

## 3.2 SOAP-Umschlag (10)

### ■ Verarbeitung in Zwischenknoten (fortges.)

#### ◆ Beispiel für veränderte Anfragenachricht

```
<?xml version='1.0' ?>

<soap:Envelope
  xmlns:soap="http://www.w3.org/2002/12/soap-envelope">
  <soap:Header>
  <u:account-ref xmlns:u="http://ufin.com/acc"
    soap:role="http://www.w3.org/2002/12/soap-envelope/role/next"
    soap:mustUnderstand="true">
    <u:original-bank>Treasure</u:original-bank>
    <u:account-number>4711</u:account-number>
  </u:account-ref>
  </soap:Header>

  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope>
```



## 3.3 Codierungsregeln (2)

### ■ Mögliche URI-Werte

#### ◆ leere URI

- keine Angabe zur Abbildungsregel

#### ◆ anwendungsspezifische URI

- z.B. zur Kennzeichnung eines bestimmten XML-Formats

#### ◆ SOAP-Encoding

- für aktuelle Version SOAP 1.2:

**http://www.w3.org/2003/05/soap-encoding**



## 3.3 Codierungsregeln

### ■ Abbildungsregeln für Body- und Header-Daten in ein XML-Dokument

#### ◆ festgelegt durch freies Attribut **soap:encodingStyle**

(im Umschlagnamensraum) bei einem Body- oder Header-Element

- Wert ist eine URI, welche die Abbildungsregel charakterisiert

#### ◆ Attribut gilt für aktuelles und alle eingebetteten Elemente

```
...
<soap:Header>
<u:account-ref xmlns:u="http://ufin.com/acc"
  soap:role=
    "http://www.w3.org/2002/12/soap-envelope/role/next"
  soap:mustUnderstand="true"
  soap:encodingStyle=
    "http://www.w3.org/2003/05/soap-encoding">
  <u:original-bank>Treasure</u:original-bank>
  <u:account-number>4711</u:account-number>
...

```



## 3.3 Codierungsregeln (3)

### ■ SOAP-Encoding

#### ◆ Rückgriff auf XML-Schema

#### ◆ Definition der Abbildung komplexer Datenstrukturen

#### ◆ Beispiel:

```
class Buch {
  int Bestand= 25;
  String Titel= "Web Services";
  String [] Autor= new String[2]= { "A. Müller", "B. Meier" };
};
```



### 3.3 Codierungsregeln (4)

#### ■ SOAP-Encoding (fortges.)

##### ◆ Abbildung des Beispiels:

```
<Buch xmlns:xsi=
  "http://www.w3.org/2001/XMLSchema-instance"
  soap:encodingStyle=
  "http://www.w3.org/2003/05/soap-encoding"
  xmlns:enc=
  "http://www.w3.org/2003/05/soap-encoding">
  <Bestand xsi:type="xsi:int">25</Bestand>
  <Titel xsi:type="xsi:string">Web Services</Titel>
  <Autor enc:arrayType="xs:string[2]">
    <a>A. Müller</a>
    <a>B. Meier</a>
  </Autor>
</Buch>
```

##### ◆ Typisierung kann auch über Verweis auf ein Schema erfolgen



### 3.4 Fehlernachrichten (2)

#### ◆ Beispiel: (fortges.)

```
<soap:Envelope
  xmlns:soap="http://www.w3.org/2002/12/soap-envelope"
  xmlns:xml="http://www.w3.org/XML/1998/namespace">
  <soap:Header>... </soap:Header>
  <soap:Body>
    <soap:Fault>
      <soap:Code>
        <soap:Value>soap:MustUnderstand</soap:Value>
      </soap:Code>
      <soap:Reason>
        <soap:Text xml:lang="de">Mindestens ein
          Pflichtfeld wurde nicht verstanden</soap:Text>
        <soap:Text xml:lang="en">... </soap:Text>
      </soap:Reason>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```



### 3.4 Fehlernachrichten

#### ■ Mitteilung von Aufruffehlern

##### ◆ im **Body**-Element gibt es genau ein **Fault**-Element

##### ◆ Beispiel: Zwischenknoten versteht Header-Element nicht

```
<soap:Envelope
  xmlns:soap="http://www.w3.org/2002/12/soap-envelope"
  xmlns:xml="http://www.w3.org/XML/1998/namespace">
  <soap:Header>
    <soap:NotUnderstood QName="u:account-ref"
      xmlns:u="http://ufin.com/acc"/>
  </soap:Header>
  <soap:Body>
    ...
  </soap:Body>
</soap:Envelope>
```

##### • Header enthält Hinweis auf nicht verstandene Elemente



### 3.4 Fehlernachrichten (3)

#### ■ Fehlercodes

##### ◆ **VersionMismatch**: Elemente passen nicht zur SOAP-Version

##### ◆ **MustUnderstand**: obligatorisches Element wurde nicht verstanden

##### ◆ **DataEncodingUnknown**: unbekannte Codierung für Datenelemente

##### ◆ **Sender**: Absender hat Fehler verursacht

##### ◆ **Receiver**: Empfänger kann Nachricht nicht bearbeiten, Fehler liegt beim Empfänger

##### ◆ Verfeinerung der Fehlercodes durch untergeordnete Fehlercodes

##### • Element **Subcode** im **Code**-Element

##### ◆ HTTP-Fehlermeldung sind nicht erlaubt

##### • Ergebnis ist immer **200 OK**



## 3.5 Abbildung auf Trägerprotokoll

- HTTP
  - ◆ Nutzung der GET- und POST-Methoden von HTTP
    - GET nur bei rein lesenden Zugriffen (erst ab SOAP v1.2)
  - ◆ Adressat bestimmt durch URL
    - URL sollte unabhängig vom Inhalt der SOAP-Nachricht Adressaten vollständig bestimmen
  - ◆ spezieller Content-Type: **application/soap+xml**
    - bei SOAP v1.1 auch **text/xml**
  - ◆ Antwort in der Regel ebenfalls SOAP-Nachricht
    - einfache Realisierung von Request-Reply-Interaktionen
- E-Mail
  - ◆ Einbetten von SOAP-Nachrichten in E-Mails
  - ◆ Nutzung der gleichen Content-Types



## 3.6 Request-Reply-Nachrichten

- Aufrufnachricht (Request)
  - ◆ Aufruf ist eine Struktur
    - Name der aufgerufenen Methode ist Name des Elements
  - ◆ alle **in** und **inout** Parameter sind Komponenten der Struktur
- Antwortnachricht (Reply)
  - ◆ Antwort ist eine Struktur
    - Name der Methode plus angehängtes **Response** ist Name des Tags
  - ◆ alle **out**- und **inout**-Parameter sind Komponenten der Struktur
  - ◆ spezielles Tag für unbenanntes Ergebnis

```
<rpc:result xmlns:rpc=
    "http://www.w3.org/2003/05/soap-rpc">...
</rpc:result>
```

    - **rpc** ist spezieller Namensraum von SOAP für RPCs



## 3.6 Request-Reply-Nachrichten (2)

- Zuordnung von Antwort zu Aufruf
  - ◆ HTTP: natürliche Zuordnung
  - ◆ E-Mail
    - Message-ID des E-Mail-Systems wird bei Antwort angegeben (*In-Reply-To-Header*)
    - im Umschlag wird jeweils eine eindeutige Identifikation codiert
      - meist im Header
      - vgl. Seriennummer bei RPC



## 3.7 Sicherheit

- XML-Encryption-Standard
  - ◆ Verschlüsselung von
    - einzelnen XML-Tags und ihrer Subtags
    - Textelementen zwischen XML-Tags
    - alle Elemente zwischen XML-Tags
  - ◆ Beispiel

```
<Payment>
  <Name>Hans Müller</Name>
  <CreditCard Limit='5,000' Currency='USD'>
    <Number>4019 2445 0277 5567</Number>
    <Issuer>Treasure Bank</Issuer>
    <Expiration>04/02</Expiration>
  </CreditCard>
</Payment>
```



## 3.7 Sicherheit (2)

### ◆ Beispiel verschlüsselt

```
<Payment>
  <Name>Hans Müller</Name>
  <EncryptedData
    Type='http://www.w3.org/2001/04/xmlenc#Element'
    xmlns='http://www.w3.org/2001/04/xmlenc#' >
    <CipherData>
      <CipherValue>A23B45C56</CipherValue>
    </CipherData>
  </EncryptedData>
</Payment>
```



## 4 WSDL

### ■ Web-Services Description Language

- ◆ Schnittstellenbeschreibung für Web-Services
- ◆ Beschreibung von
  - Typen und Elementen (*Types and Elements*)
  - Nachrichten (*Messages*)
  - Operationen (*Operations*)
  - abstrakte Schnittstellen (*Port Types*)
  - Bindungen einer Schnittstelle an ein Nachrichtenprotokoll (*Binding*)  
(zur Zeit nur SOAP über HTTP bzw. E-Mail)
  - Schnittstelle einer konkreten Web-Service-Instanz (*Port*)



## 3.7 Sicherheit (3)

### ■ Digitale Signatur

- ◆ digitale Unterschrift einer SOAP-Nachricht
- ◆ Signatur als **Header**-Element
- ◆ Beispiel: (stark verkürzt)

```
<soap:Header>
  <wssec:Security ...>...
    <ds:Signature ...>
      <ds:SignedInfo>... </ds:SignedInfo>
      <ds:SignatureValue>Ace94sYXwjag...</...>
      <ds:Reference URI='#Body'>...</...>
    </ds:Signature>
  </wssec:Security>
</soap:Header>
<soap:Body>
  <a:withdraw ID='Body' ...> ... </a:withdraw>
</soap:Body>
```



## 4 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)



## 4.1 Überblick einer WSDL-Beschreibung

### ■ Komponenten von WSDL (Version 1.x)

```
<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>
```



## 4.2 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>
```



## 4.3 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



## 4.4 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>
```



## 4.5 Bindung an das Übertragungsprotokoll

- Beispiel: Bindung an HTTP

```
<wsdl:binding name='AccountSoapBinding'  
  type='AccountPort' >  
  <soap:binding style='rpc'  
    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>
```



## 4.6 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>
```



## 4.5 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**
    - Nachricht wird als XML-Dokument zusammengesetzt
  - ◆ Einfügen mit automatisch generiertem Element bei **rpc**
    - Nachricht wird als Request-Reply-Interaktion codiert



## 4.7 WSDL 2.0

- Schnittstellenbeschreibung
  - ◆ **interface**-Element
    - fasst **message**- und **portType**-Elemente zusammen
    - Nachrichtenaustausch-Muster pro **interface**-Element
      - Attribut **pattern**
  - ◆ Definition der Nachrichten pro Operation
    - Rückgriff auf die Typen aus den **types**-Elementen
  - ◆ Definition der Fehlermeldungen

