

1 Web Service Techniken (2)

- Anforderungen an Web-Service-Techniken
 - ◆ system- und programmiersprachen-unabhängige Interoperabilität
 - ◆ unabhängige Schnittstellenbeschreibungen
 - ◆ Dienste zur Lokalisierung
 - ◆ Transaktionen
 - ◆ Sicherheit

2 Politik

- Mehrere Organisation zur Entwicklung der Web-Services-Architektur
 - ◆ Organization for the Advancement of Structured Information Standards (OASIS)
 - Mitglieder: alles was Rang und Namen hat (ähnlich OMG)
 - offene Standards
 - zur Zeit verantwortlich für [UDDI](#)
 - ◆ World-Wide Web Consortium (W3C)
 - Hüterin der Web-Standards
 - offene Standards
 - zur Zeit verantwortlich für [SOAP](#) und [WSDL](#)
 - ◆ Web Services Interoperability Organization (WS-I)
 - prominente Mitglieder: IBM, Microsoft, BEA
 - gebührenpflichtige Standards möglich

K.4 Web Services — Grundlagen und Standards

1 SGML - Standard für Auszeichnungssprachen

- SGML - Standard Generalized Markup Language
 - ◆ Eigentlicher Vorläufer aller anderen Auszeichnungssprachen
 - ◆ Bereits 1986 als ISO Standard 8879 verabschiedet
 - ◆ Sollte der Beschreibung wissenschaftlicher Dokumente dienen
- Eigenschaften von SGML
 - ◆ SGML ist eine Metasprache, d.h.
 - SGML erlaubt die Beschreibung von Sprachen (z.B. HTML)
 - DTD (Document Type Definition) legt die Grammatik fest
 - ◆ SGML-fähige Programme werten DTD aus und lernen damit die Sprache
 - Generische Werkzeuge können jedes Dokument bearbeiten
 - ◆ Wenig etabliert, aufgrund hoher Komplexität

2 XML - Extensible Markup Language

- XML ist ein vereinfachtes SGML
 - ◆ Entwickelt 1998 durch W3C (<http://www.w3c.org/xml/>)
 - ◆ Weiterhin Metasprache
 - ◆ Vereinfachte Syntax (Standard hat nur ca. 35 Seiten)
 - ◆ Leicht lesbares Textformat
 - ◆ Beschreibt keine physischen Eigenschaften, sondern nur logische
- Verwendung von XML
 - ◆ Beschreibung von Sprachen (z.B. XHTML, WML, ...)
 - ◆ Beschreibung von Datenstrukturen (z.B. in Datenbanken)
 - ◆ Beschreibung von Datenübertragungsformaten
 - ◆ ...
- Wird durch Bibliotheken in vielen Sprachen unterstützt

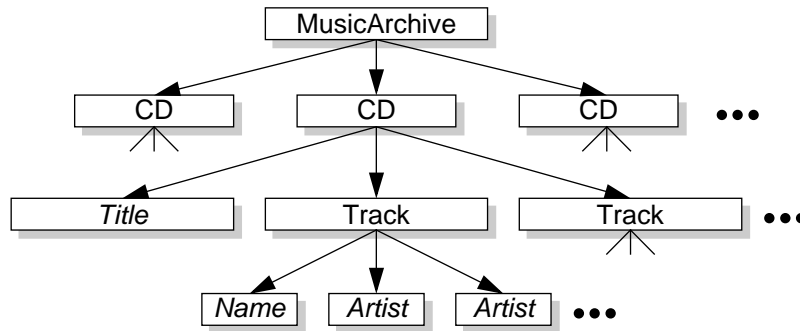
2 Einführung in XML

■ Einführendes Beispiel in XML:

◆ Problemstellung:

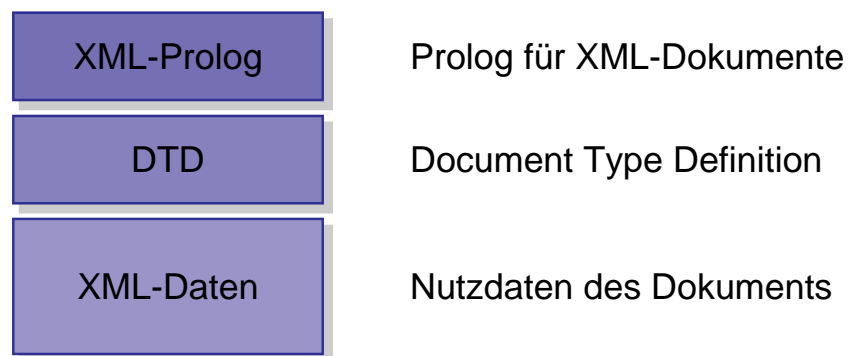
Für ein Musik-Archiv sollen für enthaltenen CDs die essentiellen Daten wie Name der CD, Titel, usw. erfasst werden.

◆ Strukturierung des Datenbestandes:



2 Aufbau eines XML-Dokumentes

■ Jedes XML-Dokument ist aus drei Teilen aufgebaut:



■ Achtung:

- ◆ XML ist „case-sensitive“: <Title> ist ungleich <TITLE>, <title>, ...

2 XML-Daten

- Ausschnitt aus einem entsprechenden XML-Dokument:

```

...
<MusicArchive>
  <CD cdno="1">
    <Title>The Rocky Horror Picture Show</Title>
    <Track no="1" length="4.32">
      <Artist>Richard O'Brien</Artist>
      <Name>Science Fiction/Double Feature</Name>
    </Track>
    <Track no="2" length="2.46">
      <Artist>Barry Bostwick</Artist>
      <Artist>Susan Saradon</Artist>
      <Name>Dammit Janet</Name>
    </Track>
    ...
  </CD>
  <CD cdmo="2">
    ...
  </CD>
  ...
</MusicArchive>

```

2 Die Document Type Definition (DTD)

- Legt die Struktur des XML-Dokumentes fest, d.h.
 - ◆ Die erlaubten Elemente
 - ◆ Die erlaubten Attribute eines Elements, incl. Default-Werte
 - ◆ Die erlaubte Schachtelung der Elemente

```

<Track no="2" length="2.46">
  <Artist>Barry Bostwick</Artist>
  <Artist>Susan Saradon</Artist>
  <Name>Dammit Janet</Name>
</Track>

```

- Spezifikation, welche DTD verwendet werden soll:

```
<!DOCTYPE MusicArchive ... >
```



← Name des Wurzelements

2 Die Document Type Description (DTD)

- Kann innerhalb des Dokuments spezifiziert werden:

```
<!DOCTYPE MusicArchive [ .... ]>
```

- ◆ Vorteilhaft bei kurzen DTDs
- ◆ Nur wenn DTD nicht in mehreren Dateien verwendet werden soll!
- ◆ Muss vor den Daten der XML-Datei stehen

- Kann auf externe DTD-Datei verweisen:

```
<!DOCTYPE MusicArchive SYSTEM "cd_db.dtd"> oder
```

```
<!DOCTYPE MusicArchive SYSTEM "http://www.myweb.de/cd_db.dtd" >
```

- ◆ Gut bei längeren DTDs
- ◆ Essentiell bei Mehrfachverwendung (Konsistenz!)
- ◆ Datei wird vom spezifizierten Ort geladen


- Mischung beider Varianten möglich

- ◆ interne Variante überschreibt externe

2 Die Document Type Description (DTD)

- Beispiel für das Musik-Archiv:

```
<!ELEMENT MusicArchive (CD)* >
```




Das **MusicArchive** besteht aus beliebig vielen CDs.

2 Die Document Type Description (DTD)

■ Beispiel für das Musik-Archiv:

```
<!ELEMENT MusicArchive (CD)* >
<!ELEMENT CD (Title, Track+) >
```




Jede **CD** besteht aus einem **Title** und beliebig vielen **Tracks**.

2 Die Document Type Description (DTD)

■ Beispiel für das Musik-Archiv:

```
<!ELEMENT MusicArchive (CD)* >
<!ELEMENT CD (Title, Track+) >
<!ELEMENT Title (#PCDATA)>
```



Der **Title** besteht aus beliebigen, nicht weiter relevanten Zeichen.

2 Die Document Type Description (DTD)

■ Beispiel für das Musik-Archiv:

```
<!ELEMENT MusicArchive (CD)* >
<!ELEMENT CD (Title, Track)+>
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Track (Artist+, Name)>
```



Ein Track besteht aus mindestens einem **Artist** und einem **Namen**

2 Die Document Type Description (DTD)

■ Beispiel für das Musik-Archiv:

```
<!ELEMENT MusicArchive (CD)* >
<!ELEMENT CD (Title, Track+) >
<!ELEMENT Title (#PCDATA)>
<!ELEMENT Track (Artist+, Name)>
<!ELEMENT Artist (#PCDATA)>
<!ELEMENT Name (#PCDATA)>
```



Artist und **Name** enthalten wieder nicht weiter relevante Zeichen.

2 XML Schema

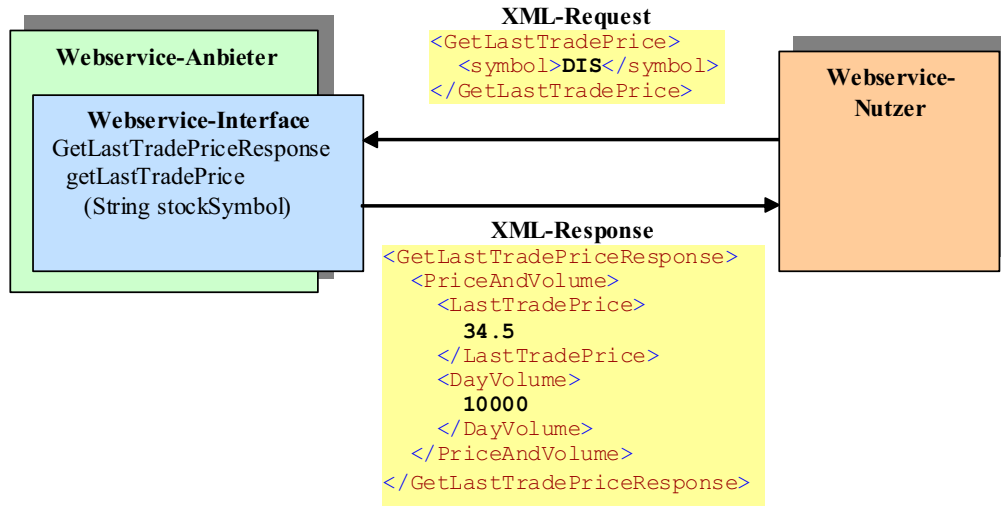
- DTDs kennen als Basistypen nur Zeichenketten
- XML Schema erlauben die Beschreibung erheblich komplexerer Datenformate
 - ▶ unterschiedliche Datentypen (vordefinierte und selbst-definierte)
 - ▶ Subtypisierung
 - ▶ Namensräume

3 XML-Resumee

- XML ist weitgehend etabliert
 - ▶ Software zur Bearbeitung von XML-Beschreibungen weit verbreitet
- XML ist beliebig mächtig
 - ▶ Beschreibungen werden schnell sehr komplex
 - ▶ Tendenzen zum Missbrauch
 - z. B. Beschreibung von Abläufen in XML (if- und while-Konstrukte, parsende Software wird damit zum Interpreter)

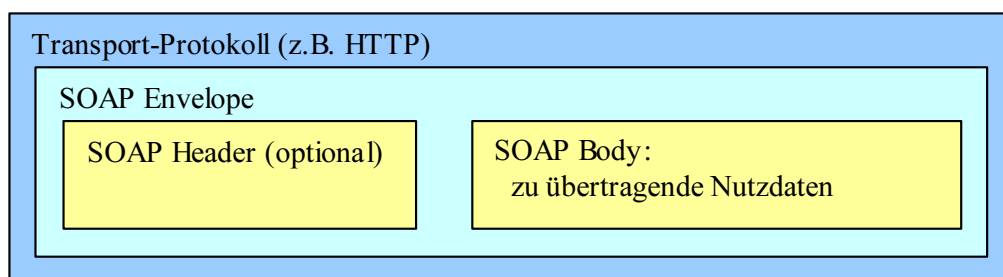
4 Simple Object Access Protocol — SOAP

- Protokoll für den Austausch von strukturierten und typisierten Daten
- Nachrichtenformat basiert auf XML (XML Schema beschreibt Typ)
- Beispiel: RPC auf XML-Basis



4 SOAP (2)

- Aufbau einer SOAP-Nachricht



- SOAP Envelope
 - beschreibt Aufbau der Nachricht
- SOAP Encoding Rules
 - beschreiben wie verschiedene Datentypen übermittelt werden
 - Basis: XML Schema

4 SOAP (3)

■ Beispiel:

```

POST /StockQuote HTTP/1.1 HTTP-Header
Host: www.stockquoteserver.com
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:GetLastTradePrice xmlns:m="Some-URI">
      <symbol>DIS</symbol>
    </m:GetLastTradePrice>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

■ Transport von SOAP-Nachrichten über HTTP

- + Kommunikation durch Firewalls relativ problemlos
 - ➔ B2B-Kommunikation einfach realisierbar
- Kommunikation durch Firewalls relativ problemlos
 - ➔ bestehende Sicherheitsmechanismen im Unternehmen werden kompromittiert
 - großes Risiko durch unkontrollierte Kommunikation

5 Web Services Description Language — WSDL

■ IDL für Web Services

■ Beschreibung von

- Datentypen
- Operationen
- Nachrichtenformaten (Messages)
- Operationen an einem Kommunikationsendpunkt (Port-Typ)
- Verbindung von einem Port-Typ mit einem Protokoll und Datenformat (Binding) — zur Zeit nur SOAP über HTTP bzw. E-Mail
- Verknüpfung eines Binding mit einer konkreten Netzadresse (Port)
- Port-Mengen (Service)

■ Ziel: unabhängige Beschreibung

- von Schnittstellen,
- den zur Nutzung auszutauschenden Nachrichten und
- der Verwendung der Nachrichten mit unterschiedlichen Protokollen

5 WSDL (2)

- Kommunikationsmuster
 - ◆ *One Way*: Nachricht vom Client zum Web-Service
 - ◆ *Request Reply*: Aufruf vom Client zum WS, Antwort zurück
 - ◆ *Solicit Message*: Bitte vom WS zum Client, Antwort zurück (noch nicht spezifiziert)
 - ◆ *Notification*: Nachricht vom WS zum Client (noch nicht spezifiziert)

5 WSDL — Überblick einer WSDL-Beschreibung

■ Komponenten von WSDL

```

<wSDL:definitions name='Account'
  xmlns:wSDL='http://schemas.xmlsoap.org/wSDL/'
  ...>
<wSDL:types> ... </wSDL:types>
<wSDL:message name='withdraw'> ... </wSDL:message>
...
<wSDL:portType name='AccountPort'> ... </wSDL:portType>
...
<wSDL:binding name='AccountSoapBinding'>...</wSDL:binding>
...
<wSDL:service name='Account'>
  <wSDL:port name='AccountPort'
    binding='AccountSoapBinding'>
    ...
  </wSDL:port>
</wSDL:service>
</wSDL:definitions>

```

5 WSDL — Typdefinitionen

■ Definition des SOAP-Rumpfs und -Kopfs

- ◆ Typisierung durch XML-Schema-Definitionen in der `types`-Sektion der WSDL-Beschreibung

```
<wsdl:types>
  <xsd:schema
    xmlns:xsd='http://www.w3.org/2001/XMLSchema'>
    <xsd:complexType name='withDrawDepositParm'>
      <xsd:sequence>
        <xsd:element name='amount' type='xsd:double'>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
    <xsd:element name='withdraw'
      type='withDrawDepositParm' />
    <xsd:element name='deposit'
      type='withDrawDepositParm' />
  </xsd:schema>
  ...
</wsdl:types>
```

5 WSDL — Nachrichtendefinition

■ Definition der SOAP-Nachrichten

- ◆ Aufführen der einzelnen Nachrichtentypen

```
<wsdl:message name='WithdrawRequest'>
  <wsdl:part name='Withdraw' element='withdraw' />
</wsdl:message>

<wsdl:message name='WithdrawReply'>
  <wsdl:part name='WithdrawResponse'
    element='withdrawResponse' />
</wsdl:message>
...
```

- Hinweis auf die in den Nachrichten enthaltenen Elementtypen

5 WSDL — Abstrakte Schnittstellen

■ Definition der Schnittstellen (*Port Types*)

- ◆ Aufführen der einzelnen Operationen und der dazugehörigen Nachrichten

```
<wsdl:portType name='AccountPort'>
  <wsdl:operation name='withdraw'>
    <wsdl:input message='WithdrawRequest' />
    <wsdl:output message='WithdrawReply' />
  </wsdl:operation>

  <wsdl:operation name='deposit'>
    <wsdl:input message='DepositRequest' />
    <wsdl:output message='DepositReply' />
  </wsdl:operation>
</wsdl:portType>
```

5 WSDL — Bindung an das Übertragungsprotokoll

■ Beispiel: Bindung an HTTP

```
<wsdl:binding name='AccountSoapBinding'
  type='AccountPort'>
  <soap:binding style='document'
    transport='http://schemas.xmlsoap.org/http' /
    xmlns:soap='http://schemas.xmlsoap.org/wsdl/soap'>
    <wsdl:operation>
      <wsdl:input><soap:body use='literal' />
    </wsdl:input>
      <wsdl:output><soap:body use='literal' />
    </wsdl:output>
    </wsdl:operation>
    ...
</wsdl:binding>
```

5 WSDL — Bindung an das Übertragungsprotokoll (2)

- Bedeutung des `use`-Attributs
 - ◆ Codierung gemäß XML-Schema bei `literal`
 - ◆ eigene Codierung bei `encoded`
- Bedeutung des `style`-Attributs
 - ◆ Einfügen der Elemente gemäß XML-Schema bei `document`
 - ◆ Einfügen mit automatisch generiertem Element bei `rpc`

5 WSDL — Dienstbeschreibung

- Beschreibung einer konkreten Dienstinstanz

```

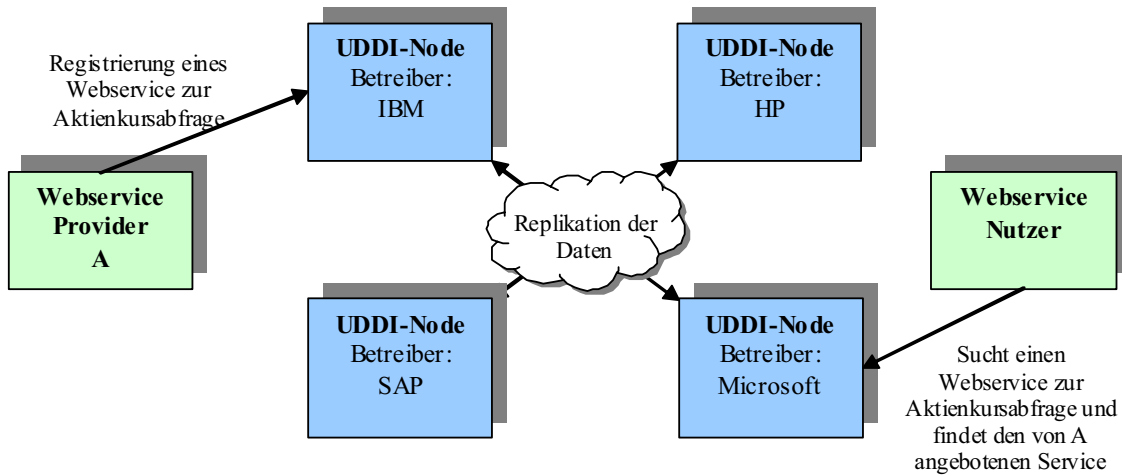
wsdl:service name='Account' >
  <wsdl:port name='AccountPort'
    binding='AccountSoapBinding' >
    <soap:address
      location='http://www.treasurebank.com/WS/Account.jsp' />
    </wsdl:port>
  </wsdl:service>

```

6 Universal Description, Discovery and Integration — UDDI

■ Nameservice und Interface Repository

- globales Web Service Verzeichnis
- Informationen über Unternehmen und die angebotenen Schnittstellen



6 UDDI (2)

- Kommunikation mit UDDI erfolgt über SOAP
- UDDI-Schnittstellen sind in WSDL beschrieben
- white pages
 - ◆ allgemeine Informationen über Unternehmen
- yellow pages
 - ◆ Kategorisierung von Unternehmen
- green pages
 - ◆ technische Informationen über Web Services und Schnittstellen

K.5 Web-Services in Java

- ★ Sun ONE (Open Net Environment)
- Java API für XML Message (JAXM) 1.1
 - ◆ Schnittstelle zum Versenden und Empfangen von SOAP-Nachrichten
- SOAP with Attachments API for Java (SAAJ) 1.1
 - ◆ Schnittstelle zur Erzeugung von SOAP-Nachrichten mit Anhängen
 - Anhänge ähneln den Anhängen von E-Mails
 - Dateien beliebigen Inhalts können angehängt werden

K.5 Web-Services in Java (2)

- Java API für XML RPC (JAXR)1.0
 - ◆ Umsetzung von Java-Remote-Interfaces auf Web-Services-Schnittstellen
 - ◆ Java/WSDL-Mapping
 - ◆ Umsetzung von Web-Services-Schnittstellen auf Java
 - ◆ WSDL/Java-Mapping
 - ◆ Tools zum Umwandeln der Mappings
 - ◆ Integration mit EJB, Servlets und JSP

K.6 Vergleich mit objektbasierter Middleware

- Web-Services ist dienstbasiert
 - ◆ Beispiel: Account
 - typisch ein Webservice, Kontonummer als Parameter
 - statt mehrere Account-Objekte
- Web-Services kennen Referenzen auf Web-Services
 - ◆ jedoch keine transparente Kommunikation von Referenzen

1 Minimalanforderungen

- Eindeutige Dienstbezeichner
 - ◆ gegeben durch Schnittstellenbeschreibung und Binding
 - typisch: URL
- Erzeugung neuer Web-Services
 - ◆ nicht vorgesehen
- Schnittstellenspezifische Stellvertreterobjekte
 - ◆ Stubobjekte aus WSDL-Beschreibung
- RPC-basiertes Kommunikationsprotokoll
 - ◆ SOAP über HTTP, E-Mail etc.
 - ◆ beliebige andere Trägerprotokolle verwendbar
 - neues SOAP-Binding erforderlich

1 Minimalanforderungen (2)

- Erzeugen neuer lokaler Stellvertreter aus übermittelten Referenzen
 - ◆ nicht vorgesehen
- Automatische Generierung der Stellvertreter
 - ◆ Werkzeuge auf verschiedenen Plattformen
- Namensdienst zum Finden von Web-Services
 - ◆ ausgefeilter Beschreibungsdienst: UDDI

K.7 weitere Standards und Entwicklungen

- Web Services Inspection Language (WSIL)
 - ▶ Auffinden von Web Services über normale Web-Seiten eines Unternehmens
- Beschreibung von Geschäftsprozessen
 - ▶ Web Services Flow Language (WSFL) – IBM
 - ▶ Web Services for Business Process Design (XLANG) – Microsoft
- Transaktionen
 - ▶ Business Transaction Protocol (BTP)
 - ▶ Transaction Authority Markup Language (XAML)
- Sicherheit
 - ▶ SOAP Security Extensions
 - ▶ XML Encryption
 - ▶ Security Assertion Markup Language (SAML)
 - ▶ Extensible Access Control Markup Language (XACML)