Web Services und Service Oriented Architectures

Grundlagen und Vergleich mit bestehenden Middlewaretechnologien

Inhalt und Motivation

- Service Oriented Architecture
- Web Services
- Basistechnologien
- Web Services vs. CORBA

Service Oriented Architectures

- Ziele
- Service Oriented Model
- Anforderungen an eine SOA

Service Oriented Architecture

- **Ziele**
  - **Code-Reuse**
    - geringerer Entwicklungsaufwand
    - getestet, zuverlässig
  - **Separation of Concerns**
    - nutzen eines Services ist idR. günstiger als ihn selbst zu implementieren
    - Experten
  - **Loose Coupling**
    - Unterscheidung zwischen notwendigen und ungewollten Abhängigkeiten
    - Minimierung der künstlichen, ungewollten Abhängigkeiten

Service Oriented Architecture

» Service Oriented Model

» Agenten

- Programme die
  - einem Besitzer gehören
  - Services von anderen Agenten anfordern
  - Services für andere Agenten leisten

» Aktionen und Nachrichten

- SOA fokussiert Interaktion zwischen Softwareagenten
- Aktion ist Tätigkeit eines Agenten
  - Senden einer Nachricht
  - Bearbeiten einer empfangenen Nachricht
  - Sonstige Tätigkeiten zur Überführung des Systems in den Zielzustand

Service Oriented Architecture

- **Service Oriented Model**
  - **Service**
    - Wird von einem Provideragenten für einen Kundenagenten bereitgestellt
    - Serviceleistung besteht aus Aktionen
    - Über Schnittstellen zur Verfügung gestellt

Service Oriented Architecture

- **Anforderungen an eine SOA**
  - **einfache und allgegenwärtige Schnittstellen**
    - generische Schnittstellen
  - **deskriptive Nachrichten**
    - Anbieter ist für Lösung des Problems zuständig
  - **Format, Struktur und Vokabular**
    - begrenztes Vokabular für effizientere Kommunikation
  - **Erweiterbarkeit**
  - **Mechanismus zum Auffinden von Serviceprovidern**
  - **Idempotent Requests**
    - Wiederholte Anfragen müssen selbes Ergebnis wie eine einzelne Anfrage liefern

Web Services

- Das Web – Heute und Morgen
- Beispiele
- Definition
- Architektur

Web Services

- **Heute**
  - interaktiver Zugriff auf Dokumente und Applikationen
  - Web-Browser, Audioabspielgeräte, Peer2Peer,…
  - Human-Centric-Web
  - Mensch-Maschine-Kommunikation
- **Morgen**
  - Application-Centric-Web
  - Maschine-Maschine-Kommunikation
- **Der Weg**
  - Web Services & Semantic Web
  - Browser für B2C $\Leftrightarrow$ Web Services im B2B

Beispiele

Komponentendienste

- Währungsumrechnung
- Kreditkartenüberprüfung
- Börsenkurse
- Frachtverfolgung
- …

Zusammengesetzte Dienste

- Reisebuchung
- Reklamationsbearbeitung
- Workflows
- …

» Definition en

Web Services are a new breed of Web application. The are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web Services perform functions that can be anything from simple requests to complicated business processes. […] Once a Web Service is deployed, other applications (and other Web Services) can discover and invoke the deployed service.

Web Services is a technology and process for discovery and connection.

A web service is a service that is available over the Internet, uses a standardized XML messaging system, and is not tied to any one operating system or programming language.

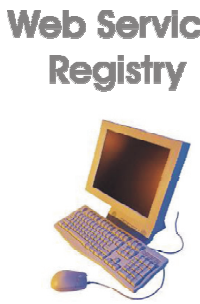
- **Definitionen**
  - **Internet** (auf Internettechnologien aufbauende Netze)

    Stärken des Internets
    - Informationsverteiler
    - Einfachheit
    - Allgegenwärtigkeit
  - **XML**
    - Trennung von Struktur, Inhalt und Form
    - Erweiterbarkeit, Unabhängigkeit
    - XSLT, Xpath, Xquery, Tool-Unterstützung
  - **Unabhängigkeit von**
    - Programmiersprachen
    - Betriebssystemen

Web Services

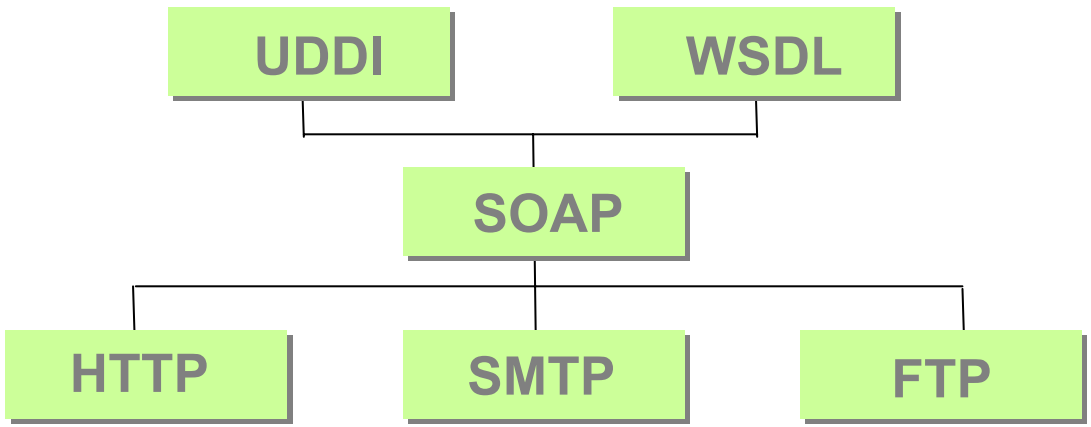
- **Definition**
  - **Modularität**
    - aus einzelnen Komponenten zusammengesetzte Dienste
    - Vorteile bekannt aus dem Bereich Software Engineering
  - **Selbstenthaltend**
    - keine weiteren Abhängigkeiten
    - Transparenz
  - **Selbstbeschreibend**
    - Beschreibung der Schnittstellen zum Service
    - *Datentypen, Transportprotokolle, Adresse, …*
  - **Veröffentlichung**
    - Logisch zentrales, verteiltes Verzeichnis

» Architektur – Web Service Roles



» Architektur – Web Service Stack

Discovery	UDDI
Description	WSDL
XML Messaging	SOAP
Transport	HTTP, SMTP, FTP



Basistechnologien

- **SOAP**
  Austauschformat für Nachrichten
- **WSDL**
  Schnittstellenbeschreibung
- **UDDI**
  Web-Service-Verzeichnis

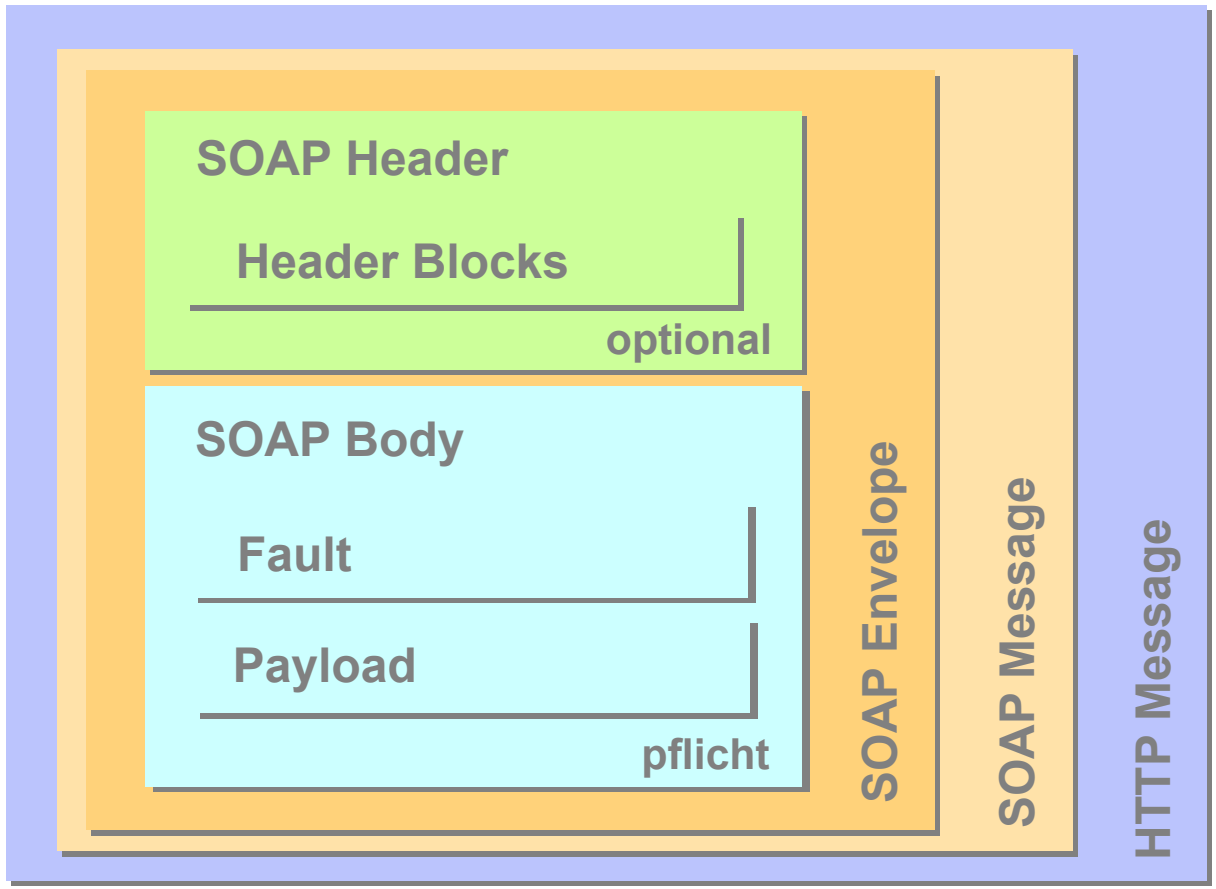
SOAP

- Überblick
- Nachrichten
- Beispiel

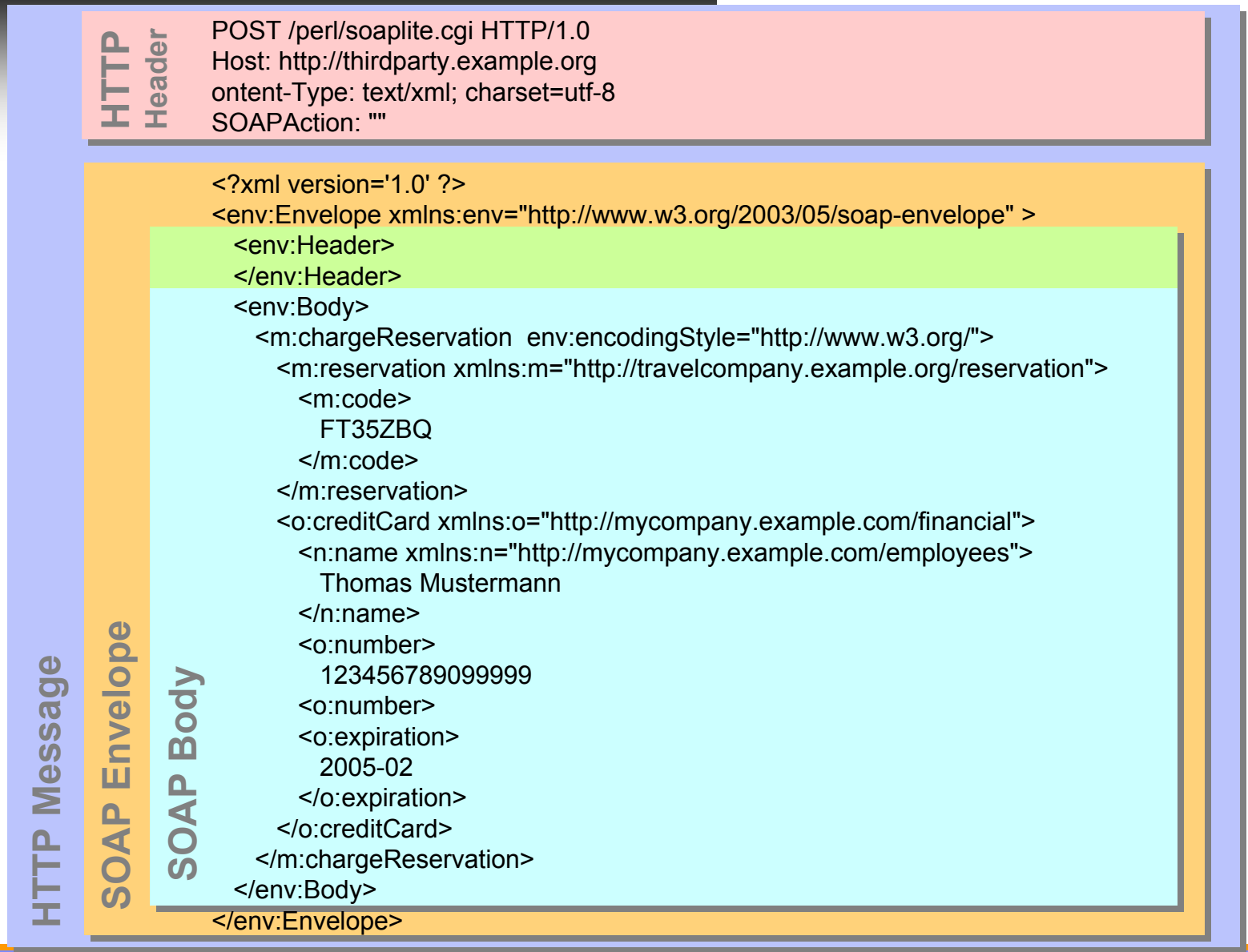
SOAP – Überblick

- Simple Object Access Protocol SOAP
- Framework zum Informationsaustauch
- XML-basierte, strukturierte und typisierte Information
- Envelope
- zustandsloses One-way Paradigma
- keine Aussagen über
  - Semantik
  - Routing
  - Verlässlichkeit

» SOAP – Nachrichen



Basistechnologien



WSDL

- Überblick
- Spezifikation
- Beispiel

WSDL – Überblick

- **Web Service Description Language**
- **Plattform zur Beschreibung der Schnittstelle(n) zu einem Web Service in einer XML-Grammatik**
- **Schnittstellenbeschreibung beinhaltet**
  - Schnittstelle zum Service
  - Verwendete Datentypen
  - Transportprotokolle
  - Adressen
- **vergleichbar mit Java Interfaces**

» WSDL – Spezifikation

<definitions>

<types>	What datatypes will be transmitted?
<message>	What messages will be transmitted?
<portType>	What operations will be supported?
<binding>	How will the messages be transmitted?
<service>	Where is the service located?

<message> What messages will be transmitted?

WSDL – Spezifikation

- **<definitions>**
  - WSDL-Wurzelelement
  - Name des Web Services
  - verwendete Namespaces
- **<types>**
  - Beschreibung aller verwendeter Datentypen
  - Nicht an ein bestimmtes Typensystem gebunden
- **<message>**
  - Beschreibt eine *einfache* Nachricht
    request oder *response*
  - Namen und Parameter bzw. Rückgabewerte

<service> Where is the service located?

WSDL – Spezifikation

- **<portType>**
  - verbinden **<message>**-Elemente zu Operationen
  - Vier Basis Patterns
    one-way, notification, request-response, solicit-response
- **<binding>**
  - Bindung an Protokolle zum Nachrichtenaustausch
    HTTP-PUT, HTTP-POST, SOAP
  - Inhalte protokollspezifisch
- **<service>**
  - Beschreibt Netwerkadressen für den Service
  - Gegliedert durch **<port>**-Elemente

» WSDL – Beispiel

<definitions>

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="HelloService"
  targetNamespace="http://www.wenzler.info/wsdl/HelloService.wsdl"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns:tns="http://www.wenzler.info/wsdl/HelloService.wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

<message>

```
  <message name="SayHelloRequest">
    <part name="firstName" type="xsd:string" />
  </message>
  <message name="SayHelloResponse">
    <part name="greetings" type="xsd:string" />
  </message>
```

<portType>

```
  <portType name="Hello_PortType">
    <operation name="sayHello">
      <input message="tns:SayHelloRequest" />
      <output message="tns:SayHelloResponse" />
    </operation>
  </portType>
```

» W

<definitions>

<binding>

```
<binding name="Hello_Binding" type="tns:Hello_PortType">
  <soap:binding style="rpc"
    transport="http://schemas.xmlsoap.org/soap/http" />
  <operation name="sayHello">
    <soap:operation soapAction="sayHello" />
    <input>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded" />
    </input>
    <output>
      <soap:body
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:examples:helloservice"
        use="encoded" />
    </output>
  </operation>
</binding>
```

<service>

```
<service name="Hello_Service">
  <documentation>
    WSDL file for HelloService
  </documentation>
  <port binding="tns:Hello_Binding" name="Hello_Port">
    <soap:address
      location="http://localhost:8080/soap/servlet/rpcrouter" />
  </port>
</service>
</definitions>
```

UDDI

- Überblick
- White, Yellow und Green Pages
- Datenstruktur

UDDI – Überblick

- Universal Description, Discovery and Integration
- Logisch zentrales, verteiltes Web Service Verzeichnis
- Veröffentlichen und Auffinden von Web Services
- wesentliche Funktionen der API
  - publish
  - find
  - bind
- UDDI Cloud Services

» UDDI – White, Yellow und Green Pages

White Pages

Wer bin ich?

Kontaktinfos

- ✓ Name
- ✓ Anschrift
- ✓ Telefonnummer
- ✓ …

Yellow Pages

Was biete ich an?

Klassifikationen

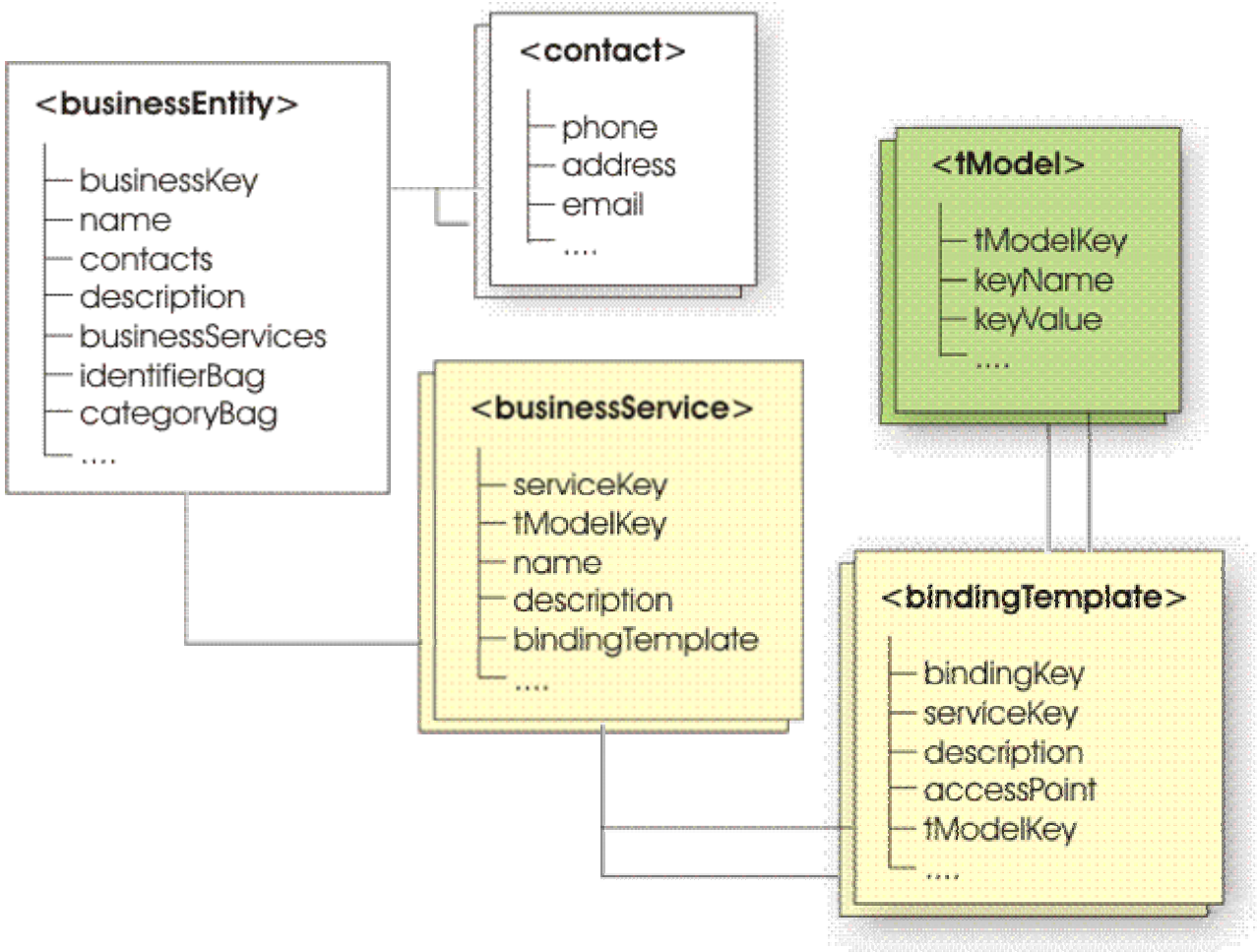
- ✓ Branche
- ✓ Produkt
- ✓ geo. Codes
- ✓ …

Green Pages

Technische Infos

- ✓ Verweis auf externe Spezifikationen
- ✓ Adressen
- ✓ …

» UDDI – Datenstruktur



Web Services vs. CORBA

- Vergleich
- Eigenschaften
- Einsatzgebiete

Web Services vs. CORBA

» Vergleich – Verarbeitungsmodell

» Datenmodell

- CORBA: enge Kopplung zwischen Client und Server
- WS: entkoppelt

» Aufruf-Semantik

- CORBA: Unterstützung der *At-most-once* Semantik
- WS: protokollabhängig, HTTP nicht unterstützt und erfordert somit einen zustandsorientierten Service

» Skalierbarkeit

- CORBA: durch Lastbalancierung skalierbar
- WS: wiederum nicht vom Standard unterstützt, somit Aufgabe des Anwendungsservers

Web Services vs. CORBA

» Vergleich – Verarbeitungsmodell

» Überprüfungen zu Compile- oder Laufzeit

- CORBA: 2 Arten von Schnittstellen IDL und DII
  bei IDL statische Überprüfungen möglich,
  bei DII nur zur Laufzeit
- WS: zur Zeit keine Unterstützung zur statischen Überprüfung
  Zukunft evtl. WSDL Anbindungen für Programmiersprachen
  Laufzeit Wohlgeformtheit des XML
  Überprüfung der Validität jedoch feiner

Web Services vs. CORBA

» Vergleich – Eigenschaften

» Überwindung von Firewalls

- CORBA: Herausforderung, da *CORBA Firewall Traversal Specification* noch nicht unterstützt
- WS: kein Problem

» Sicherheit

- Authentifizierung, Autorisierung, Verschlüsselung, Datenintegrität
- CORBA: *CORBA Security Service*
- WS: keine Standardisierung, aber Aufbau auf SSL oder XML-Signaturen möglich

Web Services vs. CORBA

» Vergleich – Eigenschaften

» Persistenz

- CORBA: *CORBA Persistent State Services*
- WS: keine Standardisierung, Aufgabe Anwendung(sserver)

» Unabhängigkeit

- CORBA und WS sind Plattform- und Programmiersprachen unabhängig

Web Services vs. CORBA

- **Einsatzgebiete**
  - **Web Interfaces**
    - Web Services aufgrund Verknüpfung XML und HTTP
  - **Mobile Endgeräte**
    - Web Services, sich ändernde Netzwerkadressen
  - **Legacy Systeme**
    - CORBA, Schnittstellen existieren bereits
    - Web Services mit Hilfe von CORBA-SOAP Gateways
  - **Thin Clients**
    - Minimale Infrastruktur
    - CORBA

Inhalt und Motivation

- Service Oriented Architecture
- Web Services
- Basistechnologien
- Web Services vs. CORBA

Vielen Dank für Ihre Aufmerksamkeit