Verteilte Systeme

3. Dienstevermittlung

3.3 Web Services als Mittel zur Realisierung einer SOA

Sebastian Iwanowski
FH Wedel

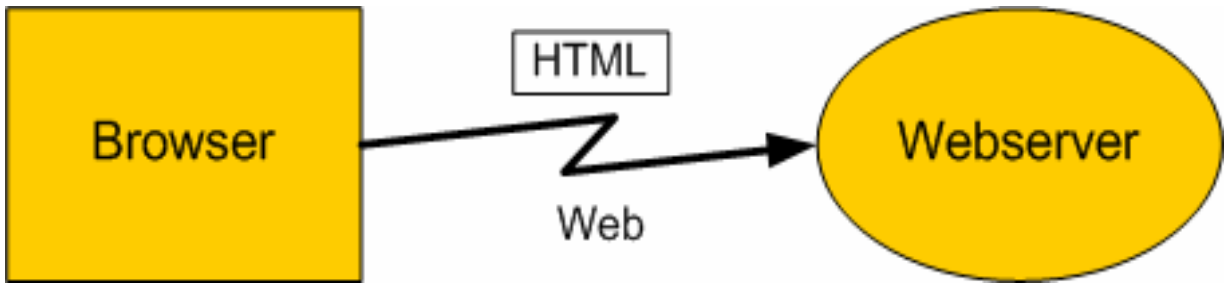
Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. WSDL
5. UDDI

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

Historische Einordnung

Webanwendungen der 1. Generation: HTML Seiten



Am Webserver liegen statische HTML Seiten.

Diese werden über HTTP in den Browser geladen.

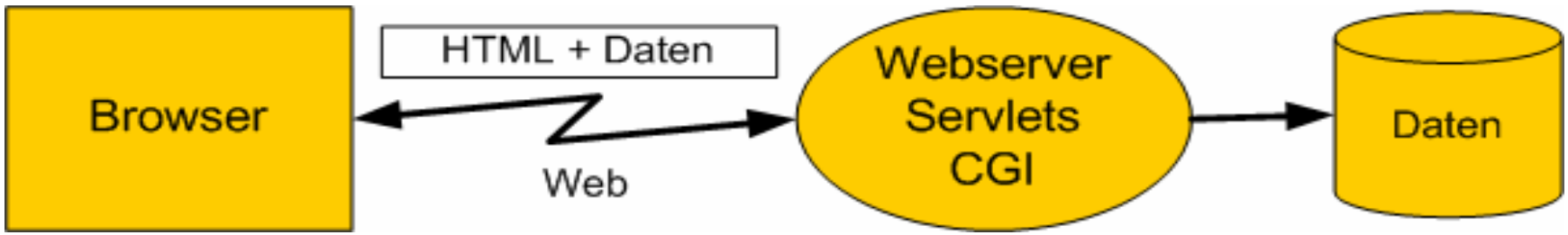
Technologie ab Ende der 1980er Jahre verfügbar.

Mensch-Maschine-Kommunikation

aus Hammerschall: *Verteilte Systeme und Anwendungen*

Historische Einordnung

Webanwendungen der 2. Generation: Dynamische HTML Seiten: CGI, Servlets



Webseiten werden dynamisch (abhängig von Anfrageergebnissen) aufgebaut.

Zugriff auf hinter der Seite liegende Daten ist möglich.

Beispiele für Technologien:

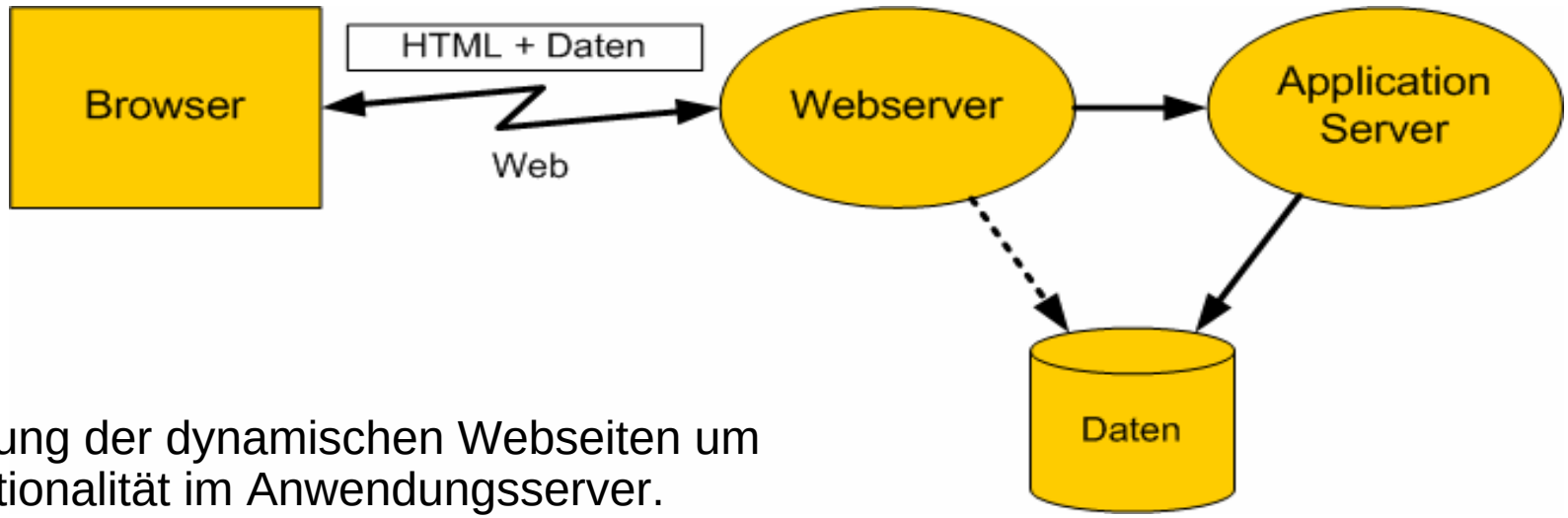
- CGI (Common Gateway Interface): Prozessbasierte Bearbeitung von HTTP-Requests über Skripten (Perl, C)
- Servlets / JSP: Threadbasierte Bearbeitung von HTTP-Requests (nur für Java)

Mensch-Maschine Kommunikation

aus Hammerschall: *Verteilte Systeme und Anwendungen*

Historische Einordnung

Webanwendungen der 3. Generation: Einführung von Mehrschichten-Architekturen



Erweiterung der dynamischen Webseiten um Funktionalität im Anwendungsserver.

Der Webserver leitet die Aufrufe an den Anwendungsserver weiter:

- bearbeitet die Anfrage.
- schickt Ergebnisse an den Webserver zurück.
- Ergebnisse werden von Webserver über HTTP an den Browser zurückgeschickt.

Vorteile: Skalierbarkeit!

Mensch-Maschine Kommunikation

aus Hammerschall: *Verteilte Systeme und Anwendungen*

Historische Einordnung

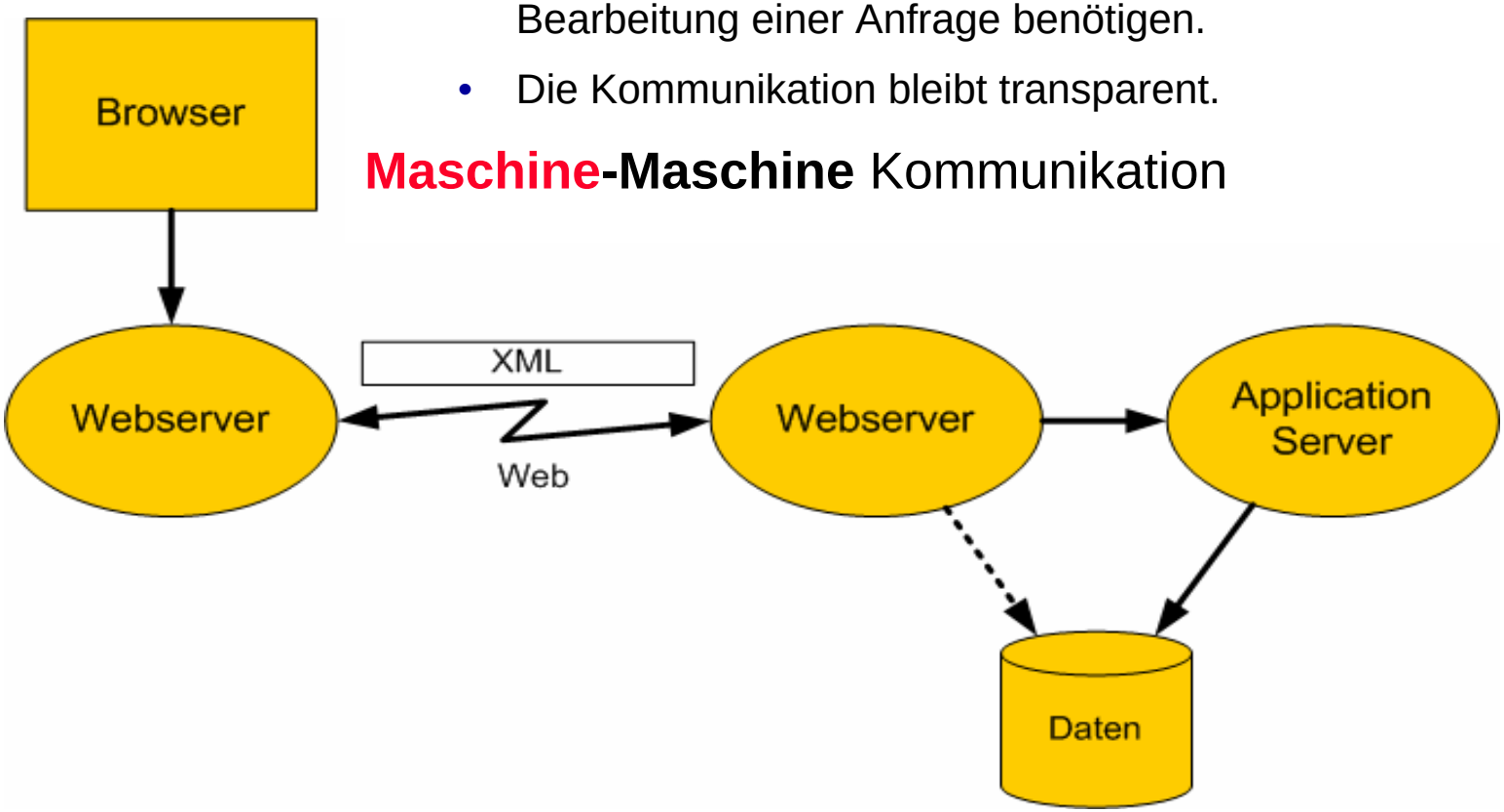
Webanwendungen der 4. Generation: Webservices

Anwendungen (auf Webservern) kommunizieren unabhängig vom Anwender.

Vision Automatisierung:

- Anwendungen suchen sich automatisch Dienste im Web, die sie zur Bearbeitung einer Anfrage benötigen.
- Die Kommunikation bleibt transparent.

Maschine-Maschine Kommunikation



aus Hammerschall: *Verteilte Systeme und Anwendungen*

Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. WSDL
5. UDDI

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

Was ist das Wesentliche eines Webservices?

A web service is any 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.

Ethan Cerami, Autor des Buchs *Web Services Essentials*, 2002

A web service is self-contained, modular, uses open interfaces and is based on internet standards.

UDDI-Konsortium (OASIS)

A web service is written in XML, is affiliated to a URI, and supports direct interaktion between software agents.

W3C

Webservice ist ein Kommunikationsstandard, mit dessen Hilfe eine SOA realisiert werden kann:

SOA

• **Verteilung**	• **prozessorientiert**	• **Sicherheit**
• **Lose Kopplung**	• **Akzeptanz**	• **Verzeichnisdienst**
• **Standards**	• **Einfachheit**	• **Dynamisches Binden**

Dostal, Jeckle et al.

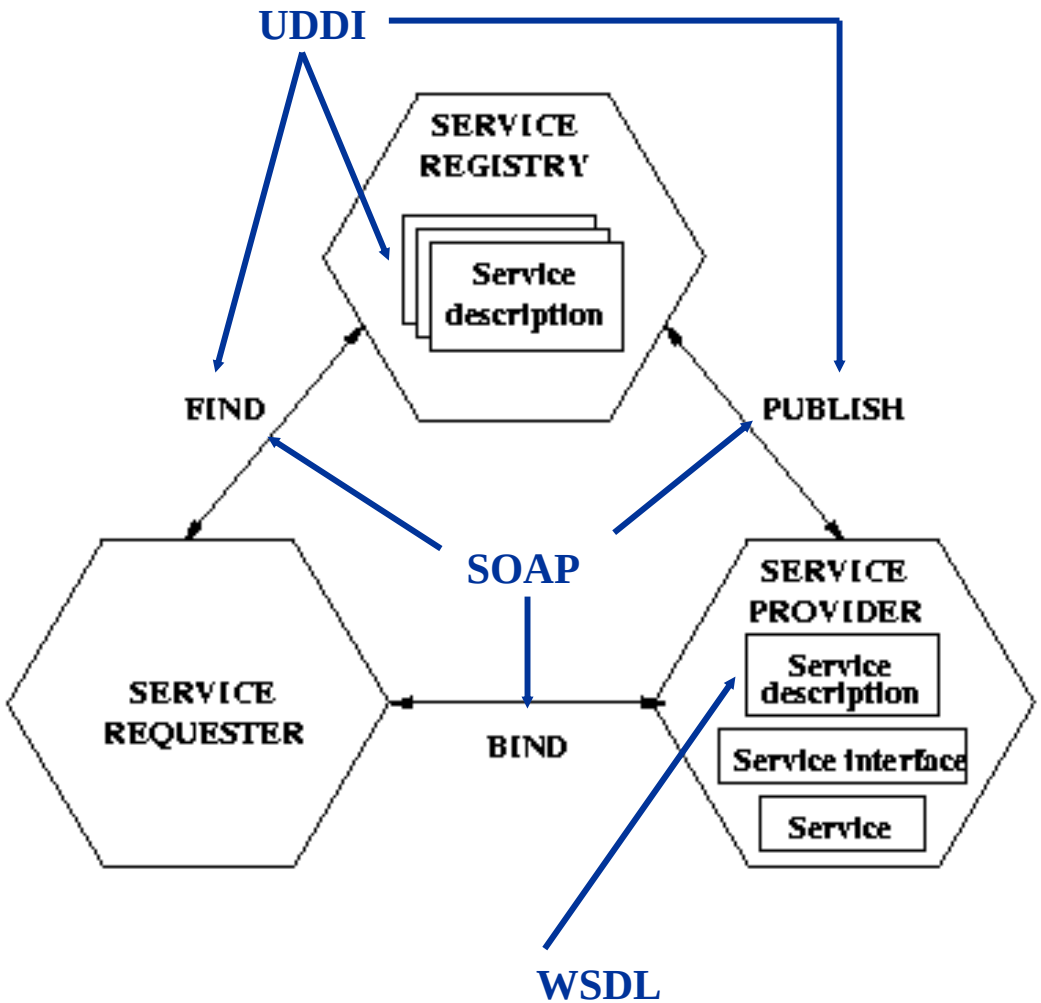
Webservices: Bestandteile

Protokollebenen:

- **Suchdienst** — **UDDI:** Universal Description and Discovery Interface
- **Nachrichtenbeschreibung** — **WSDL:** Web Services Desciption Language
- **Nachrichtenkodierung** — **SOAP:** Simple Object Access Protocol
  Ältere Version: **XML-RPC**
- **Nachrichtentransport** — **HTTP, HTTPS**

  - Die Standards bauen aufeinander auf und greifen ineinander über
  - Die Web Services Standards werden von allen Großen unterstützt (Microsoft, Sun, HP, IBM, …)
  - Für alle Standards gibt es Java-APIs (J2EE) und Microsoft-APIs (.NET)

Webservices: Bestandteile



aus Alonso / Pautasso: graduate course in Lappeenranta, http://www.inf.ethz.ch/personal/alonso/teaching.html

Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. WSDL
5. UDDI

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

SOAP

Entstehungsgeschichte und Ziele

- 1999 initiiert vom W3C, immer noch verantwortlich
- wurde geschaffen zur Realisierung entfernter Prozeduraufrufe
- sollte einfachere Alternative zu CORBA IIOP/GIOP sein
  - Trägerprotokoll: HTTP
- Später Kern zur Entwicklung weiterer Paradigmen (SOA)
  - Weitere Trägerprotokolle möglich (SMTP, Messaging Service)

aus Hammerschall: *Verteilte Systeme und Anwendungen*

SOAP

Komponenten einer SOAP-Beschreibung:

Envelope

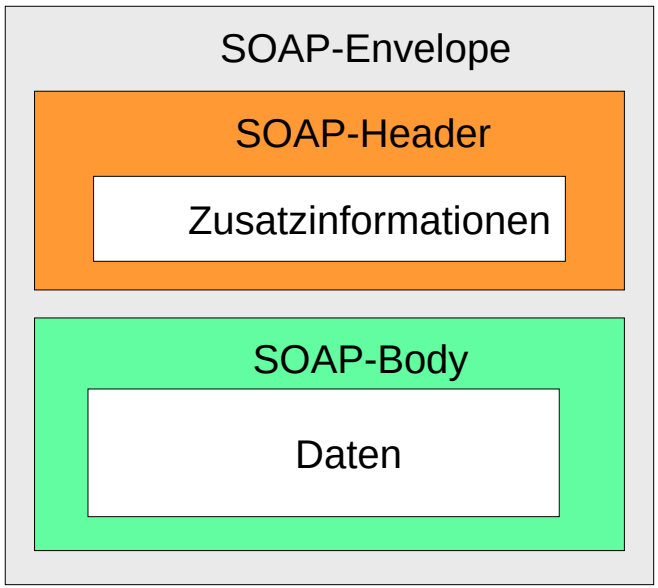
- dient als Container für die gesamte SOAP Nachricht
- muss Konventionen genügen, damit die Nachricht als SOAP Nachricht identifiziert werden kann.

Header (Optional)

- Wie der Header verwendet wird, ist weitgehend Sender und Empfänger überlassen
- Platz für weitere Standards (Sicherheit, Transaktionsmanagement)

Body

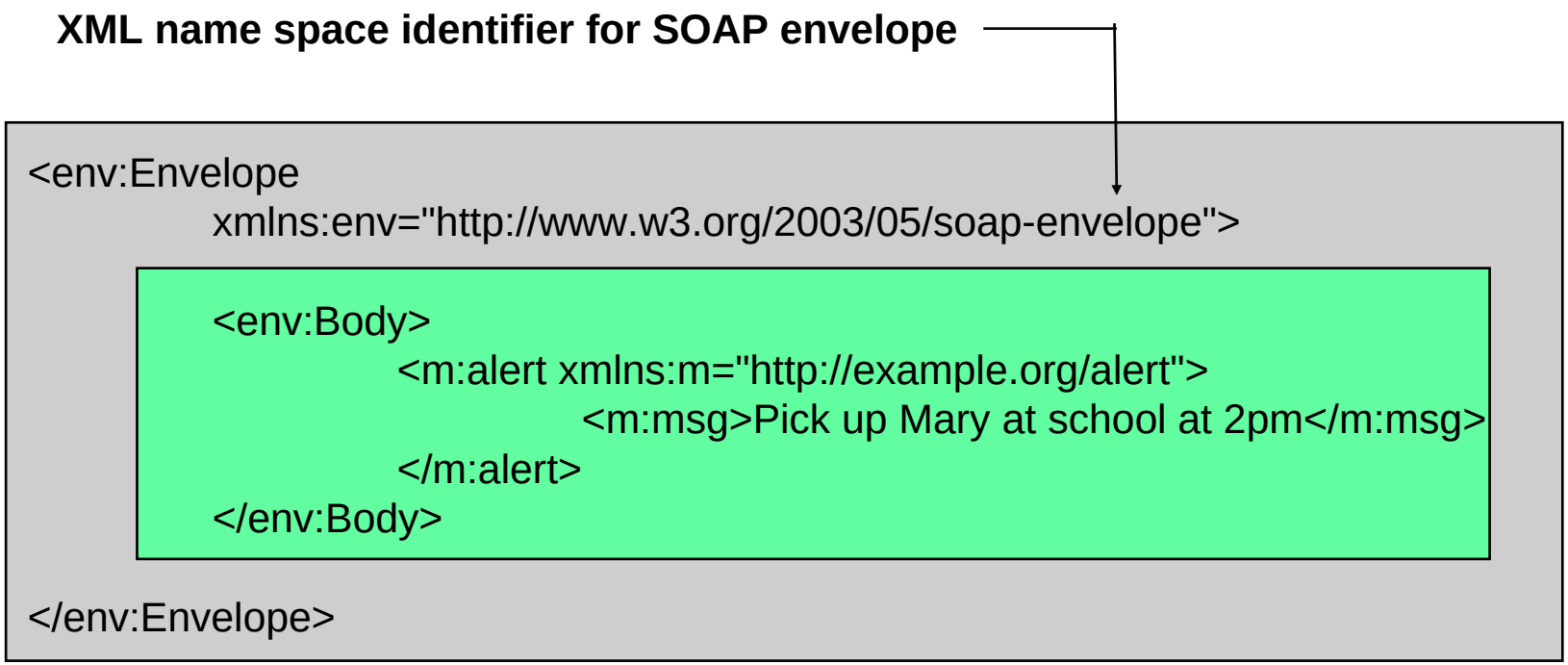
- Nutzdaten
- Funktionsaufrufe mit Daten in XML Format
- Es werden drei Arten von Nachrichten unterschieden: Request, Response, Fault.



aus Hammerschall: *Verteilte Systeme und Anwendungen*

SOAP

Minimale SOAP-Beschreibung:



aus: Simple Object Access Protocol (SOAP) 1.2 ©W3C Recommendation 24 June 2003

SOAP

SOAP-Beschreibung mit Header und Body:

```
<env:Envelope
        xmlns:env="http://www.w3.org/2003/05/soap-envelope">
    <env:Header>
        <n:alertcontrol xmlns:n="http ://example.org/alertcontrol"
                                        env:mustUnderstand="true"  >
            <n:priority>1</n:priority>
            <n:expires>2001-06-22T14:00:00-05:00</n:expires>
        </n:alertcontrol>
    </env:Header>
    <env:Body>
        <m:alert xmlns:m="http://example.org/alert">
            <m:msg>Pick up Mary at school at 2pm</m:msg>
        </m:alert>
    </env:Body>

</env:Envelope>
```

aus: Simple Object Access Protocol (SOAP) 1.2 ©W3C Recommendation 24 June 2003

SOAP

Versenden von SOAP-Nachrichten mit HTTP

HTTP implementiert ein Request/Response Modell der Kommunikation:

- SOAP-Frage wird mit HTTP-Request verschickt.
- SOAP-Antwort wird mit HTTP-Response verschickt.

Da HTTP zustandslos ist, ist auch eine SOAP Kommunikation zustandslos:

- Ein Service kann keinen Zustand zwischen zwei SOAP Aufrufen halten.
- Das kann durch explizite Implementierungen auf den Anwendungsschichten umgangen werden.

Sicherung der Verbindung:

- beispielsweise durch SSL (Secure Socket Layer)

aus Hammerschall: *Verteilte Systeme und Anwendungen*

SOAP

Versenden von SOAP-Nachrichten mit HTTP: Request

```
POST /StockQuote HTTP/1.1
        Host: www.stockquoteserver.com
        Content-Type: text/xml; charset="utf-8"
        Content-Length: nnnn
        SOAPAction: "Some-URI"

    <SOAP-ENV:Envelope
      xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
      SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">

       <SOAP-ENV:Body>
          <m:GetLastTradePrice xmlns:m="Some-URI">
             <symbol>DIS</symbol>
          </m:GetLastTradePrice>
       </SOAP-ENV:Body>

    </SOAP-ENV:Envelope>
```

aus: Simple Object Access Protocol (SOAP) 1.1. © W3C Note 08 May 2000

SOAP

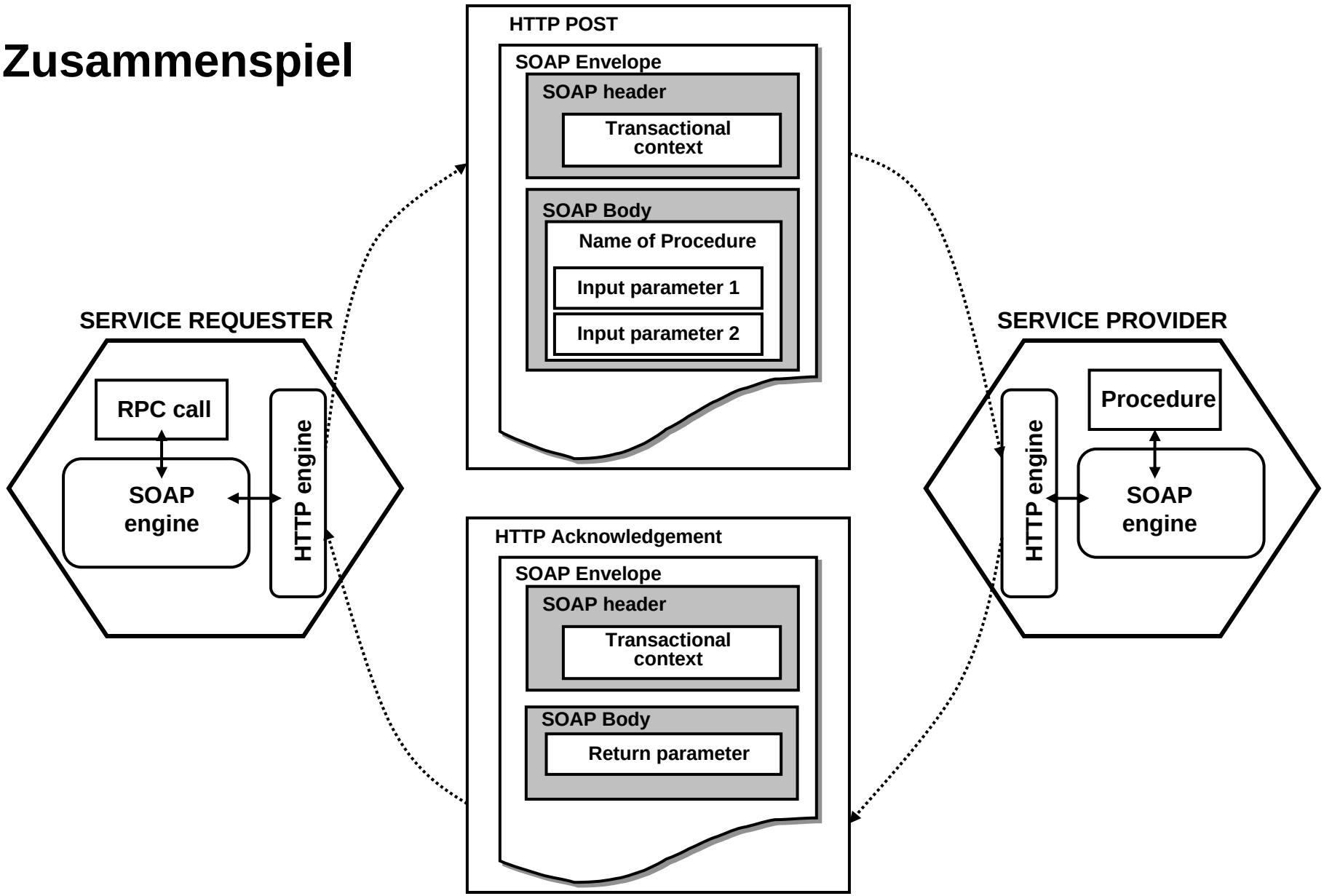
Versenden von SOAP-Nachrichten mit HTTP: Response

```
HTTP/1.1 200 OK
        Content-Type: text/xml; charset="utf-8"
        Content-Length: nnnn

    <SOAP-ENV:Envelope
      xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
         SOAP-ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>

      <SOAP-ENV:Body>
         <m:GetLastTradePriceResponse xmlns:m="Some-URI">
            <Price>34.5</Price>
         </m:GetLastTradePriceResponse>
      </SOAP-ENV:Body>

    </SOAP-ENV:Envelope>
```

aus: Simple Object Access Protocol (SOAP) 1.1. © W3C Note 08 May 2000

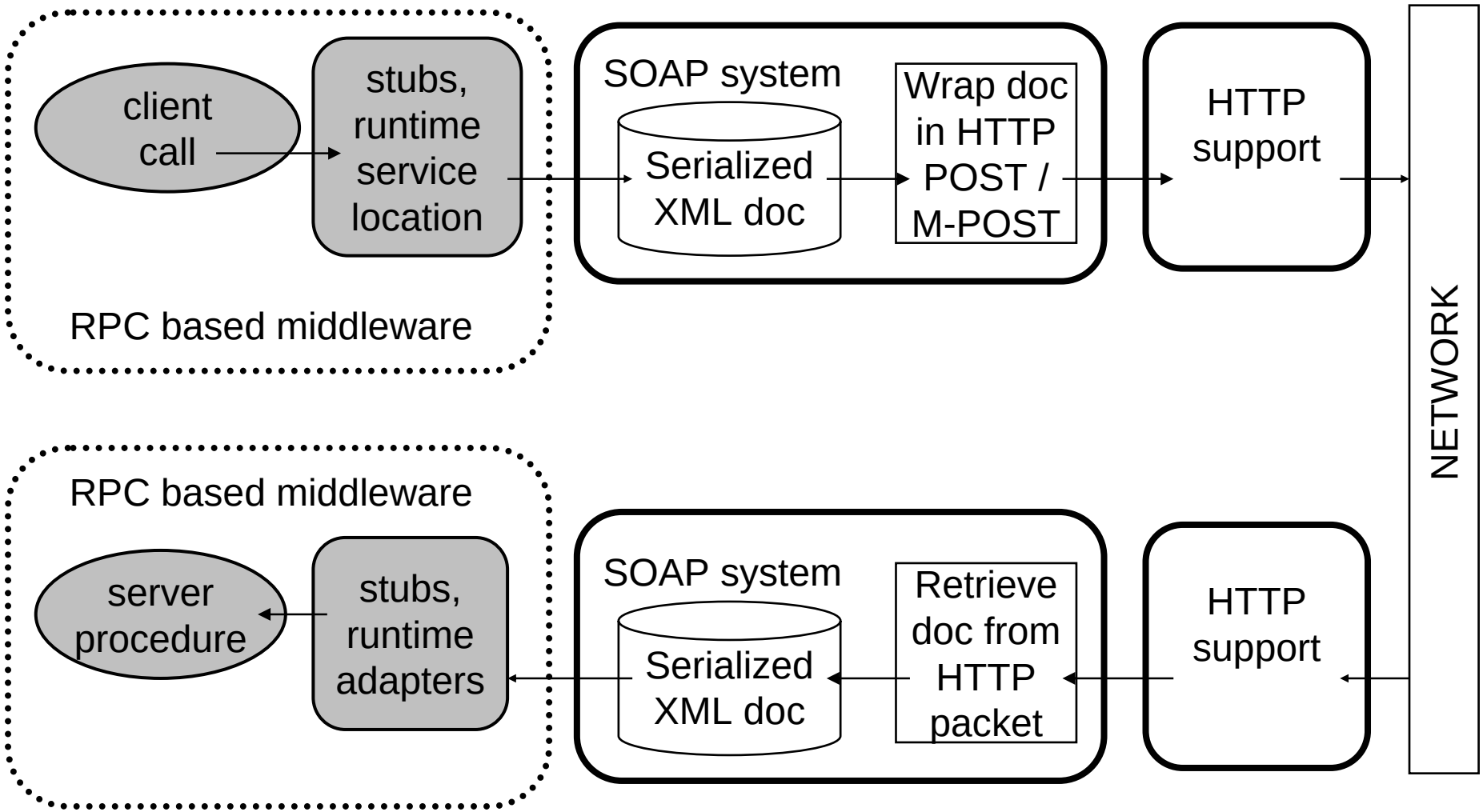
SOAP

Zusammenspiel



SOAP

RPC mit SOAP



aus Alonso / Bussler: EDBT-WebService-Tutorial,
http://www.inf.ethz.ch/personal/alonso/teaching.html

Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. **WSDL**
5. **UDDI**

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

WSDL

Was ist WSDL ?

- Schnittstellensprache zur Veröffentlichung von Diensten als Webservices

Was sind die Ziele von WSDL ?

- Schaffung eines von Applikationen lesbaren Zugangs zu angebotenen Diensten
- weitgehend automatisierte Integration von Diensten zu neuen Anwendungen
- Integration zur Laufzeit

Wie sieht eine WSDL-Beschreibung aus ?

- Vollständige Formulierung in XML
- Der WSDL-Standard definiert ein spezifisches XML-Schema.

aus Hammerschall: *Verteilte Systeme und Anwendungen*

WSDL

Aufbau einer WSDL-Schnittstelle

<definitions>

Wurzelelement eines WSDL Dokuments

definiert Namen und Namensraum des Services sowie den Namensraum der verwendeten Standards

<types>

enthält alle Datentypdefinitionen, die für den Aufruf des Services benötigt werden und nicht im Standard von XML-Schema des W3C definiert sind.

</types>

<message name="Message1">

definiert die Nachrichten, die bei einem SOAP Aufruf übertragen werden

Wenn mehrere Nachrichten vorhanden sind (z.B. für Eingabeparameter und Rückgabewert), so werden mehrere Nachrichten definiert.

Eine Nachricht kann aus logischen Teilelementen bestehen, so genannten Parts.
Ein Part definiert ein Name-Wert Paar zu den Parametern einer Nachricht.

</message>

....

aus Hammerschall: *Verteilte Systeme und Anwendungen*

WSDL

Aufbau einer WSDL-Schnittstelle

.....

<portType>

beschreibt die Methoden, die der Web Service anbietet

Zur Definition der Parameter werden die in Message definierten Nachrichten verwendet.

Für den Aufruf unterstützt WSDL vier Kommunikationstypen:

- **One-way:** Der Client sendet eine Nachricht an den Web Service, eine Antwort wird nicht erwartet.
- **Request-Response:** Der Client sendet eine Nachricht und erhält vom Web Services eine Antwort.
- **Solicit Response:** Der Server sendet eine Nachricht und erhält vom Client eine Antwort.
- **Notification:**.Der Server sendet eine Nachricht an den Client, eine Antwort wird nicht erwartet.

</portType>

.....

Anmerkung

- messages und portType gehören nur zu WSDL 1.1
- In WSDL 2.0 ersetzt durch interface (erste Beispiele in Dostal / Jeckle et al.)

aus Hammerschall: *Verteilte Systeme und Anwendungen*

WSDL

Aufbau einer WSDL-Schnittstelle

.....

<binding>

definiert Nachrichtenformate und Transportprotokoll zur Übertragung der Aufrufe
Häufig wird das SOAP Binding eingesetzt.

</binding>

<service>

definiert alle für den Zugriff auf den Dienst notwendigen Informationen
wie Netzwerkadresse und Portnummer.

</service>

</definitions>

aus Hammerschall: *Verteilte Systeme und Anwendungen*

Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. WSDL
5. **UDDI**

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

UDDI

Umfang von UDDI

Verzeichnisdienst für Webservices.

Veröffentlicht werden:

- WSDL Definitionen von Webservices
- Zusatzinformationen zu Inhalten, verantwortliche Organisation, grobe thematische Einordnung, ...

Definiert zwei APIs:

- Publishing API: Anbieter können über dieses API ihren Webservice veröffentlichen.
- Inquiry API: Anwender können über dieses API nach einem Webservices suchen.

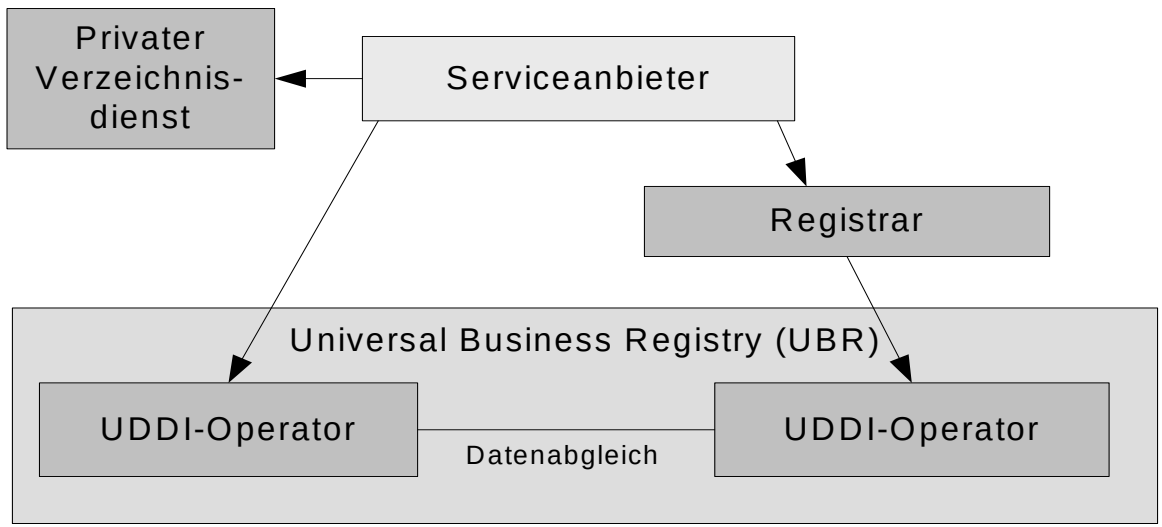
Historie von UDDI

- ursprünglich im Rahmen eines unabhängigen Projekts entwickelt: www.uddi.org (Beginn 2000).
- wurde 2002 an OASIS übergeben.

aus Hammerschall: *Verteilte Systeme und Anwendungen*

UDDI-Organisationsstrukturen

- Der UDDI Standard gibt feste Organisationsstrukturen vor.
- Ein UDDI-Operator ist ein Anbieter einer UDDI Registry, welche die Anforderungen der *UDDI Operators Specification* bezüglich Verfügbarkeit, Sicherheit und Performance erfüllt. (Anm.: Nur wenige Unternehmen sind dazu in der Lage, z.B. IBM, Microsoft, SAP)
- Die UDDI-Operatoren bilden in ihrer Gesamtheit die UBR.
- Daneben sind private Verzeichnisdienste möglich.
- Registrare unterstützen Serviceanbieter bei der Veröffentlichung ihres Webservices.



aus Hammerschall: *Verteilte Systeme und Anwendungen*

UDDI

Specifications:

http://www.uddi.org

http://uddi.org/pubs/uddi-v3.00-published-20020719.pdf

UDDI Business Registry (UBR) nodes:

IBM

Homepage: http://uddi.ibm.com/

Inquiry API:
http://uddi.ibm.com/ubr/inquiryapi

Publish API:
https://uddi.ibm.com/ubr/publishapi

Microsoft

Homepage: http://uddi.microsoft.com/

Inquiry API:
http://uddi.microsoft.com/inquire

Publish API :
https://uddi.microsoft.com/publish

SAP

Homepage: http://uddi.sap.com/

Inquiry API :
http://uddi.sap.com/uddi/api/inquiry

Publish API :
https://uddi.sap.com/uddi/api/publish

NTT

Homepage: http://www.ntt.com/uddi/

Inquiry API :
http://www.uddi.ne.jp/ubr/inquiryapi

Publish API :
https://www.uddi.ne.jp/ubr/publishapi

aus Alonso / Pautasso: graduate course in Lappeenranta, http://www.inf.ethz.ch/personal/alonso/teaching.html

Ablauf einer Webservice-Anwendung

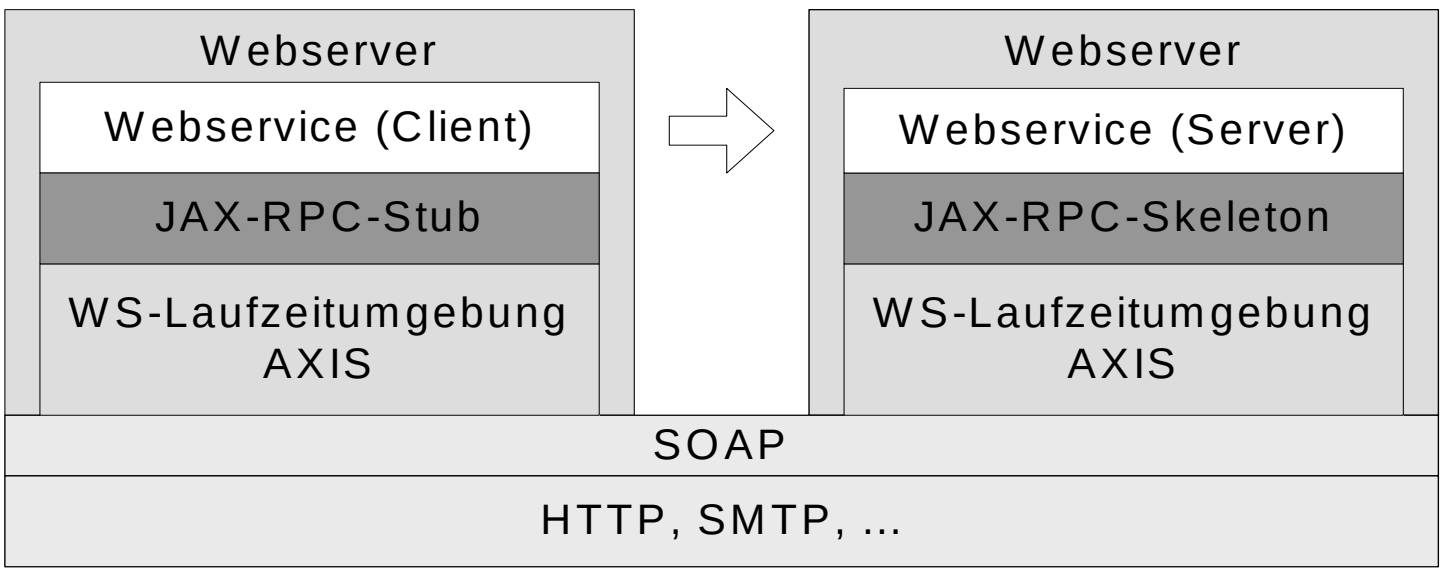
1. Der Anbieter eines Webservices publiziert den Service mit Hilfe einer WSDL-Beschreibung in einem UDDI Service.
2. Der Anwender (eine Anwendung) sucht anhand von Merkmalen einen geeigneten Service.
3. Der Anwender erhält von UDDI eine Referenz (URL) auf den Service
4. Der Anwender holt sich über die URL die WSLD Beschreibung vom Serviceanbieter.
5. Der Anwender generiert sich aus der WSDL Beschreibung den Stub und kann den Dienst nutzen (dynamic binding)

   ... oder der Anwender erhält den Stub bereits zur Compilezeit und kann jeder Zeit ohne Umweg über Verzeichnisdienst den Webservice nutzen. (static binding)

aus Hammerschall: *Verteilte Systeme und Anwendungen*

JAVA-Anbindung für Webservices

- Die Java-APIs for XML-basiertes RPC (JAX-RPC) sind der Standard einer Schnittstelle für Java-Anwendungen auf Webservices.
- SOAP wird um eine Java Schnittstelle erweitert.
- Zur Entwicklung von Webservices wird ein Service-Endpoint-Interface in Java definiert.
- Mit Hilfe eines Compilers wird daraus die WSLD-Schnittstelle generiert.



aus Hammerschall: *Verteilte Systeme und Anwendungen*

Links zu Standards

W3C: XML Specification. http://www.w3.org/TR/REC-xml/

W3C: WSDL Specification. http://www.w3.org/TR/wsdl

W3C: SOAP Specification. http://www.w3.org/TR/soap/

OASIS: WS-Security 2004, BTP, SAML, UDDIv2 Specifications:
http://www.oasis-open.org/specs/index.php

OASIS: UDDIv3 Specification: http://uddi.org/pubs/uddi_v3.htm

OASIS: BPEL: http://www.oasis-open.org/committees/wsbpel

JCP: JAX-RPC. http://java.sun.com/xml/downloads/jaxrpc.html

Apache Axis Projekt. http://ws.apache.org/axis/

aus Hammerschall: *Verteilte Systeme und Anwendungen*

Web Services

1. Historische Einordnung
2. Allgemeiner Aufbau
3. SOAP
4. WSDL
5. UDDI

Vorlesung Verteilte Systeme: Was ist klausurrelevant?

Verteilte Systeme

1. Paradigmenwechsel für Softwarelösungen durch Verteilung

1.1 Einführung durch Beispiele aus der Praxis

Personalisierte dynamische Fahrgastinformation, Marktbasierte Verkehrsleitung, Pheromonbasierte Verlehrsleitung, Touristeninformationssystem

1.2 Allgemeine Anforderungen und Techniken verteilter Systeme

Offenheit, Transparenz (Typen nach ISO), Skalierbarkeit

1.3 Pheromonbasierte Verkehrssteuerung

Vorteile des natürlichen Verfahrens: Probabilistik und Verdunstung, Unterschiede zu künstlichen Ameisen, Algorithmischer Ablauf, Funktionsweise ABC, Nachteile von ABC, Verbesserungen in AntNet, Systemkomponenten, hierarchische Verfahren

schwarz: klausurrelevanter Stoff wie im Vorjahr,
rot: klausurrelevanter Stoff, neu im Vergleich zum Vorjahr,
gold: in Vorlesung behandelt, aber nicht klausurrelevant

Verteilte Systeme

2. Die Client-Server-Beziehung und daraus resultierende Techniken

2.1 Grundlagen

Definition, Protokolle (Callback und Polling), Transaktionen, Mehrschichtenarchitektur, Socket-Schnittstelle (mit Realisierung in Java)

2.2 Nebenläufigkeitstechniken in Java

Definition, Threadkonzept, auch in Java (über Vererbung und Interfaces, Methodenaufrufe), Thread-Handling bei Callback und Polling (Prinzip und Verständnis), Risiken, Vorsorgemaßnahmen (nur Prinzip)

2.3 Entfernte Aufrufe

RPC, RMI: Begriffe, Funktionsabläufe, Unterschiede.
Java-RMI: Welche Klasse / Interface muss was realisieren.

2.4 Objektmigration

Daten, Objekte, Agenten: Unterschiede, unterschiedliche Anwendungsziele, Motivation für Migration, grundlegende Agenteneigenschaften, Abgrenzung zwischen stationären und mobilen Agenten, Migrationstypen (stark vs. schwach, pull vs. push), Agentenkommunikation, Sicherheit bei mobilen Agenten

Verteilte Systeme

3. Dienstevermittlung

3.1 Aufgabenstellungen und erste Lösungen

Forderungen an die Eigenschaften, unterschiedliche Aufgabenstellungen, Lösungsprinzipien: Tupel-Konzept von Jini, konkrete Beispiele aus dem Touristeninformationssystem

3.2 Prinzipien einer SOA

Merkmale, Semantic Web: Sinn und Zweck, Evolution der Architekturen: Begriffswelt (EAI, CRM), B2C vs. B2B: Unterschiede, Prinzip der nachrichtenorientierten Kommunikation, Geschäftsprozesse in einer SOA

3.3 Web Services

Grundlegende Eigenschaften und Zielsetzungen von Webservices, Abgrenzung von Webservices gegenüber früheren Webanwendungen, Funktionalität von SOAP, WSDL und UDDI, Aufbau von SOAP, Zusammenhang zu RPC, Überblick über den inneren Aufbau von WSDL und UDDI

Das war die Vorlesung Verteilte Systeme

Schöne Ferien !