

# Geschäftskritische SOA- Anwendungen mit Open Source-Infrastruktur

---

Prof. Dr. T. Wieland  
Centrum für innovative Softwaresysteme  
GmbH, Coburg

20. Juni 2007

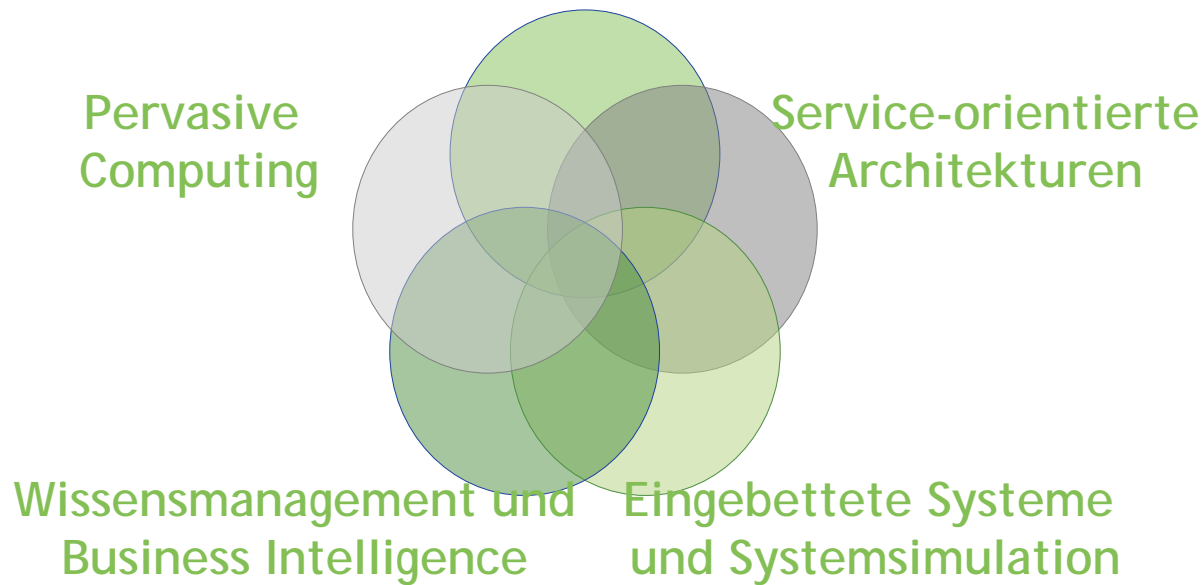


# c-fis Centrum für innovative Softwaresysteme GmbH



- Gegründet 2007, aus der Hochschule Coburg hervorgegangen
- Führt anwendungsbezogene Forschungs- und Entwicklungsvorhaben sowie Beratung und Weiterbildung durch in den Bereichen

## Open Source Software in Unternehmen



[www.c-fis.de](http://www.c-fis.de)

# Agenda

- Einführung
- Was ist "service-orientierte Architektur" eigentlich?
  - und was ist ein "Service"?
- Open Source-Bausteine
- Fazit und Empfehlungen
  - Wozu das Ganze eigentlich?

# Aktuelle Probleme in der IT

- 💣 Extreme Komplexität der Schnittstellen
- 💣 Große Lücken in der Funktionalität
- 💣 Hohe Redundanzen bei Daten und Funktionalitäten

- 💣 Hohe Kosten für Anwendungsentwicklung und Betrieb
- 💣 Lange Projektlaufzeiten
- 💣 Häufiges Scheitern von Projekten
- 💣 Heterogene, unflexible IT-Landschaft
- 💣 Abhängigkeiten von Herstellern und Werkzeugen

# Demgegenüber: Geschäftsanforderungen

**Schnelle Reaktionen  
auf immer  
schnellere Veränderungen**

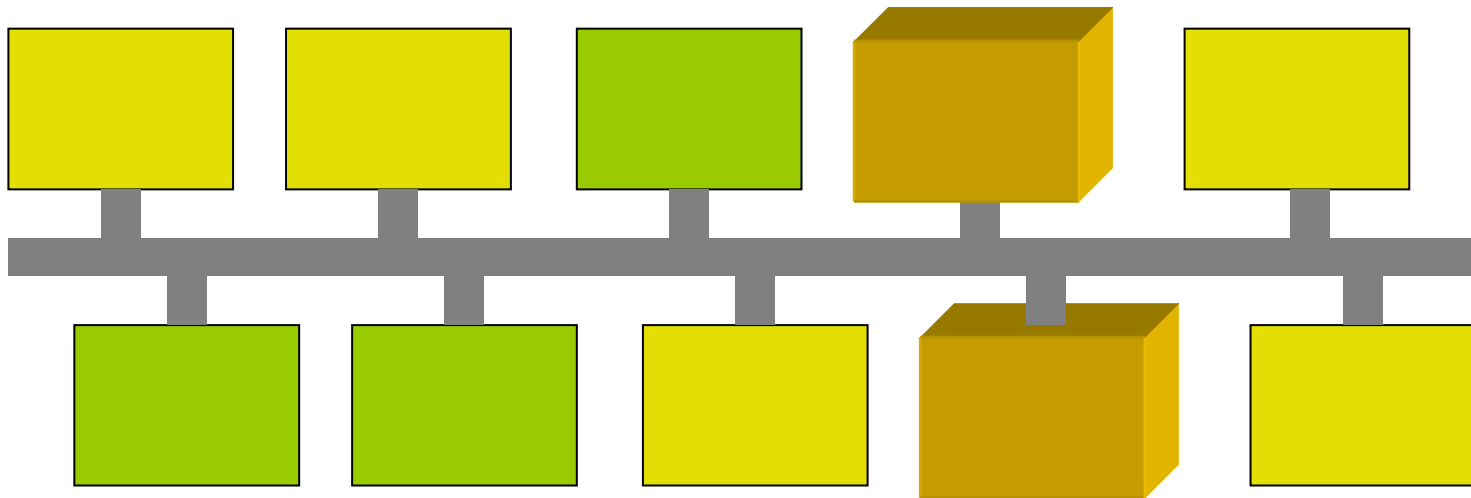
**Anpassbarkeit der Systeme  
auf neue Geschäfte  
und neue Märkte**

**Sofortige Verfügbarkeit  
aller Informationen  
überall**

**Effizienz und  
"Reibungsfreiheit"  
in allen Prozessen**

# Lösung: SOA

- **Service-orientierte Architektur:** Aufteilung der IT-Systeme in höherwertige Dienste
- Verbindung der Dienste über einen gemeinsamen "Daten-Bus"



# Was ist "service-orientierte Architektur" eigentlich?

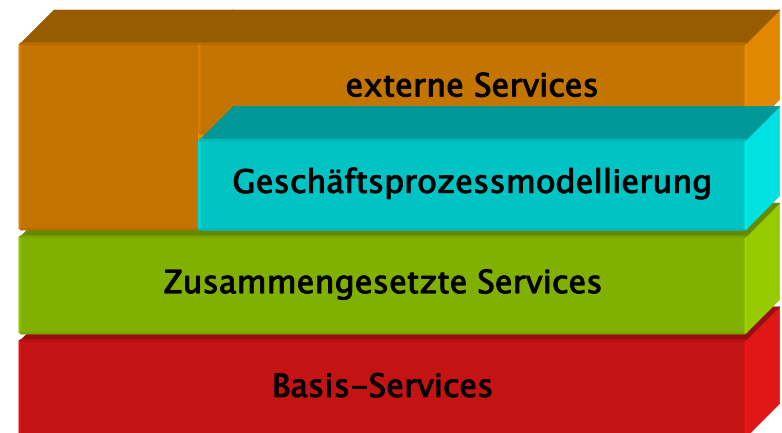
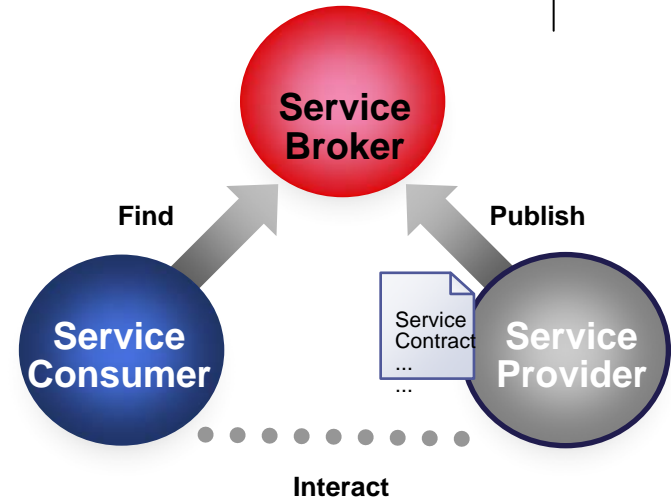
---



# Was ist ein Service?

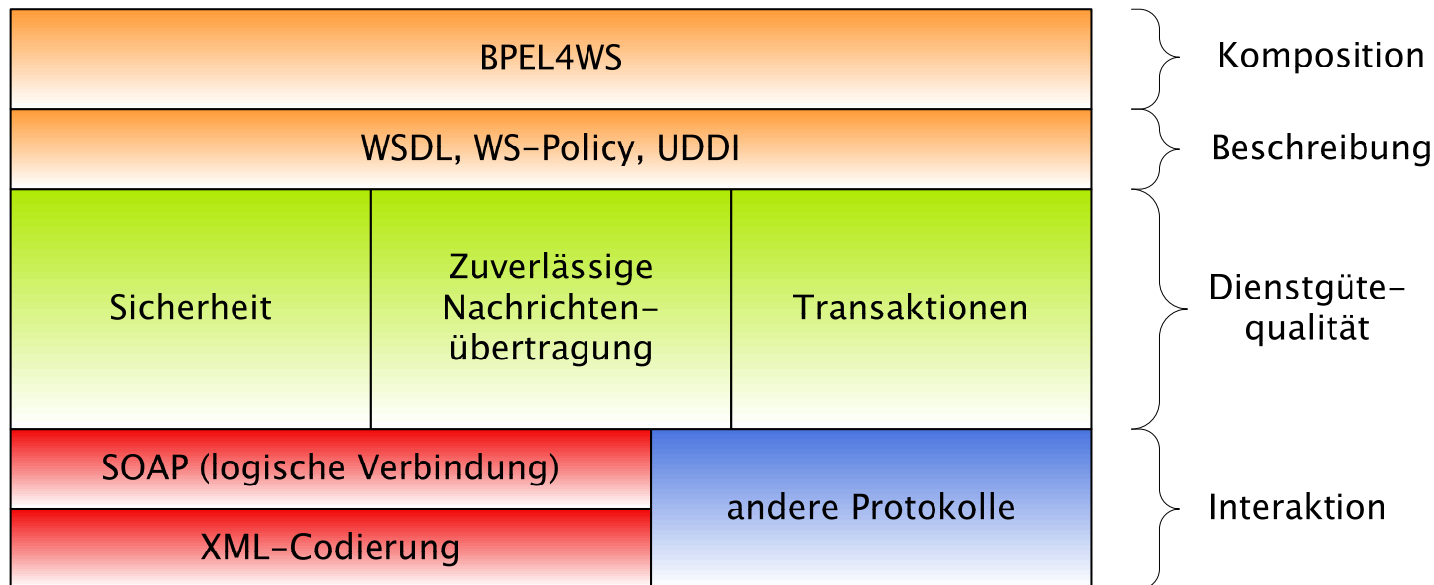
Ein Service ist

- ist eine logisch abgeschlossene, **wiederverwendbare Softwarekomponente**
- lokal oder **über ein Netzwerk** nutzbar
- nur **lose gekoppelt** mit anderen Services
- mit einer genau definierten **Schnittstelle** ausgestattet
- oft eine **Zusammenfassung** einzelner Funktionalitäten
- aufgeteilt in Anbieter ("**Provider**"), Nutzer ("**Consumer**") und Verzeichnisdienste ("**Broker**")



# Web Services

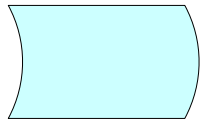
- Web Services sind *eine* Möglichkeit, einen Service zu realisieren
- Über das Internet erreichbare, **plattformunabhängige Schnittstelle**
- Entspricht grob einem entfernten Methodenaufruf über das **HTTP-Protokoll im XML-Format**
- Inzwischen große Zahl darauf aufbauender **Standards**



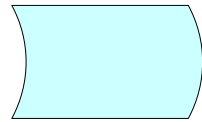
# Was ist eine SOA?

- SOA ist das **Zusammenspiel von Services** zur Realisierung einer Geschäftsanwendung
  - Durch die **Fokussierung auf Geschäftsprozesse** und den Einsatz von Standardschnittstellen verringert sich die technische Komplexität
- SOA stellt nicht eine wirklich neue Technik, aber eine **sehr neue Betrachtungsweise der IT** dar
- SOA baut meist auf herkömmlicher **Komponententechnologie** auf
- SOA eignet sich besonders gut zur **Integration von Unternehmensanwendungen (EAI)**

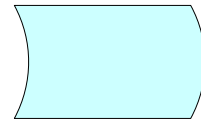
# Agiles prozess-orientiertes Design



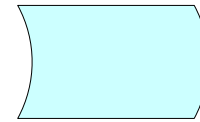
Einkauf



Verkauf

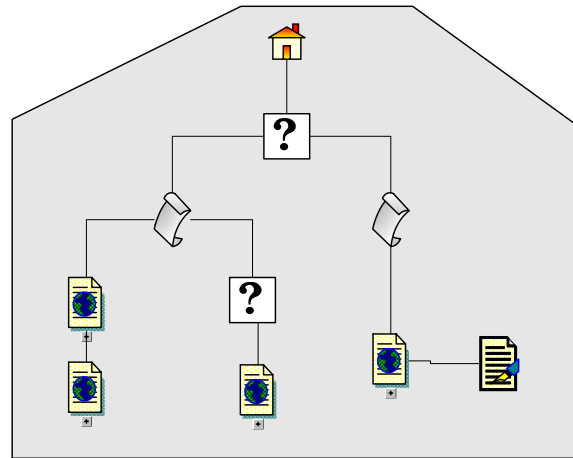


Auslieferung

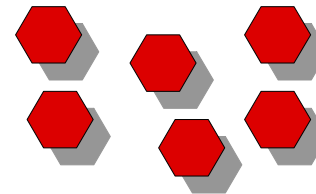
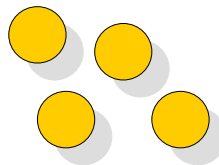
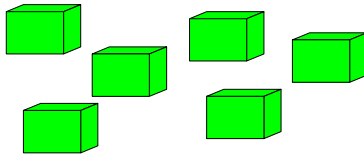


Rechnungstellung

Geschäftsprozesse

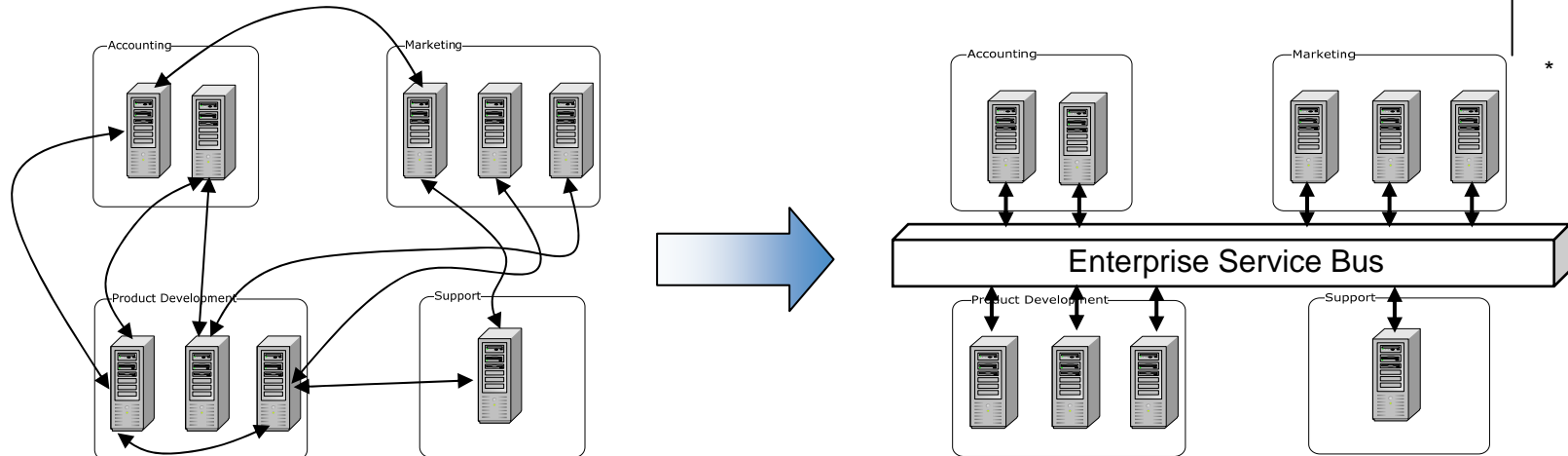


Orchestrierung



Komponenten

# Enterprise Service Bus



- Ein ESB ist ein Vermittler, der Services untereinander verfügbar macht
  - Er beherrscht zu den Applikationen hin verschiedene **Adaptervarianten**
  - Kann richtigen Ort und **richtige Version** eines Service bestimmen
  - Unterstützt **Transaktionen**, Sicherheit und Dienstgüte
- ABER: **Keine einheitliche Definition**, jeder Hersteller versteht etwas anderes darunter

# Vorteile von SOA

- **Ausrichtung der IT an den Geschäftsprozessen**
- Erhöhte **Wiederverwendbarkeit** innerhalb der IT
- **Trennung** von technischen und geschäftlichen Aspekten
  - Service-Schnittstellen zeigen nur Geschäftsdaten/-operationen
- Bessere Möglichkeiten zu **Verwaltung und Wartung**
  - Entwicklung, Versionierung, Installation, Überwachung
- **Schnellere Anpassung** bei Veränderungen des unternehmerischen Handelns
- **Schnellere Bereitstellung** von Applikationen für neue Geschäftsanforderungen

# Analystenmeinungen zu SOA

- [Gartner](#) sagt voraus (2003): Bis 2008 werden mehr als 60% der Unternehmen **SOA als leitendes Prinzip** einsetzen, wenn sie "mission-critical" Anwendungen und Prozesse erstellen werden.
- Unternehmen, die das Potential von SOA ignorieren, werden durch **deren Konkurrenten vom Platz gedrängt**, die ihre Agilität verbessert und bereits die neue Art von Unternehmen adaptiert haben.
- Studie von [AT Kearney](#) (2006): Bis 2015 können bei Großunternehmen in Deutschland aufgrund von Prozessautomatisierung, vor allem auf Basis von SOA, **bis zu 100,000 Arbeitsplätze eingespart** werden
- Davon 43% im **Finanz- und Rechnungswesen**, 29% in der **Informationstechnologie**, 13% im Personalwesen und 10% im Einkauf

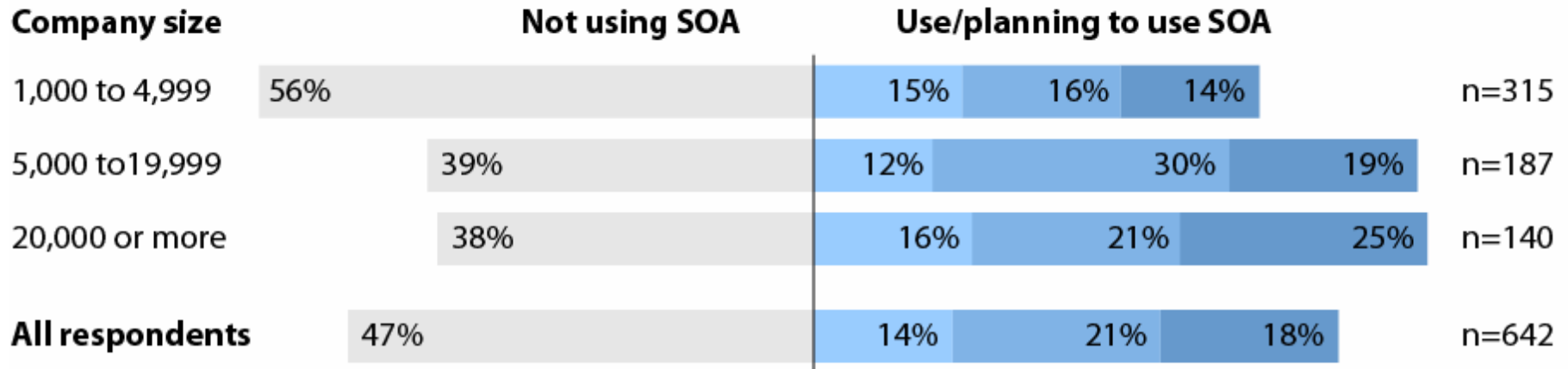
Gartner Strategic Planning (SPA-19-5971):  
'Introduction to Service-Oriented Architecture',  
14.04.2003

AT Kearney Pressemitteilung: "Deutsche Industrie baut  
100.000 Arbeitsplätze in Verwaltung ab", 08.08.2006

# Verbreitung

**“Which of the following best describes your firm’s approach to or status of SOA?”**

Not pursuing, and no immediate plans to do so
  Will pursue within 12 months
  Use selectively, without a clear strategy
  Have an enterprise-level strategy and commitment for SOA



Base: North American and European software and services decision-makers

Quelle: Forrester’s Business Technographics November 2005 North American And European Enterprise Software And Services Survey

# Open Source- Bausteine

---



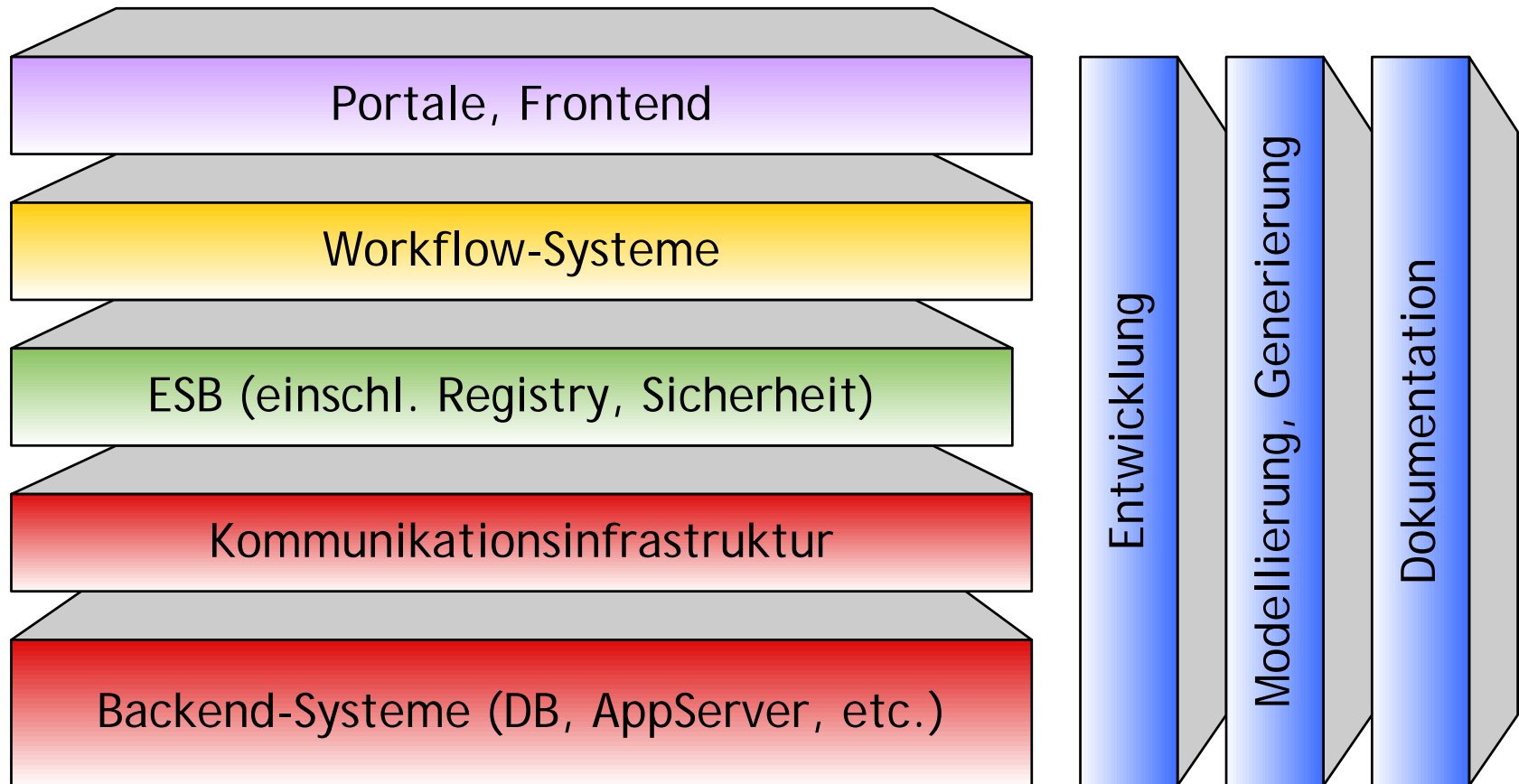
# Möglichkeiten bei der SOA-Realisierung

- Variante 1: **Nur selbst entwickelte Bausteine**  
→ Zu aufwändig in Entwicklung und Pflege!
- Variante 2: **Fertige Lösung eines Herstellers**  
→ Aber: Probleme bei proprietären Lösungen
  - Abhängigkeit vom Hersteller
  - Überfrachtung von Software-Paketen, daher hoher Preis
  - Komplexität der Produkte
  - Viele nicht benötigte Elemente
  - Eingeschränkte Interoperabilität
  - Fehlende oder kostenintensive Anpassung auf eigene Verhältnisse
- Variante 3: **Individuelle Zusammenstellung aus OSS-Bausteinen**

# Vorteile von freier/Open Source Software

- Hohe Qualität zu geringen Kosten
- Höhere Reife durch **Unabhängigkeit** von Marktzwängen
- Hohe **Sicherheit** durch Einblick in alle Abläufe
- **Wiederverwendbarkeit** von Komponenten
- **Unabhängigkeit von einem Hersteller** und seinen Produktzyklen
- Unterstützung von **offenen Standards**

# Software-Bausteine für eine SOA



# Backend-Systeme

- **Datenbanken**
  - Sollte hochverfügbar und leistungsfähig für geschäftskritische Anwendungen sein
  - MySQL, PostgreSQL, Firebird, [Ingres r3](#)
- **Application Server**
  - Geschäftslogik meist als EJB implementiert, mehrere getrennte AppServer-Instanzen möglich
  - [Jboss](#), JOnAS, Apache Geronimo
- **O/R-Mapping**
  - Abbildung der Objekte in relationale Datenbankschemata
  - [Hibernate](#)

# Kommunikationsinfrastruktur

- Für Webservice-Kommunikation: **SOAP**
  - SOAP-Protokoll inzwischen mit Erweiterungen recht komplex
  - [Apache Axis 2](#), Celtix von ObjectWeb, Java API for XML Web Services (JAX-WS) von Sun, XFire von Codehaus
- Aber: **Nicht nur auf Webservices vertrauen**
  - Gut für unternehmensübergreifene Kommunikation
  - Intern: Besser direkte Verbindungen: JMS, RMI oder POJO
- Tipp: WSIF (Web Service Invocation Framework)
  - Abstrakter Aufruf von Services, unabhängig von der Kommunikationsart
  - Bedenke: **SOAP geht nicht nur mit http!**

# Kommunikationsinfrastruktur (2)

- Wichtig: Nicht auf eine Technologie einschränken!
  - Austauschbarkeit, Integration heterogener Systeme
- Abhängigkeit der Geschäftslogik von Kommunikationsparadigmen oder gar -produkten ist zu vermeiden
- Für den Aufbau einer SOA gut: **JMS**
  - Message-oriented Middleware für Anwendungen mit hoher Kommunikationslast
  - OpenJMS von Sun, [ActiveMQ](#) von Apache, JBoss Messaging

# Enterprise Service Bus

- Nicht zwingend erforderlich!
- Hauptaufgabe: **Datenaustausch** über Aufruf von Services
- Kann **zentrale Dienste** anbieten
  - Registry, Single-Sign-On, Transaktionen, Verschlüsselungsdienste und Datenkonvertierung
- Produktauswahl an Bedarf ausrichten
  - Alle unterstützen JBI (JSR-208) und verschiedene Services
  - [ServiceMix](#) von Apache, [Mule](#) von Codehaus, OpenESB von Sun, Celtix von ObjectWeb

# Workflow-Systeme

- Bei kleinen Workflows nicht erforderlich, aber wichtig bei **Workflows mit langen Unterbrechungen** (Stunden, Tage)
- Konkurrierende Beschreibungen: **BPEL** und **XPDL**
  - Auswahl nach Geschmack und vorhandener Kompetenz
- XPEL-Engines
  - Enhydra Shark, [WfMOpen](#) von Danet
- BPEL-Engines
  - Apache Ode (gesponsort von Sybase + FiveSight), [ActiveBPEL](#) von Active Endpoints, BeeXee bei Sourceforge

# Portale, Frontend

- **Web Frontend**-Lösungen
  - Plattformunabhängig für die Clients, nur serverseitige Wartung
  - Nachteil können hohe Antwortzeiten sein
  - [Jakarta Struts](#), [Apache Cocoon](#), Jakarta Turbine, Apache MyFaces
- Beschleunigung durch **AJAX**
  - Nicht ganze Web-Seiten aufbauen, sondern nur Anzeige aktualisieren
  - Google Web Toolkit, [DWR](#) von Tibco
- **Integrierte** Frameworks
  - Für Front- und Backend, mit und ohne Appserver
  - [Spring Framework](#)

# Modellierung, Generierung

## Modellierung zur Systembeschreibung

- In UML 2.0 oder in domänenabhängiger Sprache für MDA/MDS
- [Eclipse Modeling Framework EMF/GMF](#), ArgoUML, Umbrella, evtl. Poseidon oder MagicDraw (nicht OSS, aber freie Community Edition)

## Generierung von Code und anderen Artefakten aus Modellen

- Im Sinne der MDA bzw. MDS, auch für SOA sinnvoll!
- [openArchitectureWare](#) bei Eclipse, AndroMDA, openMDX

# Entwicklungswerkzeuge und Dokumentation

- Programmierung: **Eclipse**
  - Optimal bei Java-Projekten, nur eingeschränkt bei .NET-Projekten ☺
  - Tipp: Eclipse **SOA Tools Platform Project** (<http://www.eclipse.org/stp/>)
- Versionsverwaltung: CVS, **Subversion**
- Test: **jUnit**
- Konfigurationsmanagement: **Ant**, Maven
- Projektdokumentation: **WikiWiki**, JavaDoc, OpenOffice
  - Außerdem viele CMS für Webseiten (Mamba, Joomla, Typo3, drupal etc.), Blogs, ...

# Fazit und Empfehlungen

---



# Empfehlungen (1)

- SOA behandelt die **Architektur** hinter und über den Prozessen
- Versuchen Sie so viel wie möglich zu **modellieren**
  - Die Interaktionen der Services ebenso wie die Einzelsysteme und deren Komponenten
  - Halten Sie die Modelle frei von Technologien
- Erstellen Sie mittelfristig ein **eigenes Meta-Modell**
- **Testen** Sie regelmäßig die Funktionalität und die Performanz Ihrer Services
  - Schlechte Performanz deutet auf zu feine Granularität der Services hin
- Achtung: SOA-Einführung bedeutet **Kulturwandel** bei IT- und Fachabteilung!

# Empfehlungen (2)

- Passen Sie Werkzeuge und Frameworks genau **den Bedürfnissen Ihres Unternehmens** an
- Setzen Sie jedes Werkzeuge für eine **genau bestimmte Aufgabe** ein
  - Viele Pakete haben überlappende Funktionalität
- Stimmen Sie die **Software-Systeme aufeinander ab**
  - Ansonsten entsteht ein Versions-Chaos
- Bei erstmaligem Einsatz: Am besten **externe Beratung** einholen
  - Auch für Open Source Software gibt es oft gute Betreuung durch dahinter stehende Hersteller

# Beratungsangebot

Beratung und Projektbegleitung durch:

Centrum für innovative Softwaresysteme GmbH

Sandstraße 54

D-96450 Coburg

wieland (*at*) c-fis.de

[www.c-fis.de](http://www.c-fis.de)

Tel. 09561/2482099

