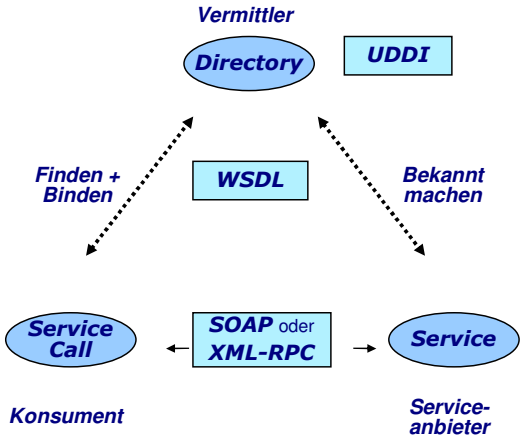
Technischer Überblick zu Web Services

Stephan Karrer

Web Services: Basistechnologien



Web Services sind keine Revolution sondern eher Evolution durch Kombination verschiedener Technologien:

- HTTP, SHTTP, SMTP
- XML

und Konzepte

Auf dieser Basis entstehen:

SOAP, WSDL, UDDI, ...

Anwendungs-Beispiele

- **Suchmaschinen-APIs: Google, Yahoo**
- **Amazon**
  - Abfrage von Produktinformationen
  - Verkaufsunterstützung für Amazon-Produkte
  - Bezahlsystem (micro payment)
  - Virtuelle Rechner- und Speicherkapazität
  - ...
- **Microsoft**
  - Identitätsverwaltung mit Windows CardSpace
  - Schnittstelle zu Microsoft Dynamics
  - MapPoint Web Services (Virtual Earth)
  - ...
- **eBay**
  - Abfrage der Katalog- und Angebotsdaten
  - Benachrichtigungsmechanismus (Auktions-Monitoring)
  - ...

Web Services Implementierungs-Plattformen

- Apache Axis (Weiterentwicklung Apache SOAP, ursprüngliche Basis „SOAP4J“ von IBM)
- IBM Websphere (Apache SOAP 2.3)
- Microsoft WCF (Windows Communication Foundation)
- SAP Netweaver Application Server
- Oracle Application Server
- Bea WebLogic
- sowie alle weiteren führenden (oder auch nicht) Hersteller von Applikationsservern
- Vielzahl von freien Implementierungen für bestimmte Zielsprachen wie PHP, Phyton, ...

HTTP als Transport-Protokoll

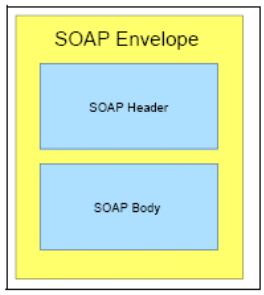
- **Vorteile:**
  - weitreichende Unterstützung und Verfügbarkeit
  - offenes Protokoll
  - HTTP-Daten passieren in der Regel vorhandene Firewalls
  - sichere Kommunikation kann via HTTPS erfolgen

- **Nachteile:**
  - HTTP bietet keine spezielle Servicequalität (QoS)
  - HTTP ist „stateless“
    - ☞ hier muß bei Bedarf auf der Applikationsebene nachgesteuert werden
    - ☞ man kann auch andere Transport-Protokolle verwenden

Weitere Transport-Protokolle (SOAP bindings)

- SOAP Over Email
  - im Body der Email
  - als Anhang (Verwendung von MIME)
- Direkte Verwendung des Transport-Protokolls
  - z.B. UDP, TCP, ...
- SOAP über JMS
  - z.B. Nutzung von Websphere MQ

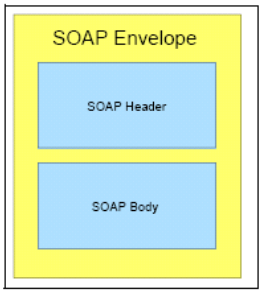
SOAP Nachricht (Simple Object Access Protocol)



- SOAP spezifiziert Nachrichten im XML-Format
- SOAP definiert einen Rahmen für die Inhalte der Nachricht und Regeln, wie diese zu verarbeiten sind
- SOAP definiert auch Regeln für die Bindung an darunterliegende Transport-Protokolle
- Es existieren Codier/Decodier-Schemata für grundlegende Datentypen

```
<?xml version="1.0">
<env:Envelope xmlns:env = http://www.w3c.org/soap-envelope>
        <env:Header> … </env:Header>*
        <env:Body>
                …
                <env:Fault> … </env:Fault>*
                …
        </env:Body>
</env:Envelope>
```

SOAP Nachricht: Bestandteile



- SOAP definiert nicht den Inhalt der einzelnen Elemente wie Header und Body
- Der optionale Header enthält Kontroll-Informationen aus der Anwendungssicht:
  - Welche Knoten müssen was bearbeiten
  - Zustellhinweise
  - Kontext-Informationen bzgl. des Inhalts
- Der Body enthält die eigentliche, applikationsspezifische Nachricht
- Im Falle einer Fehler-Signalisierung besteht der Body aus den Fehlerinformationen

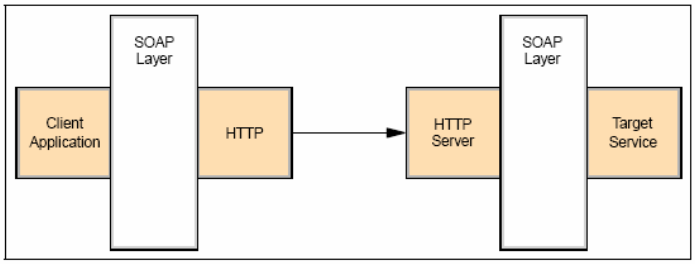
SOAP Document Beispiel:

Reisereservierung

```
<?xml version='1.0' ?>
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope">
 <env:Header>
  <m:reservation xmlns:m="http://travelcompany.example.org/reservation"
        env:role="http://www.w3.org/2003/05/soap-envelope/role/next"
         env:mustUnderstand="true">
   <m:reference>uuid:093a2da1-q345-739r-ba5d-pqff98fe8j7d</m:reference>
   <m:dateAndTime>2001-11-29T13:20:00.000-05:00</m:dateAndTime>
  </m:reservation>
</env:Header>
 <env:Body>
  <p:departure xmlns:p="http://travelcompany.example.org/reservation/travel">
     <p:departing>New York</p:departing>
     <p:arriving>Los Angeles</p:arriving>
     <p:departureDate>2001-12-14</p:departureDate>
  </p:departure>
  <q:lodging
   xmlns:q="http://travelcompany.example.org/reservation/hotels">
   <q:preference>none</q:preference>
  </q:lodging>
 </env:Body>
</env:Envelope>
```

Quelle: W3C

Verarbeitung eines Service-Calls

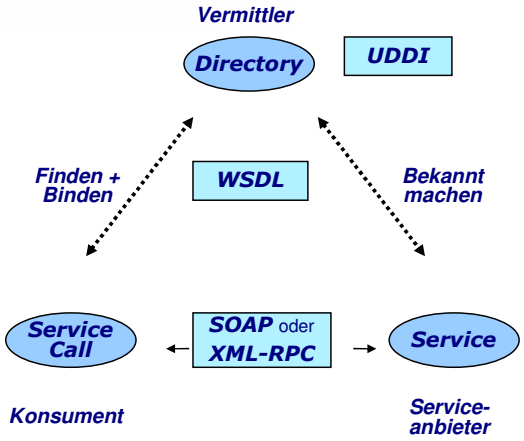


1. Client ruft Methode (Prozedur) via Proxy (Stub) auf
2. SOAP-Layer wandelt den Aufruf in XML-Format
3. SOAP-Layer packt das XML-Format in ein SOAP-Paket (ebenfalls XML-Format)
4. SOAP-Nachricht wird an URI verschickt
5. Web-Server empfängt SOAP-Nachricht
6. SOAP-Layer extrahiert den XML-Aufruf aus der SOAP-Nachricht
7. SOAP-Layer wandelt das XML-Format in einen Aufruf für die Zielumgebung
8. Der Service wird lokalisiert und aufgerufen

Generierung des SOAP-Layers

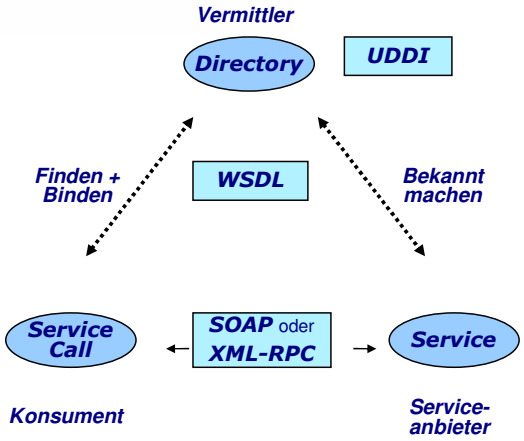
- Die Aufgaben des SOAP-Layers auf der Client- sowie Server-Seite können in hohem Maße automatisiert werden.
- Es existieren für viele Zielsprachen, Laufzeitumgebungen und Transportmechanismen entsprechende Werkzeuge für die Entwickler
- Häufig sind die Services schon vorhanden:
  Entsprechende Werkzeuge generieren dann anhand des vorhandenen Codes möglichst komplett den SOAP-Layer
  - Apache Axis
  - Microsoft .NET Tools

WSDL (Web Service Description Language)



- Wie bei früheren Konzepten kommt auch hier der Schnittstelle zwischen Aufrufer und Aufgerufenem überragende Bedeutung zu
- Der Service wird mittels eines XML-Dialekts, der WSDL, beschrieben
- WSDL entspricht den IDLs anderer Konzepte
- Damit ist quasi die Selbstbeschreibung gegeben (self description)
- Auch hier kann mittels entsprechender Werkzeuge ein erheblicher Anteil der Codierungen auf der Client- bzw. Server-Seite generiert werden
- Die neueste Version ist WSDL Version 2.0 (W3C Recommendation 26 June 2007), am Markt etabliert ist WSDL Version 1.1 (Mitte 2007)

Was ist UDDI



- Universal Description, Discovery, and Integration
- UDDI spezifiziert Protokole für:
  - Veröffentlichung von Services in Verzeichnissen
  - Suchmöglichkeiten nach Services in Verzeichnissen
  - Zugriffskontrolle auf Verzeichnisse
  - Verteilung auf verschiedene Verzeichnisse
- Bietet Lokations- und Aufruf-Metadaten für die dynamische Bindung von Konsumenten an Services zur Laufzeit
- Wird durch das OASIS Standardisierungs-Gremium verwaltet