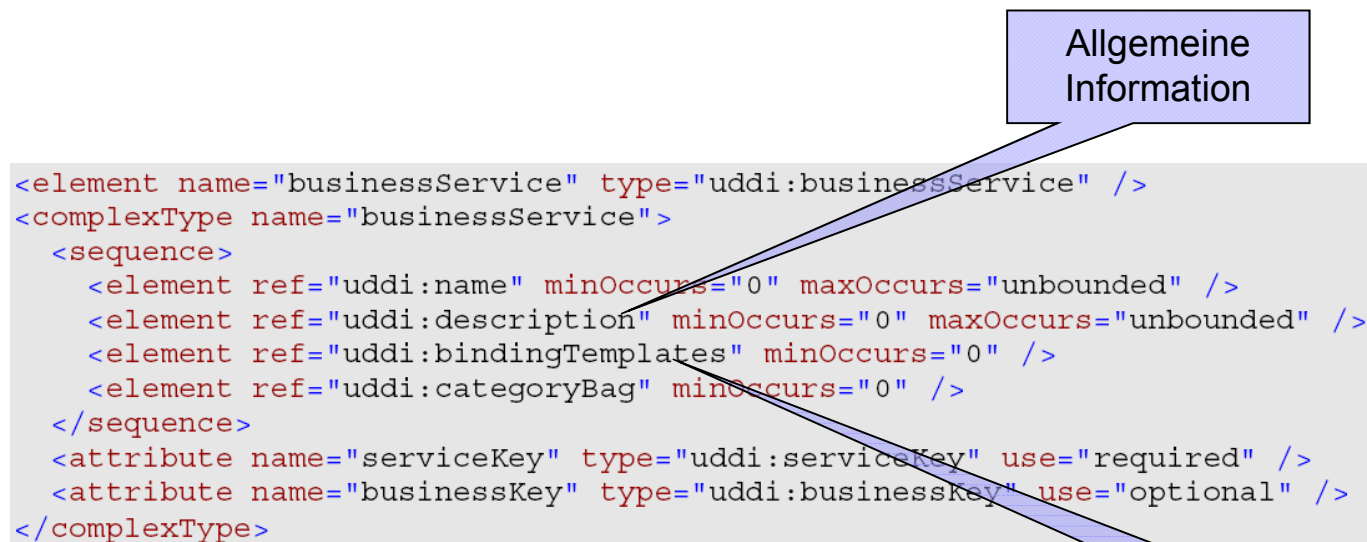
Angebote
Services

Suchbegriffe /
Kategorien

Quelle: UDDI Version 2.03 Data Structure Reference

Struktur des businessService

Der businessService liefert allgemeine Informationen über einen Service und den Verweis auf das bindingTemplate.



Quelle: UDDI Version 2.03 Data Structure Reference

Struktur des bindingTemplate

Das bindingTemplate liefert Informationen über die technische Erreichbarkeit des Services

```
<element name="bindingTemplate" type="uddi:bindingTemplate" />
<complexType name="bindingTemplate">
  <sequence>
    <element ref="uddi:description" minOccurs="0" maxOccurs="unbounded" />
    <choice>
      <element ref="uddi:accessPoint" />
      <element ref="uddi:hostingRedirector" />
    </choice>
    <element ref="uddi:tModelInstanceDetails" />
  </sequence>
  <attribute name="serviceKey" type="uddi:serviceKey" use="optional" />
  <attribute name="bindingKey" type="uddi:bindingKey" use="required" />
</complexType>
```

Adresse für den Service-Aufruf

Detailinformationen über den Service

Quelle: UDDI Version 2.03 Data Structure Reference

Beispiel für accessPoint:

<http://services.irgendwo.de/orderservice>

Struktur des tModels

Das tModel liefert inhaltliche Informationen über Funktion des Services, z.B. die WSDL-Information.

```
<element name="tModel" type="uddi:tModel" />
<complexType name="tModel">
  <sequence>
    <element ref="uddi:name" />
    <element ref="uddi:description" minOccurs="0" maxOccurs="unbounded" />
    <element ref="uddi:overviewDoc" minOccurs="0" />
    <element ref="uddi:identifierBag" minOccurs="0" />
    <element ref="uddi:categoryBag" minOccurs="0" />
  </sequence>
  <attribute name="tModelKey" type="uddi:tModelKey" use="required" />
  <attribute name="operator" type="string" use="optional" />
  <attribute name="authorizedName" type="string" use="optional" />
</complexType>
```

Kurz-
beschreibung

Überblicksinformationen
über den Service

Quelle: UDDI Version 2.03 Data Structure Reference

Beispiel für overviewDoc:

<http://services.irgendwo.de/orderservice.wsdl>

Anmelden von Services

- Publishers API
- Erfordert Authentifizierung durch den Publisher
- Verwendung von SOAP über HTTPS

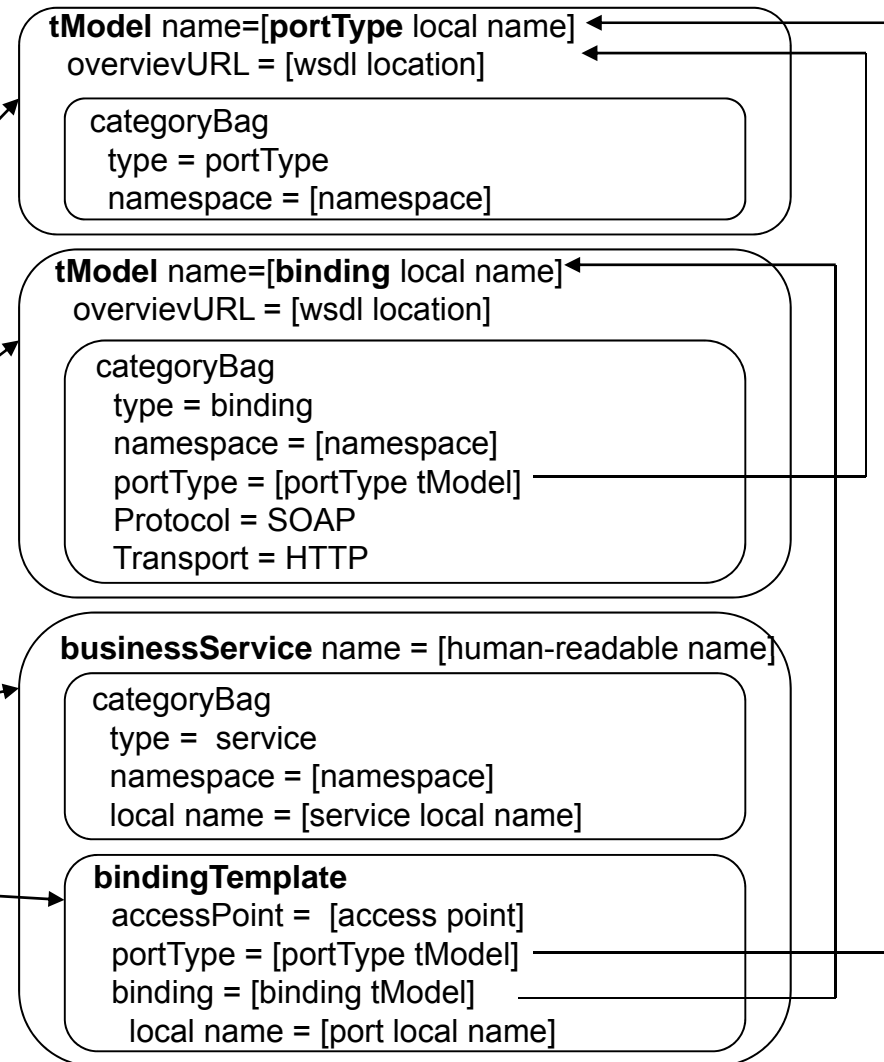
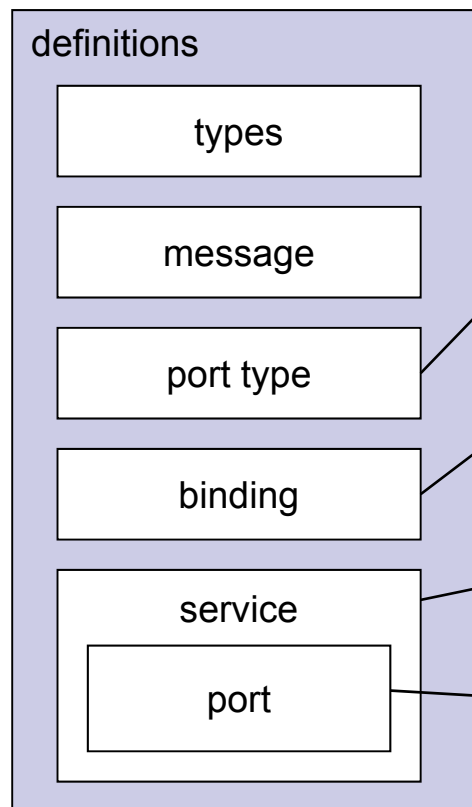
Beispielnachricht für Registrierung eines Bindings (SOAP-Body)

```
<save binding generic="1.0" xnkbs="urn:uddi-org:api">
  <authInfo>Microsoft Corporation</authInfo>
  <bindingTemplate>
    <description xml:lang="en">Production UDDI server,
      Publishing Interface</description>
    <accessPoint URLType="https">https://uddi.microsoft.com/publish</accessPoint>
    <tModelInstanceDetails>
      <tModelInstanceInfo tModelKey="UUID:64C756D1-3374-4E00-AE83-EE12E38FAE63" />
    </tModelInstanceDetails>
  </bindingTemplate>
</save_binding>
```

Neben dem binding muss auch noch das tModel veröffentlicht werden, auf das sich dieser Schlüssel bezieht

Abbildung von WSDL in UDDI

WSDL-Struktur



Quelle: Using WSDL in a UDDI Registry, Version 2.0.2,
<http://www.oasis-open.org/committees/uddi-spec/doc/tn/uddi-spec-tc-tn-wsdl-v2.htm>

Abfragen des Verzeichnisses

- **Browse Pattern**

Suche nach globalen Suchbegriffen und Verarbeitung der Suchergebnislisten
(Zeitpunkt: Entwurf oder Implementierung einer Anwendung)

- **Drill Down Pattern**

Suche mit exaktem Datenstrukturschlüssel, z.B. ServiceKey
(Zeitpunkt: Implementierung einer Anwendung)

- **Invocation Pattern**

Suche des bindings und des tModels eines bestimmten Services
(Zeitpunkt: Implementierung einer Anwendung oder zur Laufzeit)

Nachricht, um Daten zu einer Firma zu finden (SOAP-Body)

Beispiele von <http://msdn.microsoft.com/de-de/library/ms950813.aspx>

```
<find_business generic="1.0" xmlns="urn:uddi-org:api">
  <name>Microsoft</name>
</find_business>
```

Antwort

```
<businessList generic="1.0" operator="Microsoft Corporation"
truncated="false" xmlns="urn:uddi-org:api">
  <businessInfos>
    <businessInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3">
      <name>Microsoft Corporation</name>
      <description xml:lang="en">Empowering people through great software
- any time, any place and on any device is Microsofts vision. As
the worldwide leader ...</description>
      <serviceInfos>
        <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"
          serviceKey="D2BC296A-723B-4C45-9ED4-494F9E53F1D1">
          <name>UDDI Web Services</name>
        </serviceInfo>
        ...
      </serviceInfos>
    </businessInfo>
  </businessInfos>
</businessList>
```

businessKey
identifiziert
den Anbieter
von Services

Liste angebotener Services

Nachricht, um einen Service einer Firma zu finden (SOAP-Body)

```
<find_service generic='1.0' xmlns='urn:uddi-org:api'  
  businessKey='0076B468-EB27-42E5-AC09-9955CFF462A3'>  
  <name>UDDI Web Services</name>  
</find_service>
```

businessKey in der
Abfrage identifiziert
den Anbieter

Antwort

```
<serviceList generic="1.0" operator="Microsoft Corporation"  
  truncated="false" xmlns="urn:uddi-org:api">  
  <serviceInfos>  
    <serviceInfo businessKey="0076B468-EB27-42E5-AC09-9955CFF462A3"  
      serviceKey="D2BC296A-723B-4C45-9ED4-494F9E53F1D1">  
      <name>UDDI Web Services</name>  
    </serviceInfo>  
  </serviceInfos>  
</serviceList>
```

serviceKey identifiziert
eindeutig einen Service

Nachricht, um Servicedetails abzufragen (SOAP-Body)

```
<get_serviceDetail generic='1.0' xmlns='urn:uddi-org:api'>
  <serviceKey>D2BC296A-723B-4C45-9ED4-494F9E53F1D1</serviceKey>
</get_serviceDetail>
```

serviceKey in der
Abfrage identifiziert
den Service

Antwort

```
<serviceDetail generic="1.0" operator="Microsoft Corporation"
  truncated="false" xmlns="urn:uddi-org:api">
  ...
  <bindingTemplates>
    <bindingTemplate bindingKey="313C2BF0-021D-405C-8000-000000000000"
      serviceKey="D2BC296A-723B-4C45-9ED4-494F9E53F1D1"
      <description xml:lang="en">Production UDDI server,
      Publishing interface</description>
      <accessPoint URLType="https">https://uddi.microsoft.com/publish</accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="uuid:64C756D1-3374-4E00-AE83-EE12E38FAE63">
          <description xml:lang="en">UDDI SOAP Publication Interface</description>
        </tModelInstanceInfo>
      </tModelInstanceDetails>
    </bindingTemplate>
    ...
```

Liefert ein oder mehrere
bindingTemplates, da ein
Service mehrere
Operationen anbieten kann

Defizite von UDDI V3

- Die Meta-Informationen zu den Services werden nicht im Verzeichnis selbst abgelegt. Es gibt immer nur Verweise auf diese Informationen.
- Zugriffsrechte sind nur sehr grob definierbar, feingranulare Einstellungen sind teilweise sehr aufwendig
- Eine Standardisierung der Taxonomie (categories) ist nicht durchsetzbar bzw. eine falsche Kategorisierung wird nicht unterbunden



UDDI ist für eine einfache IT-Landschaft ausreichend.
Bei komplexen IT-Landschaften hat das Verfahren Grenzen.
Stattdessen können dann Registry Repositorys verwendet werden.

Registry Repositorys

- Verwaltung von Services aus fachlicher Sicht
- Meta-Informationen werden im Gegensatz zu einer reinen Registry im Repository gespeichert.
- Neuere Standards: Z.B. ebXML-Registry (OASIS/ISO-Standard)
- Herstellerspezifische Lösungen, z.B. WSSR von IBM

Alternative Lösungen

- Definiertes Verzeichnis im Filesystem, das Servicebeschreibungen enthält
- Web-Seiten, auf denen die Services dargestellt werden; dort wird dann auf die WSDL verwiesen
 - <http://seekda.com>
 - <http://www.xmethods.com/ve2/index.po>

Wesentliche Eigenschaften von Web Services

- Nutzung von Web Services geschieht überwiegend über SOAP over HTTP
- Beschreibungen von Services werden mit WSDL gemacht, die automatisiert (weiter-)verarbeitet werden können
- Hinterlegung der Servicebeschreibungen in Verzeichnisdiensten zur Abfrage zur Designzeit und zur Laufzeit

Ist damit das ursprüngliche Schnittstellenproblem gelöst?



Nur TEILWEISE!

Der Vorteil ist lediglich die Verwendung eines standardisierten Kommunikationsprotokolls

Wie sieht es mit der losen Kopplung aus?

→ Keine lose Kopplung, da HTTP ein „synchrones“ Protokoll ist

Was ist mit der Reduzierung der Schnittstellen?

→ Kaum Reduzierung der Schnittstellen, da nach wie vor Punkt-zu-Punkt-Verbindungen erforderlich sind

Was ist mit Systemen, die man nicht direkt SOAPen kann?

→ Adapter und Konvertierungsprogramme erforderlich

Wie steht es mit der Sicherheit?

→ Nur rudimentär bis überhaupt nicht gelöst



Für eine komplexe IT-Landschaft braucht man eine Kommunikationsinfrastruktur, die diese Probleme löst oder zumindest verringert!



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am _____

Serviceorientiertes eGovernment

Enterprise Service Bus (ESB), Business Process Management (BPM)

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Kommunikationsinfrastruktur

- Bei verteilten Systemen ist eine nachrichtenorientierte Infrastruktur sehr häufig anzutreffen
- Bisher waren die Lösungen in der Regel äußerst komplex und waren (leider) sehr stark herstellerabhängig
- Ein neuer Ansatz setzt auf offene Standards

→ **Enterprise Service Bus (ESB)**

Grundsatz:

Alle Komponenten der Anwendungslandschaft kommunizieren nur über die Infrastruktur und nicht mehr direkt (wie zum Beispiel bei Web-Services üblich).

Aufgaben eines ESBs

- Inhaltsbezogenes Weiterleiten von Nachrichten („content-based routing“)
- Transformation von Daten
- Flexible Steuerung von Prozessschritten eines (Teil-) Geschäftsprozesses („service orchestration“)
- Trennung der Transportlogik von der Schnittstellenimplementierung (und der Anwendungslogik)
- Monitoring von Schnittstellen und Geschäftsaktivitäten
- Sicherheitsfunktionen
- Sicherstellung eines bestimmten Qualitätsstandards der Service-Erbringung („quality of service“ [QoS])

Ausgangslage

- Heterogene IT-Landschaft mit zahlreichen unterschiedlichen Schnittstellen diverser Anwendungssysteme
- Nicht alle Systeme können und sollen als Web Services umgestaltet werden

Aufgaben

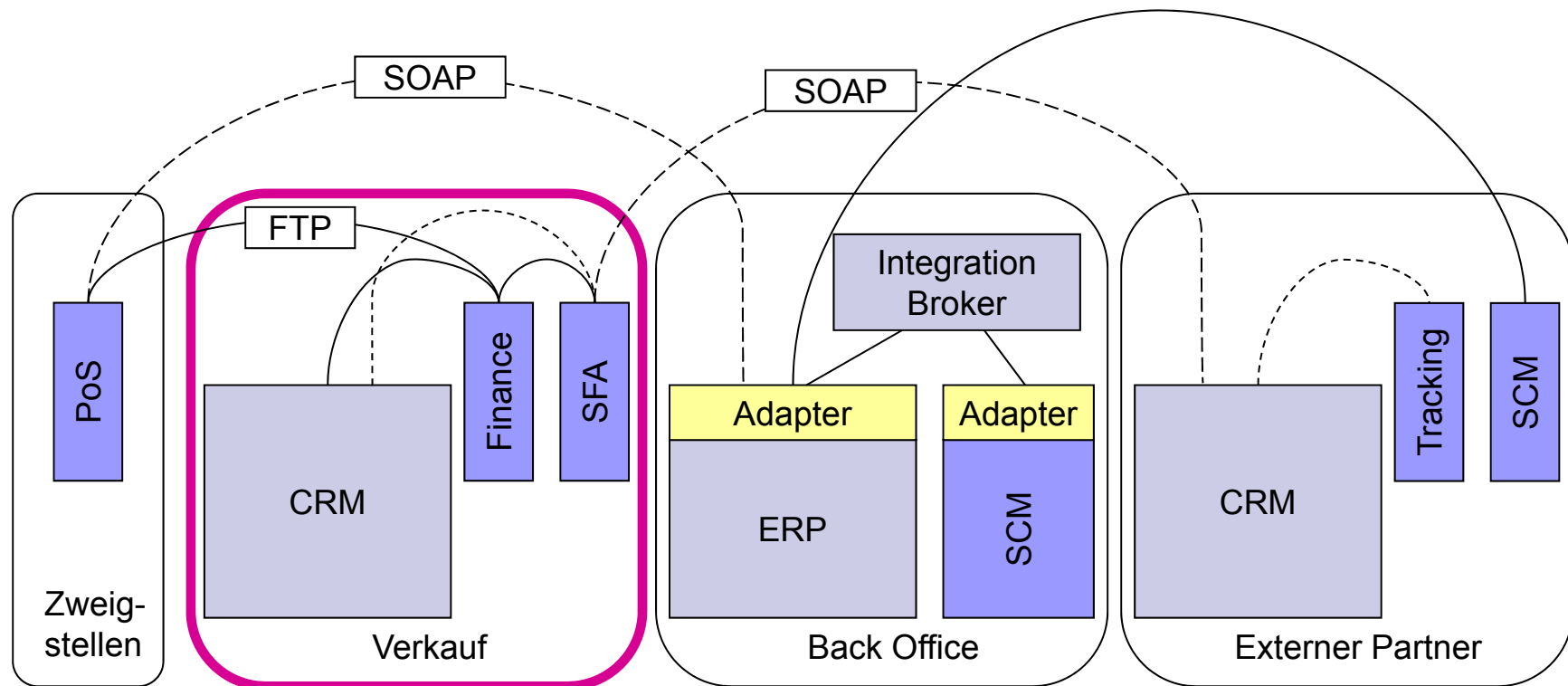
- Schrittweise Umgestaltung der IT-Landschaft
- Punkt-zu-Punkt-Verbindungen der bestehende Anwendungen möglichst transparent gestalten
- Kommunikationsprotokoll vereinheitlichen
- Kommunikationssicherheit verbessern
- Quality of Service erhöhen

Enterprise Service Bus (4)

Beispiel

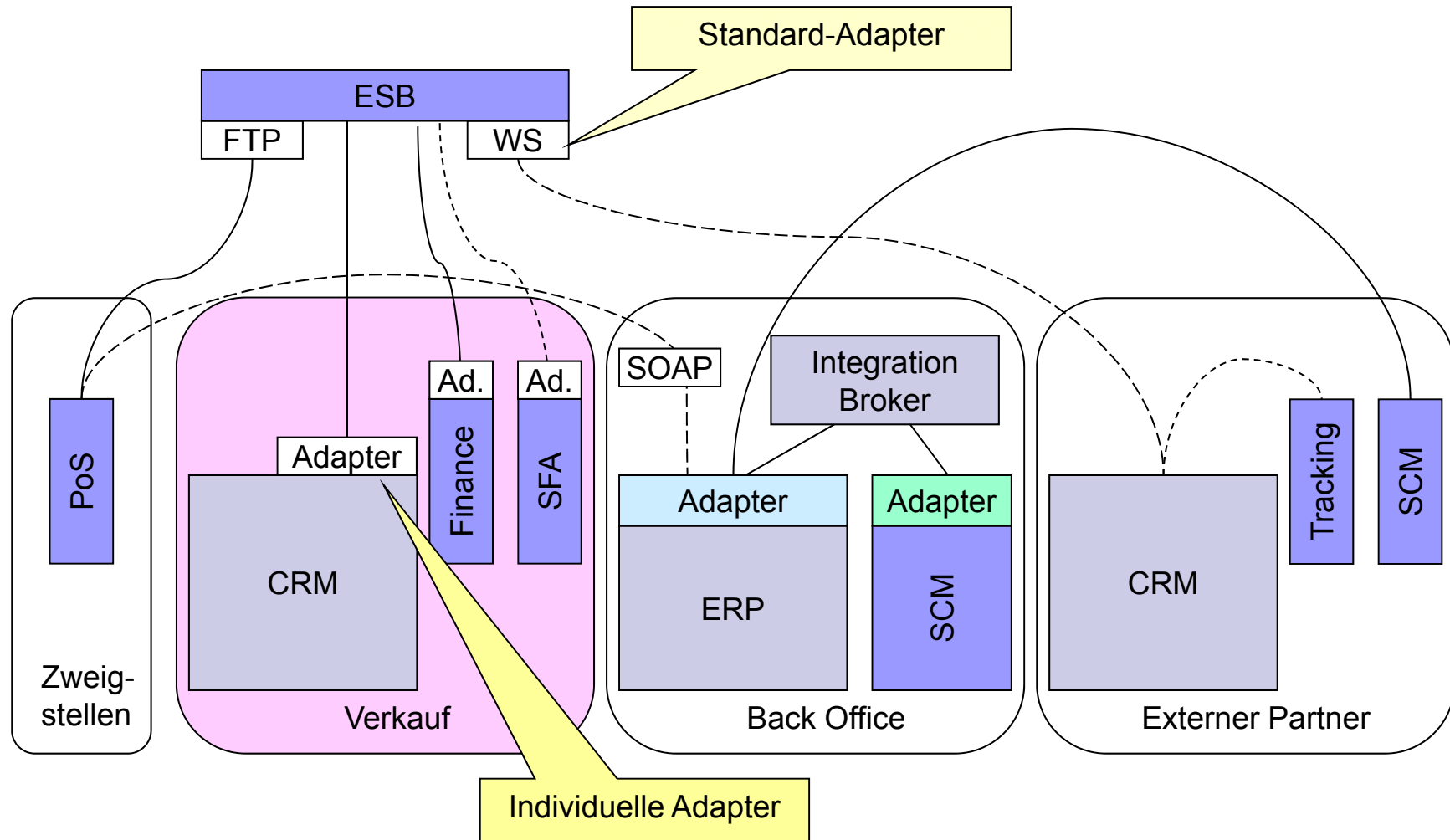
Quelle: D. Chapel, Enterprise Service Bus – Theory in Practice

IT-Landschaft mit unterschiedlichen Schnittstellentechnologien



Enterprise Service Bus (5)

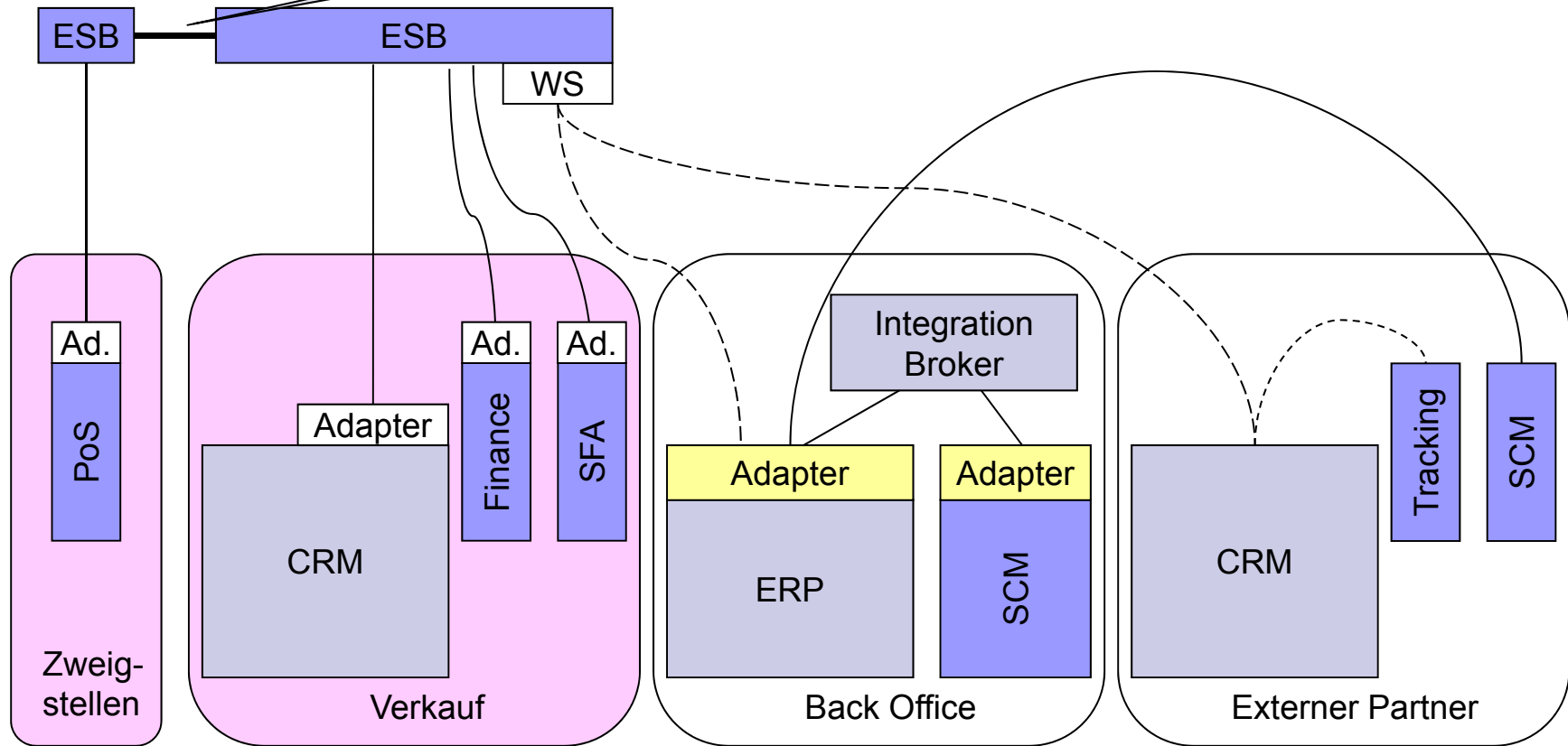
Veränderung der Einheit „Verkauf“



Enterprise Service Bus (6)

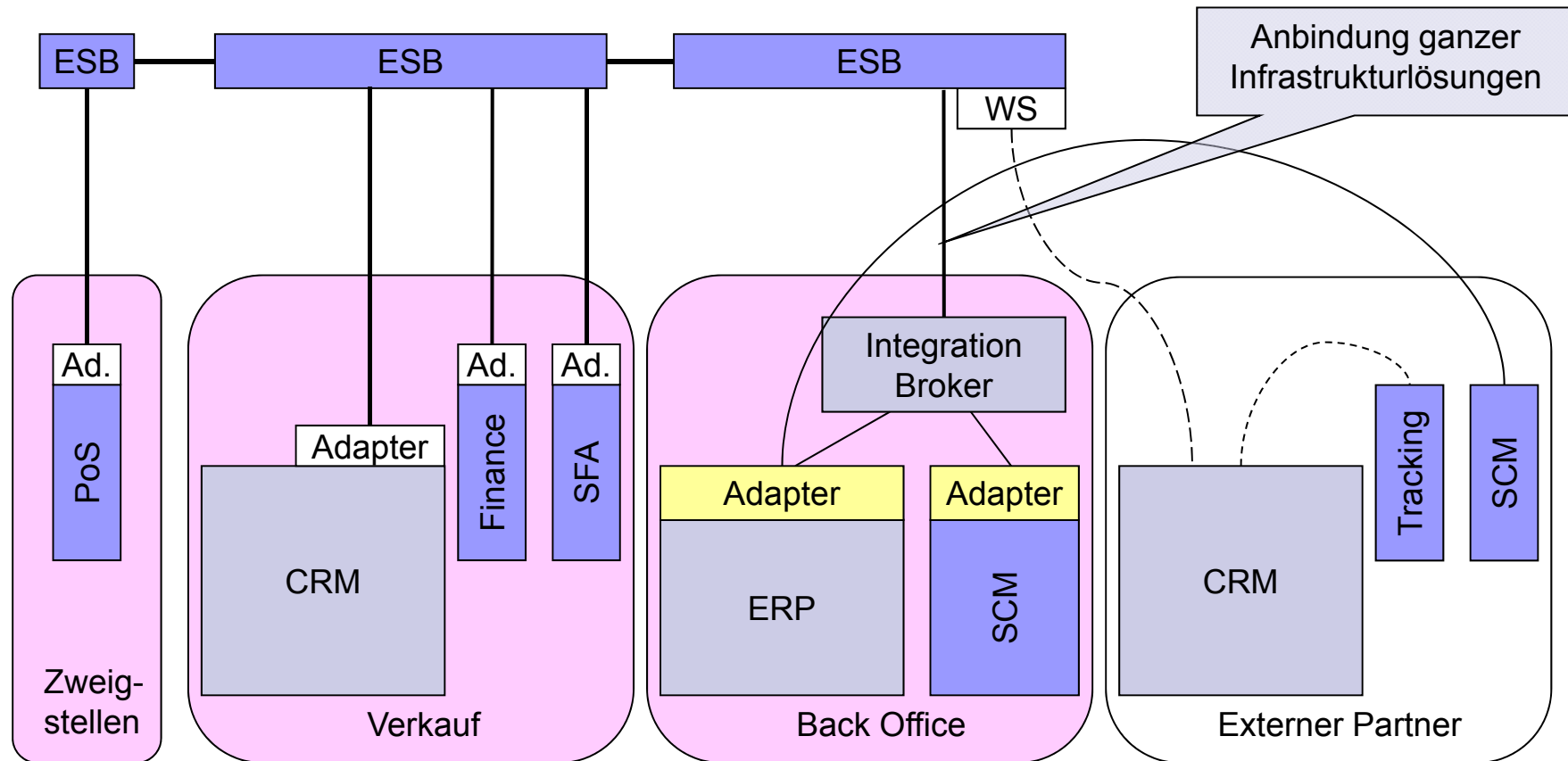
Veränderung der Einheit „Zweigstelle“

Verbindung mehrerer ESBs



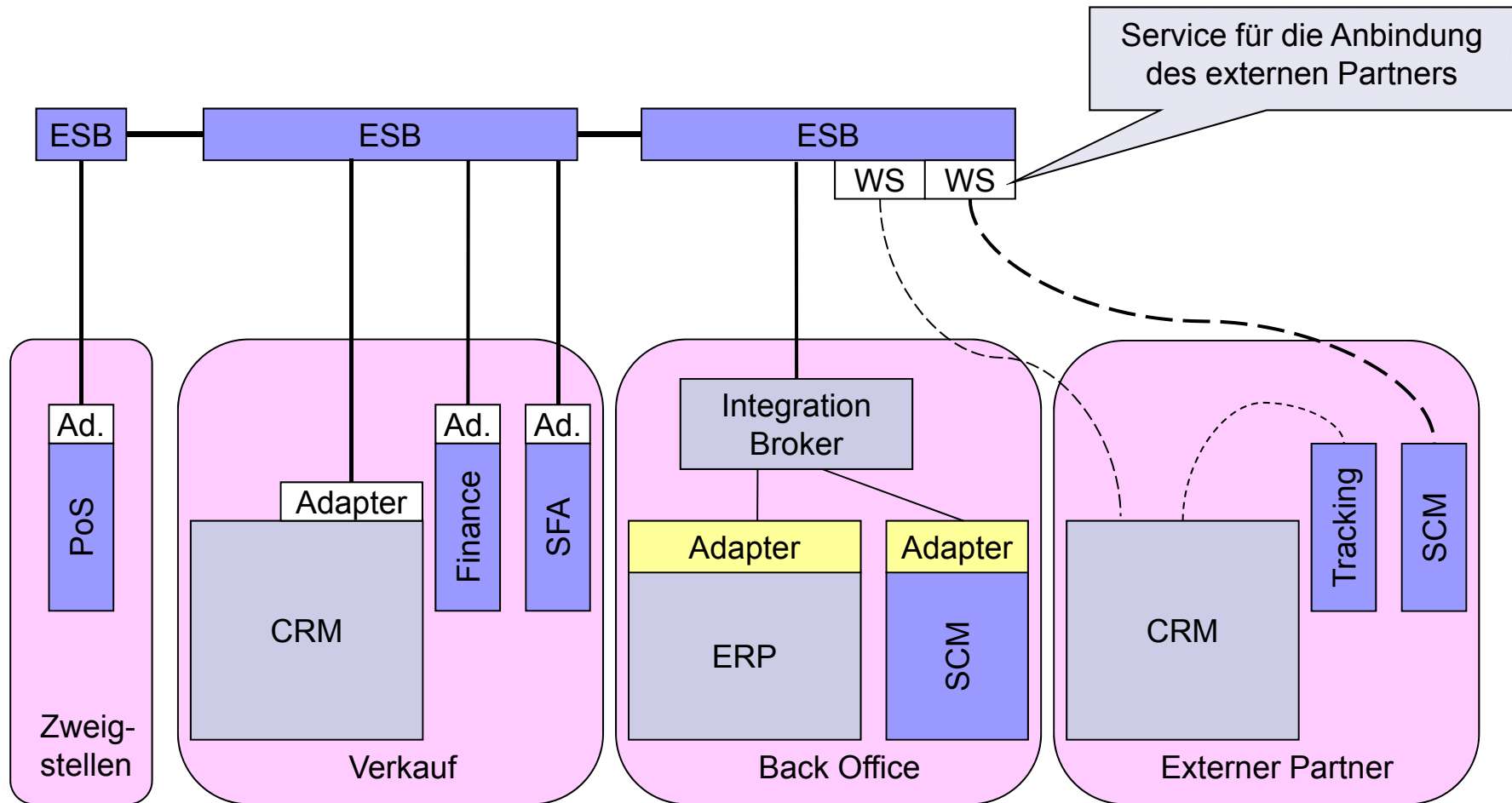
Enterprise Service Bus (7)

Veränderung der Einheit „Back Office“



Enterprise Service Bus (8)

Einrichtung weiterer Web Services „nach außen“



Wichtige Aspekte eines ESBs

- **Mehrere ESBs** können zu einem einzigen ESB **verknüpft** werden
- Innerhalb des ESBs wird in der Regel ein **einheitliches Kommunikationsprotokoll** verwendet, üblicherweise ein Protokoll, das auf XML basiert
- Der Datenaustausch muss über ein **nachrichtenorientiertes Kommunikationssystem** erfolgen, das folgende wesentliche Eigenschaften unterstützt:
 - **Sichere Zustellung** von Nachrichten (reliable messaging)
 - **Synchrone und asynchrone** Kommunikation
 - Unterstützung verschiedener **Message Patterns**
 - Zwischenspeicherung von **Prozesszuständen**
- Anwendungen werden entweder über Standardschnittstellen (Standardadapter) oder über individuelle Adapter angebunden.

Inhaltsbezogenes Weiterleiten von Nachrichten („content-based routing“)

- Daten sollen aufgrund bestimmter Dateninhalte unterschiedlich behandelt werden

Beispiel:

Bestellungen von Auslandskunden werden an ein anderes CRM geschickt als Bestellungen von Inlandskunden

Transformation von Daten

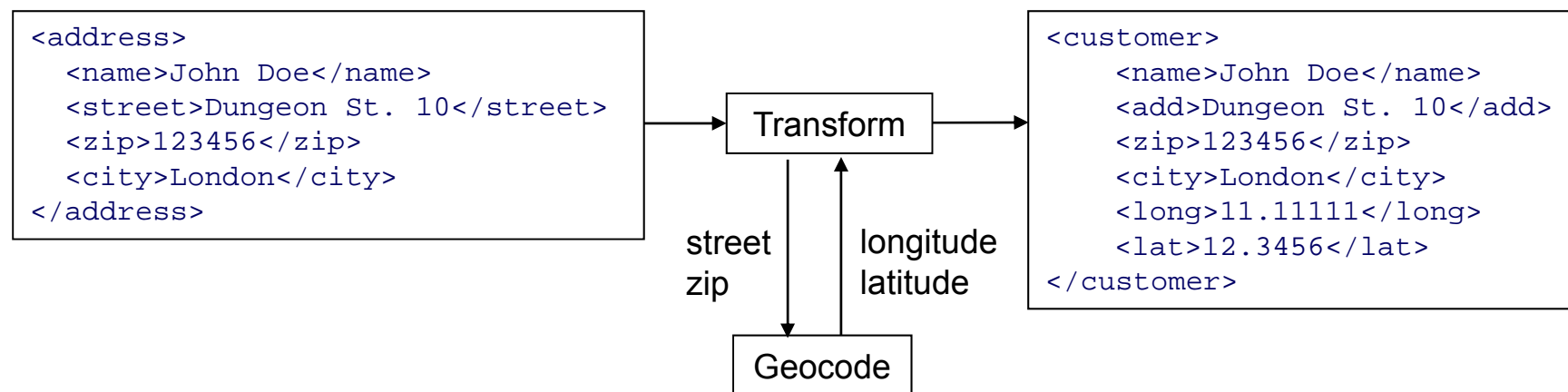
- Passen eingehende Daten z.B. nicht zur Datenstruktur des Zielsystems, muss ein Mapping und ggf. eine Ergänzung der Daten vorgenommen werden
- Die Ergänzung der Daten kann durch Aufruf weiterer Services erfolgen

Beispiel:

Die Ergänzung, welche Geolokalisationsdaten eine Adresse hat

- XML-Nachrichten werden häufig mit Hilfe von XSLT transformiert

Beispiel:



Verminderung von Schnittstellen durch einheitliches XML-Format

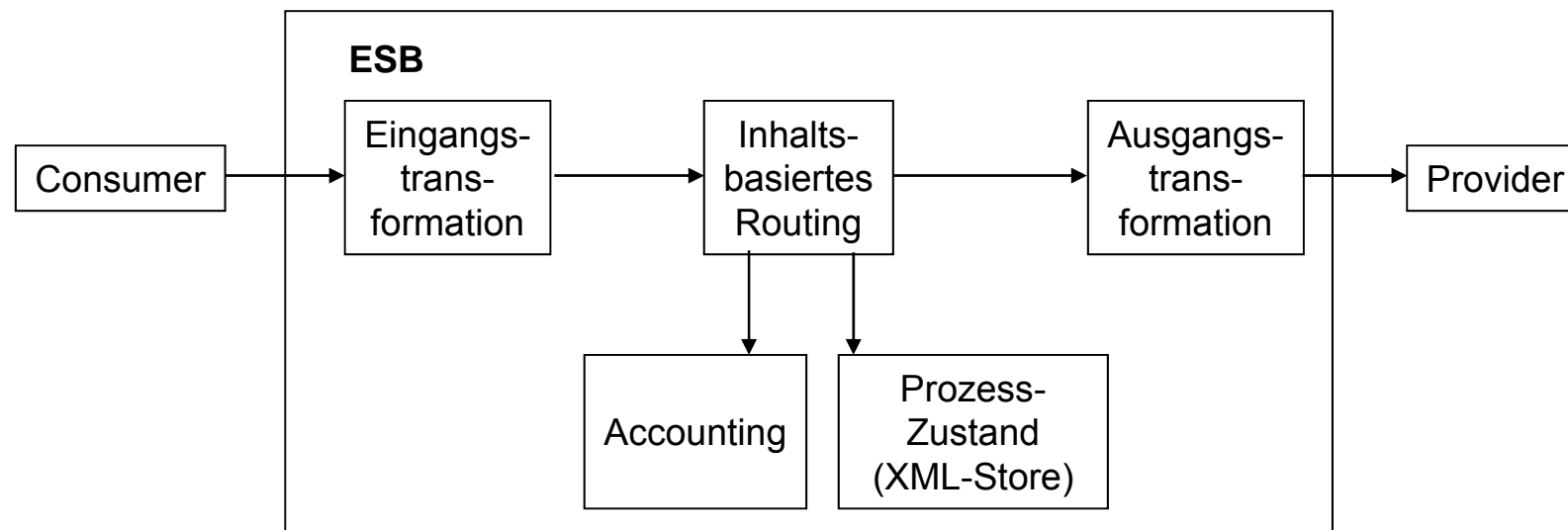
- Solange jede Einzelkommunikation zwischen Anwendungen individuell abgebildet wird, nimmt **die Anzahl der individuell zu konfigurierenden Schnittstellen** nicht ab
 - Abhilfe durch Schaffung eines **firmenweiten XML-Dialekts**, der die wesentlichen Dateninhalte definiert („kanonisches XML“)
 - Jede Nachricht wird vom ESB zunächst in dieses Format übersetzt und in dieser Form verschickt (**Eingangstransformation**).
 - Der Empfänger entnimmt daraus die Informationen, die er braucht (**Ausgangstransformation**).
- **Verringerung der Anzahl** der individuellen Schnittstellen
- **Verbesserte Übersichtlichkeit**, da Schnittstellen zentral definiert werden
- Evtl. **Aufblähung des Datentransfervolumens** und daraus resultierende Performanceprobleme

Steuerung von Prozessschritten eines (Teil-) Geschäftsprozesses („service orchestration“)

- Die Schritte eines Geschäftsprozesses können über eine **Abfolge von Services**, die durch den ESB gesteuert werden, abgebildet werden
- Bei komplexer Steuerung ist der Einsatz einer **Business Rules Engine** üblich, die die Schrittfolge inhaltsbezogen steuert
- Alternativ kann auch ein „**Orchestrierungsservice**“ verwendet werden
- Häufig ist die Zwischenspeicherung von Prozesszuständen erforderlich (z.B. mit Hilfe von XML-Storage, um Persistenz zu erreichen)

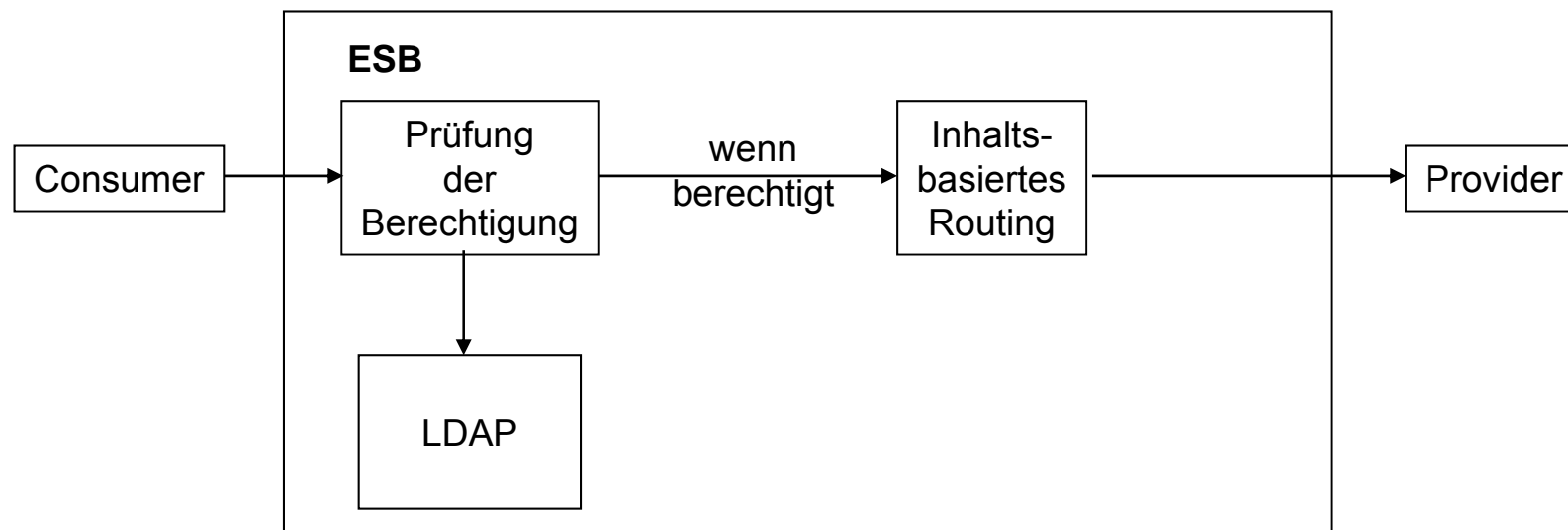
Monitoring von Schnittstellen und Geschäftsaktivitäten

- Wichtige Schnittstellen müssen hinsichtlich Auslastung, Verfügbarkeit und möglichen Fehlern **überwacht** werden (→ Kontrolle der Quality of Service)
- Für die Kostenrechnung könnte die **Häufigkeit von Serviceaufrufen** protokolliert werden
- Überwachung des Zustands von Geschäftsprozessen durch sog. **Business Activity Monitoring (BAM)**



Sicherheitsfunktionen

- Es muss sichergestellt sein, dass nur berechnigte Benutzer auf einen Service zugreifen
- Sicherheitsinformationen müssen in jeder Nachricht berücksichtigt werden (→ WS-Security oder andere Methoden wie z.B. LDAP)



Sicherstellung eines bestimmten Niveaus an Quality of Service

- Prüfung der Verfügbarkeit und der Performance durch das Monitoring
- Vorrang für bestimmte Dienste aufgrund einer vorgegebenen Priorisierung
- Transparentes Routing zu Ersatzsystemen, wenn Services ausfallen (evtl. auch durch Hardwaremechanismen)
- Transparente Verteilung der Last auf verschiedene Systeme
- Transparentes Hinzuschalten ergänzender Systeme, wenn die bestehenden Systeme die Last nicht mehr verarbeiten können

Hub & Spoke-Architektur vs. verteiltes Routing

- Das Routing von Nachrichten kann entweder **zentral** oder **dezentral** gesteuert werden:
 - a) **Zentrales Routing** bedeutet, dass für alle Nachrichten, die über den ESB geschickt werden, eine zentrale Instanz die Routing-Entscheidungen trifft (sog. **Hub & Spoke-Architektur**).
 - Evtl. hohe Belastung der zentralen Systeme
 - b) Beim dezentralen Routing (**itinerary-based routing**) enthält die Nachricht selbst die notwendigen (Routing-)Informationen, um zum nächsten Kommunikationspartner weitergeleitet zu werden.
 - Kein Engpass bei einem zentralen System
 - Verfeinerung des Konzepts der losen Kopplung

Wichtige Aspekte im Hinblick auf Web Services

- Lösung für das Problem der synchronen Aufrufe des HTTP-Protokolls
 - Request-Nachrichten werden vom ESB akzeptiert und zwischengespeichert (für den Consumer ist der Aufruf damit erledigt)
 - Der ESB sorgt dann dafür, dass die Nachricht zuverlässig an den Empfänger übermittelt wird (→ reliable messaging)
- Entkopplung der Punkt-zu-Punkt-Verbindungen
 - Zugangspunkte für die Web Services sind nicht mehr die tatsächlichen physischen Zugangspunkte der Services, sondern die Zugangspunkte des ESBs!
- Bereitstellung von fehlenden Funktionalitäten zur Zugriffskontrolle und im Hinblick auf die Sicherheit durch den ESB

Enterprise Service Bus (19)

Beispiel eines ESB-Produkts: Oracle

The image displays two screenshots related to an Oracle Enterprise Service Bus (ESB) product. The left screenshot shows the Oracle JDeveloper IDE with a service configuration for 'CustomerData.esb'. The configuration consists of several components: 'CustIn', 'CustIn_RS', 'CustOut_RS', 'CustDBOut', 'merge', 'write', 'Kundendat...', and 'Write'. The right screenshot shows the Oracle Enterprise Manager 10g 'ESB Control' console. It displays a monitoring view for the 'CustIn' service, including a diagram of the service flow and a table of instances.

Startzeit	Instance-ID	Status
09.11.08 17:30:47	OWdpXEv_92_NdYCMERvAA...	✓
09.11.08 17:29:51	FfHm2F93DAfF02:xpOaGfA...	✗
09.11.08 17:11:19	Qk86AENBAtGj_6HGfOew...	✗
09.11.08 17:09:48	7VMu3p2BW5GcGFNHo6DEQ...	✗
09.11.08 16:06:12	ZBxvUJyqT43GfWDEHfCnw...	✗
09.11.08 15:50:01	8Dy3_OFyDpSq11T6oIQMg...	✓
09.11.08 15:36:05	mmXbforUPfNPv3_TUIJdVA...	✗
08.11.08 22:03:21	o2PIng_Gsg52U91:nZEDCg...	✗
08.11.08 20:53:45	NaX:PRyK86bUn0dCND5Zg...	✗
08.11.08 20:52:51	28cWPKXIOHC955sn12eYA...	✗

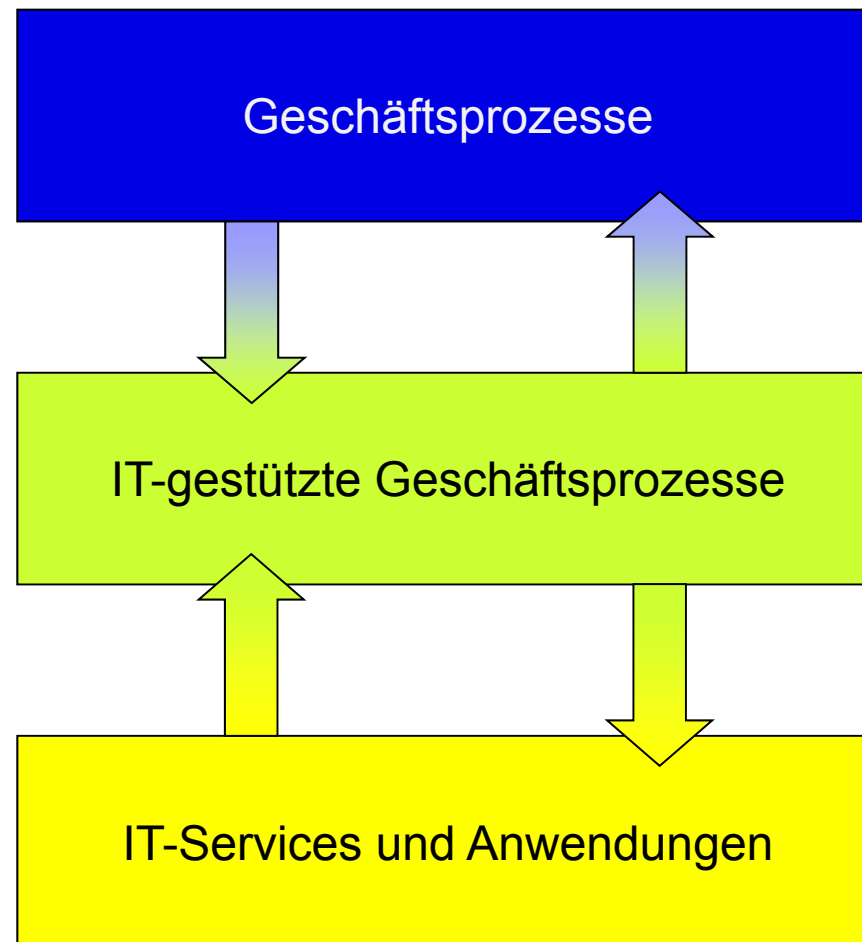
Geschäftsprozess

- Reale Aufgaben und damit verbundene Tätigkeiten
- Beschreibung eines Geschäftsprozesses:
 - nicht dokumentiert
 - informell dokumentiert
 - halbformal dokumentiert, z.B. durch einfache Arbeitsanweisung
 - formal dokumentiert, z.B. durch Ablaufplan für ISO-Zertifizierung
 - formal und in elektronisch verwertbarer Form dokumentiert (z.B. mit Hilfe eines Modellierungswerkzeugs)

IT-Services und Anwendungen

- Bestehende Systeme und deren Services, die „nach außen“ angeboten werden

Ziel



Top-Down

- Definition von Geschäftsprozessen und Entwicklung der dazu passenden IT-Services
 - In komplexen IT-Umfeldern unrealistisch, da bereits viele Anwendungen vorhanden sind, die genutzt werden müssen / sollen.

Bottum-Up

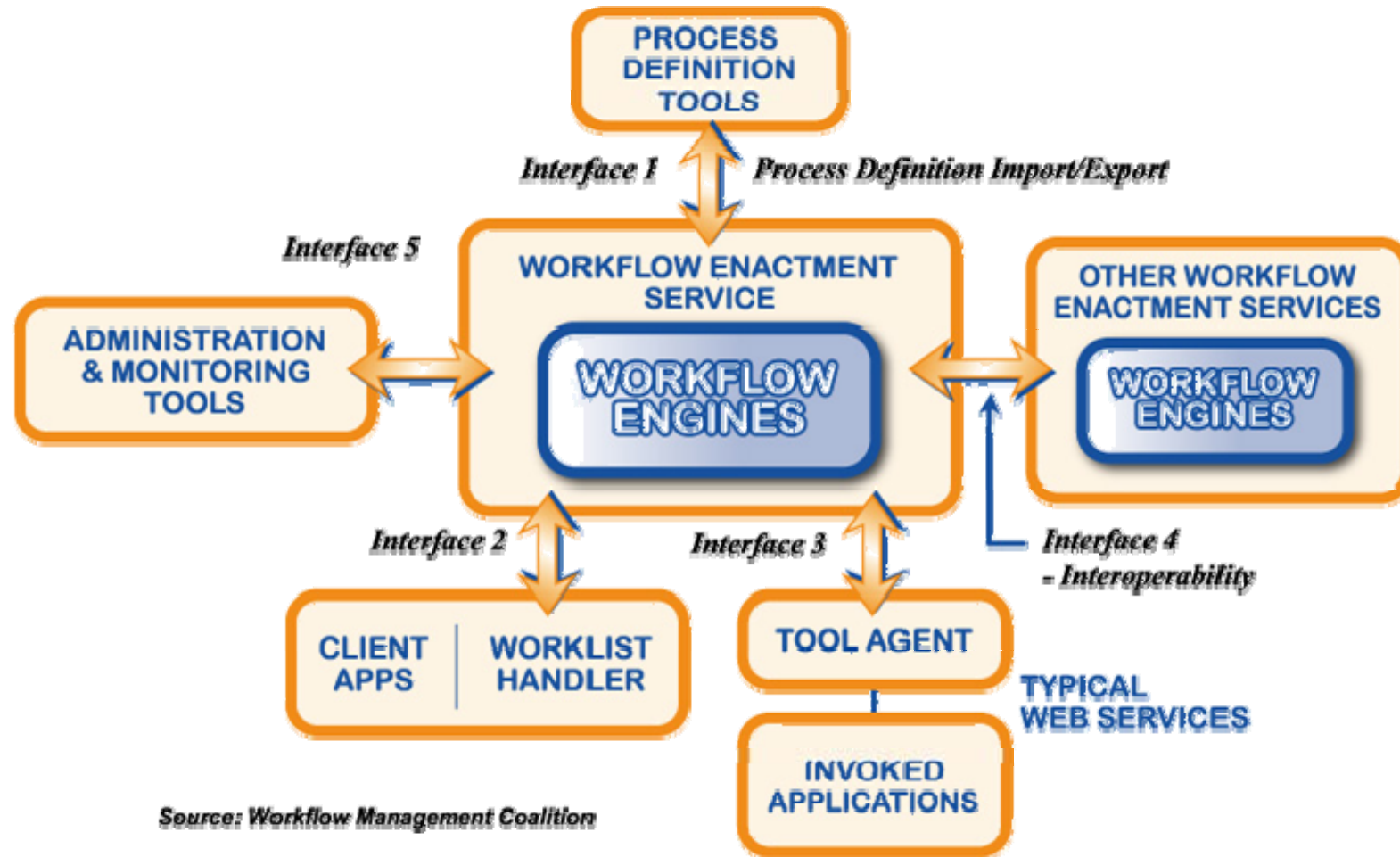
- Anpassung der Geschäftsprozesse an vorhandene Services und Anwendungen
 - Gefahr, mit veralteten Methoden weiterzuarbeiten

Kombination aus Top-Down und Bottom-Up

- Was soll erreicht werden?
- Welche Services und Anwendungen sind bereits vorhanden?
- Was kann davon verwendet werden?
- Was muss neu gemacht werden?
- Welche technische Randbedingungen gibt es, z.B. nicht mehr modifizierbare Systeme?
- Welche Funktionen sollen wieder verwendbar sein?
- ...

→ **Kreativer, iterativer Prozess**

Referenzmodell der Workflow Management Coalition (WfMC)



Workflow- oder Prozess-Engine?

Beide Engines bieten ähnliche Funktionalitäten:

- Verarbeitung von Prozessbeschreibungen
- Steuerung der angebundenen Dienste
- Speicherung von Verwaltungsinformationen
- Überwachung und Protokollierung

Was ist jedoch der Unterschied zwischen „Workflow“ und „Prozess“?

Ein Prozess beschreibt, **was** geschehen soll.

Ein Workflow beschreibt, **wie** es durchgeführt wird.

→ Diese Definition ist aber nicht durchgängig verwendet, so dass beides oft synonym verwendet wird.

Prozessbeschreibung (Process Description)

- An erster Stelle steht die Aufgabe, den Geschäftsprozess so zu beschreiben, so dass er sich auf einer Prozess- oder Workflow-Engine technisch ausführen lässt.

Für die Beschreibung wurden verschiedene Standards entwickelt

- WS-BPEL (früher BPEL4WS)
- BPEL4People
- BPML
- EPK / EPC
- XPD
- UML
- ...

Schnittstellen für Client-Anwendungsprogramme

- Aufruf von Prozessen / Teilprozessen
- Statusabfrage
- Benutzerinteraktion

Schnittstellen für die Anbindung von Services

- Nachrichtenaustausch
- Mapping verschiedener Protokolle und APIs
- Session Management

Schnittstellen zu anderen Workflow-Engines

- Aufruf externer Prozesse
- Anbindung von externen Systemen

Administrations- und Monitoring-Tools

- Benutzermanagement
- Rollenmanagement
- Protokollierung
- Ressourcenüberwachung
- Überwachung von Prozesszuständen

BPEL4WS bzw. WS-BPEL (Business Process Execution Language)

- Basiert auf XML
- Standardisierte Programmiersprache (OASIS)
- Modellierung entweder im XML-Code, in der Regel aber mit einer **nicht standardisierten** grafischen Notation
- BPEL ist in einer Workflow-Engine **direkt ausführbar**
- Breite Unterstützung in der Industrie (?)

XPDL (XML Process Definition Language)

- Standard der WfMC
- Wie BPEL direkt in Workflow-Engines ausführbar
- **Standardisierte grafische Notation**, Verwendung von BPMN

BPML (Business Process Modeling Language)

- Basiert auf XML
- Standardisierte Modellierungssprache (OMG)
- **BPMN** (Business Process Modeling Notation) ist eine **standardisierte grafische Notation**
- **BPQL** (Business Process Query Language) für die Administration
- **WSCI** (Web Services Choreography Interface) für die Prozesssteuerung
- BPMN lässt sich **auf BPEL abbilden**

UML (Unified Modeling Language)

- Standard der OMG
- **Standardisierte grafische Notation**, aber nicht stringent auf Prozessmodellierung bezogen
 - Abbildung auf Workflow-Engines ist problematisch

Weitere Modellierungssprachen

- **BPSS** (Business Process Specification Scheme, Modellierungssprache im Umfeld von ebXML)
- **EPK** (Ereignisgesteuerte Prozesskette, Modellierungssprache im SAP- und ARIS-Umfeld)
- **BPEL4People**
BPEL mit Berücksichtigung von Benutzerinteraktionen

BPEL ist eine XML-basierte Sprache zur Beschreibung von Geschäftsprozessen, deren einzelne Aktivitäten durch Web Services implementiert sind

→ Soll in erster Linie der Orchestrierung von Web Services dienen

BPEL soll das Konzept des „Programmierens im Großen“ ermöglichen (jedoch keine Unterstützung für die Interaktionen mit einem Menschen!)

Erweiterungen:

- BPELJ
- WS-BPEL4People
(liegt derzeit OASIS als Antrag vor)

Wichtige Merkmale:

- blockstrukturiert
- „Handler“ für Fehlerbehandlung
- „Compensation Handler“ für Kompensationen
- „Event Handler“ für Ereignisbehandlung
- BPEL kennt keine „Unterprozesse“!

Open Source BPEL Engines:

- Apache ODE („Orchestration Director Engine“)
- Intalio
- BPEL SE (Bestandteil des ESB von Sun Microsystems)

Kommerzielle Produkte (BPEL Engines):

- BPEL Process Manager (Oracle)
- XI Solution Manager / NetWeaver (SAP)
- WebSphere Process Server (IBM)
- BPWS4J (IBM)
- BizTalk Server (Microsoft)
- Business Mashup Server 2008 (Serena)

Zwei Programmiermodelle

- Executable Business Process Description
 - Orchestrierung
- Business Protocol Description
 - Choreografie

Ausdrücke und Werte

- Boolean Expressions → Wahrheitswerte
 - Deadline-Valued Expressions → Stichtagswerte
 - Duration-Valued Expressions → Zeitraumwerte
 - General Expression
- XPath als Sprache für die Expressions

Umgang mit Variablen

- Streng typisierte Variablen
- Wertzuweisungen

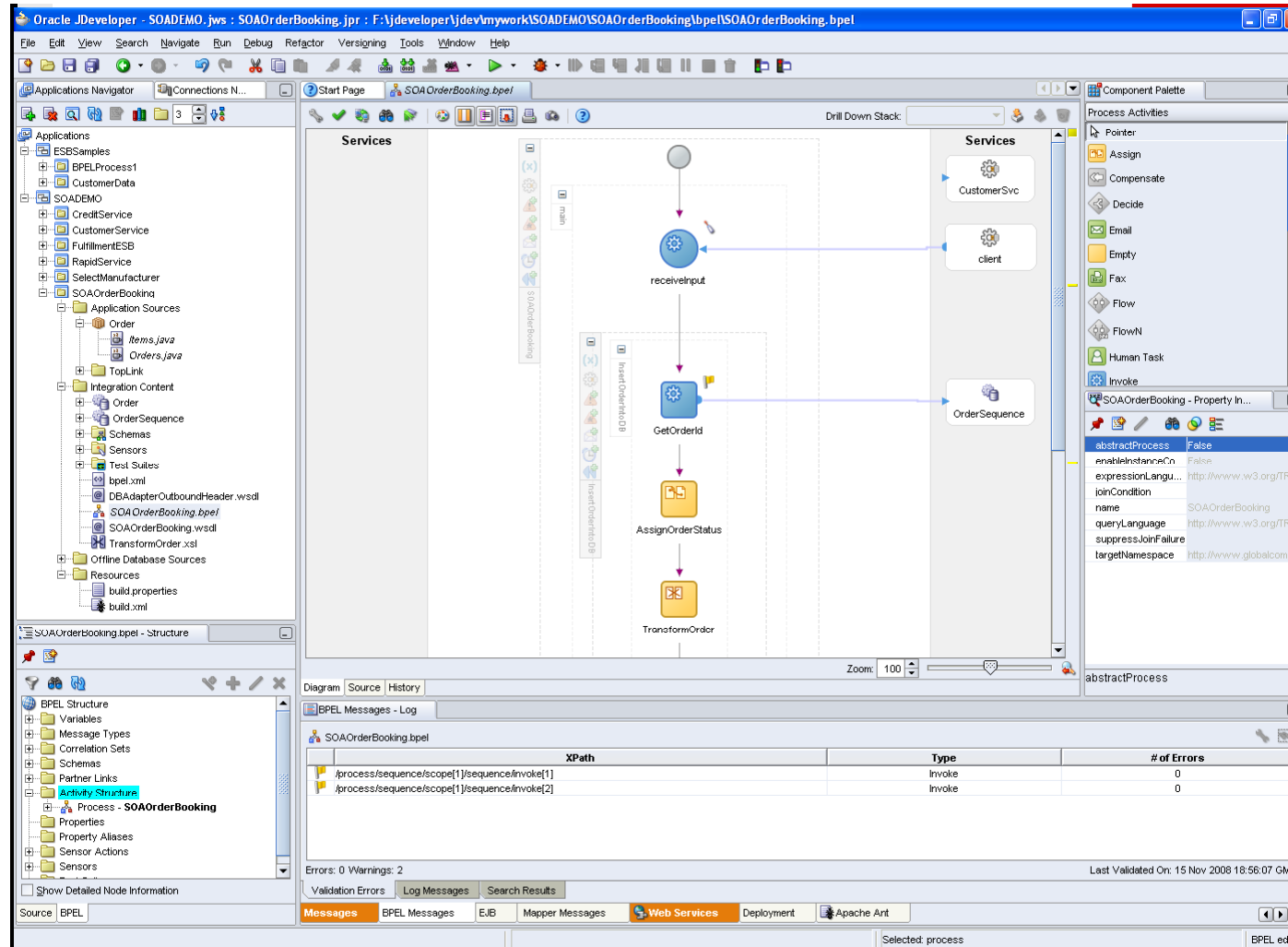
Aktivitäten

- Invoke → Aufruf von Services (synchron/asynchron)
- Receive → Warten auf Aufruf und Ergebnisrückgabe mit reply
- Reply → Rückmeldung von Ergebnissen
- Assign → Wertzuweisungen
- Throw → Auslösung von Fehlerereignissen
- Terminate → Beendigung der Verarbeitung
- Wait → wait for (warte Zeitspanne), wait until (warten bis)
- Empty → Platzhalter
- Scope → Gruppierung und Begrenzung der Fehlerausbreitung
- Compensate → Fehlerbehandlung innerhalb eines Scopes

Kontrollstrukturen

- Sequence → Folge von Aktivitäten
- While → Wiederholung solange Bedingung gültig
- Switch → Verzweigung
- Flow → Parallele Aktivitäten
- Links → Synchronisation von Aktivitäten

Beispiel: ORACLE



The screenshot displays the Oracle JDeveloper IDE interface for editing a BPEL process. The main workspace shows a flow diagram with the following activities: **receiveInput** (Start), **GetOrderId** (Invoke), **AssignOrderStatus** (Assign), and **TransformOrder** (Transform). The **Services** palette on the right lists **CustomerSvc**, **client**, and **OrderSequence**. The **Properties** window on the right shows the following details for the **abstractProcess**:

Property	Value
abstractProcess	False
enableInstanceCn	False
expressionLangu...	http://www.w3.org/TR...
joinCondition	
name	SOAOrderBooking
queryLanguage	http://www.w3.org/TR...
suppressJoinFailure	
targetNamespace	http://www.globalcom...

At the bottom, the **BPEL Messages - Log** table shows two messages:

XPath	Type	# of Errors
/process/sequence/scope[1]/sequence/Invoke[1]	Invoke	0
/process/sequence/scope[1]/sequence/Invoke[2]	Invoke	0

The status bar at the bottom indicates **Errors: 0 Warnings: 2** and **Last Validated On: 15 Nov 2008 18:56:07 GMT**.

Modellierungsobjekte

- Pools und Lanes (Verantwortungsbereiche)
- Events (Ereignisse)

- Als Startpunkt



- zu beliebigen Zeitpunkten



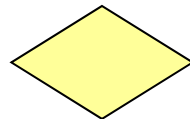
- als Endpunkt




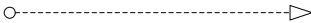
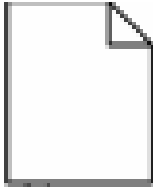
- Prozess, Sub-Prozess, Task



- Gateways

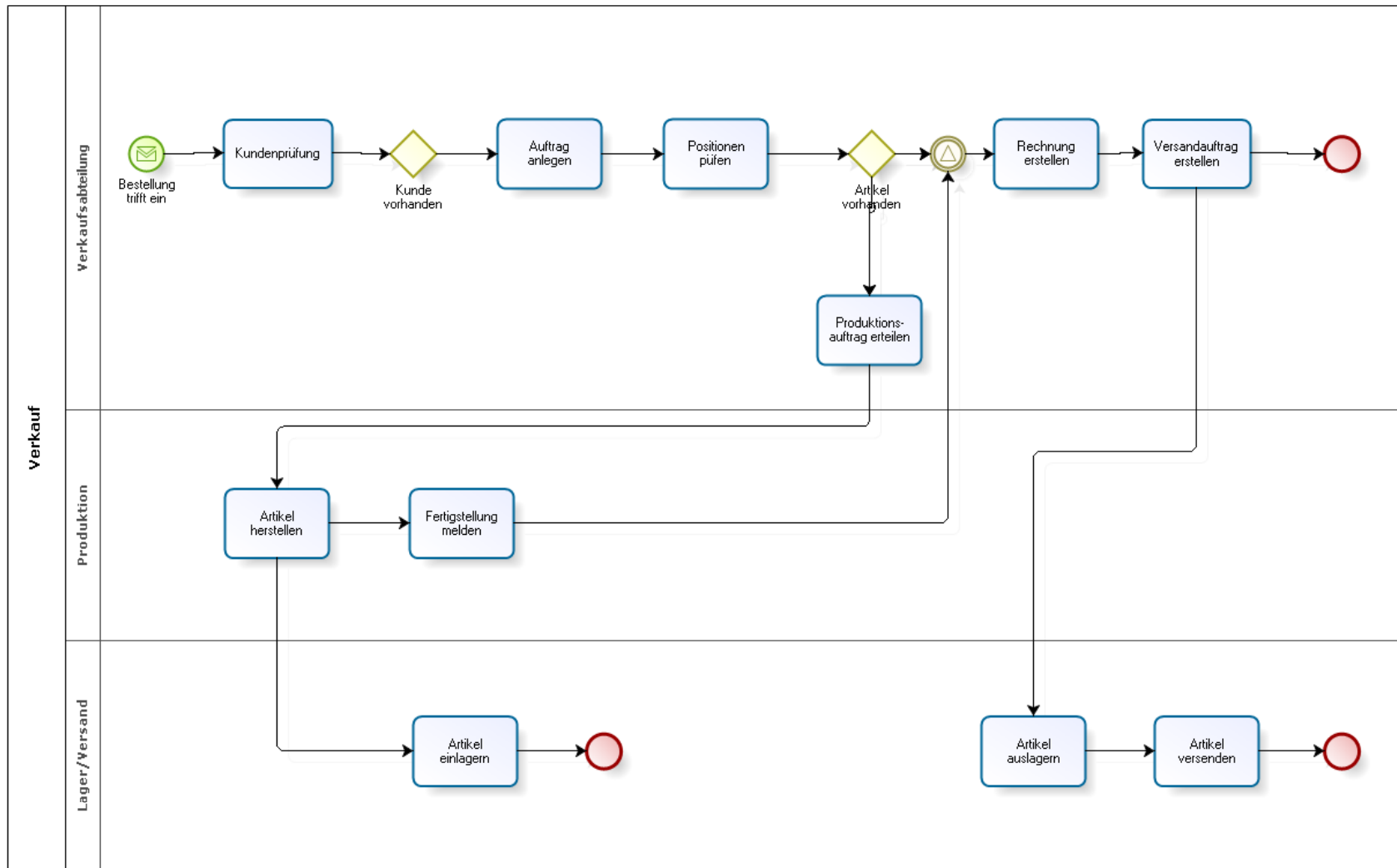


- Gateways erzeugen **parallele Abläufe** und fassen diese wieder zusammen
 - **Komplexe Entscheidungen** beim Erzeugen und Zusammenfassen **möglich**

- Sequence flow 
 - Innerhalb von Pools und Lanes
- Message flow 
 - Nur zwischen verschiedenen Pools bzw. Lanes verschiedener Pools
- Datenobjekte 
 - Datenobjekte werden als Dokumente dargestellt und sind **nur Annotationen**
 - Der Zustand kann in eckigen Klammern angegeben werden

Name
[State]

Beispiel

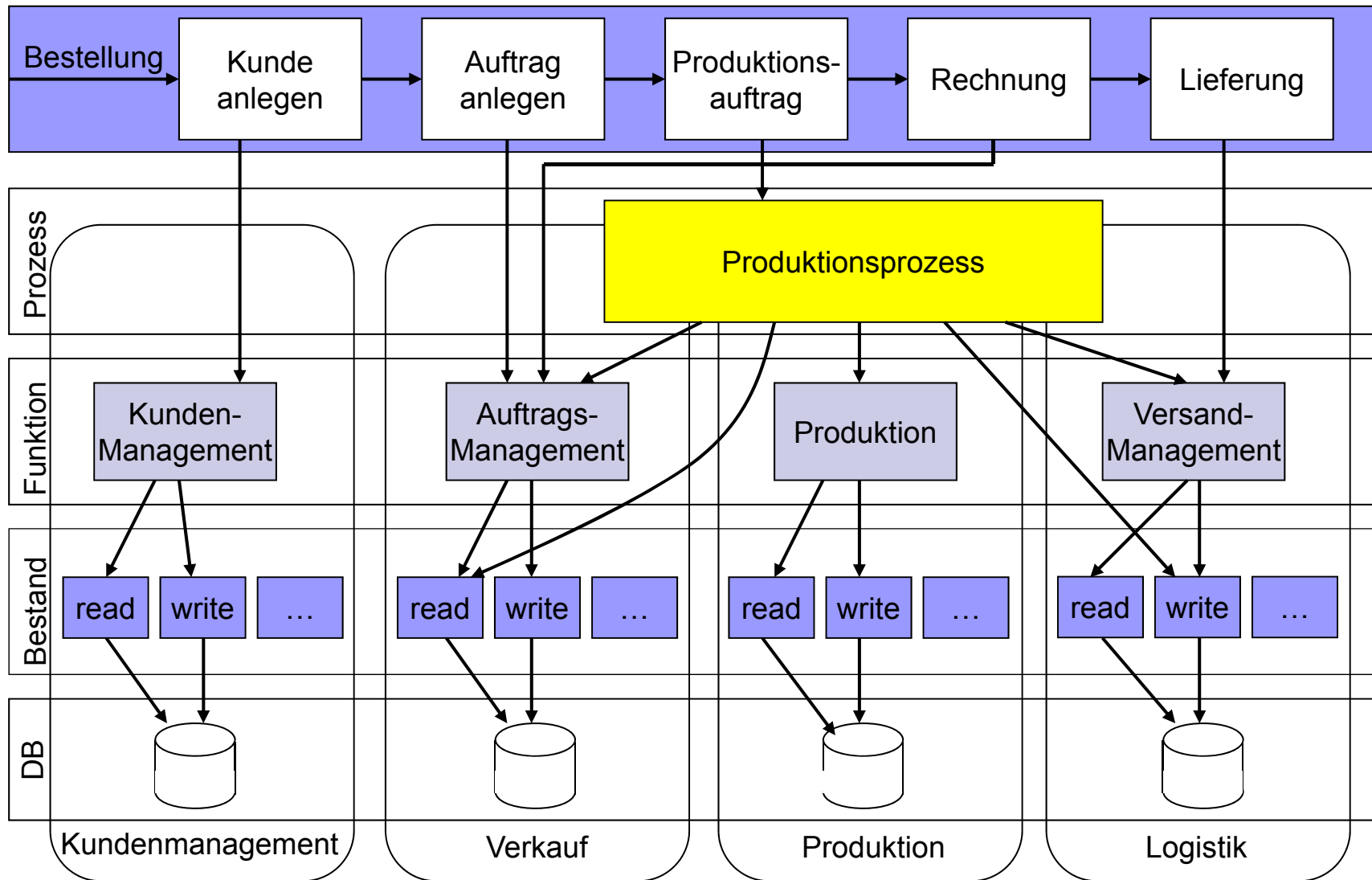


- Die Modellierung berücksichtigt in der Regel keine nicht-funktionalen Anforderungen, so z.B. keine SLAs
- Die Komplexität steigt bei Berücksichtigung von Fehlerzuständen und deren Behandlung stark an
- Bei sehr komplexen Problemen lassen sich nicht alle Aspekte adäquat abbilden
- Technische Details überfordern den Anwender in der Fachabteilung, der den Prozess aus Anwendungssicht definieren soll (die Erfahrung ist, dass sich Mitarbeiter der Fachabteilung nicht damit auseinandersetzen wollen)
- Zu grobe Beschreibungen aus Sicht der Anwender sind nicht ausreichend, um die effektive Abbildung auf die technischen Services zu automatisieren.

→ Schlußfolgerung ???

- Die fachliche Logik soll nur durch Bestands- oder Funktions-Services abgebildet werden.
- Prozessservices sollen möglichst wenig Fachlogik beinhalten und nur die Koordination der erforderlichen Schritte nach definierten Regeln durchführen.
- Prozessservices müssen einen Prozesszustand verwalten.

Beispiel: Bestellabwicklung





Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 23.6.2009

Serviceorientiertes E-Government

Identity Management und Berechtigungen

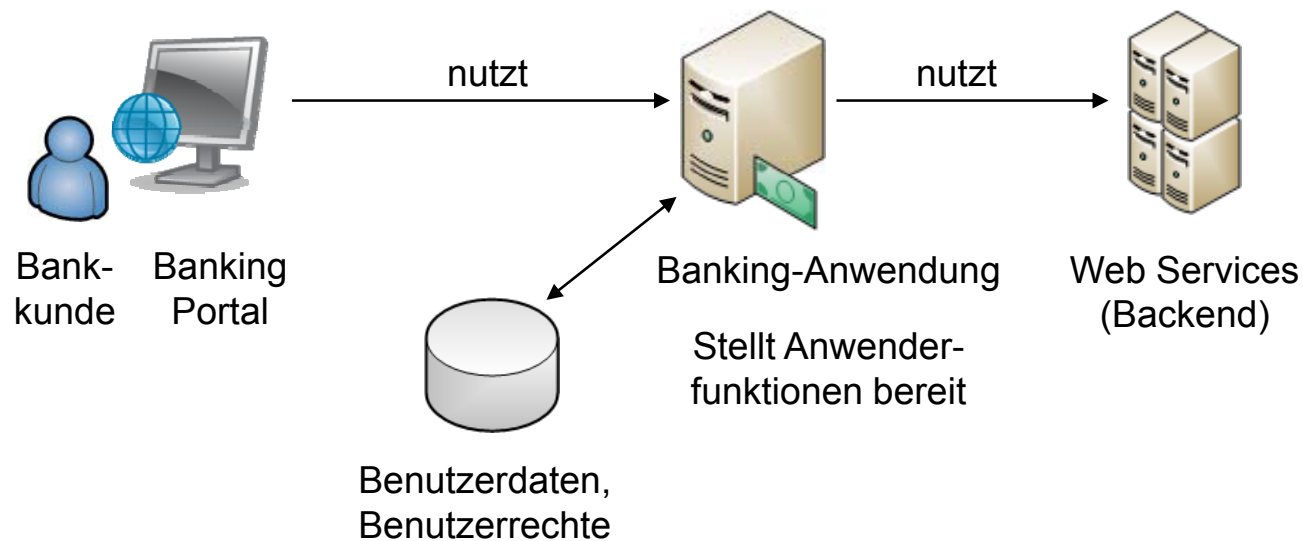
Dr. Frank Sarre

Lehrbeauftragter der LMU München

Ausgangssituation (1)

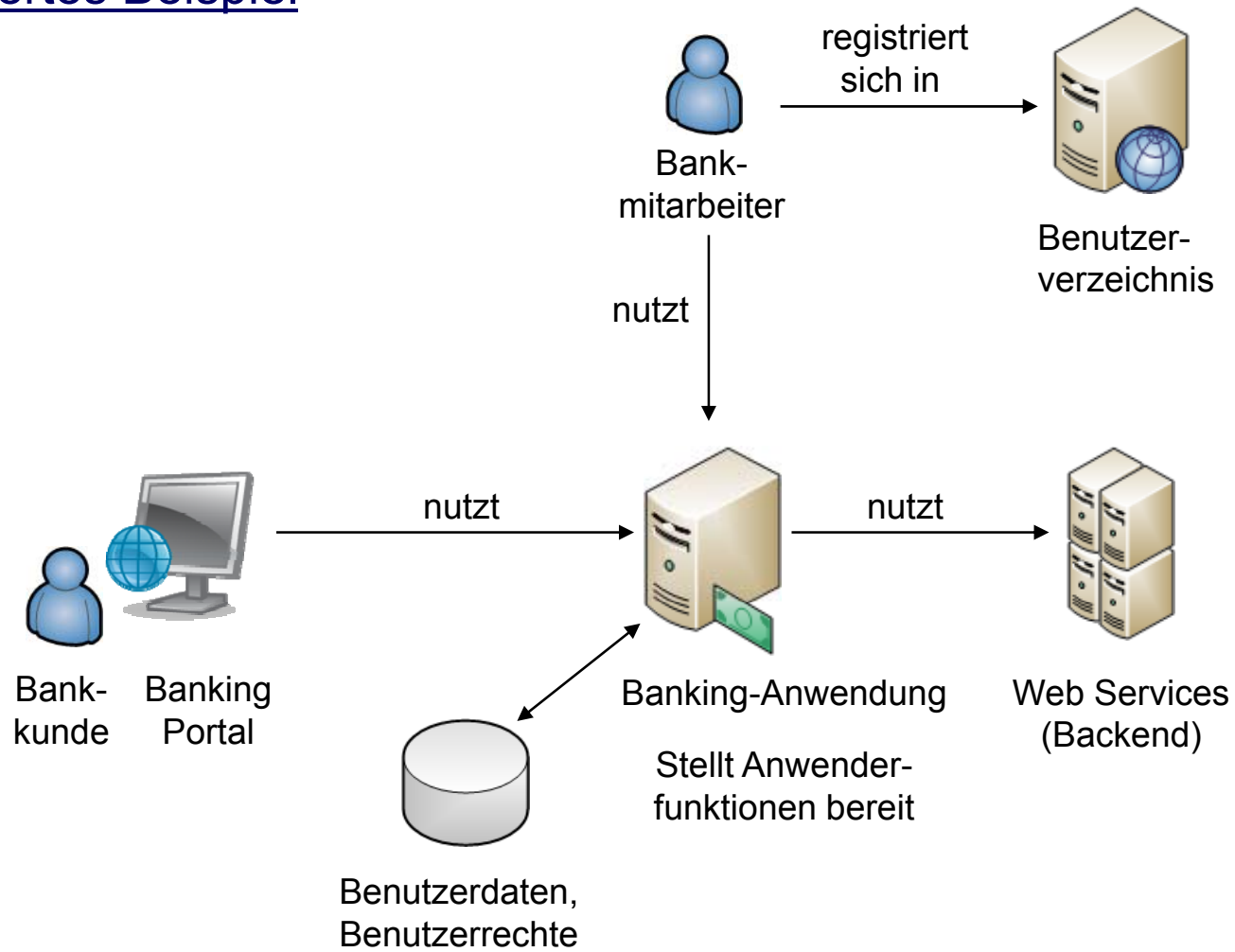
- Web Services und die verwendeten Protokolle bieten in der grundlegenden Form keine Sicherheitsmechanismen
- In der Praxis besteht jedoch immer die Notwendigkeit, dass Services gewissen Nutzungsbeschränkungen unterliegen

Beispiel



Ausgangssituation (2)

Erweitertes Beispiel



Anforderungen mit höchster Priorität

- Authentifizierung / Authentisierung
- Autorisierung
- Vertraulichkeit
- Integrität

Weitere Anforderungen geringerer Priorität

- Unleugbarkeit → Vorgang ist zweifelsfrei nachvollziehbar und kann nicht abgestritten werden
- Verfügbarkeit → Denial of Service muss verhindert werden
- Auditing → Protokollierung und Nachvollziehbarkeit

Authentifizierung / Authentisierung

- **Authentifizierung** ist die Verifikation einer behaupteten Identität
 - **Authentisierung** ist die Vorlage eines Nachweises der eigenen Identität
- Diese Ziele erreicht man üblicherweise durch **Identity Management** und **Zertifikate**

Autorisierung

- Festlegung bzw. Prüfung, ob ein Subjekt eine Aktion überhaupt ausführen darf bzw. welche Aktionen ein Subjekt ausführen darf
- Dieses Ziel erreicht man üblicherweise durch **Identity Management** in Verbindung mit **Rollen und Rechten**

Vertraulichkeit

- Informationen und Daten, die über Organisationsgrenzen hinweg verschickt werden, **dürfen nicht durch unbefugte Dritte gelesen werden können.**
- Dieses Ziel erreicht man üblicherweise durch **Verschlüsselung.**

Integrität

- Die übermittelten Informationen müssen **unversehrt übermittelt** werden bzw. dürfen zumindest **nicht unbemerkt verändert** werden können.
- Dieses Ziel erreicht man üblicherweise durch **Signaturen.**

Identität

- Systemobjekte haben in der Regel eine eindeutige Kennzeichnung und individuelle Attribute.

→ Daraus definiert sich eine sog. Identität (engl. Identity)

Objekttypen (Beispiele):

- Benutzer
 - Services
 - Maschinen
-
- Üblicherweise sind einer Identität verschiedene Rollen zugewiesen.
An diesen Rollen hängen in der Regel definierte Rechte.

Beispiele für Identitäten

- Benutzer in einem Active Directory oder LDAP-Verzeichnis
- Computer in einem Active Directory oder LDAP-Verzeichnis
- Benutzer in einer Benutzerdatenbank einer Anwendung

Beispiele für Rollen

- Systemadministrator
- Finanzbuchhalter
- Administrator für die Finanzbuchhaltung
- Verkäufer

Beispiele für Rechte

- Benutzer anlegen
- Konten in der Finanzbuchhaltung anlegen
- Konto bebuchen

Identity Management innerhalb einer Organisation

- Häufig findet man eine verteilte Verwaltung von Benutzern vor:
Die unterschiedlichen Anwendungen halten Informationen über Benutzer in eigenen Datenbanken

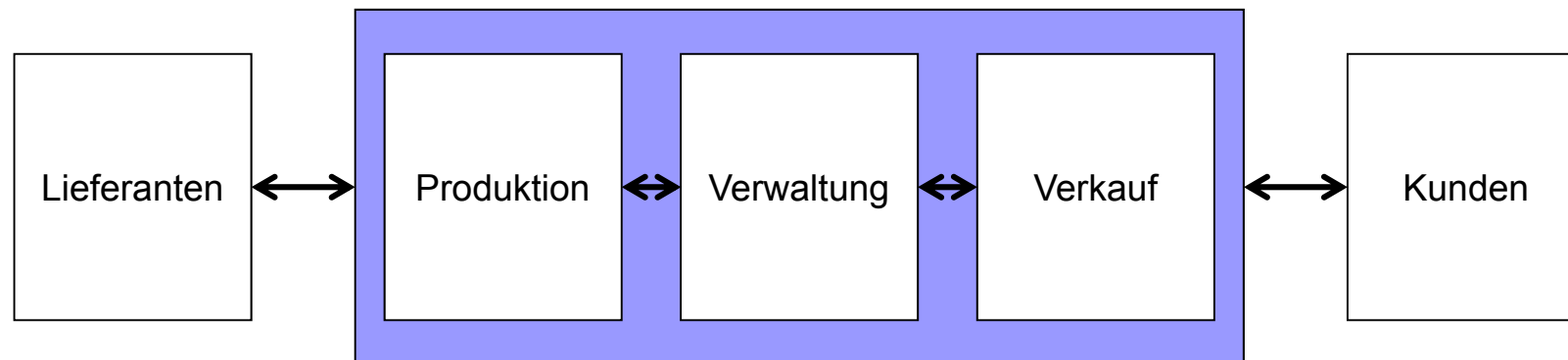
Technisch wäre es heute möglich, eine zentrale Benutzerdatenbank zu führen (z.B. mit LDAP oder Active Directory)

- Die anwendungsbezogenen Rechte werden in der Regel in den einzelnen Anwendungen verwaltet
- Meistens hat der Benutzer neben der Repräsentation im Verzeichnisdienst eine eigene Repräsentation im Anwendungssystem (z.B. für die Hinterlegung anwendungsspezifischer Einstellungen)

Identity Management über Organisationsgrenzen hinweg

- In einer SOA werden auch Services anderer Organisationseinheiten (Handelspartner, unabhängige Ressorts) genutzt, die in einem anderen technischen Umfeld existieren

Es besteht kein direkter Zugriff auf das Identitätsmanagement des externen Services bzw. umgekehrt kann der Service nicht auf das Identitätsmanagement des Consumers zugreifen



→ **Federated Identity Management**

Federated Identity Management

- Die Verantwortung für alle Identitäten einer Organisationseinheit obliegt der jeweiligen Organisationseinheit
- Die Organisationseinheiten vertrauen sich gegenseitig
- Der Service Provider hat keine Informationen über den Benutzer - Ausnahme bilden sog. „Security Credentials“, die bei der Nutzung des Services übergeben werden
- Der Service Provider lässt die Credentials des Nutzers bei der zuständigen Organisationseinheit überprüfen (diese Org-Einheit ist dann in der Rolle eines sog. **Identity Providers (IdP)**)

Identity Provider können auch vertrauenswürdige Dritte sein.

Security Credentials

Übliche Security Credentials sind:

- Benutzername / Passwort
- X.509 Zertifikat
- Kerberos-Ticket
- SAML-Token

Single Sign On

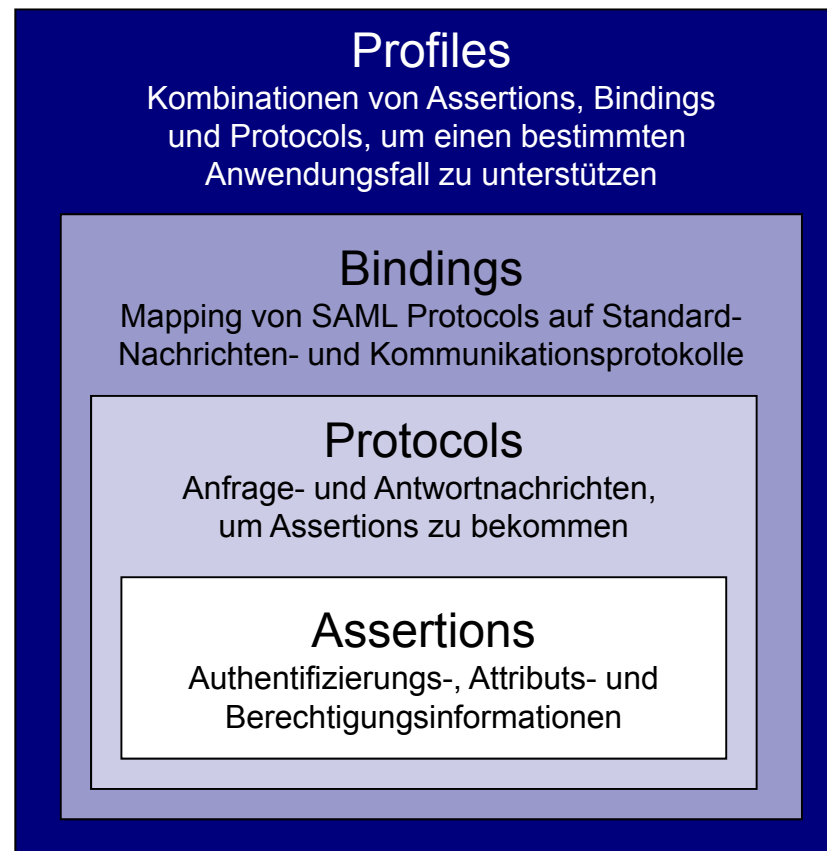
- Jede Identität, zum Beispiel ein Benutzer, soll sich nur ein einziges Mal am System anmelden und dann alle Anwendungen und Services nutzen können, für die sie/er eine Berechtigung hat

Eine Technologie, die diese Zielsetzung im SOA-Umfeld unterstützt, ist **SAML** (Security Assertion Markup Language)

Umsetzung des Federated Identity Management durch SAML

- SAML ist ein OASIS-Standard
(<http://saml.xml.org/saml-specifications>)

Quelle:
XML-Spektrum 5/2007
SAML 2.0, Ein Tutorium



Assertion

- Eine Assertion bestätigt die Authentizität eines Subjekts, die ein IdP einem Service garantiert
- Es wird auch die Information geliefert, wie das Subjekt authentifiziert wurde
- Zudem werden weitere Attribute über das Subjekt geliefert – dazu gibt es mehrere Statement -Typen:
 - Authentication → Authentifizierungsmethode
 - Attribute → Hinterlegung beliebiger Attribute
 - Autorization Decision → Rechte des Subjekts
- Weiterhin kann es weitere Informationen zur Gültigkeit einer Assertion geben, z.B. Zeitraum der Gültigkeit oder berechtigter Nutzer der Assertion usw.

Protocols

Insgesamt gibt es 6 definierte Abfolgen von Abfragen und Bestätigungen.

Die wichtigsten sind:

- Assertion Query and Request
 - Anfrage neuer Assertions
- Authentication Request
 - Prüfung einer bestehenden Assertion
- Artifact Resolution
 - Abfrage einer Assertion, die nicht direkt, sondern nur als Verweis übergeben wurde
- Single Logout
 - (Beinahe) simultaner Logout aus verschiedenen aktiven Sessions

Bindings

- SOAP 1.1
 - SAML-Nachricht über SOAP
- HTTP Redirect
 - SAML-Nachricht über HTTP-Redirect
- HTTP Post
 - SAML-Nachricht base64-encoded in HTML-Formularen
- HTTP Artifact
 - Referenz bzw. Adresse einer SAML-Nachricht
- SAML URI
 - Referenz bzw. Adresse einer Assertion

Profiles

- Insgesamt gibt es 13 „Profile“ für verschiedene Einsatzszenarien von SAML
- In einem Profil wird eine Kombination von Assertions, Bindings und Protokollen für den jeweiligen Anwendungszweck empfohlen

Beispiel: Web Browser SSO Profile

- Ein Internet-Nutzer greift auf eine Ressource eines Serviceproviders zu
(dazu muss er sich zuvor bei einem Identityprovider authentisieren oder der Serviceprovider leitet eine noch nicht erfolgte Authentifizierung seinerseits ein)
- Dieser Vorgang erzeugt eine Assertion, die der Serviceprovider akzeptiert
- Die Implementierung erfolgt mit **Authentication Request Protocol** in Verbindung mit **HTTP Redirect**, **HTTP POST** oder **HTTP Artifact Binding**

Verwendung digitaler Signaturen

- Um sicherzustellen, dass eine Assertion gültig und authentisch ist, signiert der IdP die Assertion
- Standard: Signaturen mit X.509-Zertifikaten
- Die Signatur wird mit Hilfe eines asymmetrischen Verschlüsselungsverfahrens (Public Key Verfahren) erzeugt

Asymmetrische Verschlüsselung

- Verwendung eines **Schlüsselpaares**:
 - **Öffentlicher** Schlüssel (public key)
 - **Privater** Schlüssel (private key)
- **Verschlüsselung** von Daten / Nachrichten durch Verwendung des **öffentlichen Schlüssels des Empfängers**. Nur mit dem privaten Schlüssel des Empfängers können die Nachrichten wieder entschlüsselt werden.
- Die **Signatur** einer Nachricht ist ein Hash-Wert, der mit dem **privaten Schlüssel des Absenders** verschlüsselt wird.
- Der Empfänger erzeugt seinerseits den Hash-Wert der (entschlüsselten) Nachricht und vergleicht diesen Wert mit dem Hash-Wert der (entschlüsselten) Signatur
 - Anhand der Signatur lässt sich die **Integrität** einer Nachricht überprüfen

Zertifikate und „Certification Authorities“

- Bei der Übermittlung einer signierten Nachricht wird auch ein **Zertifikat** übermittelt, das Informationen über den Aussteller des Zertifikats und den **öffentlichen Schlüssel des Absenders** enthält.

Zertifikate sowie das Schlüsselpaar werden von einer sog. **Certification Authority (CA)** ausgestellt.

- Dieser Certification Authority müssen sowohl Sender als auch Empfänger von Nachrichten **vertrauen**.
- Anhand eines Zertifikats kann der Nachrichteneempfänger prüfen, ob der behauptete **Benutzer** und damit der übermittelte Schlüssel **authentisch** ist.
- Certification Authorities können externe Dritte sein, z.B. Verisign. Es können aber auch firmenintern eigene Zertifikate erstellt und herausgegeben werden.

Entscheidung, einen Service nutzen zu dürfen

- Bei dem Aufruf eines Services muss geprüft werden, ob der Consumer die Berechtigung dazu hat
 - „**Policy Enforcement Point**“
- Ein Service kennt in der Regel keine User oder Berechtigungen, d.h. die Entscheidung muss außerhalb des Services getroffen werden
 - „**Policy Decision Point**“
- Eine Autorisierung kann verschiedenen Subjekttypen zugeordnet werden
 - Benutzer
 - Maschine
 - Anwendung / Service
- Es wird ein **Regelwerk** benötigt, mit dem Berechtigungen definiert und abgefragt werden können

Berechtigungen (2)

XACML (eXtensible Access Control Markup Language)

- Beschreibung von Autorisierungsinformationen nach einem definierten Schema
- Verwendet SAML

Beispielszenario

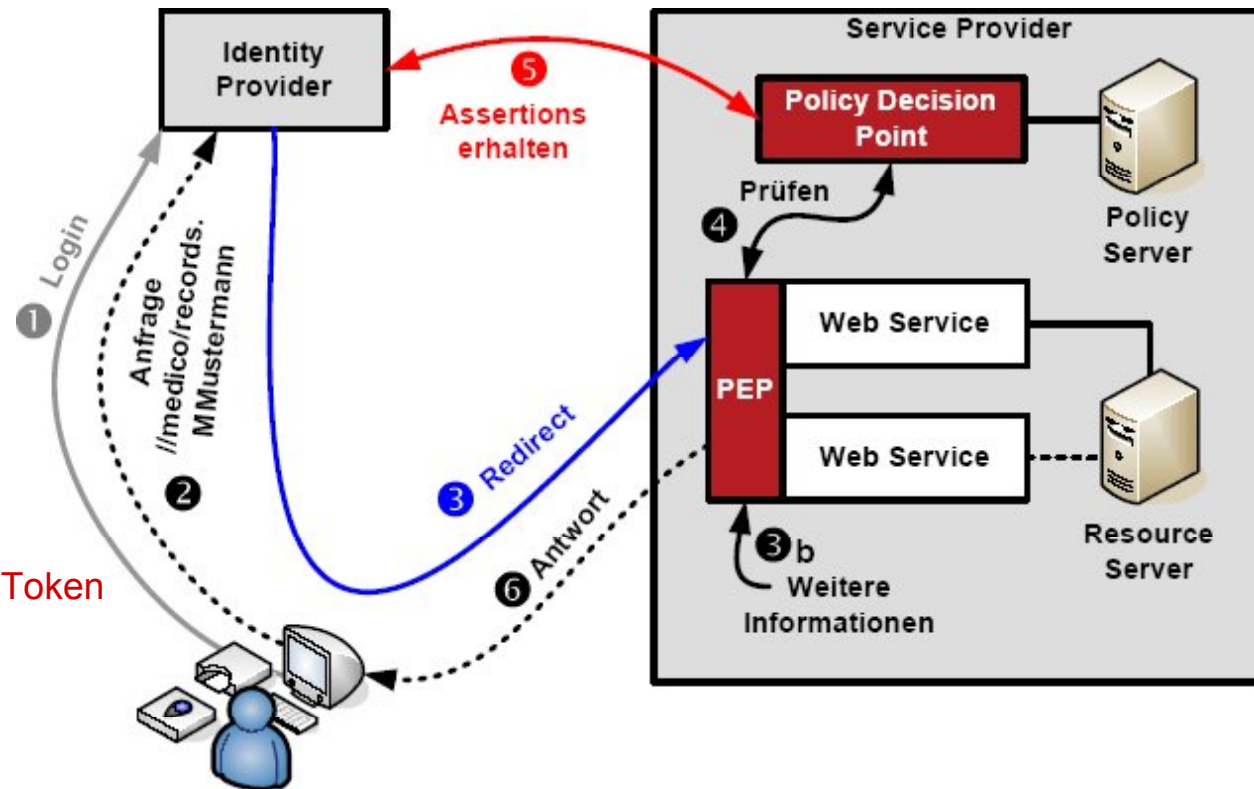
Quelle: <http://www.iam-wiki.org/XACML>

Wichtige Schritte

Schritt 1: Authentisierung

Schritt 3: Liefert User-SAML-Token

Schritt 5: Autorisierung



Weitere Federated Identity Management Frameworks

- Liberty ID-FF 1.2
 - Zusammenschluss verschiedener Organisationen und Firmen, um ein weltweites Identity Management mit Single Sign On für das E-Business aufzubauen
 - Schaffung eines „Circle of Trust“
 - Mapping von Identitäten anhand von Pseudonymen
 - Viele Aspekte sind bereits in SAML 2.0 eingeflossen, deshalb wird SAML genutzt und unterstützt

- Shibboleth Projekt
 - Umsetzung von SAML für Webanwendungen
 - Erweiterungen um Privacy-Funktionen
 - Single Sign On und Attribute-Austausch

Transport Layer Security

- Die Verbindung zwischen Consumer und Serviceprovider wird auf der Ebene der Transportprotokolle verschlüsselt
- Verwendung von HTTPS bzw. SSL / TLS
 - Aufbau eines sicheren Kanals (Secure Channel)
- Vorteil ist die für die Services die transparente Verschlüsselung
- Nachteilig ist, dass nur eine Punkt-zu-Punkt-Sicherheit möglich ist
 - keine oder nur beschränkte Verwendung von Intermediären möglich (z.B. ESB), da diese Subsysteme Teile der Information für die Verarbeitung / Weiterleitung benötigen
- Keine asynchrone Kommunikation möglich, da synchrones Protokoll
- Nur in relativ einfachen Umgebungen einsetzbar

Message Layer Security

- Verschlüsselung und Signierung der Nachrichteninhalte
 - Herstellung einer „End-to-End-Sicherheit“
- Im Allgemeinen das geeignete Verfahren für Web Services:
- Verschlüsselung des SOAP-Bodys ganz oder in Teilen
 - XML Encryption
 - Signatur des SOAP-Bodys ganz oder in Teilen
 - XML Signature
 - Signatur von Teilen des SOAP-Headers

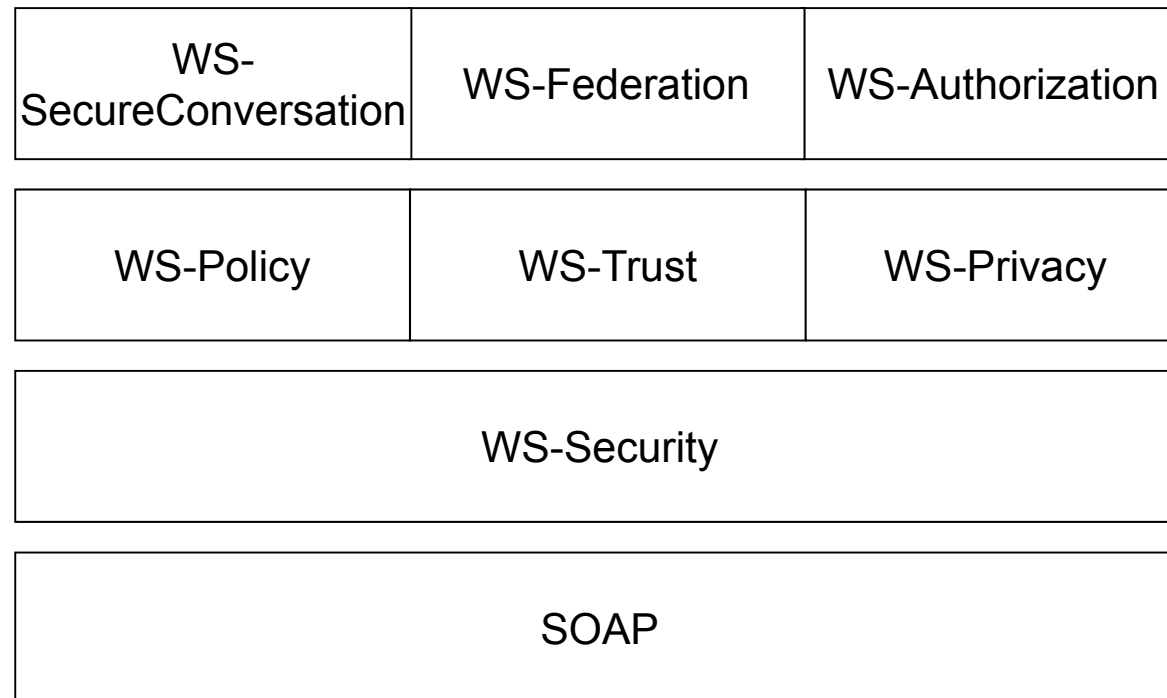
XML Encryption

- Verschlüsselung ganzer XML-Dokumente oder nur einzelner Teilstrukturen
- Symmetrische oder asymmetrische Verschlüsselung möglich
- Mindeststandards sind Advanced Encryption Standard (AES) und TripleDES
- Die gewählte Methode wird in der XML-Datei selbst definiert
- Initialisierungsparameter für die Ver- bzw. Entschlüsselung werden ebenfalls integriert
- Bei symmetrischer Verschlüsselung werden die Schlüssel häufig vorab mit asymmetrischer Verschlüsselung ausgetauscht

WS-Security

- Festlegung, wie Sicherheitsinformationen in SOAP-Nachrichten untergebracht werden
- Verwendung der verschiedenen Standards
 - XML Encryption
 - XML Signature
 - SAML

Erweiterungen von WS-Security



WS-Policy

- Festlegung der **Sicherheitsanforderungen** für eine Kommunikation
- Definition ist bereits in der Servicebeschreibung möglich

WS-Trust

- Festlegung des **Formats** der sicherheitsrelevanten Daten, z.B. SAML-Token
- Definition von **Vertrauenskett**en
- Definition von **Namensräumen**

WS-SecureConversation

- Verfahren zum Austausch eines Session Keys für symmetrische Verschlüsselung mit Hilfe eines asymmetrischen Verfahrens (Diffie-Hellman) → **Schaffung eines Sicherheitskontextes**
- Verwendung von Sicherheitstoken aus WS-Trust

WS-Federation

- Unterstützung der **Herstellung gegenseitigen Vertrauens**
- Baut auf Methoden von WS-Policy, WS-Trust und WS-SecureConversation auf
- Wird für die Schaffung einer Single-Sign-On-Lösung benötigt

WS-Privacy

- Definition der Anforderungen an die **Vertraulichkeit** von Nachrichten
- Methoden zur Sicherstellung der Einhaltung der Anforderungen

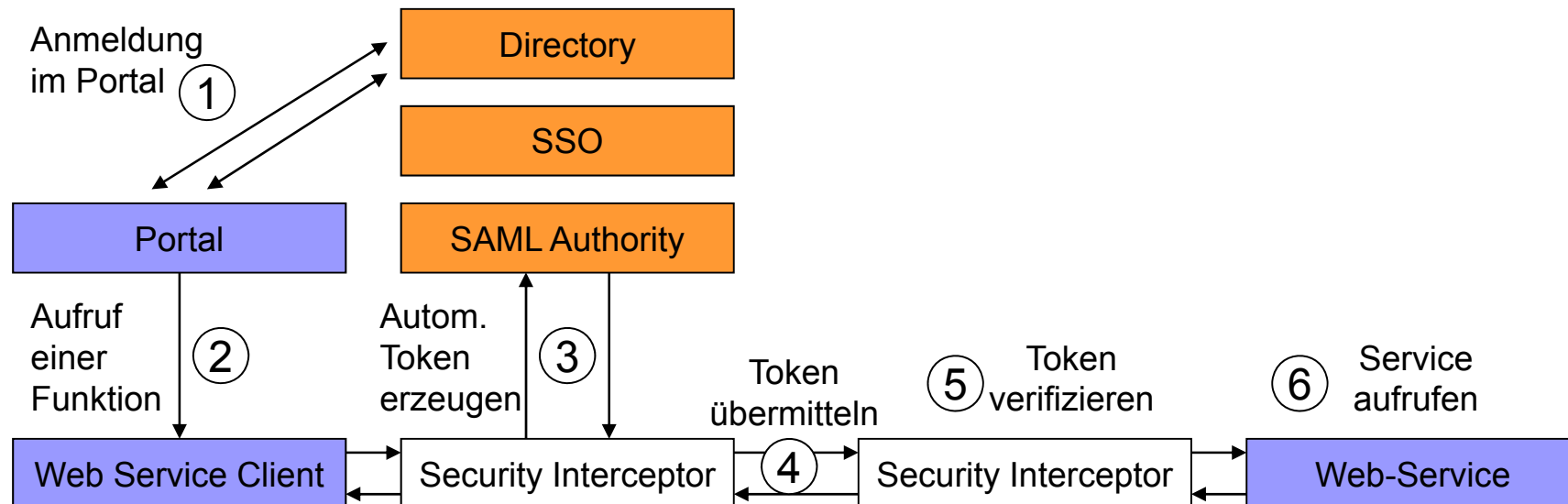
WS-Authorization

- Definition und Auswertung von **Zugriffsrechten**
- Erweitert WS-Trust um Berechtigungsattribute in den Sicherheitstokens

Security Proxy

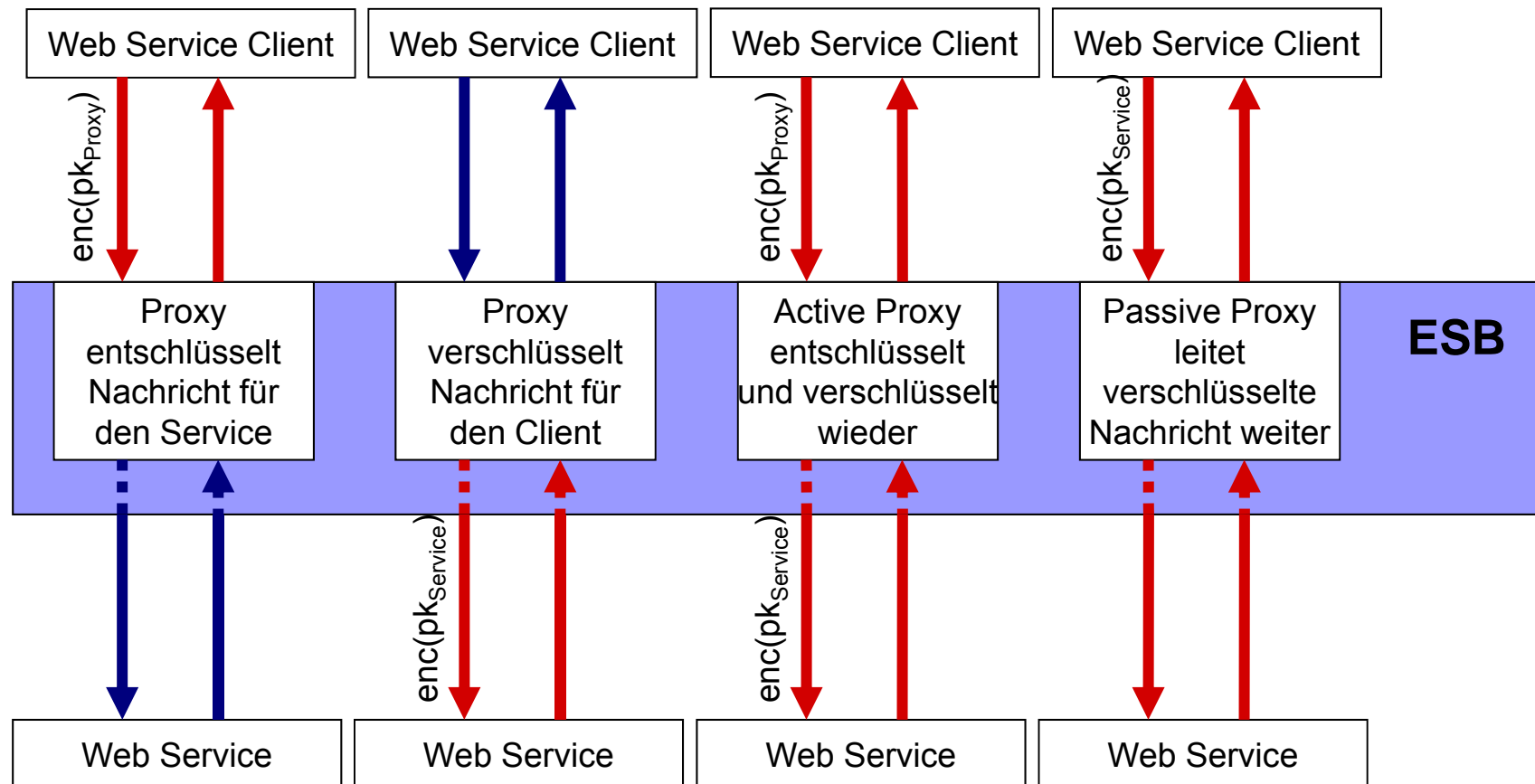
- Services müssen die Sicherheitsrichtlinien nicht selbst umsetzen
→ Hierzu kann ein sog. **Security Proxy** vorgeschaltet werden
(andere Bezeichnungen sind **SOAP-Proxy** oder **Security-Interceptor**)

Beispiel: Authentifizierung



Intermediary (Security) Proxy

Beispielszenarien für Verschlüsselung mit ESB



$enc(pk \dots)$ = Verschlüsselung mit Public key von ...



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 30. Juni 2009

Serviceorientiertes E-Government

Die E-Government-Ausschreibung des Freistaats Bayern

Dr. Frank Sarre

Lehrbeauftragter der LMU München

Unter anderem:

- **Reduzierung der Komplexität** der Anwendungslandschaft(en) im Freistaat Bayern
- **Standardisierung** von (Querschnitts-) Diensten
- Einfache, standardisierte **Kopplung von Anwendungen**
- Medienbruchfreie, **durchgängige IT-Unterstützung** von Verwaltungsprozessen
- Stärkere **Nutzerorientierung**
- Klar **geregelter Zugriff** auf Anwendungen und deren Daten (Access Management)
- **Schnellere Umsetzung** von neuen Anforderungen (→ „Agilität“)
- **Kostenreduzierung** durch die fachliche Wiederverwendung von (vorhandenen) Komponenten

Festgeschriebene Ziele des Vorhabens (grob):

1. Gesamtziel: Aufbau einer E-Government-Infrastruktur
2. Erprobung der Praxistauglichkeit anhand von Pilotprojekten
(= Beispielanwendungen)
3. Weiterentwicklung des E-Government-Architekturkonzepts
4. Know-how-Aufbau bei den Mitarbeitern des Freistaats Bayern

Grundlegende Idee:

1. **Aufbau des SOA-Know-hows** beim Auftraggeber
2. Einrichtung eines **Kompetenzzentrums**
3. **Verteilung des Know-hows** an Behörden im Freistaat Bayern

1. Prototyping und Grobkonzeption

- Fachliche und technische **Feinspezifikation** von Pilotanwendungen
- Aufbau einer prototypischen, betriebsfähigen **Infrastruktur**
- Realisierung eines (einfachen) **Identity Managements**
- Anbindung von besonderen **Querschnittskomponenten** wie ePayment und Geodatenserver
- Technische **Standardisierung**
- Ausarbeitung einer **Entwicklungsmethodik**

2. Weiterer Ausbau der Integrationsplattform

- Analyse der Anforderungen für die **Integrationsplattform**
- **Beschaffung** der notwendigen Infrastrukturkomponenten
- **Dokumentation** der Infrastruktur

3. Vervollständigung der Infrastruktur

- **Beschaffung** weiterer Infrastrukturkomponenten
- Integration in **www.bayern.de**
- Aufnahme **ressortübergreifender Geschäftsprozesse**
- Einbindung von weiteren **Basiskomponenten** und **Fachverfahren**
- **Bereitstellung von Services** über die Plattform
- Vervollständigung der **Dokumentation**

Dienstleistungsauftrag ...

Teilnahmeanträge sind zu richten an ...

Gegenstand der Bekanntmachung: Abschluss einer **Rahmenvereinbarung**

Laufzeit der Rahmenvereinbarung: **3 Jahre**

Gegenstand des Verfahrens ist die Realisierung von **eGovernmentverfahren** auf Basis einer **serviceorientierten Architektur** (SOA) inklusive der Beschaffung der hierfür benötigten **IuK-Infrastrukturkomponenten** (Middleware-Produktsuite) sowie ein zeitlich befristeter **Betrieb** des Gesamtsystems. Im Rahmen der Beschaffungsmaßnahme soll ein **Rahmenvertrag** mit einem leistungsfähigen Lieferanten oder Hersteller geschlossen werden, der die Lieferung, Inbetriebnahme und den initialen technischen Betrieb einer Middleware-Produktsuite sowie die Implementierung einzelner **Fachverfahren** und **Basiskomponenten** nach einer **Feinkonzeptionierung** umfasst. Die Middleware-Produktsuite muss als **Hauptbestandteile** folgende Bereiche abdecken, die erst bei Bedarf beschafft werden sollen:

- Applikationsserver,
- Dienstebus (ESB) inkl. Dienstverzeichnis / Repository,
- BPEL-Engine,
- Portal-Server und CMS,
- Benutzerverzeichnis und
- Entwicklungsumgebung

Bekanntmachung vom 14.2.2009 [2]

Für diesen Umfang soll eine flexible **Lizenzvereinbarung** getroffen werden, die dem Freistaat Bayern eine unbegrenzte Nutzung der Kernelemente der Produktsuite zu definierten Konditionen und mit geringem Verwaltungsaufwand ermöglicht. Gleichzeitig soll ausreichend **Flexibilität** vorhanden sein, um auf veränderte Anforderungen, neue Rahmenbedingungen und technologische Entwicklungen insbesondere durch **Skalierung** reagieren zu können.

Aufteilung in Lose: Nein.

Varianten/Alternativangebote sind zulässig: Ja.

Umfang des Auftrags

- Konzeption einer zentralen **Plattform für Online-Verwaltungsverfahren** (Basissystem) auf einer serviceorientierten Architektur (SOA) unter den technischen Rahmenbedingungen des RZ-Süd
- Lieferung und Überlassung von konfigurierter **Standardsoftware** für einen ersten Minimalausbau für eine Produktivumgebung
- Installation, Parametrisierung und Customizing auf einer vom Auftragnehmer vorgeschlagenen und optional beschafften und in der **Systemumgebung** des Rechenzentrum Süd betriebenen Hardwarekonfiguration
- **Feinkonzeptionierung** auf Werkvertragsbasis der Anbindung bzw. Erstellung von einzelnen Fachverfahren und Basiskomponenten zum Festpreis
- Programmtechnische Realisierung auf Werkvertragsbasis der **Anbindung bzw. Erstellung der einzelnen Fachverfahren** und Basiskomponenten nach Abnahme der Feinkonzepte durch den Auftraggeber auf der gelieferten SW-Umgebung zum Festpreis
- Technische **Betriebsverantwortung** für eine noch festzusetzende Übergangszeit

Umfang des Auftrags (Fortsetzung)

- **Pflege und Support** der höchsten Kategorie einschließlich Unterstützung vor Ort
- Rahmenvertrag für **Lizenerweiterungen** und weitere Werkleistungen
- **Dokumentation** für Aufbau und Betrieb
- **Projektmanagement**
- **Know-How-Transfer** für den selbständigen Ausbau und Betrieb inklusive Betriebsübergabe.

Optionen: ...

Teilnahmebedingungen

- **Unternehmensdarstellung**
- Darstellung (**Org-Chart**) sowie Anzahl der fest angestellten Mitarbeiter, aufgeteilt in den Bereichen Lösungen (Support/Consulting), Support (Hotline/Service) sowie technischen und fachlichen Betrieb
- Angaben über die durchschnittliche **Anzahl fest angestellter IT-Mitarbeiter** im Support (von 2006 - 2008)
- **Qualifikationsprofile** der Mitarbeiter in der Schlüsselposition Projektleiter, SOA-Architekt, Betriebsleiter
- **Mindestqualifikationen** der Projektmitarbeiter
- **Schulungsangebote**
- **Gewerbezentralregisterauskunft** (oder gleichwertige Bescheinigung einer Gerichts- oder Verwaltungsbehörde im Heimatland)
- **Handelsregisterauszug** (oder gleichwertige Bescheinigung einer Gerichts- oder Verwaltungsbehörde im Heimatland).

Teilnahmebedingungen, Fortsetzung

- **Jahresumsatz** bezogen auf den Gegenstand des Verhandlungsverfahrens von 2005 bis 2007
- **Erklärung Bewerbergemeinschaft**
- Kenntlichmachung der **Fabrikations-, Betriebs- und Geschäftsgeheimnisse**
- **Verpflichtungserklärung**
- Bereitstellung von (externen) **Ressourcen**
- **Eigenerklärung Zuverlässigkeit / Sozialversicherung**
- Eigenerklärung § 7 Nr. 5 Buchst. a und b VOL/A
- **Eigenerklärung Berufshaftpflicht**
- Erklärung Sicherheitsüberprüfungsgesetz
- Erklärung zur Verhütung von Manipulation im Verdingungswesen bei Liefer- und Dienstleistungen
- **Erklärung Eignungskriterien Subunternehmer**
- ...

Teilnahmebedingungen, Fortsetzung

- **Referenzen Realisierung**
- **Unternehmensreferenzen** im Bereich Lieferung und Pflege einer Middleware Plattform Suite
- **Sonstige vergleichbare Referenzen**
- **Referenzen Betrieb**
- ...

- Verfahrensart: **Verhandlungsverfahren**
- Bewerber sind bereits ausgewählt worden: Nein.
- **Beschränkung der Zahl der Wirtschaftsteilnehmer**, die zur Angebotsabgabe bzw. Teilnahme aufgefordert werden:
Geplante Mindestzahl: 3
Höchstzahl: 5
- **Objektive Kriterien für die Auswahl der begrenzten Zahl von Bewerbern:**
Es gelangen nur diejenigen Teilnahmeanträge in die Prüfung und Wertung, die sämtliche Anforderungen nach diesen Verdingungsunterlagen erfüllen. Grundlage für die Wertung der Teilnahmeanträge sind die in den Eignungsanforderungen genannten Kriterien.
...
- **Zuschlagskriterien:**
Wirtschaftlich günstigstes Angebot in Bezug auf die Kriterien, die in den Verdingungs-/ Ausschreibungsunterlagen, der Aufforderung zur Angebotsabgabe oder zur Verhandlung bzw. in der Beschreibung zum wettbewerblichen Dialog aufgeführt sind.

1. Wir wollen die eGovernment-Initiative fortsetzen mit dem Ziel, alle wichtigen Verwaltungsleistungen für die Bürger und die Wirtschaft auch online über das **Internetportal** anzubieten. Gleiches gilt für verwaltungsinterne Abläufe auch **über die Ressortgrenzen hinweg**. Um diese Ziele zu erreichen, werden wir für die bayerische Verwaltung eine zentrale Stelle schaffen, die für die grundlegende **strategische Steuerung** des Einsatzes von Informationstechnik zuständig ist („CIO“). Wir werden in Ausfüllung des eGovernment-Pakts mit der kommunalen Familie die ebenenübergreifenden, prozessoptimierten elektronischen **Verwaltungsabläufe standardisieren**.
2. Wir werden die **Integration der Rechenzentren** für die Staatsverwaltung sowie die **Einführung von Standardkomponenten** im Jahr 2011 abschließen. Innerhalb der nächsten fünf Jahre sollen **alle IT-Verfahren zentral abgewickelt werden** und – soweit möglich – **standardisierte Komponenten** nutzen.
3. Wir werden vor der Einführung von neuen IT-Verfahren und vor der Modernisierung bestehender Verfahren die zugrundeliegenden **Verwaltungsabläufe analysieren und optimieren**.

1. Wir wollen die eGovernment-Initiative fortsetzen mit dem Ziel, alle wichtigen Verwaltungsleistungen für die Bürger und die Wirtschaft auch online über das **Internetportal** anzubieten. Gleiches gilt für verwaltungsinterne Abläufe auch **über die Ressortgrenzen hinweg**. Um diese Ziele zu erreichen, werden wir für die bayerische Verwaltung eine zentrale Stelle schaffen, die für die grundlegende **strategische Steuerung** des Einsatzes von Informationstechnik zuständig ist („CIO“). Wir werden in Ausfüllung des eGovernment-Pakts mit der kommunalen Familie die ebenenübergreifenden, prozessoptimierten elektronischen **Verwaltungsabläufe standardisieren**.
2. Wir werden die **Integration der Rechenzentren** für die Staatsverwaltung sowie die **Einführung von Standardkomponenten** im Jahr 2011 abschließen. Innerhalb der nächsten fünf Jahre sollen **alle IT-Verfahren zentral abgewickelt werden** und – soweit möglich – **standardisierte Komponenten** nutzen.
3. Wir werden vor der Einführung von neuen IT-Verfahren und vor der Modernisierung bestehender Verfahren die zugrundeliegenden **Verwaltungsabläufe analysieren und optimieren**.

Unterhalb des EU-Schwellenwerts

Verfahrensarten

- Öffentliche Ausschreibung
- Beschränkte Ausschreibung
- Freihändige Vergabe

Verfahren

Gem. VOL/A VOB/A
(Basisparagrafen)

Kein Rechtsschutz

(nur Dienstaufsichtsbeschwerde)

Oberhalb des EU-Schwellenwerts

Verfahrensarten

- Offenes Verfahren
- Nicht-offenes Verfahren
- Verhandlungsverfahren
- Wettbewerblicher Dialog

Verfahren

Gem. VOL/ A VOB/A „a“ Paragraphen bzw.
im Bereich der Sektorenauftraggeber gem.
VOL/A VOB/A „b“ Paragraphen, VOL

Rechtsschutz (zwei Instanzen)

Nachprüfungsverfahren bei Vergabekammer
Beschwerde an den Vergabesenat des OLG

Schwierigkeiten bei Ausschreibungen [1]

- Der Auftraggeber kann den **Beschaffungsgegenstand** nicht genau beschreiben, weil er im Vorfeld das erforderliche technische Know-how nicht besitzt.
- Der Auftraggeber tut sich sehr schwer, einen geeigneten **Bewertungsmaßstab** zu definieren.
- Der Auftraggeber möchte verschiedenen technischen **Lösungsansätzen** nicht vorgreifen.
- Die Angebote der Anbieter sind **schwer vergleichbar**, da der genaue Umfang der Leistungen sowie die sich ergebenden Konsequenzen unklar sind.
- Es fehlt dem Auftraggeber das **methodische Know-how**, die Ausschreibungsunterlagen zusammenzustellen.

Schwierigkeiten bei Ausschreibungen [2]

- Der Auftraggeber kann die **Fachkunde** der verschiedenen Anbieter nicht richtig bewerten.
- Aufgrund unterschiedlicher **Zuständigkeiten** gelingt es dem Auftraggeber nur unzureichend, wichtige Entscheidungen zeitnah herbeizuführen.
- **Pilotanwendungen** können von dem Auftraggeber nicht abschließend **spezifiziert** werden, weil die technische Basis im Vorfeld der Ausschreibung noch sehr vage ist.

Wesentliche Erfolgsfaktoren für das Projekt

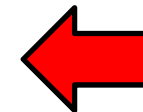
1. Der beauftragte Anbieter muss für das Projekt Mitarbeiter bereitstellen, die das Projekt **dauerhaft** und **kompetent** unterstützen.
2. Der Auftraggeber muss seinerseits geeignete, **gut ausgebildete Mitarbeiter** für das Projekt abstellen.
3. Auch seitens des Auftraggebers müssen **klare Organisationsstrukturen** definiert werden.
4. Der Auftraggeber muss für das Projekt einen geeigneten **technischen Rahmen** schaffen, zum Beispiel auch einen Identitäts- und Rechtemanagement etablieren.
5. **Offenen Standards** muss eine hohe Priorität eingeräumt werden, um die Abhängigkeit zu einem bestimmten Hersteller zu vermeiden.

Merkmale des Verhandlungsverfahrens

- Das Verhandlungsverfahren ist ein **Ausnahmeverfahren**, das nur unter bestimmten Voraussetzungen durchgeführt werden darf.
- Es wird **kein förmliches Verfahren** eingehalten, sondern über die Auftragsbedingungen **verhandelt**.
- Wenn eine Vielzahl von Bewerbern zu erwarten ist, kann ein **öffentlicher Teilnahmewettbewerb** durchgeführt werden.

Grober Ablauf des Verhandlungsverfahrens

- a) Bekanntmachung
- b) Teilnahmewettbewerb
- c) Versand der Verdingungsunterlagen
- d) Abgabe der Angebote
- e) Auswahl des wirtschaftlichsten Angebots
- f) Zuschlag



**Wir sind hier
(Stand Ende Juni 2009)**

Dann in der Folge:

- g) Durchführung des Auftrags
- h) Abnahme

Mitteilung an die Anbieter

- Aufforderung zur Abgabe eines Angebots
- Mitteilung zu den Fristen
- Mitteilung über die Vergabeart (Verhandlungsverfahren)

Überblick über die Verdingungsunterlagen

- Anschreiben der ausschreibenden Stelle
- Allgemeine Hinweise zur Ausschreibung
- Leistungsbeschreibung und Leistungsverzeichnis
- Spezifikationen
- Beschreibung von bestehenden Systemen
- Rahmenbedingungen und Richtlinien
- Vertragsbedingungen
- Vordrucke für die Angebotserstellung
- Fragenkatalog für Bieterfragen

- Grundsätzliche Hinweise
- Ausschreibende Stelle
- Angebotsabgabe
- Fristen
- Bietergemeinschaften, Subunternehmen
- Aufteilung der Leistung
- Hauptangebote / Nebenangebote
- Aufbau, Form und Inhalt des Angebots
- Änderungen, Berichtigungen oder Rücknahme von Angeboten
- Entschädigung für die Bearbeitung des Angebots
- ...

Grundsätzlicher Ablauf der Prüfung:

1. Formale Prüfung der Angebote auf Einhaltung der Richtlinien und Vorgaben
2. Bewertung der Leistung
3. Ausschluss von Angeboten, die Ausschlusskriterien verletzen
4. Ermittlung des Preises / der Preise
5. Berechnung der Wirtschaftlichkeit
6. Prüfung auf Angemessenheit der Angebote

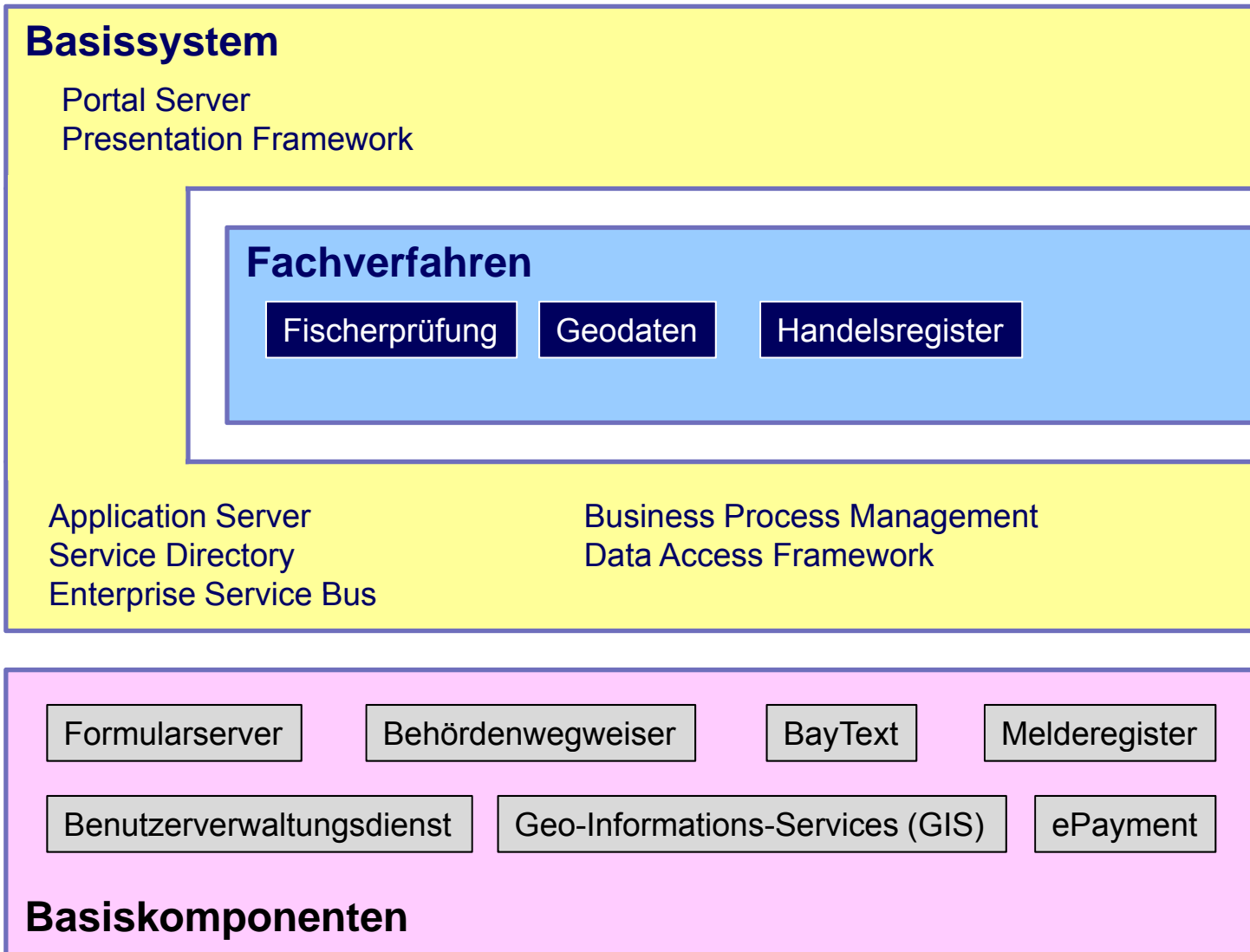
1. Ggf. Reduzierung der in Frage kommenden Anbieter
2. Verhandlungen mit den Anbietern
(Finalisierung der Vertragskonditionen, abschließende Bestimmung der Leistung)
3. Aufforderung zur Abgabe des letzten Angebots („last call“)
4. Abgabe von letzten Angeboten durch die Anbieter auf Basis des verhandelten Leistungsumfangs und der vertraglichen Konditionen
5. Auswertung der letzten Angebote und Bestimmung des wirtschaftlichsten Anbieters
6. Information an alle Anbieter (wg. Einspruchsmöglichkeit)
7. Erteilung des Zuschlags

- Standards und Architekturen für E-Government-Anwendungen
- Richtlinie BayITR-02 zur Durchführung von IuK-Projekten
- V-Modell XT
- Richtlinie „Barrierefreiheit“
- IT-Standards der Bayerischen Staatsverwaltung
- Richtlinie „Werkzeuggestützte Modellierungssprachen“
- Arbeitsschutzgesetz
- Bildschirmarbeitsverordnung
- IT-Grundschutzkataloge des BSI
- E-Government-Handbuch des BSI
- Allgemeine Empfehlungen des BSI zum sicheren Einsatz von aktiven Inhalten in vertrauenswürdigen Webanwendungen
- BSI-Studie „Sicherheit von Webanwendungen“

Richtlinien und Standards [2]

- Entwicklungsrichtlinien für Anwendungen (z.B. StMELF)
- Dokumentationsrichtlinien (z.B. StMELF)

Gesamtarchitektur der E-Government-Plattform



- Allgemeine Anforderungen und nicht-funktionale Anforderungen
 - Systemarchitektur
 - Richtlinien und Standards
 - Betrieb des Systems
 - Performance
 - Verfügbarkeit
 - Quellcode

- Funktionale Anforderungen
 - Benutzerverzeichnis
 - Enterprise Service Bus (ESB)
 - Web Services
 - Prozessdienst
 - Portal und Content Management
 - Portlets
 - Entwicklungsumgebung
 - Integration von Fachverfahren und Querschnittskomponenten

- Betriebs- und Serviceleistungen
 - Softwareunterstützung durch Hotline
 - Fehlerbehebungszeiten
 - Softwarepflege
 - Fernwartung der Software

- Sonstige Leistungen
 - Schulung und Einweisung
 - Hardware
 - Erstellung und Lieferung der Vertragssoftware
 - Erstellung von feinen Spezifikationen und weiteren Konzepten
 - Installation und Implementierung der Vertragssoftware
 - Dokumentationen
 - Produktivstartbegleitung

- Optionale Leistungen
 - Konnektoren
- Projektorganisation
 - Projektorganisation
 - Projektplanung
- Präsentation im Verhandlungsverfahren
 - Präsentation der Lösung und Kompetenz

Auftraggeber

Bayerische Staatskanzlei
(in Zusammenarbeit mit der ZIL)

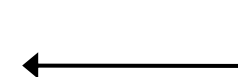


Steuerkreis

Auftragnehmer

Anbieter _____

Projektleiter,
Betriebsleiter





Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 7. Juli 2009

Serviceorientiertes E-Government

SOA Governance

Dr. Frank Sarre

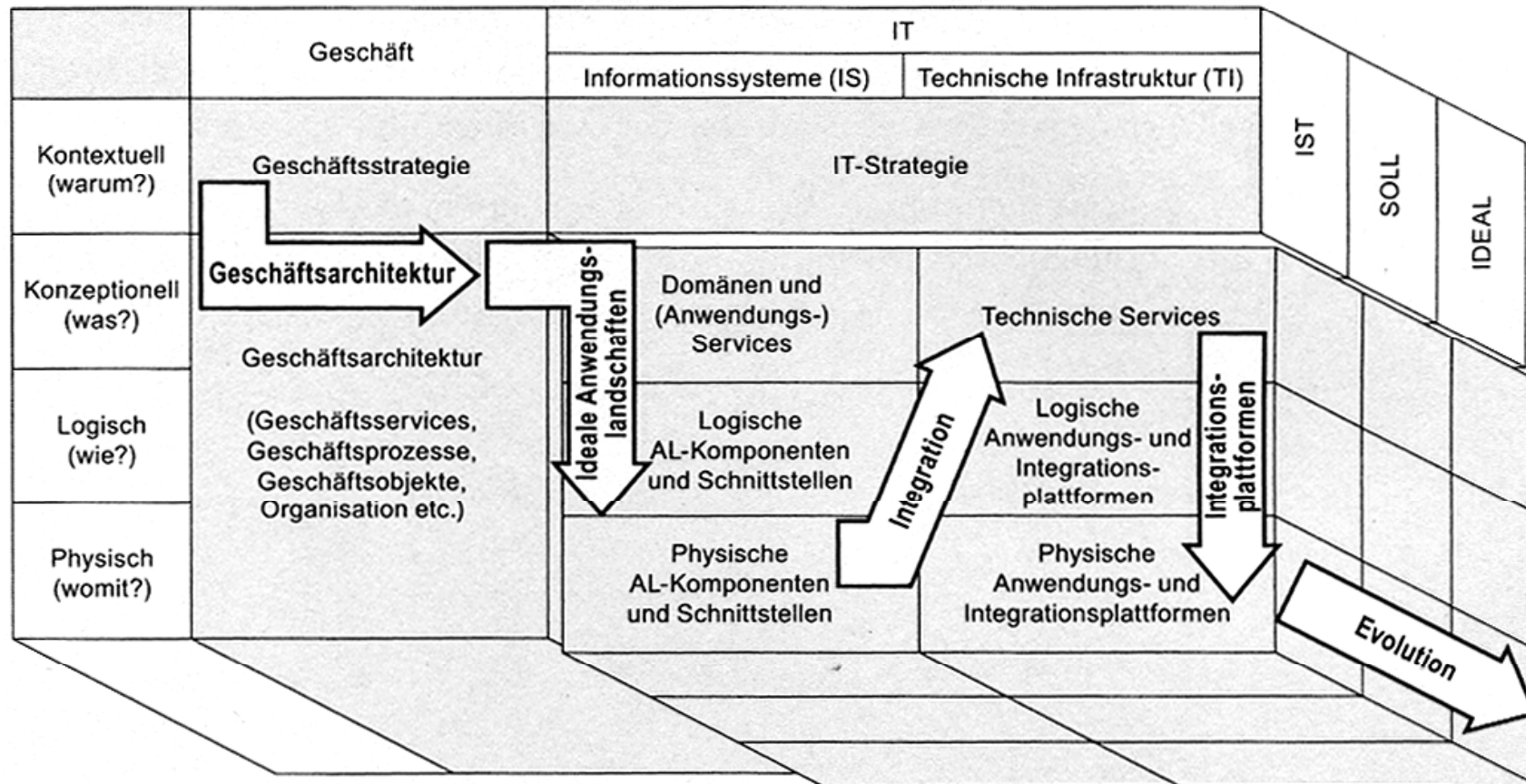
Lehrbeauftragter der LMU München

Ziele einer Serviceorientierten Architektur

- Kosteneffizienz
- Effektivität
- Agilität
- Innovation
- Quality of Service

Wiederholung [2]

Lösungsweg

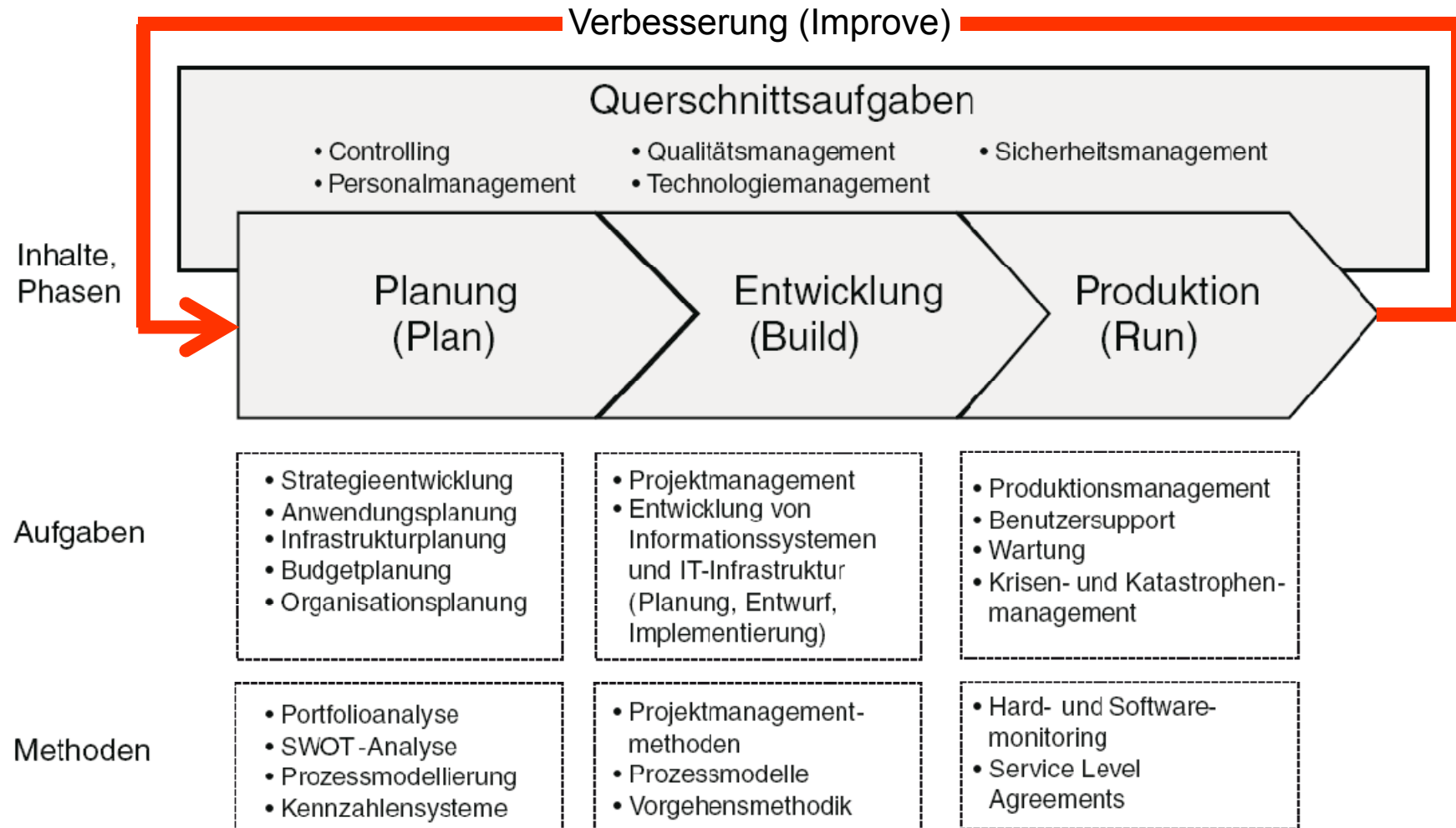


Quelle: „Quasar Enterprise“

- **Kein** eindeutig definierter Begriff !
- SOA Governance betrifft sowohl **organisatorische** als auch **technische Themen**
- Stark vereinfacht beschreibt eine SOA Governance:
 - **was** zur Einführung und Aufrechterhaltung einer SOA zu tun ist und
 - **wie** die einzelnen Beteiligten dies zu tun haben.
- SOA Governance ist ein **Teil des Informationsmanagements** einer Behörde (oder eines Unternehmens)

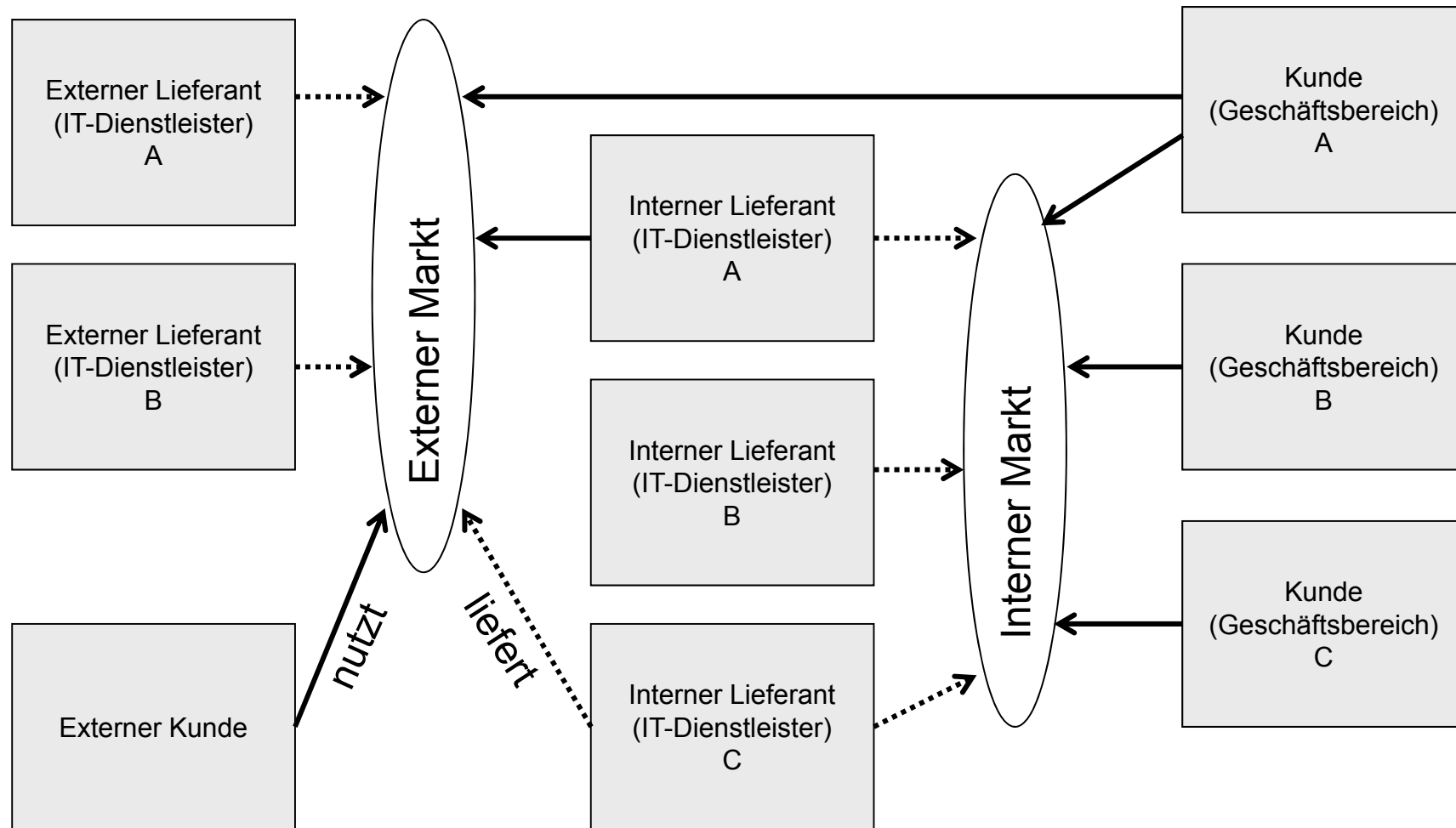
Interessante Definition der Software AG (für Manager)

- Eine **SOA** ist wie ein LEGO-Baukasten. Die Blöcke haben verschiedene Größen, Formen und Farben, lassen sich aber in vielfältiger Weise kombinieren.
- **SOA Governance** ist das Verfahren, die **richtigen LEGO-Blöcke** zu produzieren und diese in geeigneter Weise anzubieten.

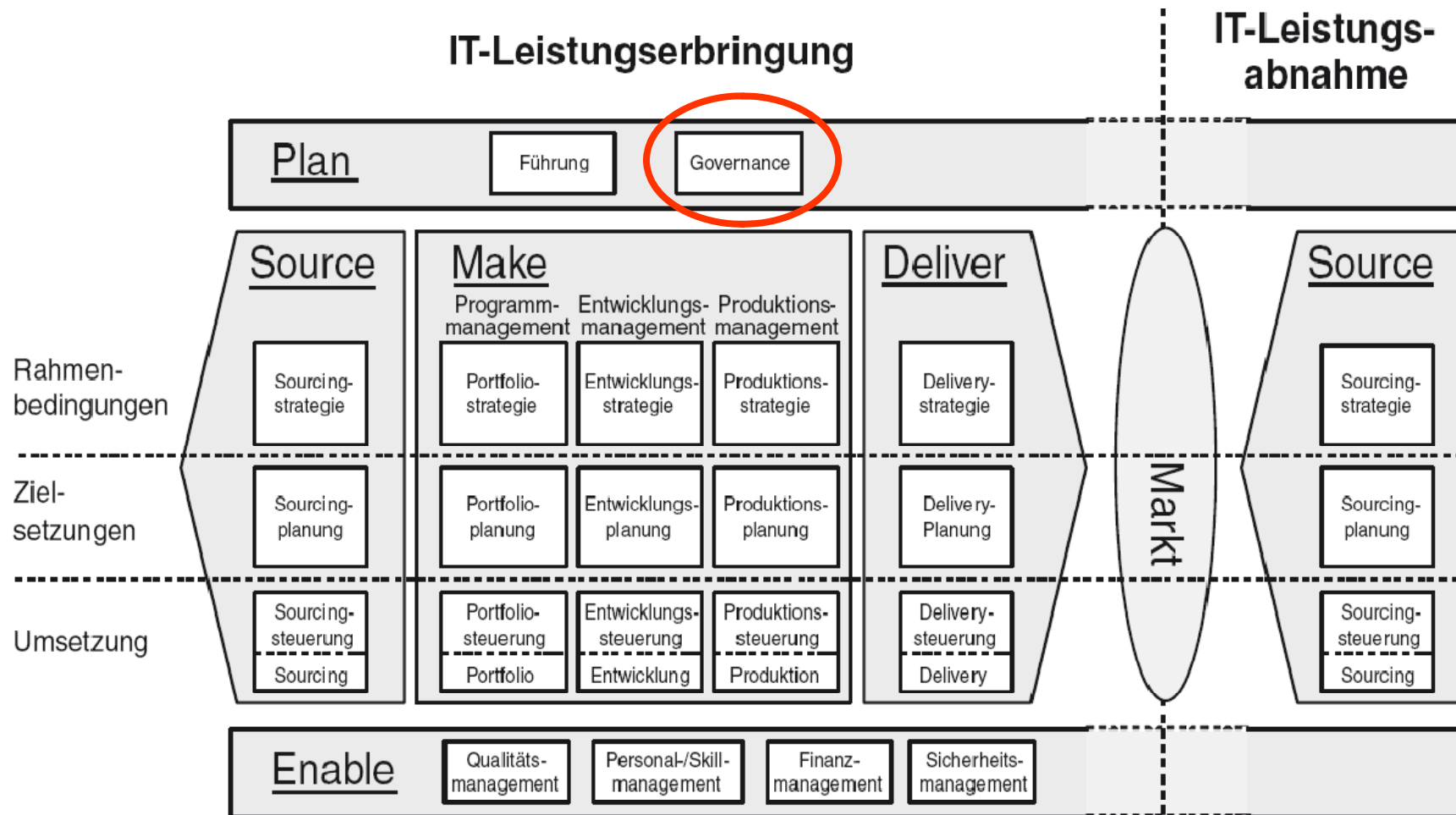


Quelle: Zarnekow, Integriertes Integrationsmanagement – Vom Plan, Build, Run zum Source, Make, Deliver

Kunden-/ Lieferantenbeziehungen



Angelehnt an: Zarnekow, Integriertes Integrationsmanagement – Vom Plan Build Run zum Source, Make, Deliver



Quelle: Zarnekow, Integriertes Integrationsmanagement – Vom Plan Build Run zum Source, Make, Deliver

- Der Modellierungsprozess lässt existierende Services außer Acht
→ zu **geringe Wiederverwendung**
- Fehlende Qualitätssicherungsprozesse, bevor ein Service für den Betrieb freigegeben wird
→ **mangelhafte Servicequalität**
- Fehlender Gesamtblick, wie das Geschäft und die EDV verzahnt sind
→ **ineffiziente Organisation und Servicenutzung**
- Keine Berücksichtigung von Auswirkungen, wenn Dienste geändert werden
→ **Versionskonflikte, unzureichende Servicequalität**
- Fehlende Verantwortlichkeiten bezüglich der Schaffung von Diensten und ihrer Nutzung
→ **unkoordinierter Software Lifecycle**
- Die Einhaltung von Richtlinien wird nur manuell sporadisch und unstrukturiert kontrolliert
→ **unzureichende Sicherheit, erhöhte Aufwände**

Ein Service bietet Methoden für die **Pflege und Verwendung von Adressdaten** in einer Behörde.

Dieser Service wird von folgenden Anwenderkreisen genutzt:

- Finanzexperten
 - Rechtsabteilung
 - Zentraler Einkauf
- Der Service wird von verschiedenen Stellen zu unterschiedlichen Zwecken genutzt
- Die Nutzung unterliegt unterschiedlichen Richtlinien und Rahmenbedingungen
- Für den effizienten Einsatz des Adressdatenmanagements muss eine Abstimmung hinsichtlich Inhalt, Verwendbarkeit und Zulässigkeit der Verwendung stattfinden.

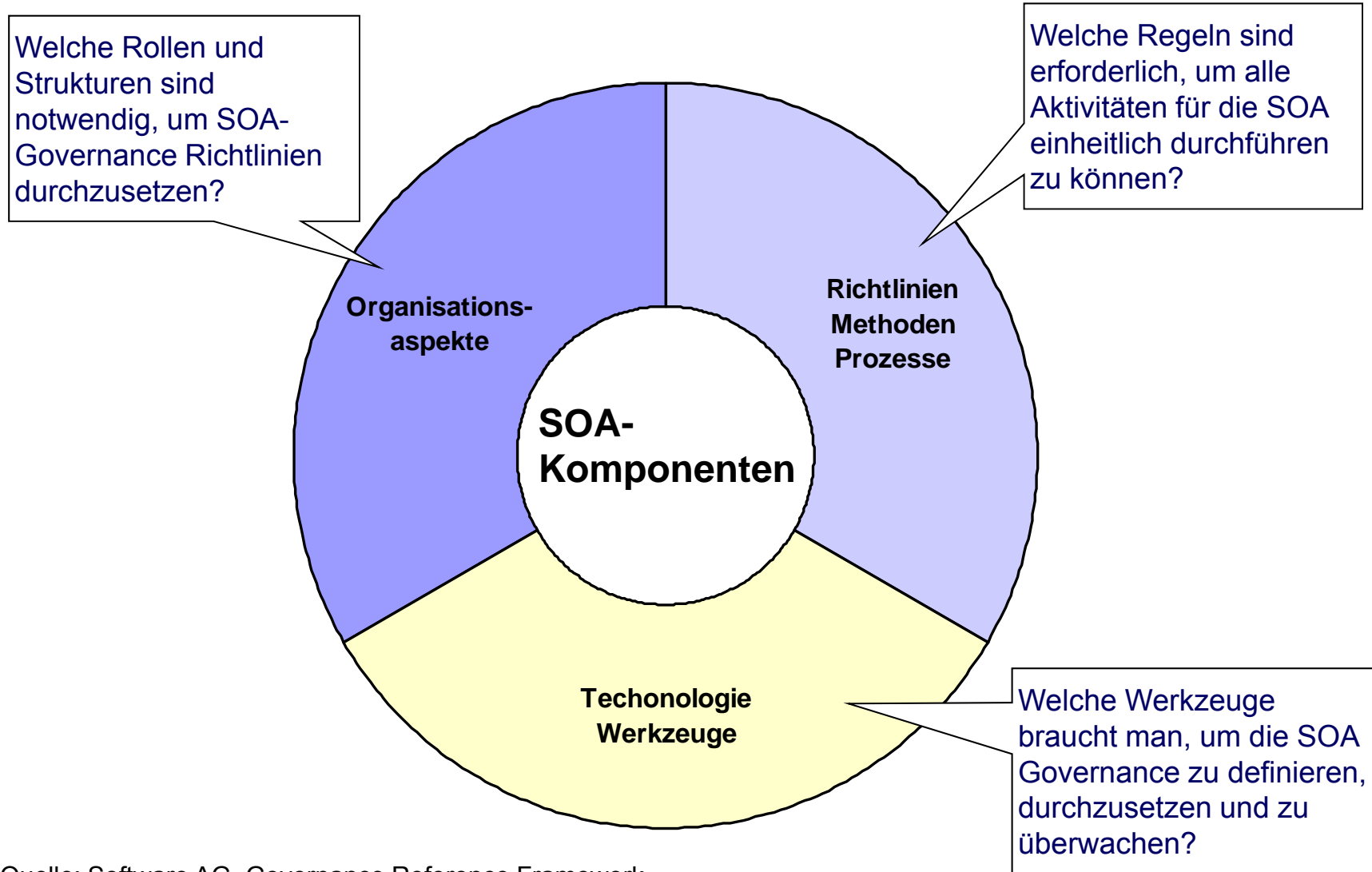
Informationsmanagement beinhaltet auch Methoden und Prozesse, um Veränderungen an bestehenden Strukturen vorzunehmen.

- a) Organisatorische Veränderungen
 - Abteilungsstrukturen
 - Aufgabenverteilungen
 - Geschäftsprozesse
 - ...

- b) Technische Veränderungen
 - Austausch veralteter Systeme
 - Hinzufügen zusätzlicher Systeme
 - Einführung neuer Technologien
 - ...

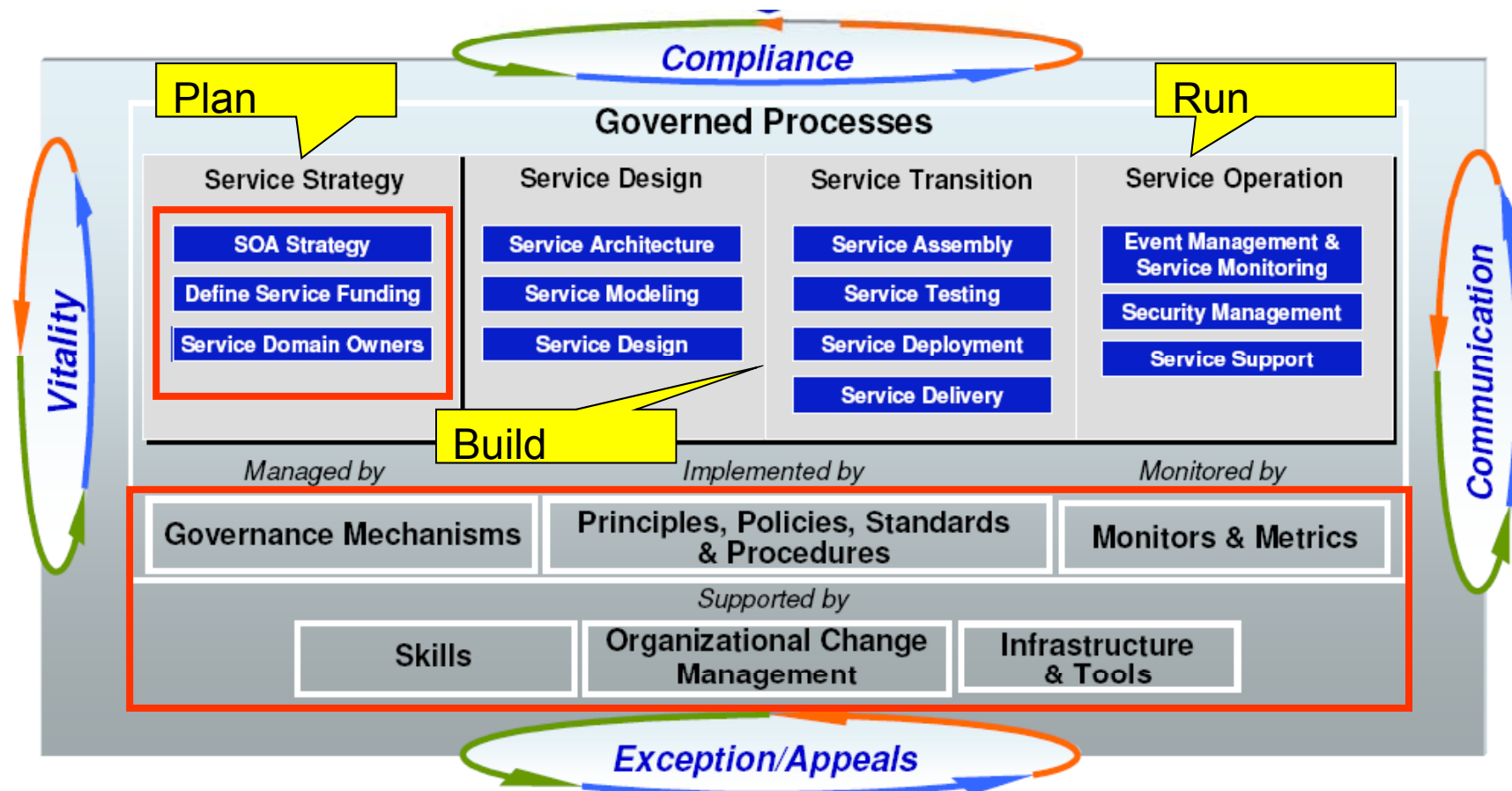
→ **Change-Management**

SOA Governance [1]



Quelle: Software AG, Governance Reference Framework

Governance für einen Service Lifecycle (Vorschlag von IBM)



Quelle: IBM, Advancing the Business/IT linkage with SOA Governance and Service Lifecycle Management

SOA-Strategie

- **Führungsentscheidung**, dass eine SOA umgesetzt werden soll
- Definition **globaler Zielsetzungen** (auch, was **nicht** Ziel sein soll)
- Definition der **Rahmenbedingungen** für eine SOA
- Einrichtung von **Kontroll- und Entscheidungsgremien**
- Festlegung von **Kompetenzen**
- Festlegung von Grundsätzen zur **Finanzierung**

→ **Top-Level-Management**

Domain-Verantwortung

- Zentraler Aspekt für die Umsetzung
- Festlegung, welche Organisationseinheit für die jeweiligen Services verantwortlich ist
- Häufig problematisch, da Geschäftsprozesse bei der SOA-Betrachtung mehrere Organisationseinheiten betreffen
- Erfordert die Bereitschaft der höheren Führungsebenen, sinnvolle Lösungen auch durchzusetzen
- Schlüsselaspekt: Der **Aufwand** und die damit verbundene **Finanzierung** bzw. **personelle Ausstattung**

Kontroll- und Entscheidungsgremien

Aufgaben

- Festlegung von **Detailzielen**
- Festlegung von **Regeln, Methoden und Verfahren**
- Verabschiedung von **Architekturleitlinien**
- Festlegung und Überwachung von **Qualitätskriterien**
- Abstimmung der **Finanzierung**
- **Change Management**

Voraussetzungen

- Hinreichende Kompetenz
- Berechtigung zur Durchsetzung von Regeln (Policy Enforcement)

Problem

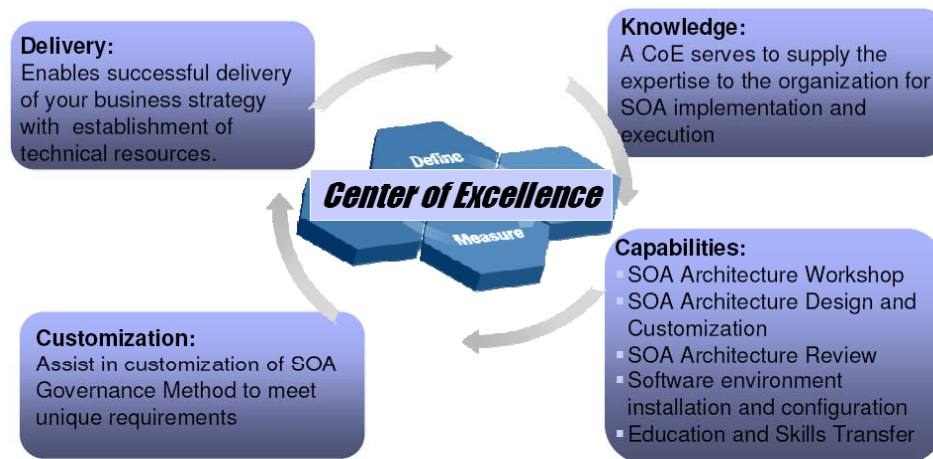
- Nicht immer vorhandenes technisches Know-how
- Unzureichende Methodenkenntnisse bei den Verantwortlichen

Grundsätzliche Probleme und beispielhafte Fragestellungen

- Welche **Services** stehen zur Verfügung?
- Wie sind bestimmte **technische Lösungen** zu bewerten?
- Was kann **wiederverwendet** und was muss neu gemacht werden?
- Was ist zu tun, damit **Services wiederverwendet** werden können?
- Wie sieht eine **optimale Architektur** aus?
- Was wären **sinnvolle Schritte** auf dem Weg dorthin?
- Welche Services sind bereits in **Planung**?
- Wie wird die **Leistung/Nutzung** eines Services gemessen?
- Welche **Qualitätsanforderungen** gibt es?

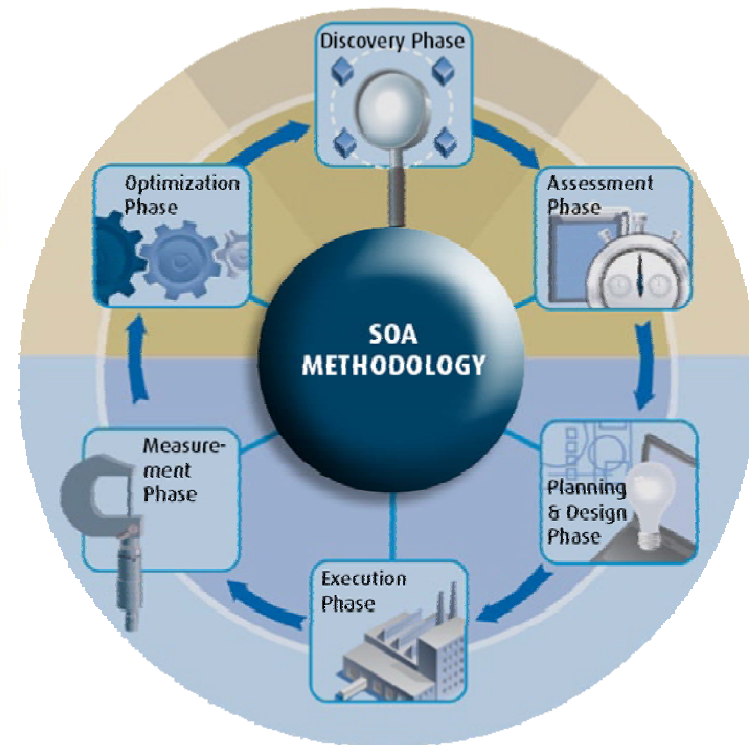
→ **SOA-Kompetenz-Center**

Kompetenz-Center zur Unterstützung aller Beteiligten



IBM: „Center of Excellence“

Software AG:
SOA Competency Center™

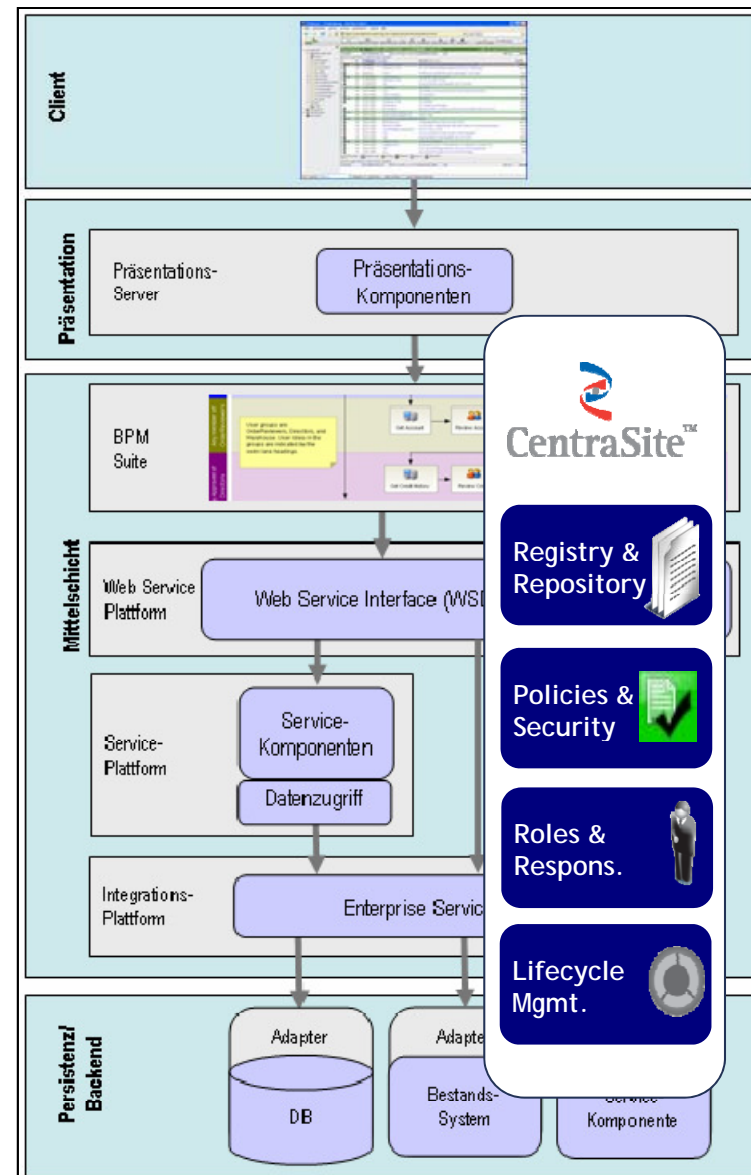


Typische Aufgaben des SOA-Kompetenz-Centers

- Auswahl und Erarbeitung von **Methoden und Richtlinien**
- **Analyse von Geschäftsprozessen**
- Entwicklung von **Architekturvorschlägen**
- Definition von **Metriken**
- Auswahl von **Werkzeugen**
- **Unterstützung bei der Umsetzung**
- **Unterstützung des Systembetriebs**
- **Ausbildung** aller Beteiligten
- **Beratung** der Verantwortlichen
- ...

Tools zur Unterstützung für SOA Governance

- Top- Anbieter für Tools zur SOA-Governance laut Forrester und Gartner :
 - IBM (Rational Asset Manager, WebSphere Service Registry and Repository)
 - HP (Systinet)
 - Software AG (CentraSite)
 - In der Regel **erweiterte Service-Repositories**
 - Keine voll integrierten und vollständigen Produkte, sondern **Tool Suiten**
- **Software-/Service-Lifecycle Mgmt**



- a) Die unkontrollierte Entwicklung einer SOA verfehlt regelmäßig die grundlegende Zielsetzung einer SOA
- b) Die SOA Governance steuert und überwacht die Entwicklung
- c) Zentrale Aspekte sind dabei Planung, Umsetzung, Überwachung und kontinuierliche Verbesserung
- d) Die SOA-Governance umfasst sowohl die Unternehmens- als auch die IT-Organisation
- e) Der Einsatz diverser Methoden des Informations- und Change-Managements ist wichtig!
- f) Die anspruchsvollen Aufgaben müssen durch ein Kompetenz-Team unterstützt werden
- g) Unterstützung durch Tools ist erforderlich:
In der Regel ein zentrales Repository mit einem darauf aufbauenden Werkzeugkasten



Ludwig-
Maximilians-
Universität
München



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 7. Juli 2009

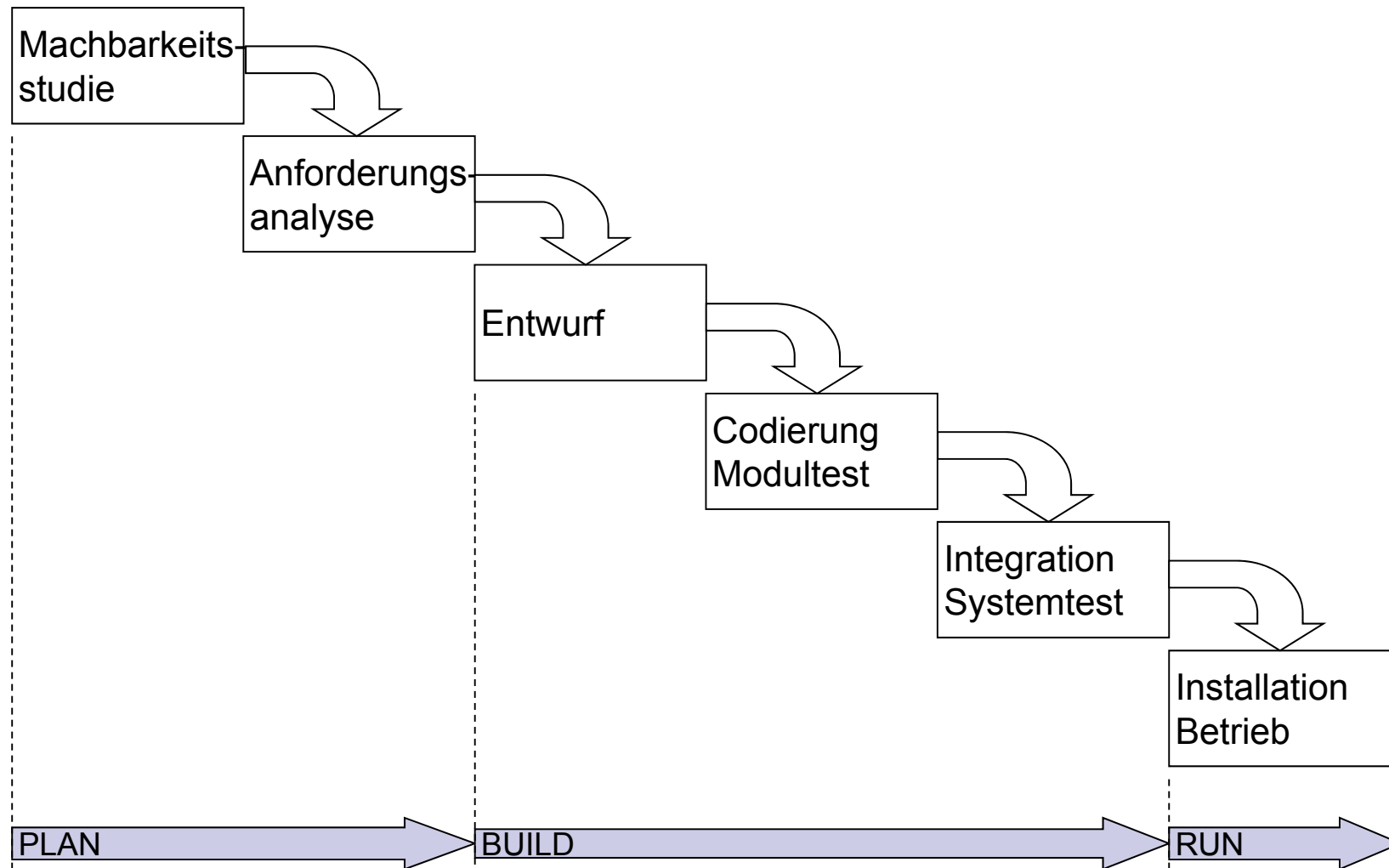
Serviceorientiertes E-Government

Software Lifecycle mit SOA

Dr. Frank Sarre

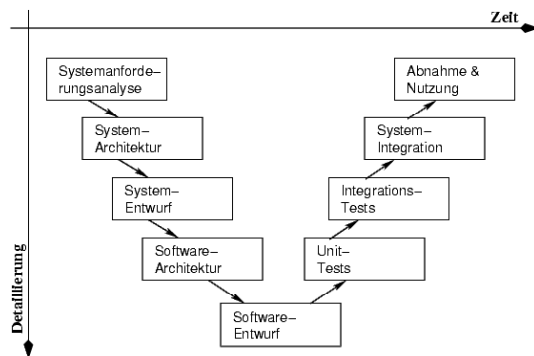
Lehrbeauftragter der LMU München

Klassisches Wasserfallmodell

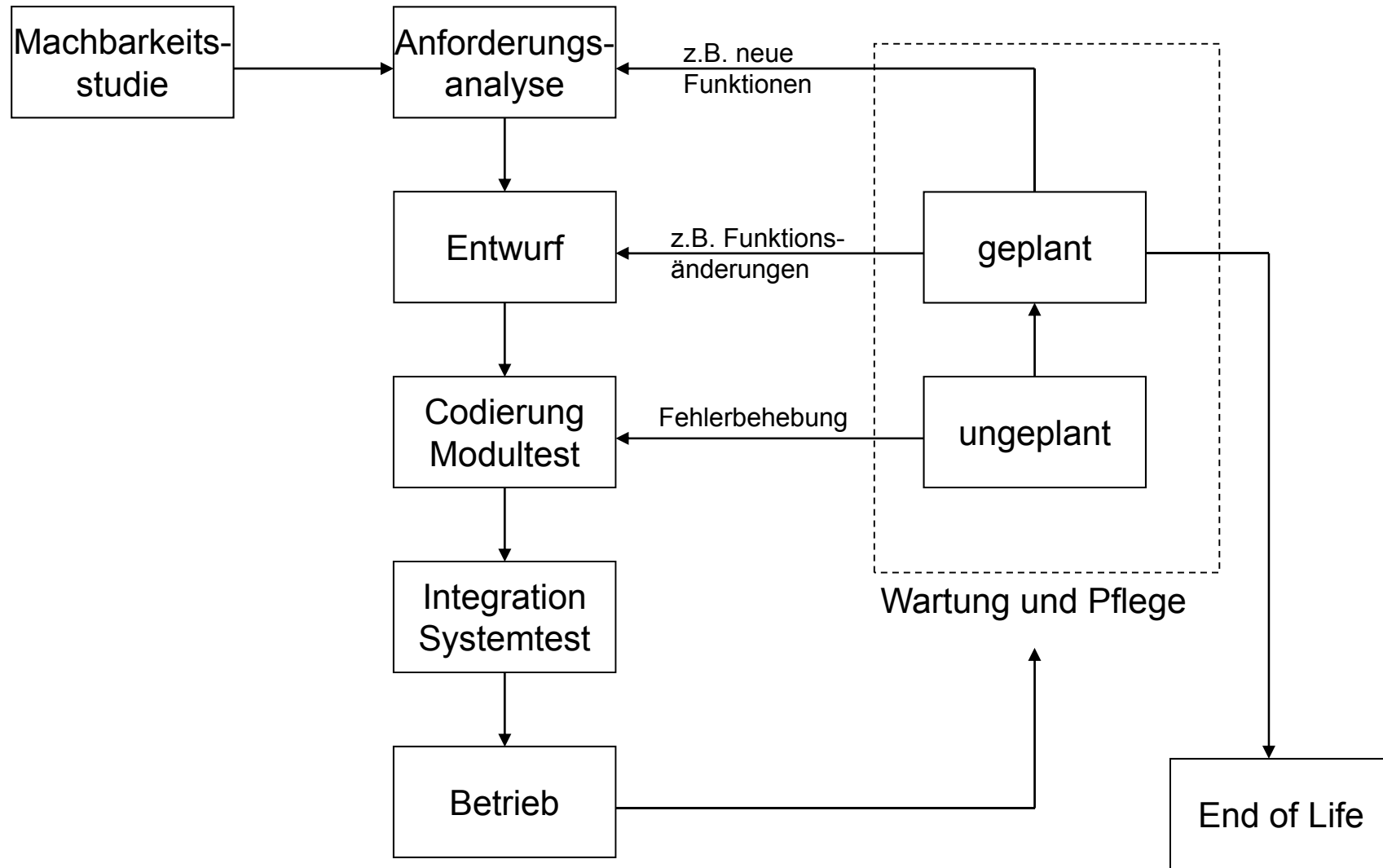


Software Lifecycles [2]

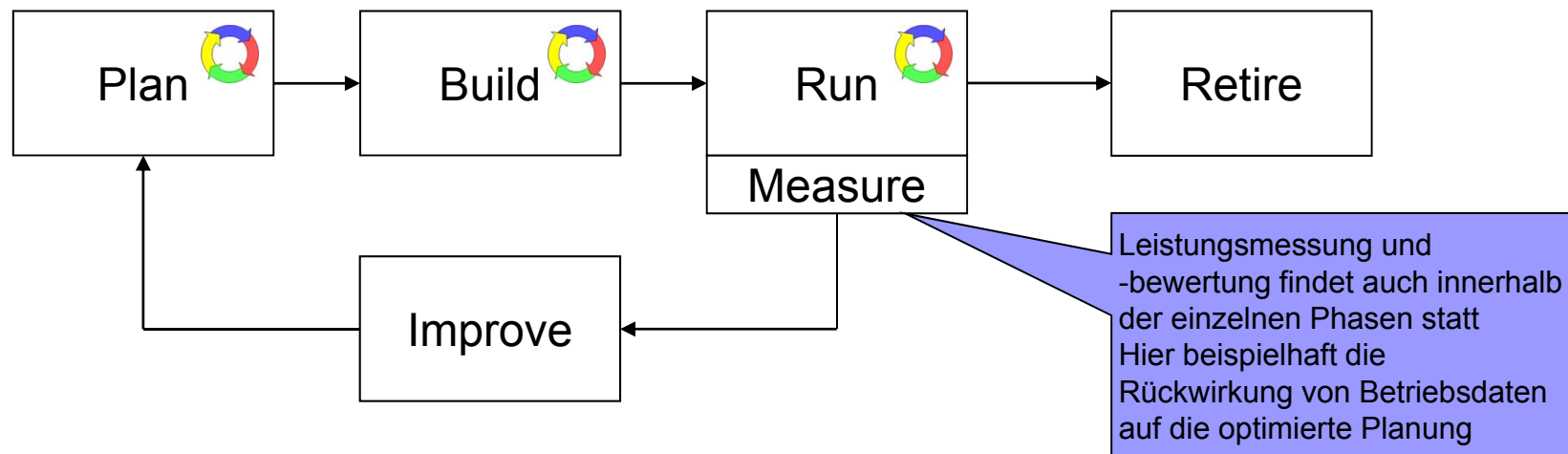
- (Rapid) Prototyping
- Spiralmodell
- Extreme Programming (XP)
- Agile Software Development
- V-Modell XT
- Capability Maturity Model (CMM)
- Test Driven Development
- Rational Unified Process (RUP)
- ...



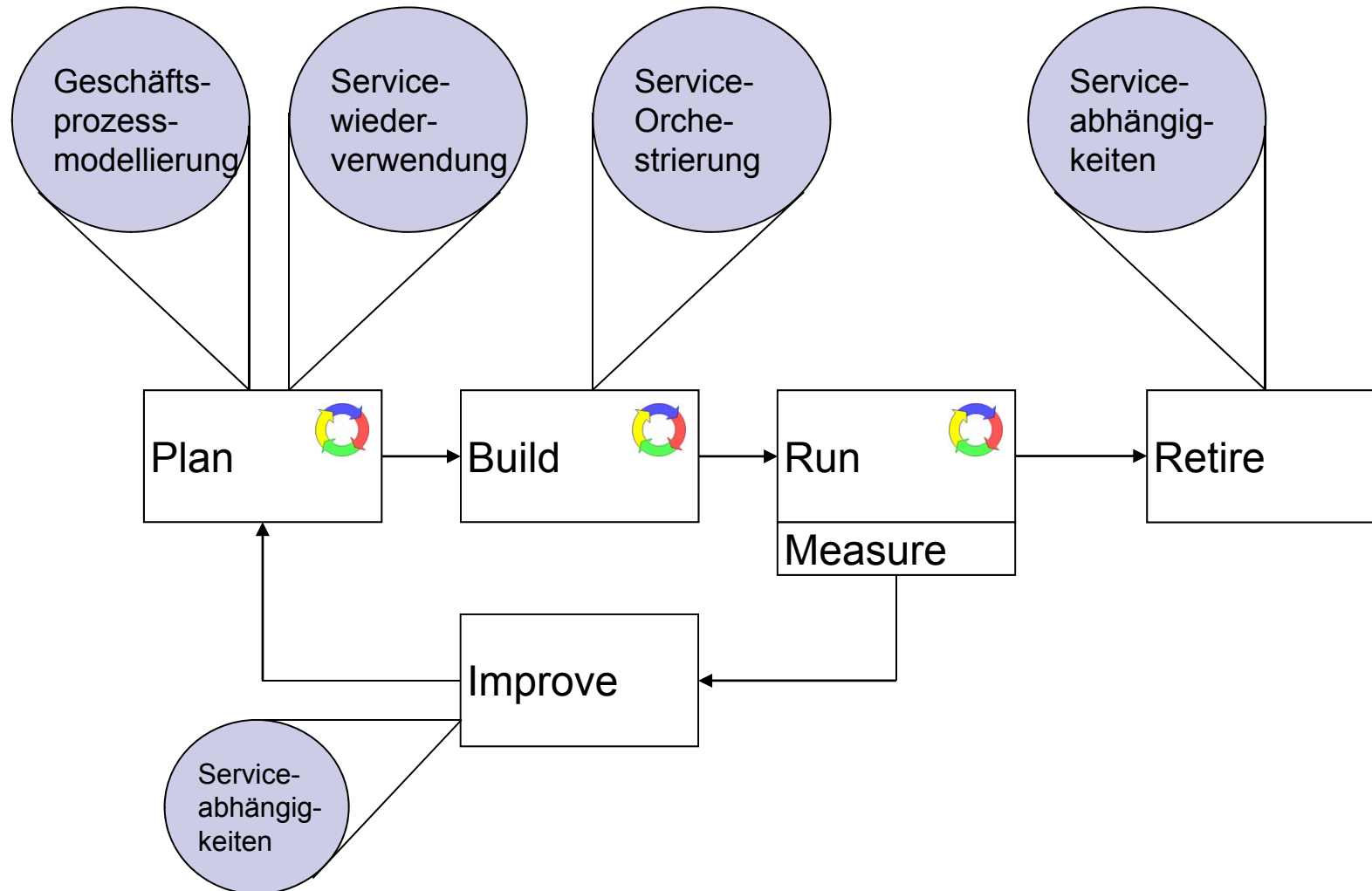
Software Lifecycles [3]



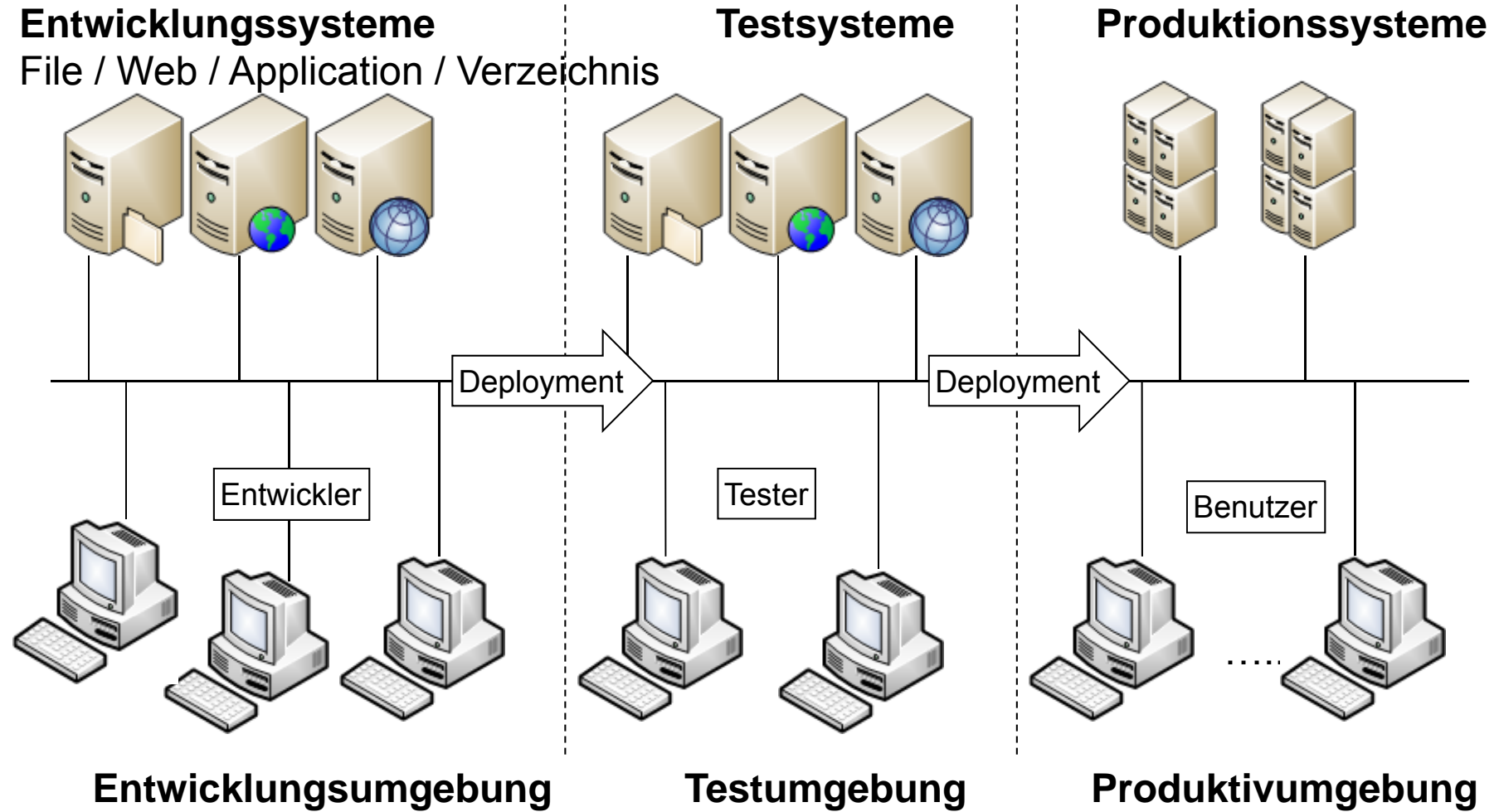
- In den meisten Entwicklungsmodellen gibt es in der Regel immer gleiche **Phasen**, auch wenn diese im Detail anders ausgeprägt sind
 - Die meisten Modelle berücksichtigen **nicht** die Aktivitäten nach der Fertigstellung und Inbetriebnahme (Betrieb, Wartung, ...)
- Einsatz ergänzender Methoden, z.B. ITIL



SOA-Einflüsse



Von der Entwicklung zum produktiven Einsatz



Die Implementierung und das Deployment von Software durchlaufen üblicherweise folgende Stufen:

- (1) Entwicklung und Modultest in einer Entwicklungsumgebung
- (2) Integrationstests in einer Testumgebung
- (3) Produktivsetzung in der Produktionsumgebung
- (4) Fallweise wird auch noch eine weitere Stufe, ein sog. *Staging System* vor der Produktivsetzung eingeführt, in der ein möglichst realitätsnahes Abbild der Produktivumgebung für den Integrationstest verwendet wird

Übliche Probleme

- Verfügbarkeit realistischer Dummies für die Nachbarsysteme bzw. Schnittstellen
- Definierter „Transport“ der entwickelten Komponenten von einer Umgebung in die andere

Idealisierte Ausgangslage

- Anwendungslandschaft ist in fachliche Komponenten zerlegt
- Basisservices behandeln Geschäftsobjekte und einfache fachliche Funktionen
- Funktionsservices bilden spezielle Fachanwendungslogik ab
- Prozessservices steuern aufgabenübergreifende Abläufe
- Services werden von unterschiedlichen Betreibern zur Verfügung gestellt
- Services sind teilweise intern und teilweise extern angesiedelt
- Servicenutzung unterliegt definierten Regeln und Kosten
- Services haben eine definierte Leistungsfähigkeit und Verfügbarkeit

Planung neuer Geschäftsprozesse

- Gibt es schon ähnliche Geschäftsprozesse?
- Können diese als Muster verwendet werden?
 - Verwendung von Templates oder „Best Practice“ Ansätzen
- Gibt es Überschneidungen mit anderen Geschäftsprozessen?
- Welche Services werden dort verwendet?
 - Wiederverwendung bestehender Services
- Kann ein Geschäftsprozess komplett auf Basis bestehender Services abgebildet werden oder müssen neue bzw. modifizierte Services geschaffen werden?



Die Abstimmung zwischen der Geschäftsprozessmodellierung und vorhandenen IT-Services ist sinnvoll und notwendig.

Planung neuer Services

- Gibt es schon ähnliche Services?
- Wem gehört ein bestehender Service?
- Zu welchen Konditionen kann man diesen nutzen?
 - Wiederverwendung bestehender Services,
evtl. durch Modifikation / Erweiterung eines bestehenden Services
- Kann ein neuer Service evtl. für andere nützlich sein?
 - Berücksichtigung erweiterter Anforderungen
- Wer darf neue Services zu welchen Konditionen nutzen?
 - Berücksichtigung des Berechtigungskonzepts
- Welche Service Levels sind erforderlich?
 - Definition eines Quality of Service

Planung neuer Services

- Gibt es Services **in Entwicklung**, die der neue Service bald nutzen kann?
- Gibt es **geplante** Services, die der neue Service in absehbarer Zeit nutzen kann?
- Soll ein potenziell genutzter Service evtl. bald wieder abgeschaltet werden?
 - Wiederverwendung von Services
 - Status der Services muss bekannt sein

Entwurf, Implementierung und Test von Services

- Entwickler brauchen Zugriff auf die Serviceinformationen inklusive aller Randbedingungen für die Nutzung
- Wie in der klassischen Softwareentwicklung wird auch hier ein CVS benötigt, das die Entwicklung in einem Team ermöglicht
- Für Modultest sind geeignete Testframeworks erforderlich
- Für Integrationstests müssen Service-Dummys bzw. Kopien der echten Services zur Verfügung stehen
 - Wer liefert die Dummys?
 - Müssen mit jedem Service immer auch Testschnittstellen entwickelt werden?

Entwurf, Implementierung und Test von BPEL-Prozessen

- Bei der Entwicklung von BPEL-Prozessen werden die Servicebeschreibungen aller genutzten Services benötigt
 - Zugriff auf Verzeichnisdienste
- Beschreibungen müssen beim Entwurf und bei der Implementierung noch nicht vollständig sein, z.B. muss binding und port erst zur Laufzeit verfügbar sein
 - Berücksichtigung von Services im Planungs- und Entwicklungsstadium
- Für den Funktionstest müssen binding und port auf Testservices „gemapt“ werden
 - Dummy-Services bzw. Kopien der echten Services erforderlich
 - Gibt es Testaccounts? Wer verwaltet diese?

Identity- und Rechtemanagement

- Im Rahmen der Entwicklung müssen **berechtigte Nutzer** und die entsprechenden Zugriffsrechte berücksichtigt werden
- In **Entwicklungsumgebungen** haben meistens alle Benutzer alle Rechte, die es gibt
- **Testsysteme** spiegeln häufig die reale Produktionsumgebung nicht vollständig wider
- Häufig fallen erst in der Produktion Fehler auf, die aufgrund falscher Berücksichtigung von Rollen und Rechten entstehen
 - Zugriff auf das zentrale Identitätsmanagement
 - Verwaltung von Rollen und Rechten im Entwicklungssystem!

- Automatisiertes Deployment von der Testumgebung in die Produktivumgebung
 - In der Regel skriptgesteuerte Lösungen, z.B. auf Basis von ANT
- Überwachung der Services zur Laufzeit:
 - Antwortzeit
 - Durchsatz
 - Latenzzeit
 - Verfügbarkeit
 - Häufigkeit der Serviceaufrufe
 - Häufigkeit einzelner Prozessschritte
 - Dauer einzelner Prozessschritte
 - ...
- Rückkopplung der Ergebnisse zur Planung und zurück in die Entwicklungsumgebung

Modifikationen an Services

- Optimierung von Abläufen
- Korrektur von Fehlern
- Einbringen neuer Funktionen

Wichtige Fragestellungen dabei:

- Welche anderen Services nutzen den zu ändernden Service?
- Was passiert, wenn Schnittstellen geändert werden?
- Was passiert, wenn die Verfügbarkeit geändert wird?
- Was passiert, wenn Nutzungsberechtigungen geändert werden?
 - Servicenutzer und Abhängigkeiten müssen bekannt sein

Abschaltung von Services

- Kann ein Service überhaupt abgeschaltet werden?
- Welche Services nutzen den Service sonst noch?
 - Servicenutzer und Abhängigkeiten müssen bekannt sein

- Ein Service kann auch durch einen neuen Service ersetzt werden
 - Auch hier müssen die Abhängigkeiten klar sein, wenn sich z.B. die Schnittstelle ändert

Service Repository zur Verwaltung von

- Servicedefinitionen
- erweiterte Serviceattribute (Metadaten)
- Serviceabhängigkeiten
- Benutzer
- Rollen / Rechte
- Service Level Agreements
- Security Policy
- Monitoringdaten
- ...

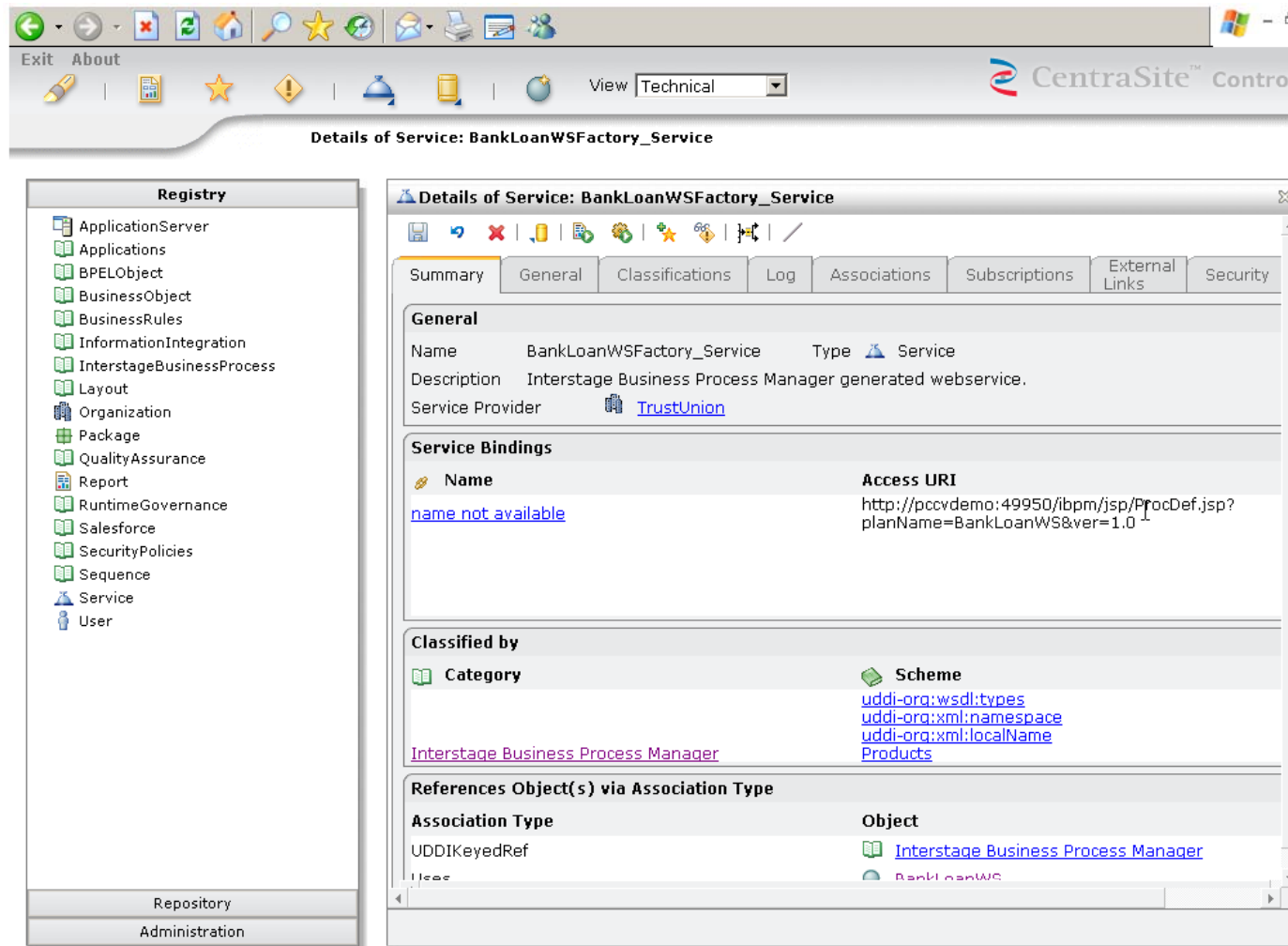


Repository allein ist nicht hinreichend!



Beispiel:
CentraSite der Software AG

Repository liefert Informationen für SOA-Architekten



The screenshot shows the CentraSite Control interface. The main window displays the details for the service **BankLoanWSFactory_Service**. The interface includes a navigation pane on the left with a tree view of the registry structure, and a main content area with several tabs: Summary, General, Classifications, Log, Associations, Subscriptions, External Links, and Security. The **General** tab is active, showing the following information:

- Name:** BankLoanWSFactory_Service
- Type:** Service
- Description:** Interstage Business Process Manager generated webservice.
- Service Provider:** TrustUnion

The **Service Bindings** section shows:

Name	Access URI
name_not_available	http://pccvdemo:49950/ibpm/jsp/ProcDef.jsp?planName=BankLoanWS&ver=1.0

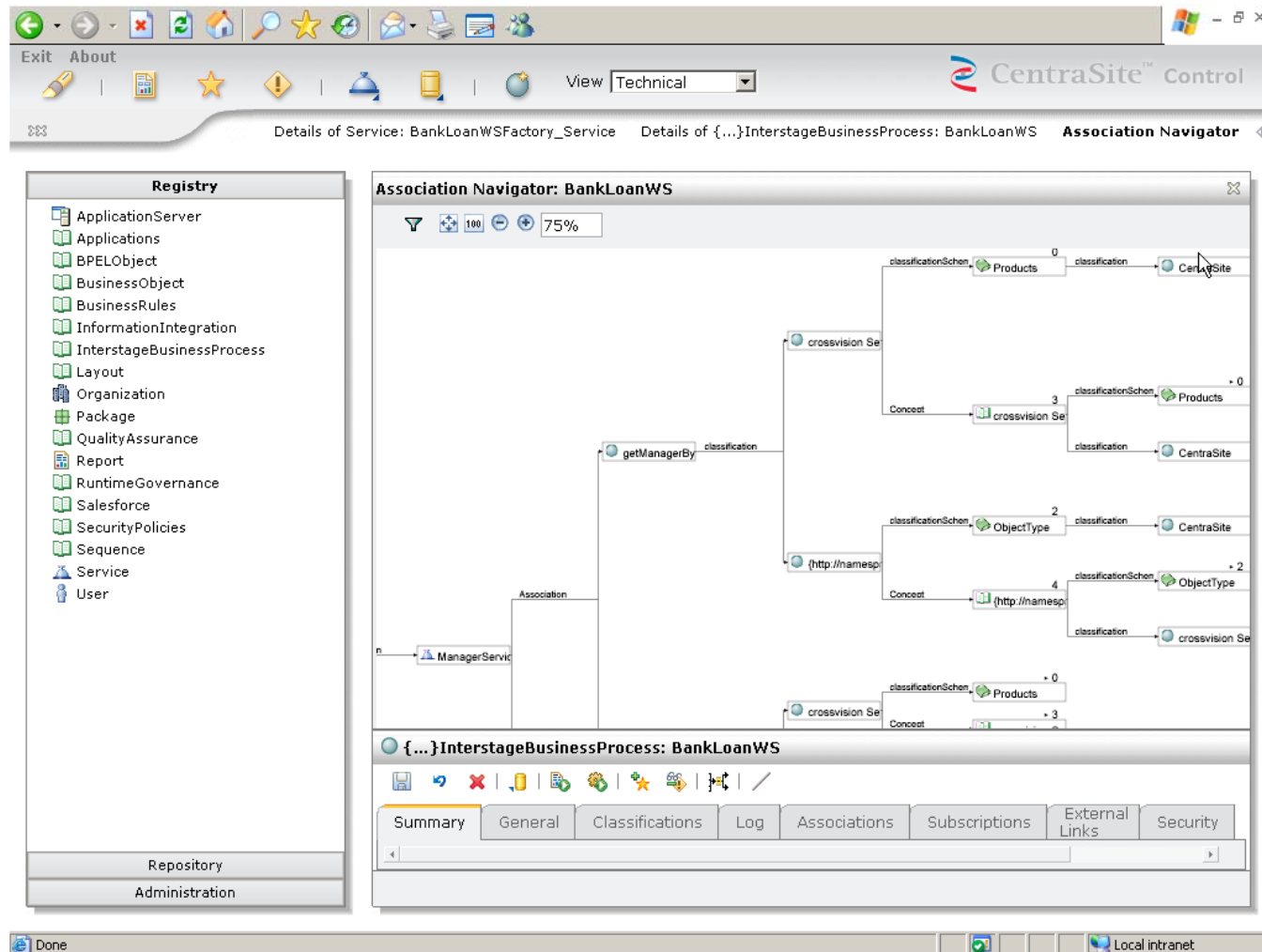
The **Classified by** section shows:

Category	Scheme
Interstage Business Process Manager	uddi-org:wSDL:types uddi-org:xml:namespace uddi-org:xml:localName Products

The **References Object(s) via Association Type** section shows:

Association Type	Object
UDDIKeyedRef	Interstage Business Process Manager
Uses	Bank Loan WS

Beispiel: Darstellung von Service-Abhängigkeiten



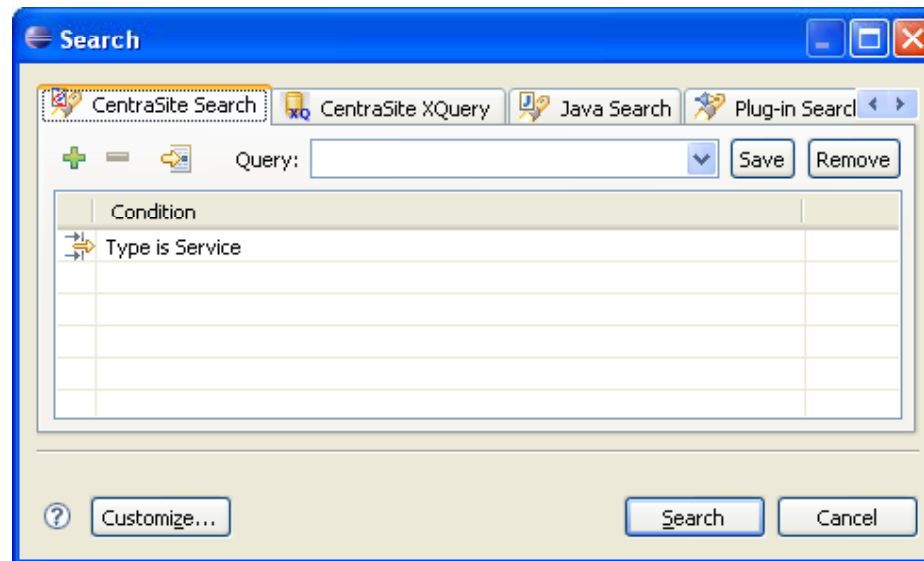
The screenshot displays the CentraSite Control interface. On the left is the 'Registry' tree with categories like ApplicationServer, Applications, BPEObject, BusinessObject, BusinessRules, InformationIntegration, InterstageBusinessProcess, Layout, Organization, Package, QualityAssurance, Report, RuntimeGovernance, Salesforce, SecurityPolicies, Sequence, Service, and User. The main window is titled 'Association Navigator: BankLoanWS' and shows a hierarchical dependency graph. At the top level, 'ManagerService' (n) is associated with 'getManagerBy' (classification). This leads to 'crossvision Se' (Concept), which further branches into 'classificationSchen' (Object Type) and 'Concept' (Concept). The 'classificationSchen' nodes are associated with 'Products' (0) and 'Object Type' (2), which in turn are associated with 'CentraSite' and 'crossvision Se' (0, 2) respectively. The 'Concept' nodes are associated with 'Products' (3) and 'Object Type' (4), which are associated with 'CentraSite' and 'crossvision Se' (0, 3) respectively. The bottom of the window features a navigation bar with tabs: Summary, General, Classifications, Log, Associations, Subscriptions, External Links, and Security. The status bar at the bottom shows 'Done' and 'Local intranet'.

Was wird neben dem Repository noch benötigt?

- Das Repository ist das Werkzeug für SOA-Architekten und Planer
- Entwickler, Tester und Betreiber der Services und Prozesse brauchen weitere Werkzeuge
 - Programmierumgebungen
 - SOA-Test-Frameworks
 - Management- und Monitoring-Tools
 - ...

Programmierumgebungen

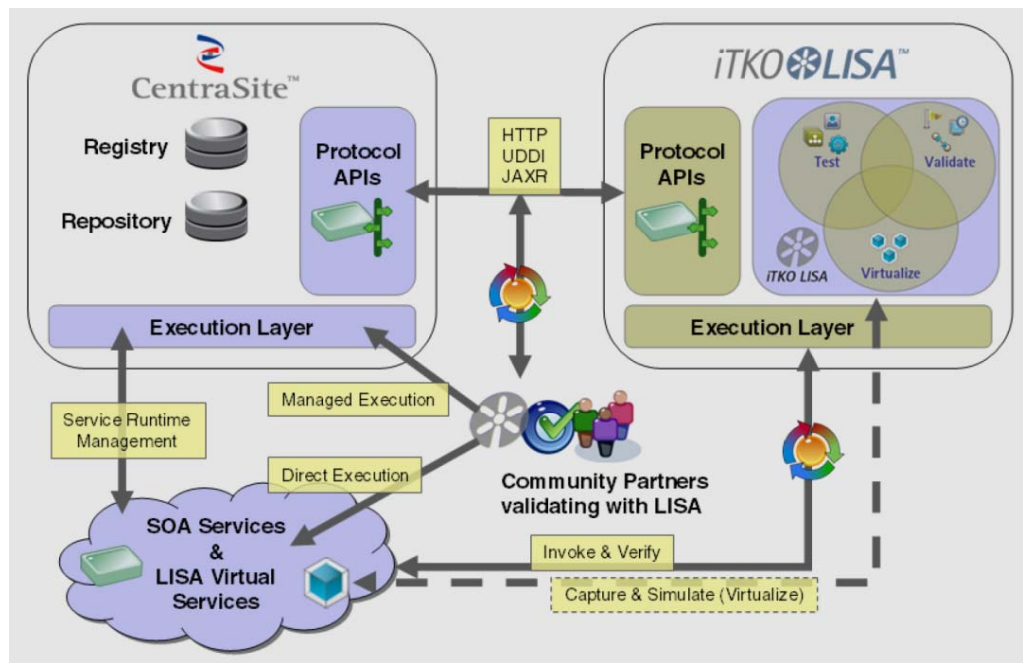
- Eclipse, JDeveloper, Visual Studio, ...
- Integration mit dem Repository erforderlich, um die Informationen im Entwicklungsprozess abgreifen zu können
- Aktualisierung der entwickelten Services im Repository



Beispiel Service Query in Centrasite über Eclipse

SOA-Test-Framework

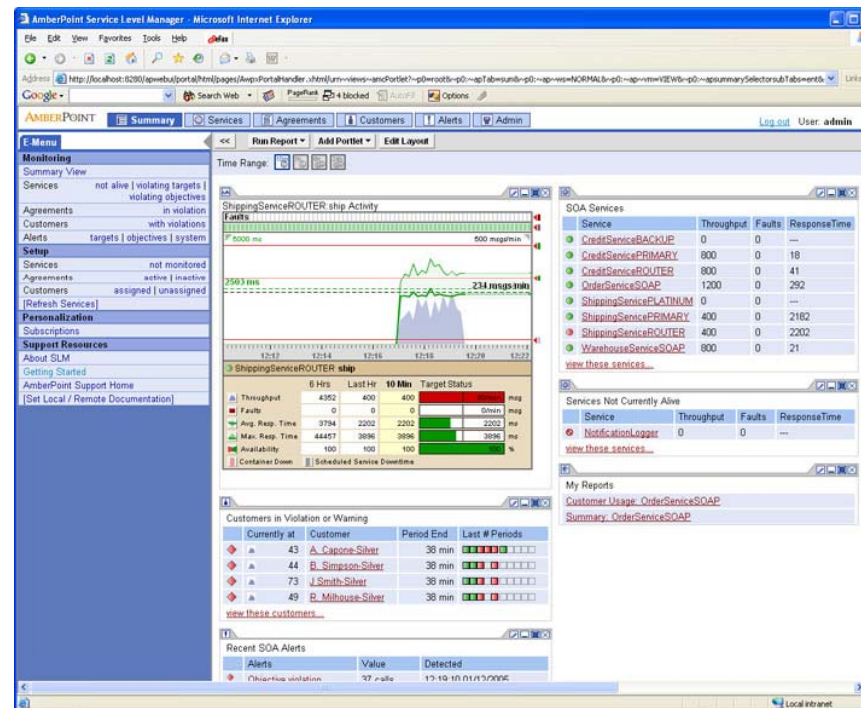
- iTKO LISA, IONA Interface Simulation and Testing Framework ...
- Besonderheit gegenüber herkömmlichen Test-Suites ist der intensive Integrationstest mit Serviceschnittstellen
- Integration zum Service Repository für die Nutzung vorhandener Serviceinformationen



Beispiel:
Integration von Centrasite und LISA

Management- und Monitoring-Tools

- AmberPoint, Actional, ...
- Überwachung der Services im Betrieb (Policies, ...)
- Monitoring-Ergebnisse müssen an das Repository zurückgeliefert werden oder über das Repository verlinkt sein



Integration von Repository und Tools

- Repositories liefern Integrationstools für gängige Programmierumgebungen wie z.B. Eclipse gleich mit
- Für Servicedefinitionen und Daten aller Art gibt es einen standardmäßigen XML-Datenaustausch
- Problem bei Daten und Informationen, die nicht über WS-Standards definiert werden!
 - Erweiterbarkeit des Repositorys über eigene XML-Schemata und Plugin-Technologien (proprietäre Herstellererweiterungen)

- Zentrale Punkte eines SOA-Software-Lifecycles sind:
 - flexible **Orchestrierung**
 - **Wiederverwendung von Services**
 - **kontrollierbarer und effizienter Betrieb**
- Effizientes Life Cycle Management muss diese Anforderungen in geeigneter Weise unterstützen
- Daraus ergeben sich hohe Anforderungen an die **Gesamtsicht** über alle verfügbaren Services und Prozesse
- Mit Hilfe von **Service Repositorys** werden alle technischen und organisatorischen Informationen zentral verwaltet
- **Alle Werkzeuge für Planung, Entwicklung und Betrieb müssen an dieses Repository angebunden sein!**



Lehr- und Forschungseinheit für Programmierung und Softwaretechnik

Vorlesung am 14. Juli 2009

Serviceorientiertes E-Government

Marktgängige Produkte und Ansätze

Fazit zur Vorlesung

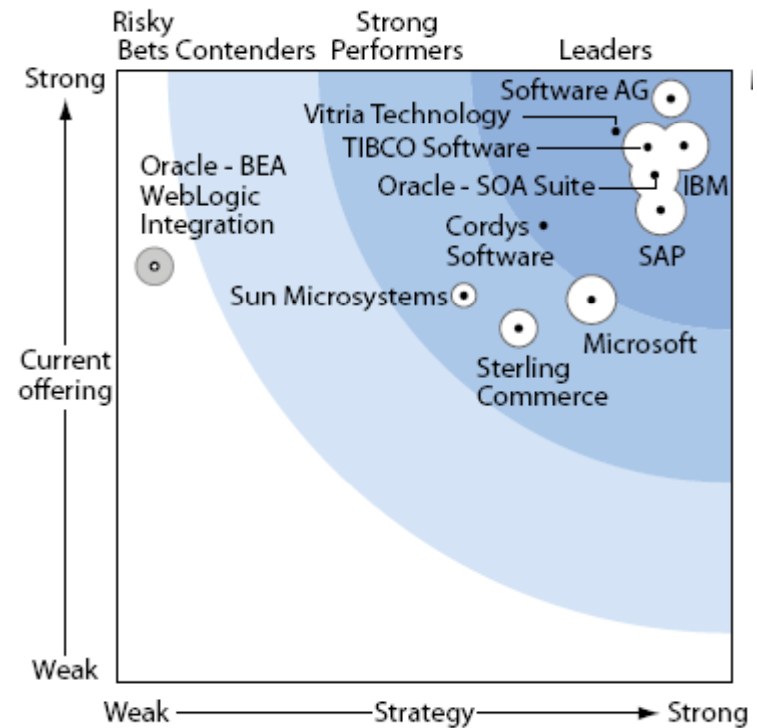
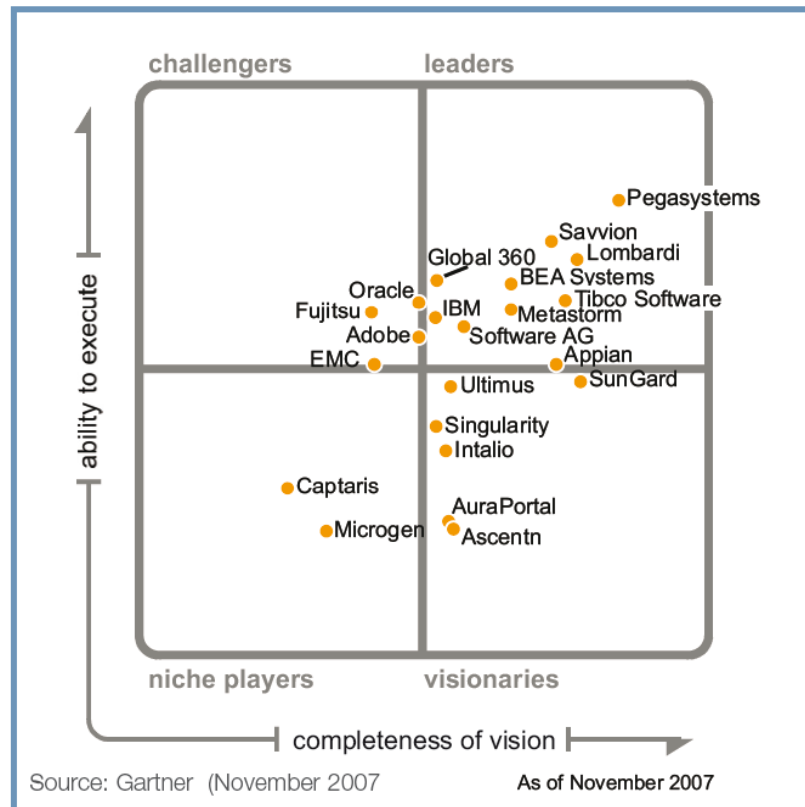
Dr. Frank Sarre

Lehrbeauftragter der LMU München

Hauptproduktbereiche von „SOA-Stacks“

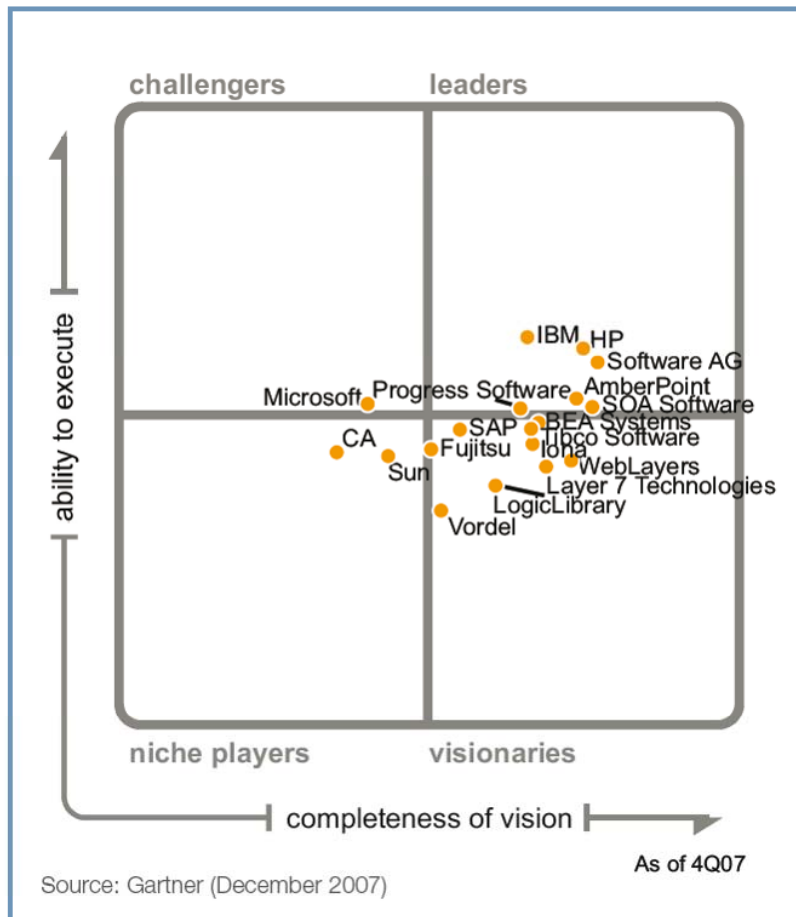
- Enterprise Service Bus
- Verzeichnisdienste / Repository
- Business Process Management
- Monitoring / Betrieb
- Identity- und Accessmanagement

Business Process Management

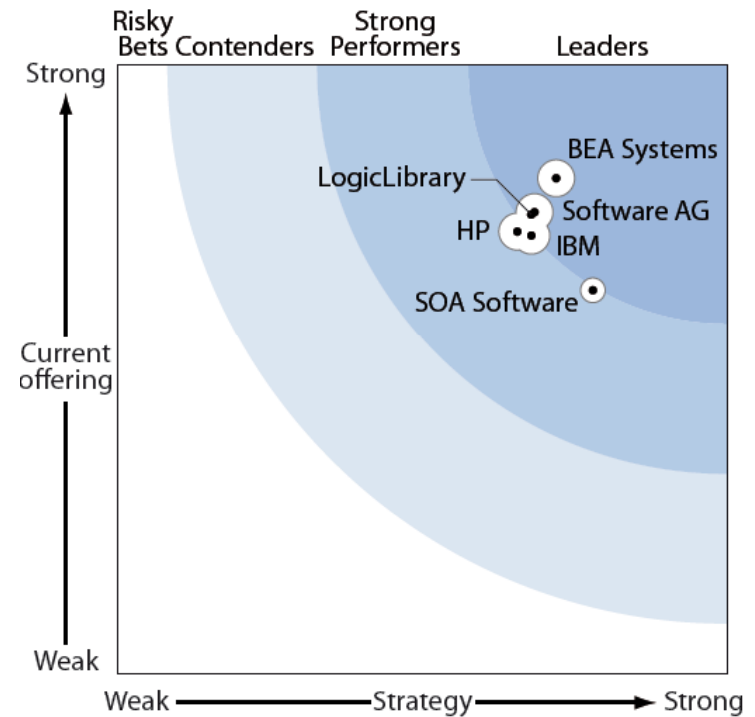


Studien von Gartner und Forrester [2]

Integrated SOA Governance

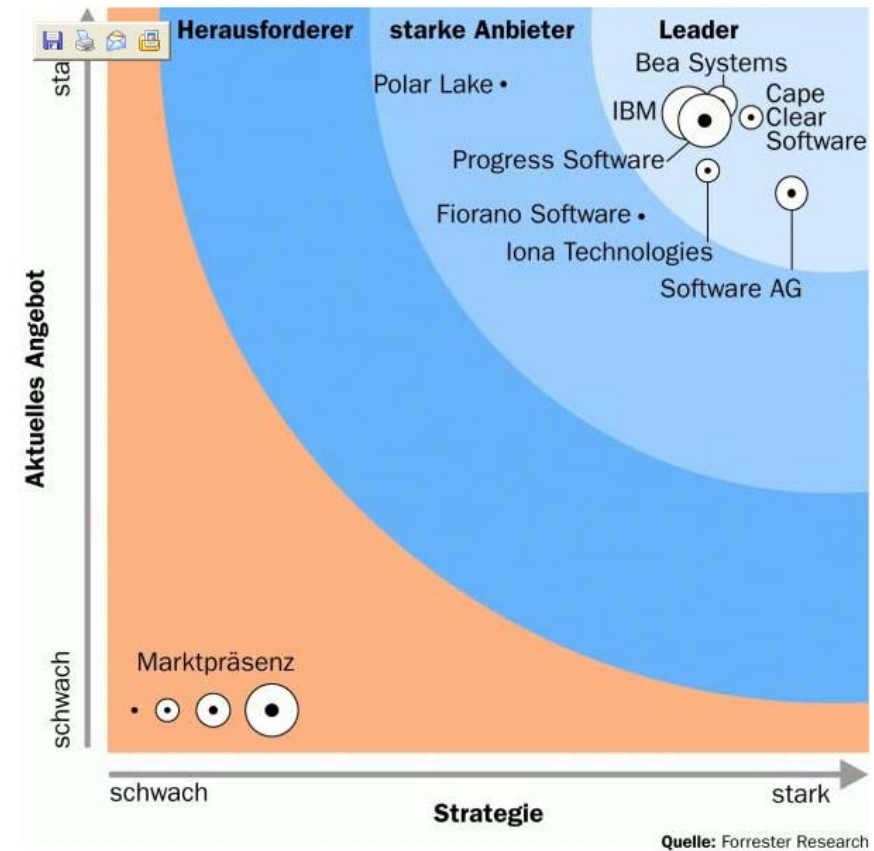


SOA Service Lifecycle Management



Quelle: Forrester Wave-Studie, Q1 2008

Enterprise Service Bus



Erkenntnisse:

- Die Studienergebnisse zeigen einige Anbieter, die in fast allen Bereichen unter den „**Leaders**“ sind
 - Angebot einer SOA-Lösung „aus einer Hand“ möglich
 - Jedoch: Nicht in allen Bereichen die beste Lösung
- In einzelnen Aufgabenbereichen gibt es Spezialisten, die eine „Best-of-Breed“-Lösung liefern könnten
 - Aufgrund von offenen Standards sollte die Integration solcher Lösungen möglich sein
- Neben den Herstellern treten auch Softwareberatungshäuser als Marktteilnehmer auf, die teils herstellerabhängig, teils herstellerunabhängig sind

- Anbieter, die das komplette Spektrum oder einen großen Teil aller Funktionalitäten anbieten:
 - IBM
 - ORACLE
 - Software AG
 - SAP
 - TIBCO
 - Microsoft

- Beispiele spezialisierter Anbieter:
 - BPM: Pega
 - ESB: IONA, Progress
 - Repository / Governance: Amberpoint, SOA Software, HP

IBM

- Umfangreiches Portfolio von Anwendungen aller Art bereits seit langem vorhanden
 - Modellierung
 - Implementierung
 - Betrieb
- Unterschiedliche Produktfamilien werden kombiniert bzw. werden konsolidiert
 - Rational
 - Websphere
 - Tivoli
- Angebot eines Produktportfolios von SOA-Produkten unter dem Label „**SOA Foundation**“

IBM SOA Foundation

Modellierungsphase

- WebSphere Business Modeler
- Rational Software Architect

„Zusammenstellungsphase“ (Begriff von IBM)

- WebSphere Integration Developer
- Rational Application Developer
- Lotus Component Designer
- Lotus Forms
- Lotus Domino Designer
- WebSphere Portlet Factory
- Rational Tester for SOA Quality

IBM SOA Foundation (Fortsetzung)

Implementierungsphase

- WebSphere DataPower SOA Appliances
- WebSphere Process Server
- WebSphere ESB
- WebSphere Message Broker
- WebSphere Partner Gateway
- WebSphere Adapters
- WebSphere Portal
- InfoSphere Information Server
- WebSphere Application Server
- WebSphere Extended Deployment
- WebSphere Business Services Fabric
- Lotus Expeditor
- FileNet P8 Platform

IBM SOA-Foundation (Fortsetzung)

Betrieb

- WebSphere Business Monitor
- Tivoli Composite Application Manager for SOA
- Tivoli Composite Application Manager for WebSphere
- Tivoli Identity Manager
- Tivoli Access Manager
- Tivoli Federated Identity Manager
- Tivoli Provisioning Manager

Governance und Best Practices

- WebSphere Service Registry and Repository

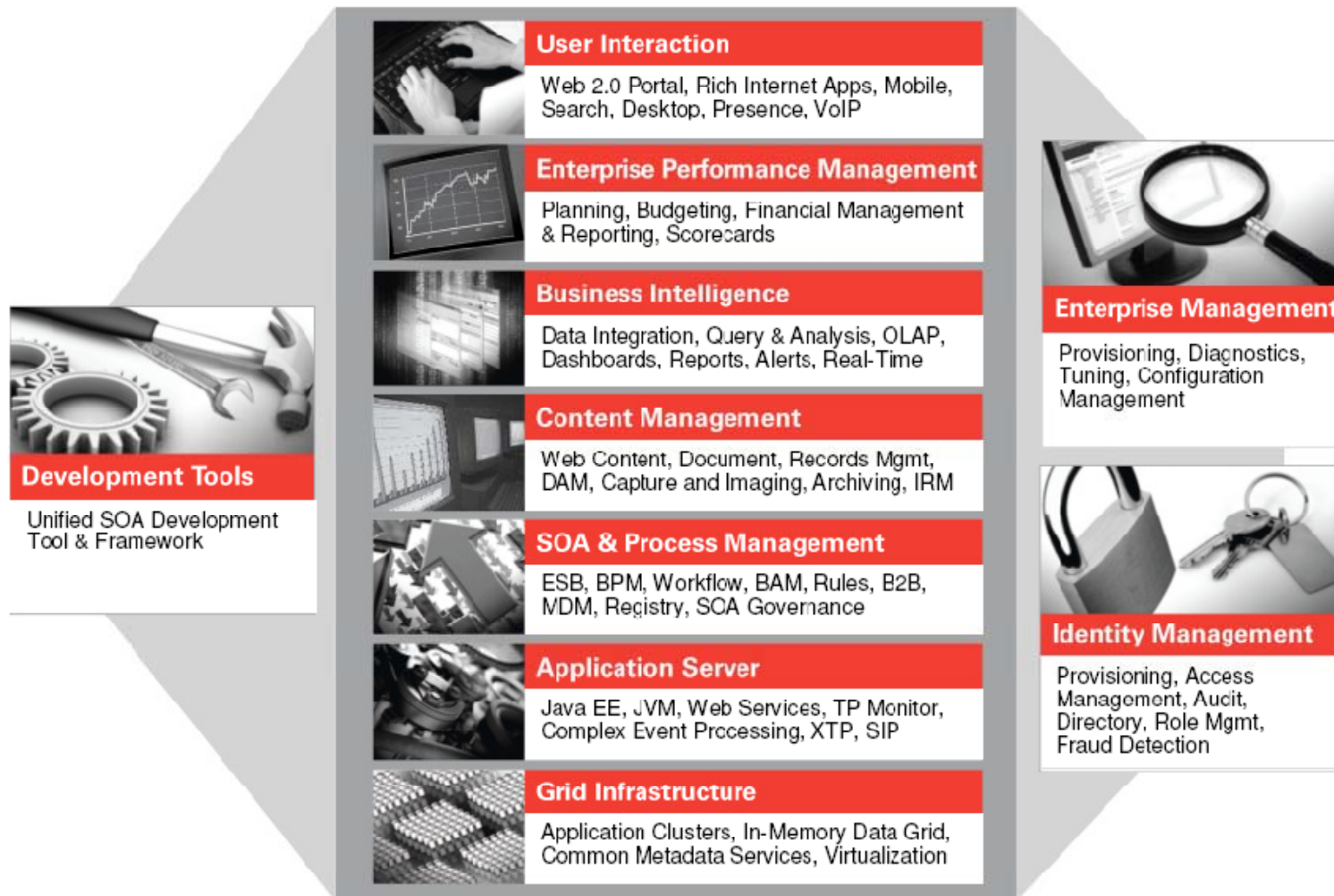
ORACLE

- ORACLE hat die eigene Marktposition im Segment „SOA-Technologien“ mit dem Zukauf von BEA wesentlich verstärkt

- Die ORACLE **SOA Suite** ist eine Sammlung der typischen Anwendungen im SOA-Bereich:
 - Oracle JDeveloper 10g (Implementierung)
 - Oracle Business Rules (Implementierung)
 - Oracle BPEL Process Manager (Modellierung, Implementierung)
 - Oracle Enterprise Service Bus (Integration)
 - Oracle Business Activity Monitoring (Betrieb)
 - Oracle Web Services Manager (Betrieb)

- Die Suite integriert sich in die **ORACLE Fusion Middleware**

ORACLE Fusion Middleware



ORACLE (Fortsetzung)

SOA-Governance-Tools

- Oracle Enterprise Repository (BEA)
- Oracle Web Services Manager
- Oracle Enterprise SOA Management Pack
- Oracle Service Registry (UDDI V3-Registry)

Offensichtliche Strategie der Firma ORACLE

- Akquisition von marktführenden Spezialprodukten, z.B. BEA
Problem: Integration unterschiedlicher Produkte
Integration von BEA jedoch innerhalb eines halben Jahres gut gelöst

Software AG

- Bisheriges Geschäftsfeld „Enterprise Transaction Systems“ mit der Datenbank *Adabas* und der Programmiersprache *Natural*
- Neue Ausrichtung auf SOA mit dem Kauf der *webMethods-Suite*:
 - webMethods CAF (Content Application Framework)
(Entwicklung von Web-Oberflächen und Portalen)
 - webMethods ESB
(Integration und Kommunikation)
 - webMethods BPM
(Modellierung mit BPMN, nicht BPEL)
 - webMethods BAM (Business Activity Monitoring)
(Monitoring und Auswertung aller Systemfunktionen)

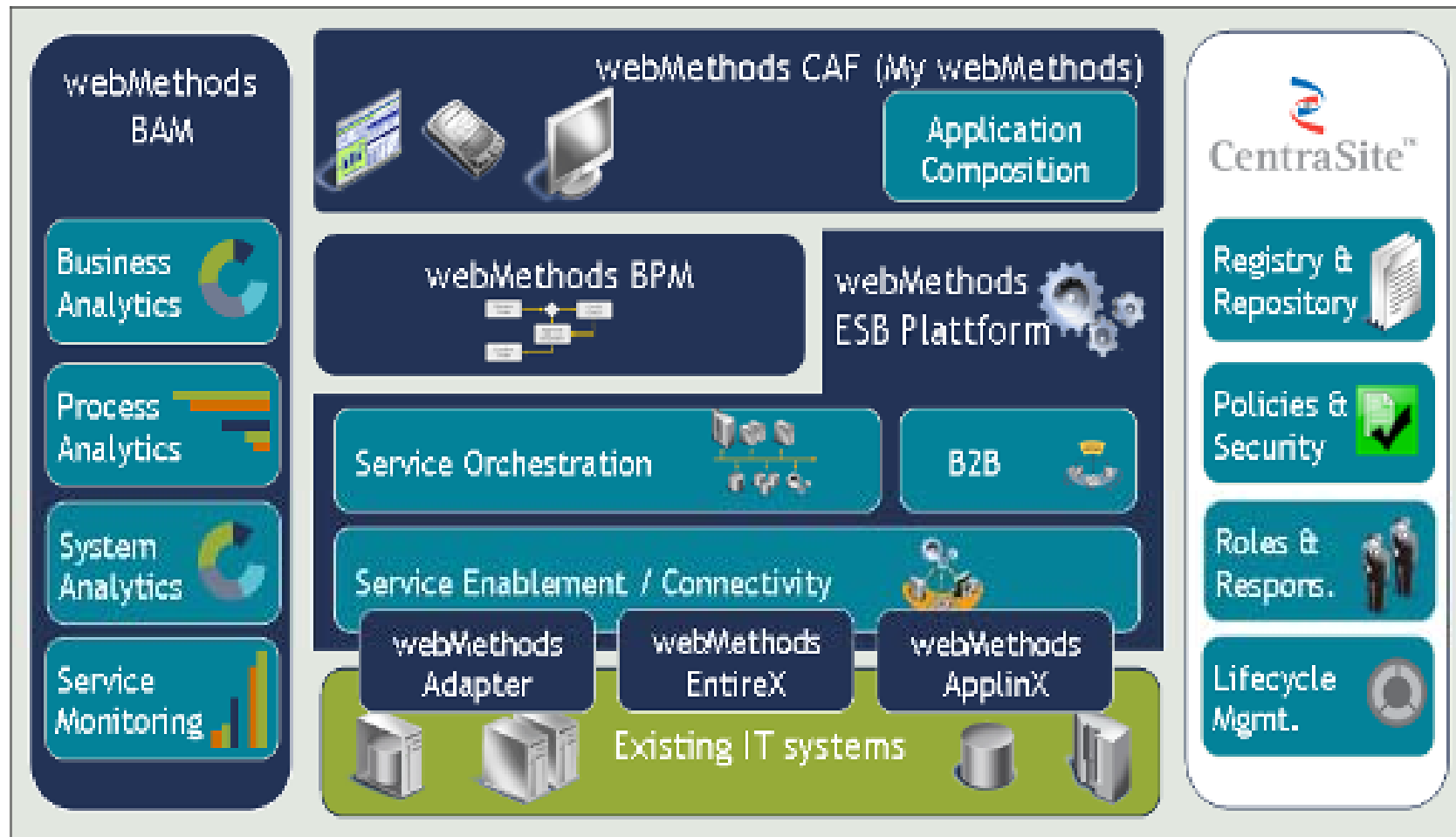
Software AG (Fortsetzung)

- Integration eigener Produkte in das SOA-Portfolio
 - EntireX
(Kapselung bestehender Programme in Services)
 - ApplinX
(Erstellung von Web-Oberflächen auf Basis bestehender Anwendungsoberflächen, z.B. auch Großrechneranwendungen)
 - Tamino XML-Server
(Native XML-Datenbank)
 - **CentraSite**
(Registry/Repository und SOA-Governance)

- Es wird vor allem die Umgestaltung vorhandener Systemlösungen in eine SOA adressiert („**Application Modernization**“)

Software AG (Fortsetzung)

Quelle: Software AG



Stärken

- Der Bereich des Business Process Management ist durch die Integration von webMethods eine besondere Stärke
- SOA-Governance ist durch flexibel erweiterbares Repository umfangreich möglich

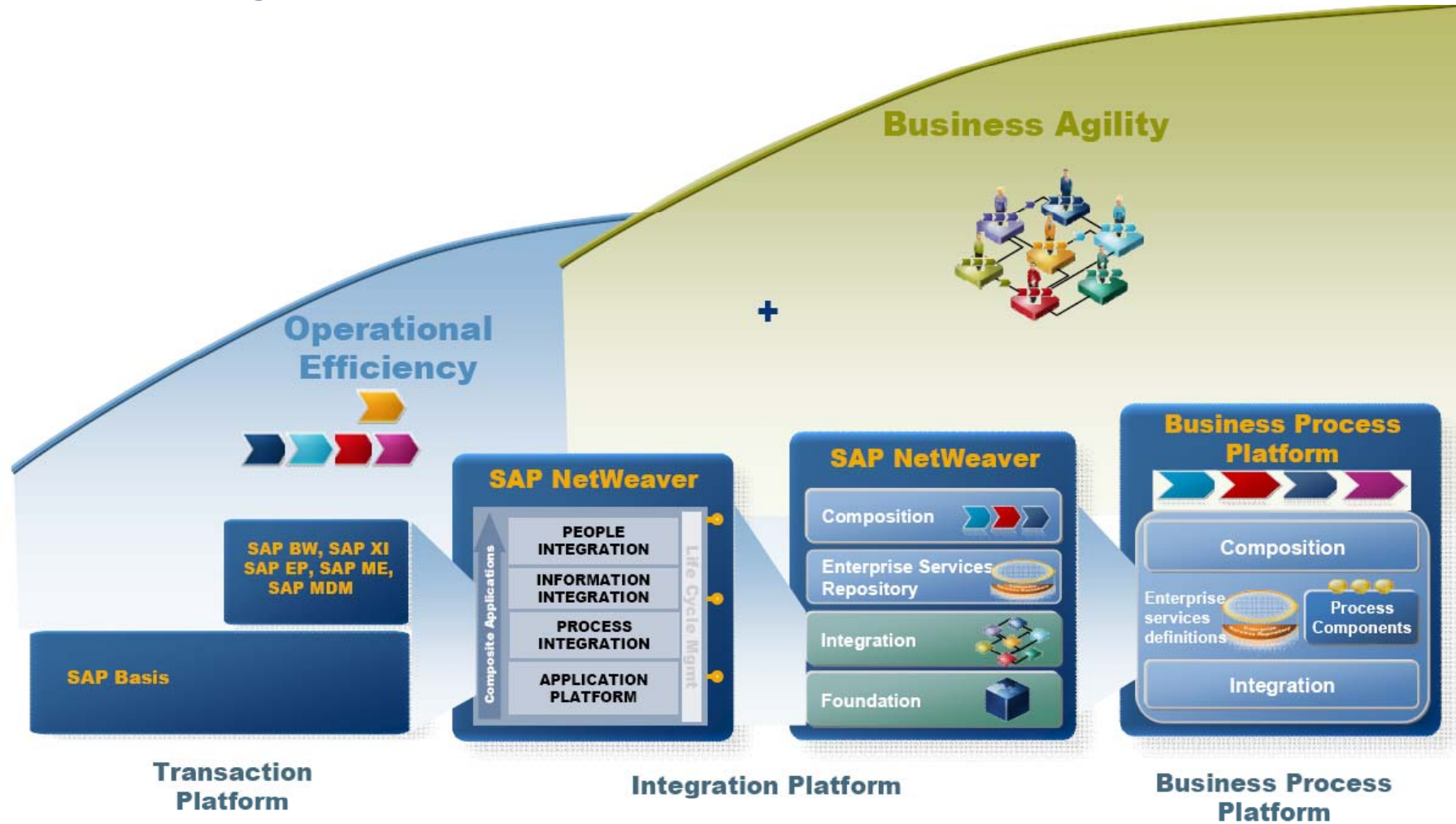
Schwächen

- Portfolio ist nicht so vollständig wie bei IBM oder ORACLE

SOA-Suites: SAP [1]

SOA-Strategie

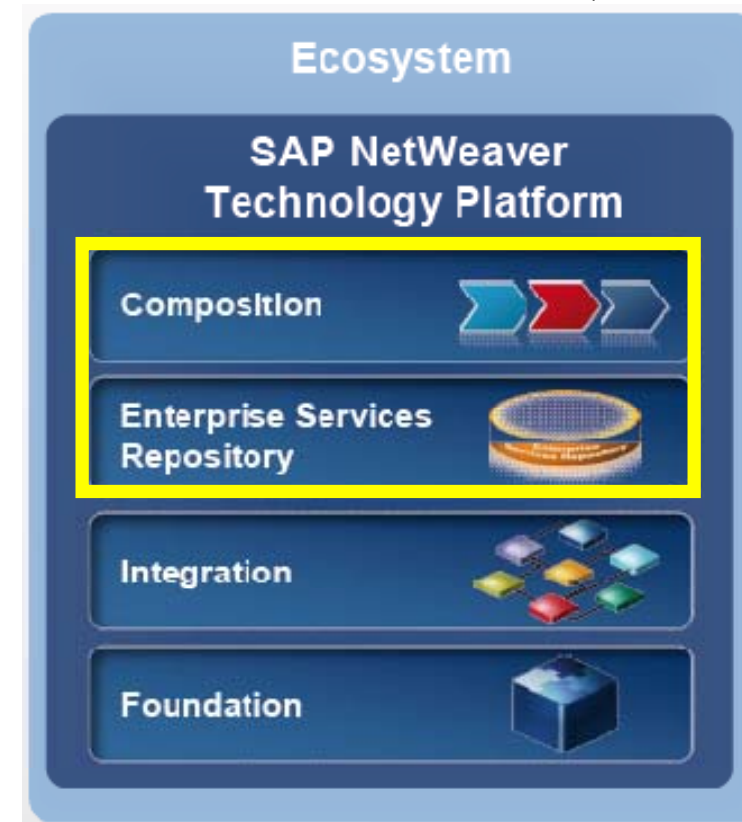
Quelle: SAP



SAP Netweaver

- Die bestehenden Netweaver-Technologien *Integration* und *Foundation* wurden ergänzt:
 - Composition
(Geschäftsprozesse,
Service-Orchestrierung)
 - Enterprise Service Repository
(Service / Software-Lifecycle,
SOA-Governance)

Quelle: SAP



- SAP bietet alle Funktionalitäten des bestehenden ERP-Systems:
 - Fachkomponenten in Form von „Enterprise Services“
 - Identity- und Accessmanagement inkl. Single Sign On (SSO)
 - Software-Lifecycle-Management
 - Portal-Lösung
 - ...
- „Out of the Box“-Geschäftsprozesse („Enterprise Service Bundles“)
- Durch die offenen Web-Service-Standards wird es möglich, eigene Anwendungen mit den SAP-Services und den Standardprozessen zu verbinden
- „Composition Environment“ (Java-Entwicklungsumgebung auf Basis von Eclipse)
- SAP setzt einen Schwerpunkt auf „Globale Datentypen“

Enterprise Service Bundles (SAP Core)

Sales

- [Account and Contact Management](#)
- [Activity Management](#)
- [Customer Fact Sheet](#)
- [Customer Service Execution](#)
- [Customer Quote Management](#)
- [Interactive Selling](#)
- [Opportunity Management](#)
- [Order to Cash](#)
- [Order to Cash with CRM](#)
- [Product Master Data Management](#)
- [Quote to Order for Configurable Products](#)
- [Rebate Management](#)
- [Sales Contract Management](#)
- [Sales Incentive and Commission Management](#)
- [Territory Management](#)

Service

- [Complaint Management](#)
- [Customer Service Execution](#)
- [Installed Base Management](#)
- [Service Contract Management](#)
- [Service Order Management](#)

Marketing

- [Campaign Management](#)
- [Lead Management](#)

Channel Management

- [Request for Registration Processing](#)

Human Capital Management

- [HCM Enterprise Learning](#)
- [HCM Master Data](#)
- [HCM Organizational Management](#)
- [HCM Time Management](#)

Corporate Services

- [Asset Configuration](#)
- [Easy Inspection Planning](#)
- [Environment, Health, and Safety \(EH&S\)](#)
- [Integration of Quality Management Systems](#)
- [Maintenance Processing](#)
- [Maintenance Service Collaboration](#)
- [Project System](#)
- [Real Estate - Room Reservation](#)
- [Travel Management](#)

E-Commerce

- [Product Catalogue Processing with CRM](#)

Demand and Supply Planning

- [Demand Planning](#)
- [Service Parts Management](#)

Financials

- [Bank Communication Management](#)
- [Credit Management](#)
- [Dispute Management](#)
- [Electronic Bill Presentation and Payment](#)
- [External Cash Desk](#)

Procurement

- [Central Contract Management](#)
- [External Requirement Processing](#)
- [Procure to Pay](#)
- [Service Procurement](#)
- [Sourcing](#)
- [Supplier Order Collaboration with SRM](#)
- [Trade and Commodity Management](#)

Supply Network Collaboration

- [Customer Collaboration for the Supply Chain](#)
- [Outsourced Manufacturing](#)
- [Supplier Collaboration for the Supply Chain](#)

Order Fulfillment

- [ATP Check](#)
- [Availability Issue Resolution and Backorder Processing](#)

Supply Chain Visibility

- [Business Event Handling for Process Tracking](#)

Product Development and Manufacturing

- [Batch Traceability and Analytics](#)
- [Integration of Manufacturing Execution Systems](#)
- [Manufacturing Work Instructions](#)
- [Technical Document Management Connectivity](#)
- [Responsive Product Development and Launch](#)

Transportation, Warehousing

- [Cross-Industry RFID Enabled Core Logistics](#)
- [Integration of External Warehouse Management Systems](#)
- [Integration of Transportation Management System](#)
- [Inventory Lookup](#)
- [Inventory Management](#)
- [Transportation Collaboration and Subcontracting](#)
- [Yard and Storage Management Processes](#)

RFID Enablement

- [Kanban Processing](#)
- [Management of Devices through Enterprise Services](#)
- [Management of Tag IDs and Tag Observations](#)

Stärken

- Der SAP-Ansatz ist gut geeignet für die SOA-Migration bestehender SAP-Installationen, da umfangreiche fachliche Komponenten zur Verfügung stehen
- Aufgrund der zahlreichen Anwendungskomponenten existiert ein großes Potential bei der Geschäftsprozessmodellierung
- Der Repository-Ansatz ist zwar erst in den Anfängen, aber ein starker Ausbau ist zu erwarten

Schwächen

- Die Einführung der Netweaver-Technologie ist in der Regeln nur in Verbindung mit dem SAP-ERP-Umfeld sinnvoll

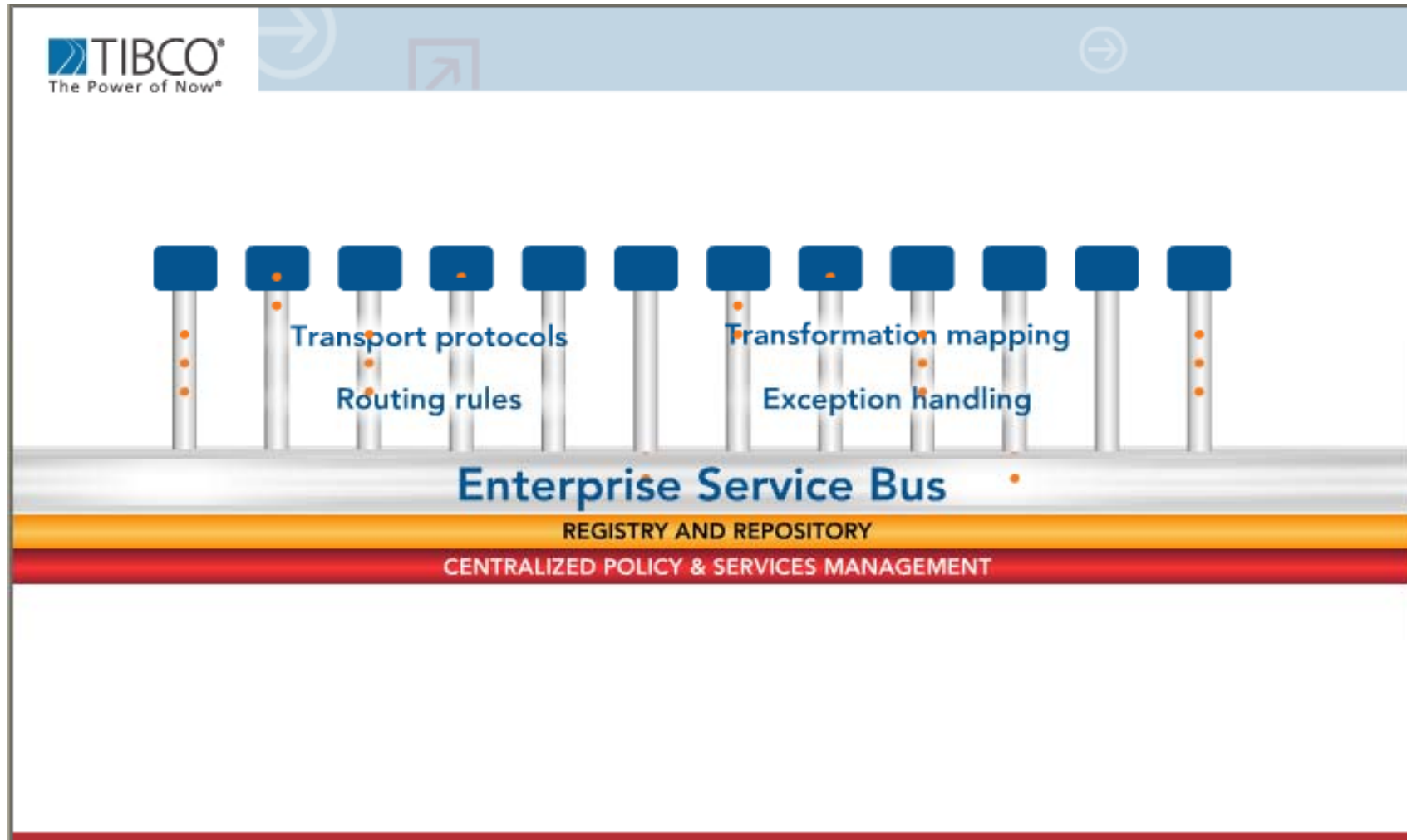
ActiveMatrix 2.0

- Service Bus
- BusinessWorks (Business Process Management)
- Policy Manager
(Security-Proxy-Management für Services und in Verbindung mit ESBs oder Registry / Repositorys)
- Registry
- Service Grid
(Service-Container / Service-Kapselung für bestehende Anwendungen)
- Performance Manager
(Monitoring)

Erweitertes Portfolio

- Neben typischen SOA-Produkten existieren weitere Tools aus dem Enterprise-Software-Bereich:
 - Diverse Tools zum Business Process Management
 - Großrechneranbindung
 - Master Data Management
 - Portal Builder
 - System Monitoring und Management

SOA-Infrastruktur



Stärken

- Kommunikations- und Serviceabstraktionsschicht
- Virtualisierung von Services
- Kapselung bestehender Anwendungen in Service-Containern
- Policy-Management
(ohne Veränderungen an bestehenden Services vornehmen zu müssen)
- Ausrichtung ähnlich wie bei der Software AG

Schwächen

- Portfolio ist nicht so vollständig wie bei IBM oder ORACLE

BizTalk

- Bestehende Produkte werden in Richtung SOA ausgerichtet
 - Service-Entwicklung: **.NET, Sharepoint**
 - Prozesse: **BizTalk**
 - Verzeichnisdienste: **Fremdprodukte (Amberpoint, SOA Software)**
 - Betrieb: **System Center, Fremdprodukte (Amberpoint, SOA Software)**
 - ESB:
Kombination aus **BizTalk**, Windows 2003, .net-Framework und SQL-Server
 - Identitätsmanagement: Active Directory, ADFS
 - Kommunikationsbasisdienste: Windows Communication Foundation

- Projekt „Oslo“:
 - Microsoft Modeling Platform
 - „Oslo“-Repository

SOA Suites: Microsoft [2]



Standardprotokolle und Adapter

BizTalk Services – “Interop in the Cloud”

Federated Identity

Federated Connectivity

Federated Workflow

BizTalk Server – “Interop In the Box”

LOB formats:

SAP
 Siebel Applications
 PeopleSoft Enterprise
 JD Edwards
 OneWorld
 JD Edwards
 Enterprise One

B2B formats:

XML
 EDI (EDIFACT, ANSI
 X12, HIPAA)
 AS2
 SWIFT
 HL7
 RosettaNet

Messaging formats:

WebSphere MQ
 MSMQ/MSMQT
 TIBCO Rendezvous
 TIBCO Enterprise
 Message Service

Legacy formats:

IBM CICS
 IBM IMS
 IBM OS/400
 IBM DB2
 IBM VSAM

Other formats:

SQL Server
 SharePoint
 Oracle DBMS
 File, FTP
 HTTP, SOAP
 POP3 / SMTP

Windows Communication Foundation – “Interop in the Framework”

Unified protocols:

ASMX
 WSE
 .NET Remoting
 COM+
 MSMQ

Messaging:

SOAP, WS-Addressing, MTOM

Metadata:

WSDL, WS-MetadataExchange,
 WS-Policy

Security:

WS-Security, WS-SecureConversation, WS-Trust

Reliability and Transactions:

WS-Reliable Messaging, WS-Coordination,
 WS-AtomicTransaction

Web 2.0

POX
 REST
 JSON
 RSS/ATOM

Stärken

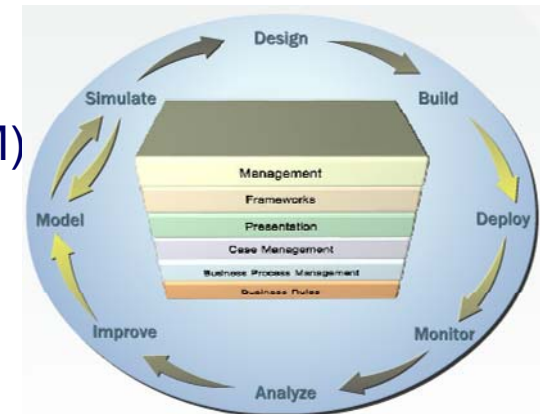
- Microsoft ist einer der Anbieter, die die Themen **Web-Services**, **Verzeichnisdienste** und **WS-Standards** forciert haben
- BizTalk ist ein umfangreiches Basisprodukt für ESB, BPM und Monitoring. Es werden zahlreiche WS-Standards unterstützt.
- Visual Studio Team System ist eine umfangreiche Entwicklungs-, Test- und Deployment-Umgebung

Schwächen

- Es fehlt ein geeignetes **Repository**, das jetzt erst im Projekt „Oslo“ entwickelt wird.
- Verschiedene Technologien sind Microsoft-spezifisch

Pega – SmartBPM-Suite

- Laut Gartner und Forrester eines der führenden Werkzeuge für **Business Process Management (BPM)**
- Process Analyzer
(Online-Analyse und -Auswertung)
- Process Simulator
(Simulation von Prozessen, bevor sie tatsächlich eingesetzt werden)
- Enterprise Integration
(Enterprise Connectors und Adapters, u.a. auch BPEL)
- Case Management
(Out-of-the-Box Lösung für eine Fall-Management-Anwendung)
- Content Management Integration
(Einbindung von Daten aus CMS und anderen Systemen)
- Portal Integration



- **IONA – Artix**
 - Artix laut Forrester einer der Market-Leader im Bereich ESB
 - Im Juni 2008 von Progress aufgekauft
 - Open Source ESB „FUSE“ auf Basis von *Apache Service Mix* (siehe Open Source Produkte)

- **Progress – SONIC / Actional**
 - **SONIC** laut Forrester auch einer der Market-Leader im Bereich ESB
 - **Actional** für Monitoring und Policy Management; integriert sich in ESB und Service Repositorys anderer Hersteller
 - Lücken im Portfolio bei Registry / Repository und BPM
 - Umfangreiche Features für Integration in Fremdprodukte

Amberpoint – SOA Management System

- SOA Discovery
(Erfassung und Darstellung von Service-Abhängigkeiten)
- Runtime Management
(Policy Management und Enforcement)
- SOA Security
- Service Level Management
(Monitoring und Management von Leistungswerten)
- Exception Management
(Regelbasierte Fehlerbehandlung auf allen Abstraktionsebenen)
- Transaction Tracking
(Protokollierung für Analysezwecke)
- Metadata Federation
(Abgleich zwischen verschiedenen Repositorys bzw. Registrys)

SOA Software

- **Policy Manager**
 - UDDI v3 Registry
 - Policy Definition
 - Contract Management
 - Monitoring
 - Wird von Microsoft mit BizTalk integriert
- **Repository Manager**
 - Erweiterung der Registry mit Metadaten
 - Federation mit HP Systinet, IBM WSSR oder TIBCO ActiveMatrix möglich
 - Synchronisation von Servicedefinitionen, Metadaten und Zustandsinformationen



SOA Software

- **Service Manager**
 - Überwachung und Enforcement von Polycys in Zusammenarbeit mit dem *Policy Manager*
 - Teilfunktionen eines ESB
 - Ergänzung von Sicherheitsfunktionen
- **SOLA**
 - Kapselung von CICS-Anwendungen als Services



HP Systinet

- Registry / Repository
- Policy Management
(Policy Management und Enforcement)
- Contract Management
(Monitoring und Management von SLAs)
- Lifecycle Management
(Erfassung und Darstellung von Service-Abhängigkeiten)

→ Wird von BEA (ORACLE) als *Aqualogic Service Registry* angeboten

Zusammenfassung (kommerzielle Produkte)

Leistungsbereiche der betrachteten SOA-Anbieter

	IBM	ORACLE / Bea	Software AG	SAP	TIBCO	Microsoft	Pega	IONA / Progress	Amberpoint	SOA Software	HP
Enterprise Service Bus (ESB)	X	X	X	X	X	X		X		O	
Verzeichnisdienste / Repository	X	X	X	X	X			X	X	X	X
Business Process Management	X	X	X	X	X	O	X	O			
Monitoring / Betrieb	X	X	X	X	X	X	X	X	X	X	X
Identity- und Accessmanagement	X	X		O		O					
Portal	X	X	X	X	X	O	X				

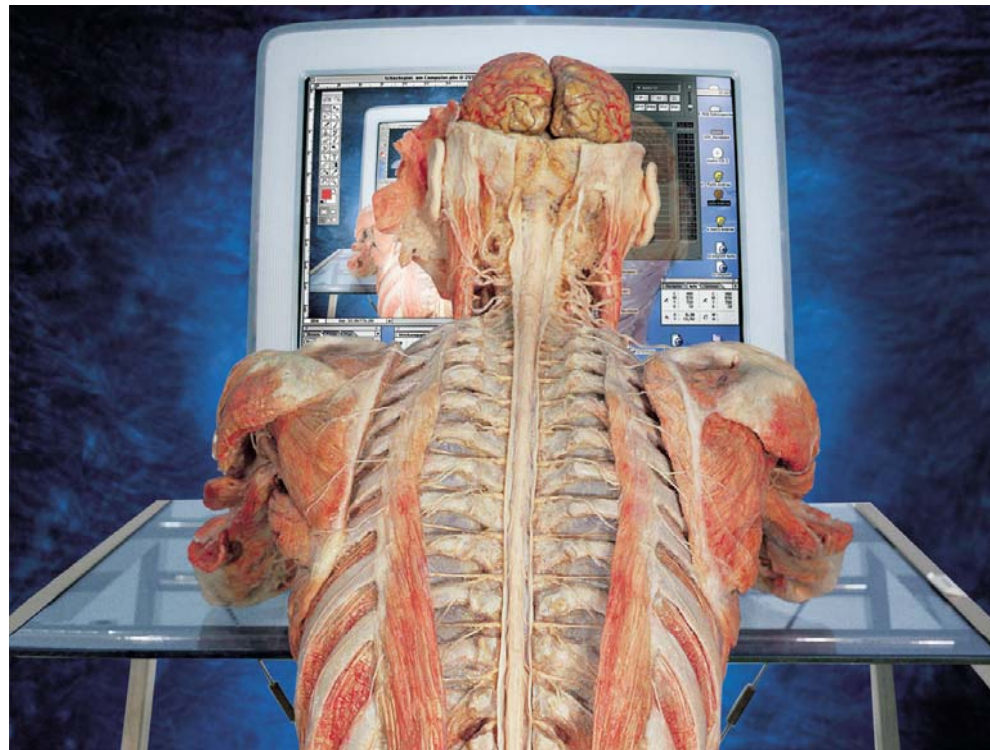
X = verfügbar O = verfügbar, aber mit Einschränkungen

Technologisches Know-how

- **Kommerzielle Produkte** werden im Allgemeinen mit einer komfortablen **Benutzeroberfläche** und umfangreicher **Dokumentation** zur Verfügung gestellt.
Die Komplexität der einzelnen Komponenten wird „unter der Haube“ versteckt.
- Bei **Open-Source-Produkte** gilt in aller Regel:
 - Die Produkte werden als **einzelne Funktionskomponenten** im **Binär- und Quellcode** zur Verfügung gestellt.
 - Individuelle Aufgabenstellungen und Probleme müssen entweder **mit eigenem Know-how** oder **Informationen aus der Entwicklungscommunity** gelöst werden.
 - **Herstellersupport** mit vereinbarten SLAs, **ist teilweise nicht zu bekommen** oder nur gegen entsprechendes Entgelt.

Kriterien für den Einsatz von Open-Source-SOA [2]

- Beim Einsatz von Open-Source-Tools lernt man die technischen Aspekte von SOA bis in die Innereien kennen!



Gunther von Hagen: Hacker

Verwendung von Standards

- **Open-Source-Produkte** müssen stark auf allgemein anerkannten **offenen Standards** aufsetzen. Ob die Interoperabilität mit anderen Produkten dadurch schon gegeben ist, hängt häufig noch von „Einzelheiten“ ab.
- Im Gegensatz dazu verwenden **proprietäre Lösungen** oft „optimierte“ Standards, also veränderte Standards, die eine problemlose Interoperabilität verhindern.
- Einige Standards sind in Open-Source-Produkten abhängig vom **Entwicklungsstand**, jedoch manchmal (noch) nicht vollständig umgesetzt.



Die Nutzung von **Standards** ist eine notwendige Voraussetzung für Interoperabilität, aber keine Garantie dafür!

Produktentwicklung

- Im **industriellen Umfeld** ist **Relaseplanung** und **Produktsupport** von hoher Wichtigkeit.
- Die Entwicklung einzelner **Open-Source-Produkte** ist oft **nicht vorhersehbar**, da es häufig **keinen Releaseplan** gibt bzw. ein vorhandener Plan in der Regel als unverbindlich anzusehen ist.
- Bei manchen Produkten reagiert die **Community** sehr schnell auf neue Anforderungen oder aufgetretene Fehler, bei manchen jedoch nicht.
- Aufgrund der in aller Regel besseren Interoperabilität **lassen sich Open-Source-Produkte leichter austauschen** als kommerzielle Produkte.

Lizenztypen

- Die **unterschiedlichen Lizenztypen**, z.B. GPL, LPGL, CPL oder EPL haben unterschiedliche Konsequenzen hinsichtlich der Nutzung und der Weitergabe!
- Bei reiner **Nutzung einer SOA-Infrastruktur** sind die Unterschiede zwar auch gegeben, jedoch nicht so gravierend.
- Hingegen gilt es signifikante Unterschiede bei den Lizenzen zu beachten, wenn SOA-Lösungen **modifiziert und vertrieben** werden!
- Bei bestimmten Lizenztypen werden **eigene Entwicklungen** automatisch ebenfalls zu Open-Source, bei anderen dagegen nicht!



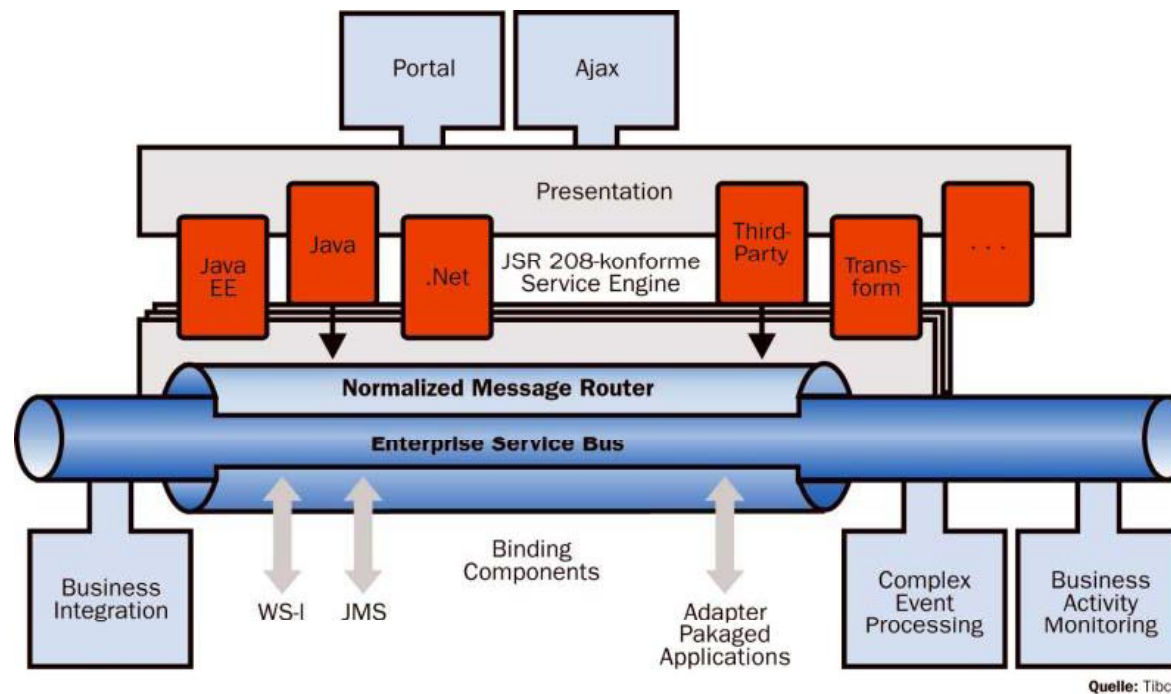
Open-Source ist kein rechtsfreier Raum!

- Im Open-Source-Sektor haben sich überwiegend Systeme auf Basis von **Java-Technologien** etabliert.
- Es gibt Standards für
 - Application Server (z.B. J(2)EE, ...)
 - Enterprise Service Bus (z.B. JBI, JMS)
 - Business Process Management (z.B. JPDL)
 - Identity- und Access-Management (z.B. JAAS, SAML)
 - Monitoring (z.B. JMX, Mbeans)
 - Portal (z.B. Portlets, JSF)

Die meisten Lösungen und Projekte setzen auf diesen Basis-Technologien auf

Beispiel: JBI – Java Business Integration

- Kernstück ist der **Normalized Message Router** (NMR) für Übermittlung und Transformation von Nachrichten, die zwischen Service-Containern ausgetauscht werden
- Implementierung auf dieser Basis: Apache ServiceMix, SUN OpenESB



SCA (Service Component Architecture)

- **SCA** ist ein **Programmiermodell** für die Entwicklung und Komposition von Geschäftsfunktionen in Komponenten
- Die Komponenten einer SCA laufen in einer SCA-Runtime-Umgebung.
- Für die unterschiedlichen **Bindings** der Services werden wie bei JBI ebenfalls **Service-Container** genutzt.
- SCA ist **keine (reine) Java-Technologie**, sondern unterstützt unterschiedliche Programmierumgebungen
- **SCA wird von fast allen kommerziellen Lösungen unterstützt**
- Im Open-Source-Bereich wird versucht, JBI und SCA zu unterstützen. Einige Projekte legen aber den Schwerpunkt auf eine bestimmte Technologie.
- Eine gute Einführung zu SCA ist zu finden unter http://www.davidchappell.com/articles/Introducing_SCA.pdf

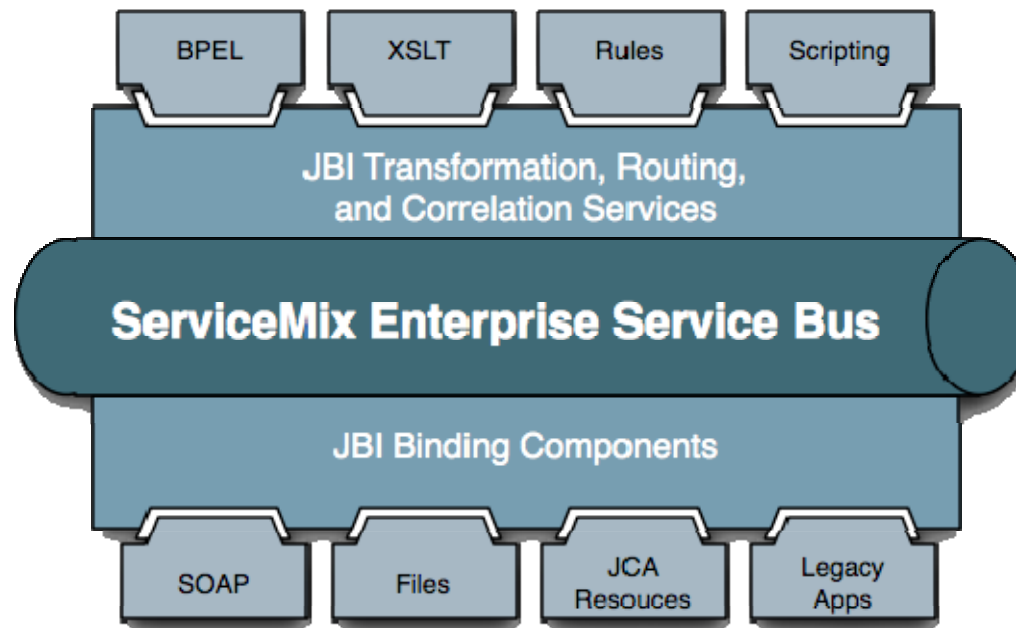
- Komponenten mit direktem SOA-Bezug:
 - Apache ServiceMix - Enterprise Service Bus
 - Apache ActiveMQ – Messaging
 - Apache Camel – Class Library für Enterprise Design Patterns
 - Apache CXF – Entwicklungsframework für Services
 - Apache ODE (Orchestration Director Engine) – Workflow Engine

- Neben den SOA-Komponenten gibt es eine Reihe von Web-Service-Tools (<http://ws.apache.org/>), unter anderem:
 - AXIS – Implementierung des SOAP-Protokolls
 - jUDDI – UDDI 2.0 Registry
 - WSIF – Aufruf von Services über verschiedene Bindings

- Die Apache-Komponenten sind wiederum Teil anderer SOA-Lösungen wie z.B. FUSE oder SOPERA ASF

Beispiel: Apache ServiceMix (<http://servicemix.apache.org>)

- Wird in einigen SOA-Lösungen, z.B. SOPERASF verwendet
- Basiert auf JBI
- Es existiert bereits eine Anzahl von JBI-Komponenten, die die Integration in den Service Bus ermöglichen



Quelle: <http://servicemix.apache.org/home.html>

- Im Open-Source-Bereich gibt es nur wenige Anbieter, die den gesamten „SOA-Stack“ abdecken

Die einschlägigsten Anbieter sind:

- Sun Microsystems – JavaCAPS
 - Red Hat – jBoss Enterprise SOA Platform
 - SOPER A
- Beispielhafte Anbieter mit reduziertem SOA-Stack:
 - Progress – FUSE
 - MuleSource - Mule

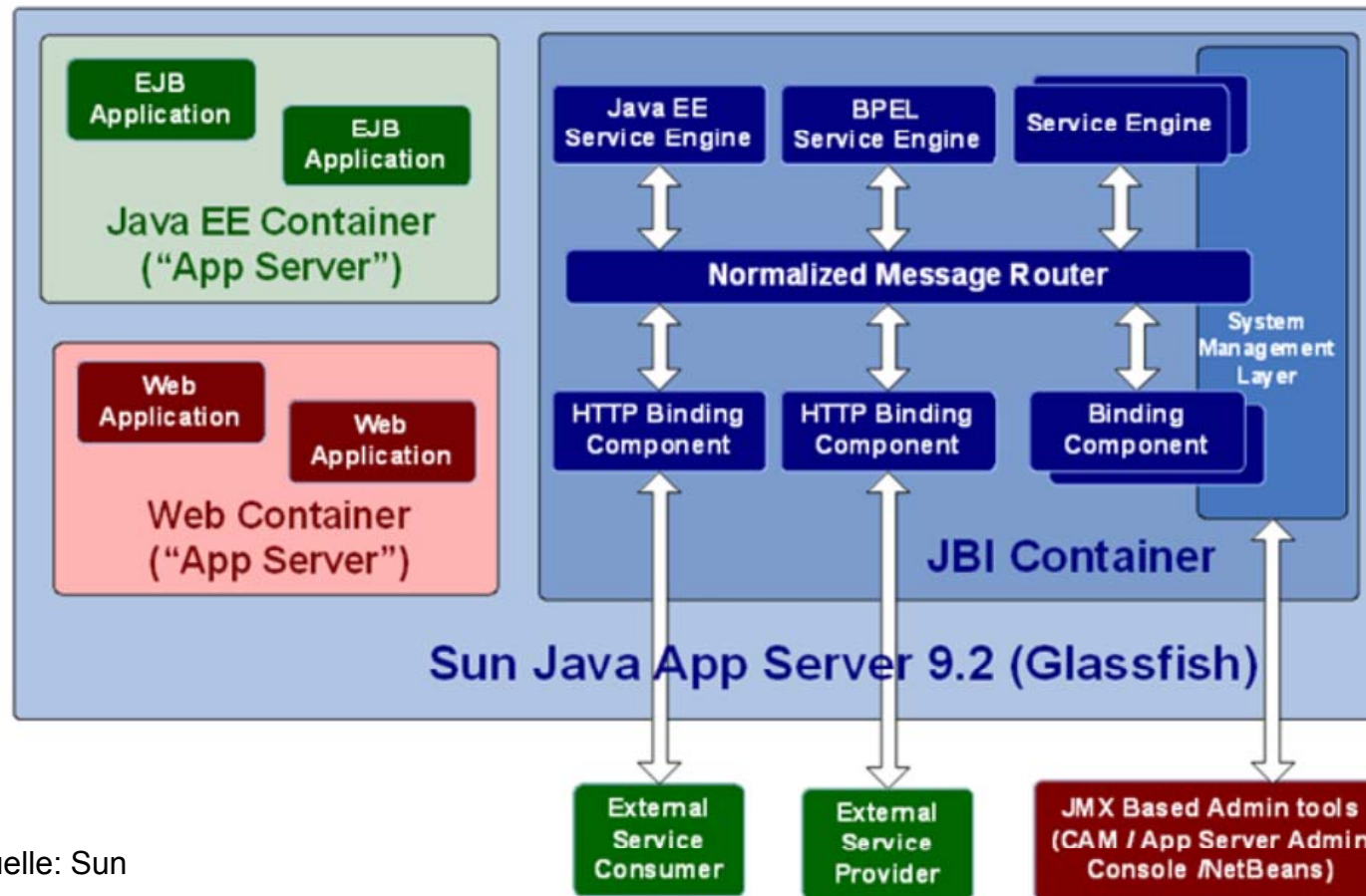
JavaCAPS (Java Composite Application Platform Suite)

- Sun Enterprise Service Bus
- Sun Business Process Manager
- Sun Intelligent Event Processor
- Sun Master Index
- Sun Data Integrator
- Sun Adapters
- Sun GlassFish Enterprise Server
- Sun Java System Portal Server
- Sun Java System Directory Server Enterprise Edition
- Sun Java System Access Manager
- NetBeans Integrated Development Environment (IDE)

- Sun fährt eine zweigleisige Strategie:
 - i. Kostenlose Open Source Software, z.B. OpenESB, Glassfish
 - ii. Kostenpflichtiges JavaCAPS-Paket mit Professional Services und Support
- Sehr gute Darstellung der Funktionalität von OpenESB / Glassfish unter Verwendung von JBI und der Unterschied zu JavaCAPS unter <http://wiki.open-esb.java.net/attach/OpenESBIntroductionScreencast/openesbIntroPreso.swf> (Video, 38 Min.)
- JavaCAPS wird derzeit als das **vollständigste und am weitesten entwickelte Produktportfolio im Open-Source-Bereich** angesehen.
- Die Entwicklung über die für Sun typische Netbeans-Oberfläche wird von manchen Benutzern als Nachteil bewertet.



OpenESB Architecture



Quelle: Sun



OpenESB Functionality

- Service Engines
 - > BPEL SE
 - > XSLT SE
 - > JavaEE SE
 - > IEP (BAM) SE
 - > ETL SE
 - > SQL SE
 - > Workflow SE
- Binding Comps
 - > MQSeries BC
 - > HL7 BC
 - > SAP BC
 - > SMTP BC
 - > HTTP BC
 - > JMS BC
 - > File BC
 - > CICS BC
 - > DCOM BC
 - > CORBA BC
 - > ...
- Other
 - > Clustering
 - > CASA
 - > JBI Mock
 - > WSIT Tech
- In Progress
 - > CAM
 - > Aspect SE
 - > Encoding SE
 - > Rules SE
 - > Scripting SE

Sun to provide a supported version though JavaCAPS releases

Quelle: Sun



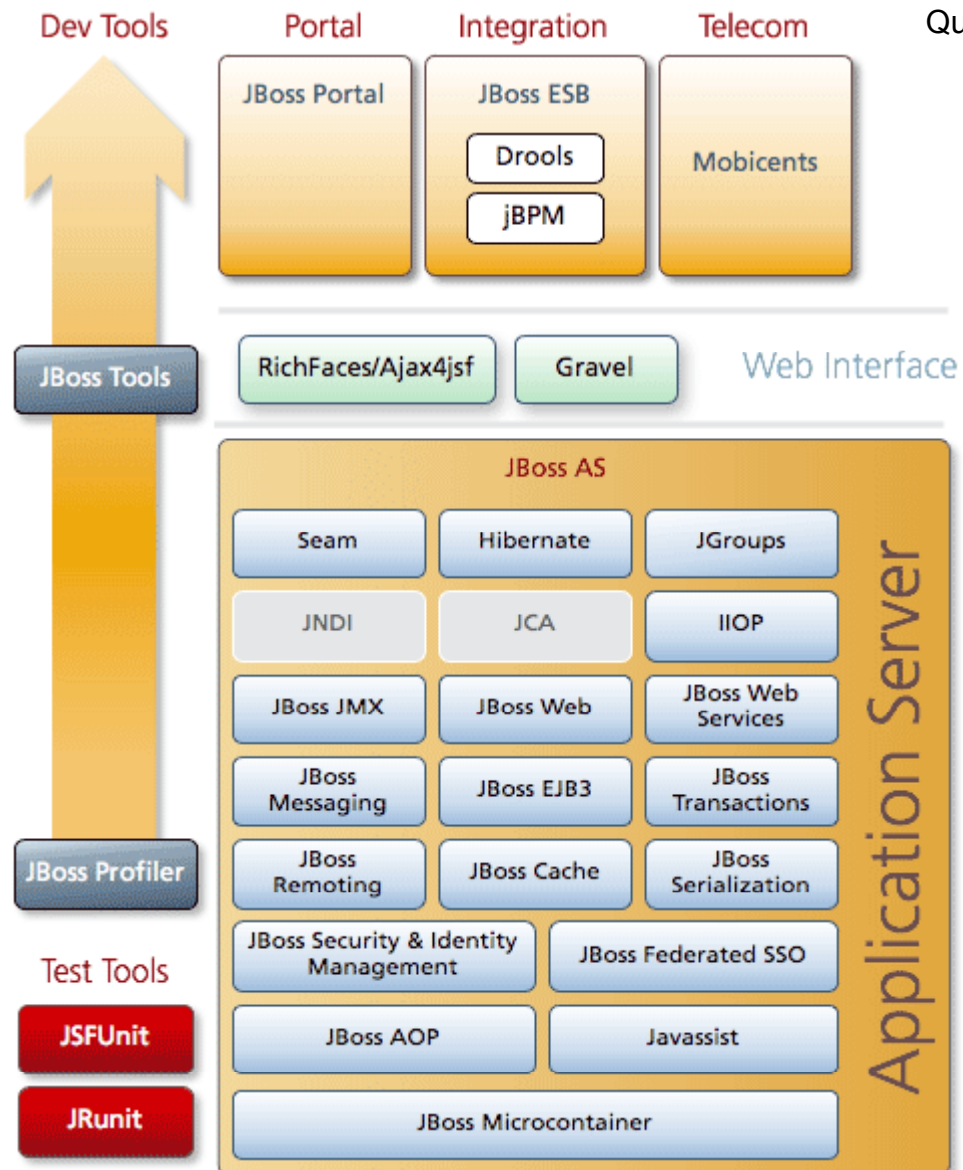
OpenESB and JavaCAPS

- Open ESB
 - > JBI Runtime
 - > Glassfish v2 AppServer
 - > Full collection of OpenESB components
 - > NetBeans v6 based tooling
 - > Combination of Sun and 3rd party components
 - > Constantly evolving
 - > Community Support
- JavaCAPS 6 ++
 - > JBI Runtime + JavaCAPS 5.1 Runtime
 - > Glassfish v2 AppServer
 - > Selection of OpenESB components
 - > NetBeans v6.1 based tooling (incl Enterprise Designer components)
 - > Combination of Sun and 3rd party components
 - > Sun Support

Quelle: Sun

jBoss-Produktfamilie

Quelle: jBoss



jBoss Enterprise SOA Platform

- Komponenten
 - JBoss Enterprise Application Platform – JEE Application Server
 - JBoss Enterprise Service Bus – ESB und UDDI 2.0 Registry
 - JBoss jBPM – Eclipse Plugin für Modellierung bevorzugt in JPD
 - JBoss Rules – Rules Engine

- Neben den als SOA-Plattform angebotenen Komponenten gibt es auch Lösungen für:
 - Portal
 - Security- und Identity-Management

jBoss Enterprise SOA Platform

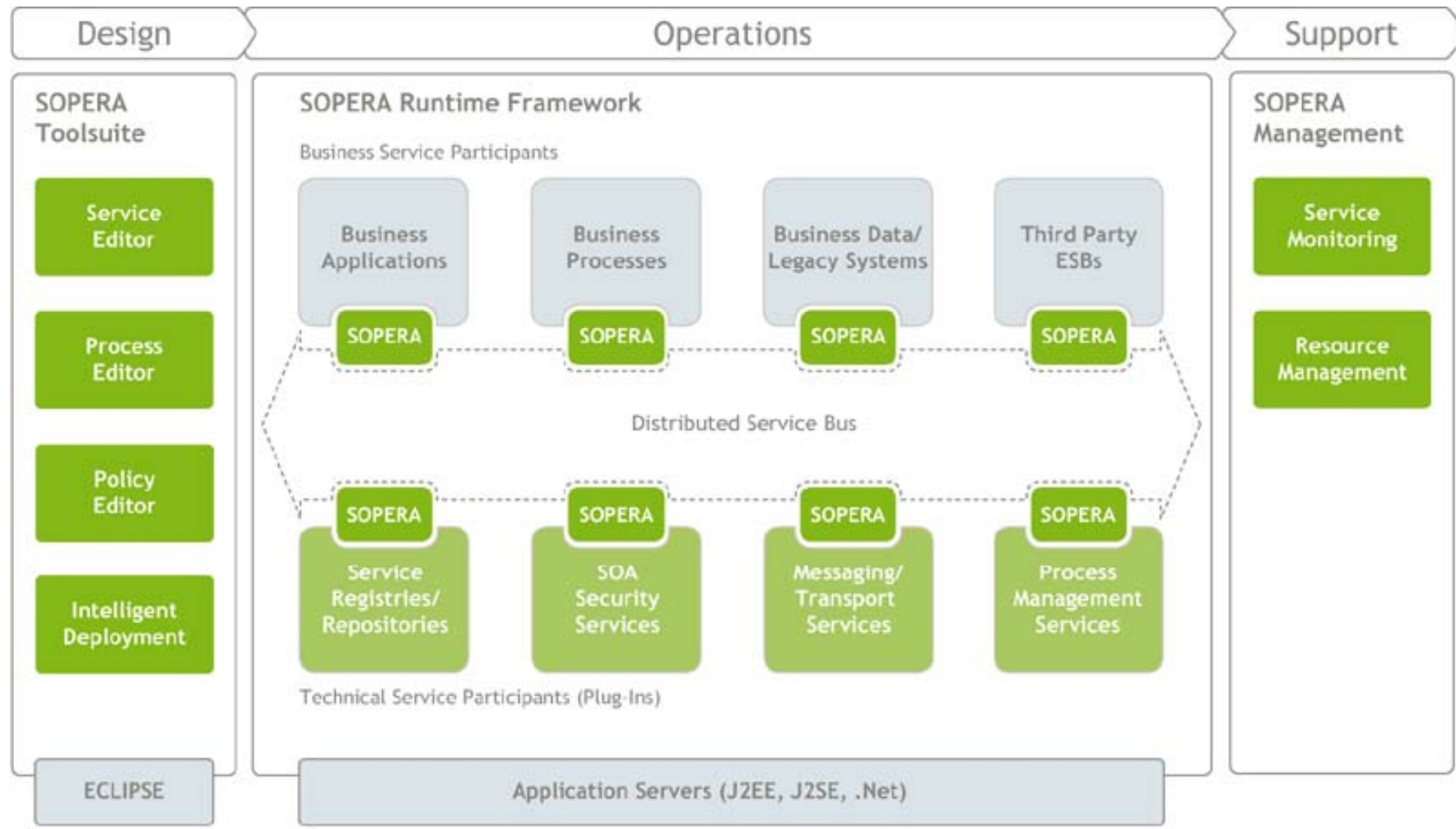
- Unterstützte Standards (laut Datenblatt):
 - SOAP 1.1
 - WS-Security
 - MTOM
 - WSDL 1.1
 - WS-Addressing 1.0
 - JAX-WS
 - UDDI 2.0
 - Java Management Extension (JMX) 1.2
 - Full J2EE 1.4 compliance



BPEL 1.1 wird nicht vollständig unterstützt
BPEL 2.0 noch nicht fertig

- JBoss hat einzelne Stärken (Application Server).
Die gesamte SOA-Plattform ist aber **noch nicht ausgereift** und
noch in der Entwicklung!

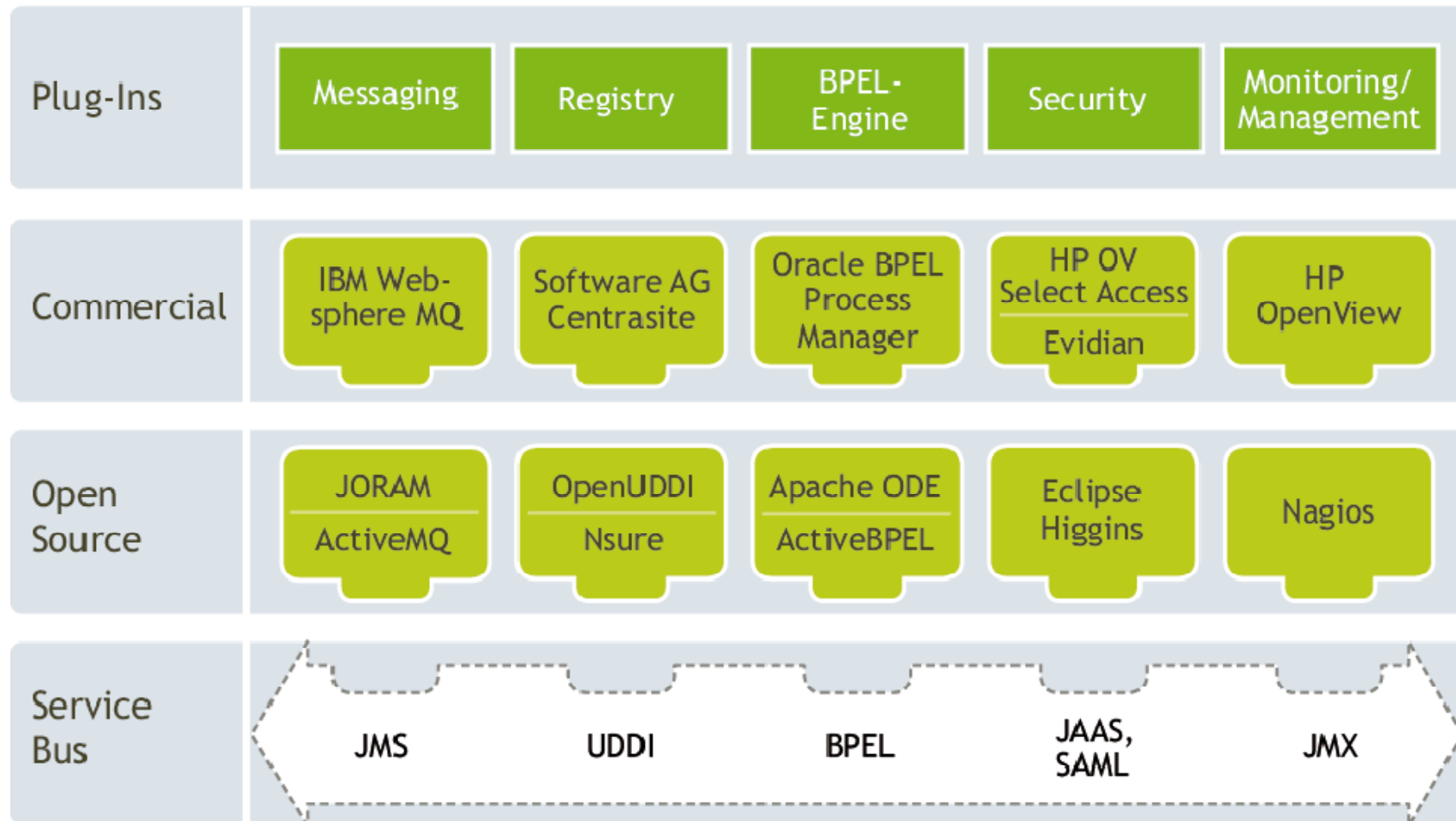
SOPERA [1]



■ SOPERA Advanced Services Framework (ASF)
 ■ Plug-Ins (Delivered with SOPERA ASF)
 ■ Third Party Components

Quelle: SOPERA

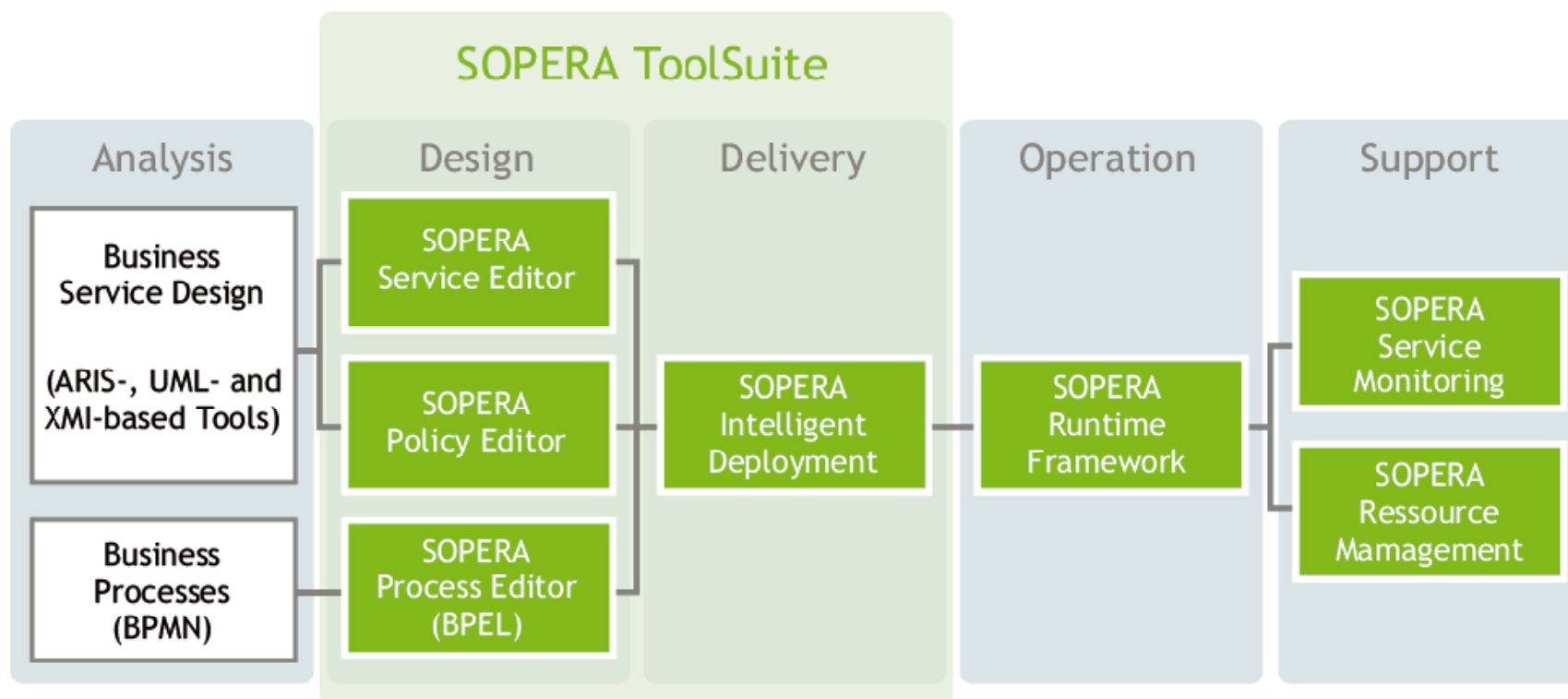
SOPERA ASF im Vergleich zu kommerziellen Lösungen



Quelle: SOPERA

SOPERA ASF-Tools

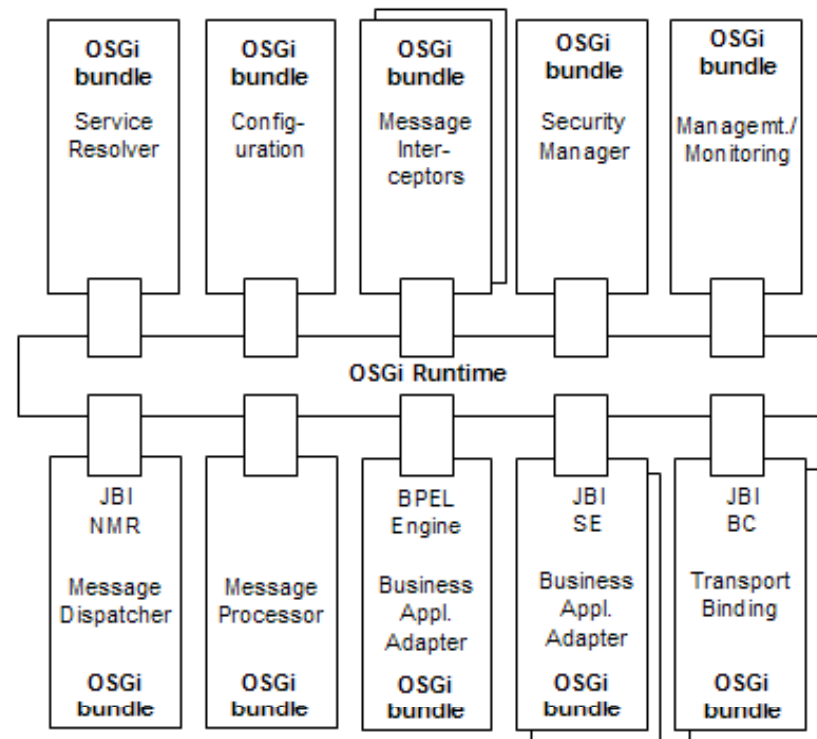
- Die Lösung bietet Tools für einen großen Teil des Service-Lifecycles
- Keine Modellierungswerkzeuge für Geschäftsprozesse



Quelle: SOPERA

- SOPERA bietet neben der kostenlosen Edition eine kostenpflichtige Enterprise-Edition an, inklusive:
 - Hotline-Support
 - SLAs mit zugesagten Reaktionszeiten
 - Zusätzliche kommerzielle Plugins
 - .NET-Support
 - SAP-Support
 - Business Activity Monitoring
 - Diverse weitere Tools
- SOPERA bietet ein **separates Open-Source-Tool** für das Management des Apache ServiceMix ESB an
- SOPERA entwickelt im Rahmen der Eclipse Foundation das Projekt **Swordfish** federführend weiter.

- **Swordfish** ist ein Projekt der Eclipse Foundation, das ein **Runtime-Framework** für SOA-Komponenten auf Basis von OSGi (Open Services Gateway Initiative) zur Verfügung stellt.
- Große Teile der Lösung stammen aus einem Infrastrukturprojekt der Deutschen Post.



Quelle: <http://www.eclipse.org/swordfish/learn/papers.php>

- Swordfish ist **keine fertige SOA-Lösung**
- Einzelne Infrastrukturkomponenten (sog. Bundles) sind bereits verfügbar, andere müssen noch entwickelt werden
- Anwendungskomponenten werden über die Bundles eingebunden
- OSGi erweitert die Java-Umgebung um spezielle Runtime-Funktionen für die Anbindung von Services, z.B. Hot-Plugging von Services oder die Einbindung von Embedded Systems
- OSGi soll die schwergewichtigen Application Server ersetzen
- Entwicklungsumgebung ist Eclipse mit den Erweiterungen des *SOA Tools Project (STP)* und *Web Tools Project (WTP)*
- Das Projekt ist im Stadium 1.0 und es gibt noch wenig Dokumentation.

Interessante Einstiegspunkte zum Thema:

<http://www.eclipse.org/swordfish/>

http://searchsoa.techtarget.com/news/article/0,289142,sid26_gci1294154,00.html#

FUSE (<http://fusesource.com/>)

- Enterprise Service Bus auf Basis von Apache ServiceMix
- Message Broker auf Basis von Apache ActiveMQ
- Service Framework auf Basis von Apache CXF
- Mediation Router auf Basis von Apache Camel
- HQ – Monitoring auf Basis von Hyperic HQ Enterprise
- Integration Designer – Eclipse basierte Entwicklungsumgebung (jedoch noch in Vorbereitung)



Komponenten für Registry und BPM fehlen und müssen durch andere Produkte ergänzt werden.

- Interessante Video-Webinars unter <http://fusesource.com/resources/video-archived-webinars/>

MuleSource (www.mulesource.org)

- Mule ESB
(Mule ESB ist in verschiedene andere Lösungen integriert)
- Mule Galaxy
(ein Repository ähnlich wie das kommerzielle Produkt *Centrasite*.
Es wird aber kein UDDI unterstützt und das mit Absicht
(„UDDI sucks“))
- Mule HQ – Monitoring
(Das Tool ist nur in der Enterprise Edition verfügbar)
- Es gibt jeweils Enterprise Editionen, die Herstellersupport,
erweiterte Qualitätssicherung und diverse Optionen für
Hochverfügbarkeit und Skalierung beinhalten.

freebXML Registry

- Implementierung des OASIS ebXML Registry 3.0 Standards
- JAXR (Java API for XML-Registries)
- Registry Browser als Java-Applikation
- Registry Browser als Web-Applikation
- Diverse administrative Tools
- Keine UDDI-Unterstützung

Intalio BPMS (www.intalio.com)

- Komponenten
 - Designer – Modellierung mit BPMN auf Eclipse-Basis (in das SOA Tools Project (STP) eingegangen!)
 - Server – BPEL 2.0 Engine basierend auf J2EE
 - Workflow – Portalkomponente für Benutzerinteraktionen
- Editionen
 - **Community** – stark limitierte Version nur für einfache Anwendungen geeignet
 - **Enterprise** – Unterstützung der für eine SOA im Enterprise-Umfeld üblicherweise erforderlichen Funktionalitäten
- Von der Zeitschrift „Infoworld“ zum besten Open-Source-Produkt im Bereich BPM bewertet

ActiveBPEL (<http://www.activevos.com/community-open-source.php>)

- Reine BPEL-Engine
- Wird in anderen Lösungen eingebunden (z.B. SOPERa)
- Stellt kein Modellierungstool für Orchestrierung von Services zur Verfügung
- Der Hersteller bietet das kostenpflichtige Modellierungstool *ActiveVOS* an

Weitere Open-Source BPM-Tools

- Bonita – XPDLe-Engine mit grafischer Modellierung
- Enhydra Shark – XPDLe-Engine nach WfMC-Standard
- Apache ODE (Orchestration Director Engine) – BPEL Engine

- Es gibt zahlreiche **Monitoring Tools** auf Netzwerkebene (z.B. Nagios)
- Es gibt Standardtechnologien für die **Überwachung von Java-Applikationen** (z.B. JMX, MBeans oder NetBeans)
- Komfortable Tools für die **Überwachung der Servicenutzung** oder der aktuellen **Zustände von Business-Prozessen** gibt es im Open-Source-Bereich in der Regel nur in den kostenpflichtigen Editionen.
- Für Sun *Open ESB* gibt es ein *BAM Dashboard* auf Basis von Netbeans, JasperReports Library und iReport als Report-Designer.
- Bei den kostenfreien Tools muss **viel Aufwand** in Auswahl, Test und Konfiguration und evtl. zusätzliche Eigenentwicklung gesteckt werden.
- **Kommerzielle Tools** bzw. kostenpflichtige Open-Source-Lösungen sind in diesem Anwendungsbereich derzeit eindeutig **im Vorteil**.

- Zahlreiche Tools und Projekte
- Guter Einstiegspunkt
<http://www.iam-wiki.org/Opensource>
- Nennenswerte Beispiele:
 - OpenSSO von Sun – Single Sign On für Webanwendungen
 - OpenLDAP – Verzeichnisdienst
 - OpenSAML – SAML-Implementierung in Java und C++

Liferay (www.liferay.com)

Auszug verwendete Produkte

- Apache ServiceMix
- Hibernate (Persistence Framework)
- Java J2EE/JEE
- jBPM (jBoss BPM)
- JGroups (Multicast Messaging)
- Lucene (Fulltext Search Engine)
- MuleSource ESB
- PHP
- Ruby
- Seam (jBoss Web 2.0 Technologie)
- Spring & AOP
- Struts & Tiles

Auszug verwendeter Standards

- AJAX
- JSR-168 (Portlets)
- JSR-127 (JSF)
- OpenSearch
- JSON
- REST
- RMI
- WSRP
- WebDAV

- Für einen **ESB** gibt es ausgereifte Standardtechnologien, die in Kombination einen industriell einsetzbaren Service-Bus ergeben.
- Für **BPM** gibt es geeignete BPEL- und XPD-Engines. Komfortable Modellierungswerkzeuge und BAM-Tools sind in aller Regel kostenpflichtig.
- Für die **Service-Entwicklung** und die **Einbindung in Portale** oder Anwendungsoberflächen gibt es zahlreiche ausgereifte Frameworks.
- Es sind diverse Technologien für **IAM** verfügbar.
- Das „**Tooling**“ ist komplex, da die einzelnen Produkte zwar in der Regel über Eclipse integriert werden, sich aber meist sehr unterschiedlich darstellen.
- Die **Lernkurve** kann je nach Vorkenntnissen sehr steil sein.
- **Für komplexe IT-Landschaften erforderliche Features** sind in der Regel nur in kostenpflichtigen Editionen verfügbar.

Warum scheitern SOA-Projekte?



Computerwoche, 3.10.2008

Anforderungen an eine SOA aus „Managementsicht“

- Kurzfristig Kosten senken
- Mittelfristig Kosten senken
- Langfristig Kosten senken
- Geschäftsprozesse optimieren
- Geschäftsprozesse flexibilisieren
- Erlöse optimieren
- Alle Maßnahmen sollen in möglichst kurzer Zeit Wirkung zeigen

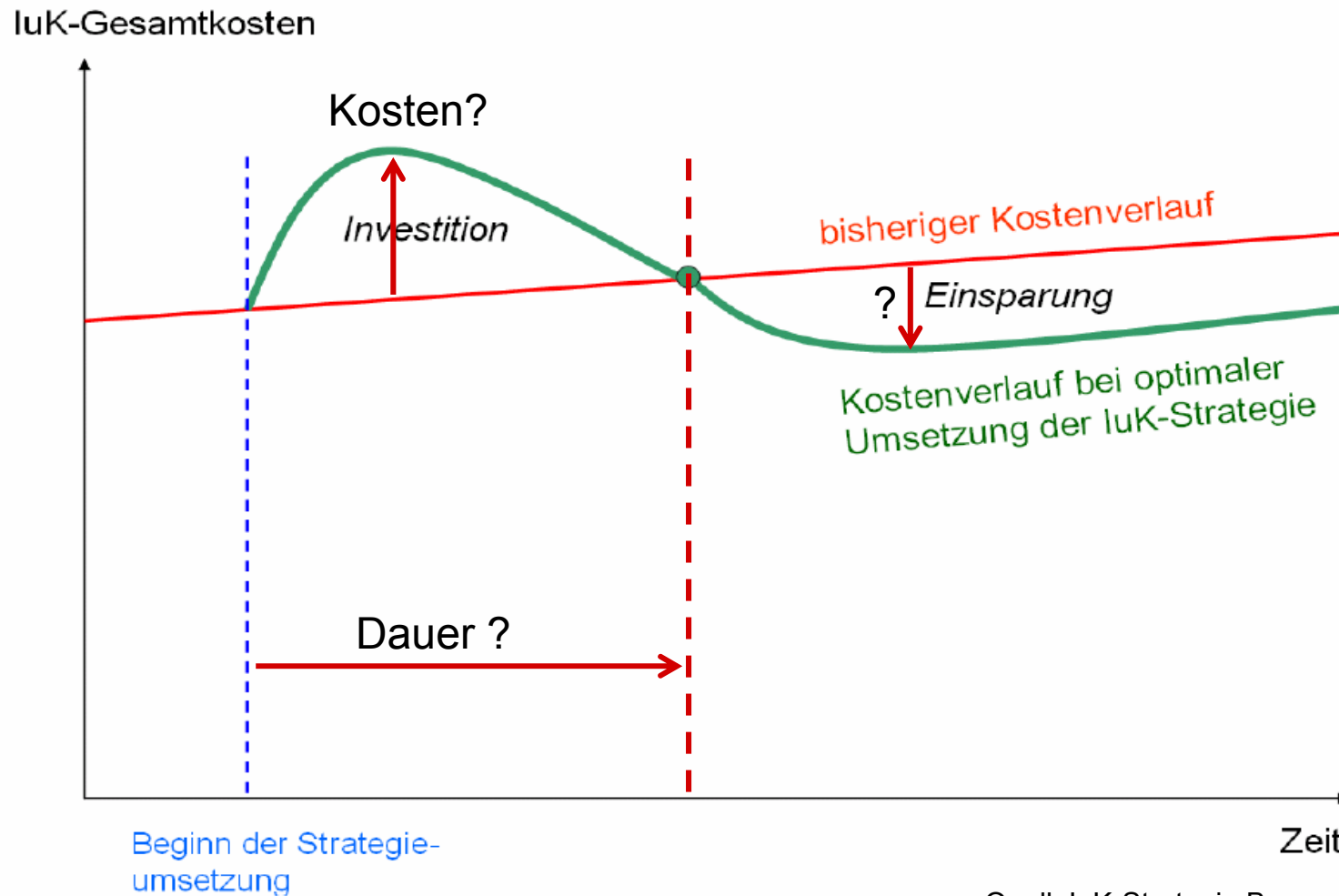


In dieser Welt kommt der Technologieaspekt nicht vor!

Eine SOA ist eine **langfristige Aufgabenstellung** und kann nur selten kurzfristigen Effekte erzielen.

Eine SOA ist in aller Regel mit einer **nicht unerheblichen Investition** verbunden.

Beispiel: Freistaat Bayern



Quell: IuK-Strategie Bayern

Projekte scheitern aufgrund

- **unterbliebener Wirtschaftlichkeitsberechnungen**, weil man SOA-Projekten pauschal eine verbesserte Wirtschaftlichkeit zurechnet
 - Überzogene Anforderungen an die Effekte einer SOA
 - Rückzug aus SOA-Projekten, wenn kurzfristig keine Einsparungen erkennbar werden
- **zu geringer finanzieller Mittel**
 - Notwendige Maßnahmen können nicht durchgeführt werden
 - Der langfristige Erfolg bleibt aus und das Projekt wird ebenfalls wegen unzureichendem Effizienzgewinn beendet



Eine umfassende Wirtschaftlichkeitsberechnung ist mangels Erfahrungen und geeigneter Methoden nur schwer durchführbar

- Optimierung der Geschäftsprozesse ist im Wesentlichen eine Frage der Ablauf- und Aufbauorganisation
- Finanzielle Vorteile stellen sich ein, wenn
 - Strukturen vereinfacht
 - Abläufe optimiert
 - die Organisation auf die Geschäftsprozesse abgestimmt wird
- Wenn im Rahmen der Planung einer SOA auch die Geschäftsprozesse bearbeitet werden, kommt es zwangsläufig zu Konflikten hinsichtlich
 - Zuständigkeiten
 - Kostenverantwortung
 - Anordnungsbefugnissen

Projekte scheitern aufgrund

- falschem Projektansatz, da SOA oft als **reines Technologieprojekt** durchgeführt wird, ohne die Organisation zu berücksichtigen
- mangelnder **Einigungsfähigkeit** zwischen Abteilungen
- **Besitzstandswahrung**
- **fehlender Durchsetzungsmöglichkeiten** für Umstrukturierungen durch die SOA-Verantwortlichen
- **fehlender Unterstützung durch das Top-Level-Management**, das die notwendigen Umstrukturierungen anordnen könnte



Oft wird die **SOA-Governance**, die auch die organisatorischen Regelungen beinhaltet, unterschätzt.

- Die Planung einer SOA ist sowohl **organisatorisch** als auch **technologisch** komplex
- Entwicklung der Architektur stellt **hohe Anforderungen** an die Fähigkeiten aller Beteiligten
 - Mitarbeiter der Fachabteilungen
 - Mitarbeiter der IT-Abteilungen
 - Projektleiter
- Insbesondere das **Projektmanagement** ist eine neue Art von Herausforderung, da viele Beteiligte für einvernehmliche Lösungen eingebunden werden müssen

Projekte scheitern aufgrund

- unzureichendem Projektmanagement
- unterschätzter Komplexität
- unzureichender Methodenkenntnisse bei den Beteiligten
- Verzettelung in Details
- zu langer Projektdauer
- nicht realisierter Quick-Wins bzw. nicht erzielter Ergebnisse

- Eine umfassende SOA deckt eine Fülle von **technischen Aufgabenstellungen** ab, die erst einmal beherrscht werden müssen:
 - Zuverlässiges Messaging
 - Adaption von Schnittstellen
 - Abbildung von Prozessen
 - Entwicklung von (Web-) Services
 - Hinreichendes Identity- und Access-Management
 - Integration verschiedenster Anwendungen in Portalen
 - Zuverlässiger Betrieb mit definierter Qualität
 - ...

Projekte scheitern aufgrund

- unzureichendem Reifegrad der am Markt verfügbaren Produkte, der für eine problemlose Integration aller Komponenten erforderlich wäre
- fehlendem Know-how der Mitarbeiter
- zu komplexer und teurer Lösung von Lieferanten



Bei fehlender SOA-Erfahrung der Beteiligten ist es schwierig, eine geeignete Lösung für eine Ausschreibung zu spezifizieren und in einem späteren Schritt einzuführen

Durchführung der Ausschreibung

- Juni 2006 Architekturstudie der IBM im Auftrag der STK fertig
- Oktober 2006 Beschluss für die Durchführung des Projekts
- Dezember 2006 Finaler Projektauftrag an das LfStaD
Erstellung Ausschreibungsunterlagen auf Basis der Studie
Auswahl der Pilotanwendungen durch die STK bzw. den Ministerrat
- Februar 2008 Auftrag an die Projective Expert Group zur Qualitätssicherung der Ausschreibungsunterlagen
- September 2008 Neuerstellung der Leistungsbeschreibung fertig
- April 2009 Bekanntmachung
- Juni 2009 Anbieter geben Angebote ab

- Leistungsbeschreibung **nach fast zwei Jahren immer noch nicht fertig**
- **Uneinheitliche Anforderungen** hinsichtlich der beabsichtigten Systemteile der Infrastrukturlösung
- Leistungsbeschreibung basiert auf zahlreichen „Erläutern Sie“-Positionen, wobei ein **Bewertungsmaßstab fehlt**
- **Pilotanwendungen sind noch nicht spezifiziert**
- Verantwortliche Ressorts für die Pilotanwendungen haben **keine verbindlichen Zusagen** gegeben und auch **keine Manpower**
- **Technische Rahmenbedingungen** für die Schaffung geeigneter Services **ungeklärt**, da Ressorts die Anwendungen nicht selbst entwickeln, sondern über Dienstleister realisieren lassen
- **Finanzielle Kompensation** für Anpassung von Anwendungen **teilweise ungeklärt**

- **Keine eigene Erfahrung** mit den Entwicklungsparadigmen verteilter Systeme bzw. einer SOA beim LfStaD
 - Hersteller soll sagen, wie die SOA aussehen muss („Erläutern Sie ...“)
 - Die Erstellung der Leistungsbeschreibung hätte ausgeschrieben werden müssen → SOA-Architekt
- Kein definitives **Commitment** der beteiligten Ressorts
 - Zuarbeiten unzureichend
 - Aktivitäten in den einzelnen Ressorts nicht abgestimmt
- Keine institutionalisierte **SOA-Governance**
 - Aufgrund der Ressorthoheit kann das LfStaD den einzelnen Ministerien keine verbindlichen Vorgaben machen

Beispielhafte ungeklärte Aspekte für den Enterprise Service Bus

- Welche konkreten Sicherheitsanforderungen müssen erfüllt werden?
- Welche Adapter werden benötigt?
- Welche Art von Transaktionsfähigkeit wird benötigt?

Beispielhafte Probleme mit der Geschäftsprozessmodellierung

- Modellierungstool „Innovator“ wurde für die Fischerprüfung getestet, aber aufgrund der Tool-Komplexität dann nicht mehr verwendet.
- Die Ausbildung zur Bedienung des Tools wurde nicht hoch priorisiert.

Beispielhafte Probleme beim Identitätsmanagement

- Kein zentrales Identitätsmanagement aus Gründen der **Ressorthoheit**
- Kein fertiges Konzept für ein **verteiltes Identitätsmanagement**

Beispielhafte Probleme bei den Pilotanwendungen

- **Zentraler E-Payment-Service nicht fertig spezifiziert** und die bisher spezifizierte Lösung ist erkennbar unzureichend.
- Für die Anwendung **E-Notar** ist nicht klar, ob die Anwender diese überhaupt brauchen bzw. in dieser Form wollen

Dr. Frank Sarre

**Anschrift: Ludwig-Maximilians-Universität München (LMU)
Institut für Informatik
Lehr- und Forschungseinheit für
Programmierung und Softwaretechnik (PST)
c/o Fr. M. Diem (Sekretariat von Hrn. Prof. Dr. M. Wirsing)
Oettingenstr. 67
80538 München**

**Telefon: Tel. 089 / 2180 -9151 (Fr. Diem) oder
direkt unter Tel. 089 / 18 92 37 -01**

Email: frank.sarre@pst.ifi.lmu.de