An Introduction to XML and Web Technologies

Web Services

Anders Møller & Michael I. Schwartzbach

Objectives

- SOAP – exchanging XML messages on a network
- WSDL – describing interfaces of Web services
- UDDI – managing registries of Web services

What is a Web Service?

- Web Service:
  “software that makes services available on a network using technologies such as XML and HTTP”
- Service-Oriented Architecture (SOA):
  “development of applications from distributed collections of smaller loosely coupled service providers”

Why a New Framework?

- CORBA, DCOM, Java/RMI, ... already exist
- XML+HTTP: platform neutral, widely accepted and utilized

What do We Need?

- We already know how to
  - represent information with XML
  - communicate with HTTP

- Fault tolerance
- Intermediaries
- RPC
- Interface descriptions
- Locating services
- ...

ad hoc solutions
vs.
use of standards?

A Recipe Server with XML and HTTP

- Ad hoc, RPC-style:
  - `Recipes getRecipes()`
  - `Lock lockRecipe(ID)`
  - `void writeRecipe(Lock,Recipe)`
  - `void unlockRecipe(Lock)`

Example Request (writeRecipe)

```
POST /personal/jdoe/recipeserver HTTP/1.0
Host: www.widget.inc
Content-Type: text/xml
Content-length: 5714

<?xml version="1.0"?>
<call xmlns="http://www.brics.dk/ixwt/xmlrpc"
      xmlns:rcp="http://www.brics.dk/ixwt/recipes">
  <operation>writeRecipe</operation>
  <arg>4DHX5ZV3D871AQ09</arg>
  <arg>
    <rcp:recipe id="r105">
      <rcp:title>Cailles en Sarcophages</rcp:title>
      <rcp:date>Tue, 26 Sep 06</rcp:date>
      ...
    </rcp:recipe>
  </arg>
</call>
```

Example Response (lockRecipe)

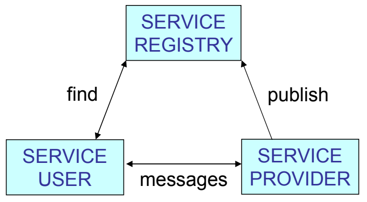
```
HTTP/1.1 200 OK
Date: Tue, 26 Sep 2006 22:29:08 GMT+1
Content-Type: text/xml
Content-Length: 101

<?xml version="1.0"?>
<return xmlns="http://www.brics.dk/ixwt/xmlrpc">
  4DHX5ZV3D871AQ09
</return>
```

XML-RPC

- A (too) simple RPC protocol based on XML and HTTP
- Close to the ad hoc approach in the Recipe Server...

Web Service Standards

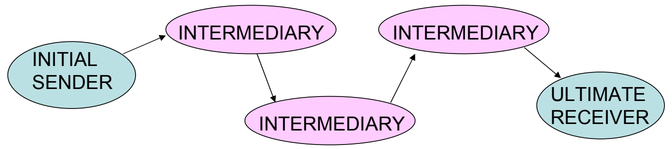
- SOAP
- WSDL
- UDDI
- WS-*
  - WS-Addressing
  - WS-ReliableMessaging
  - WS-Security, WS-Policy
  - WS-Resource
  - WS-Choreography (WS-CDL)
  - WS-BPEL (aka. BPEL4WS)
  - WS-Coordination, WS-AtomicTrans
  - ...



SOAP

- Used to be "**S**imple **O**bject **A**ccess **P**rotocol", but no longer an acronym...
- Processing Model
- Data Representation and RPC
- Binding to transport protocols (e.g. HTTP)

The SOAP Processing Model



SOAP Envelope:

```
<Envelope xmlns="http://www.w3.org/2003/05/soap-envelope">
  <Header>...</Header>
  <Body>...</Body>
</Envelope>
```

Envelope Headers

- Encryption information
- Access control
- Routing
- Auditing
- Data extensions
- ...

A SOAP Message

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
              xmlns:w="http://www.widget.inc/shop"
              xmlns:n="http://notaries.example.org">
  <env:Header>
    <w:ticket>54B42CF401A</w:ticket>
    <n:token>
      <n:value>32158546</n:value>
      <n:issuer>http://notarypublic.example.com</n:issuer>
    </n:token>
  </env:Header>
  <env:Body>
    <w:buy>
      <w:product>light gadget</w:product>
      <w:amount>430</w:amount>
    </w:buy>
  </env:Body>
</env:Envelope>
```

Special SOAP Header Attributes

- `role`
  - `next`
  - `ultimateReceiver`
  - `none`
- `mustUnderstand`
- `relay`
- `encodingStyle`

Another Example

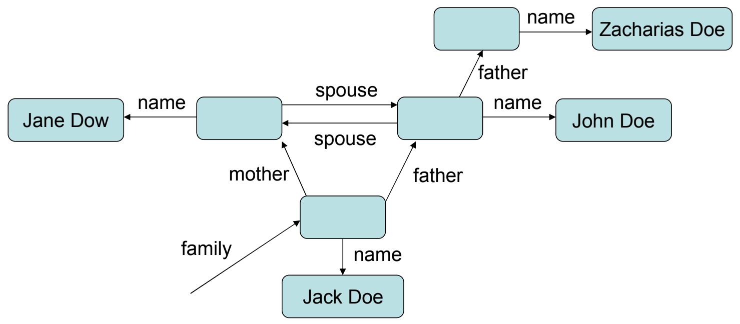
```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
              xmlns:c="http://encodings.example.org"
              xmlns:r="http://routings.example.org">
  <env:Header>
    <c:encoding env:role="http://encodings.example.org/decoder"
                env:mustUnderstand="true">
      gzip+base64
    </c:encoding>
    <r:route env:relay="true"
             env:role=
             "http://www.w3.org/2003/05/soap-envelope/role/next">
      <r:node>130.225.16.12</r:node>
      <r:node>10.11.40.201</r:node>
    </r:route>
  </env:Header>
  <env:Body>
    H4sICACI/0EAA3EA80jNycnXUSjPL8pJUeQCABinVXsOAAAA
  </env:Body>
</env:Envelope>
```

Faults

```
<env:Envelope xmlns:env="http://www.w3.org/2003/05/soap-envelope"
              xmlns:w="http://www.widget.inc/shop">
  <env:Body>
    <env:Fault>
      <env:Code>
        <env:Value>env:Sender</env:Value>
        <env:Subcode>
          <env:Value>w:InvalidBuyRequest</env:Value>
        </env:Subcode>
      </env:Code>
      <env:Reason>
        <env:Text xml:lang="en">
          The value of 'amount' is invalid!
        </env:Text>
        <env:Text xml:lang="da">
          Værdien af 'amount' er ugyldig!
        </env:Text>
      </env:Reason>
    </env:Fault>
  </env:Body>
</env:Envelope>
```

SOAP Encoding

XML serialization of data graphs



SOAP Encoding, cont.

```
<family xmlns:env="http://www.w3.org/2003/05/soap-envelope"
        xmlns:enc="http://www.w3.org/2003/05/soap-encoding"
        env:encodingStyle=
               "http://www.w3.org/2003/05/soap-encoding"
        xmlns="http://www.widget.inc/encoding">
  <name>Jack Doe</name>
  <father enc:id="1">
    <name>John Doe</name>
    <father>
      <name>Zacharias Doe</name>
    </father>
    <spouse enc:ref="2"/>
  </father>
  <mother enc:id="2">
    <name>Jane Dow</name>
    <spouse enc:ref="1"/>
  </mother>
</family>
```

RPC in SOAP

```
<env:Envelope
       xmlns:env="http://www.w3.org/2003/05/soap-envelope"
       xmlns:rs="http://www.brics.dk/ixwt/recipeserver"
       xmlns:rcp="http://www.brics.dk/ixwt/recipes">
  <env:Body>
    <rs:writeRecipe env:encodingStyle=
             "http://www.w3.org/2003/05/soap-encoding">
      <rs:lock>4DHX5ZV3D871AQ09</rs:lock>
      <rs:recipe env:encodingStyle=
             "http://xml.apache.org/xml-soap/literalxml">
        <rcp:recipe id="r105">
          <rcp:title>Cailles en Sarcophages</rcp:title>
          <rcp:date>Tue, 26 Sep 06</rcp:date>
          ...
        </rcp:recipe>
      </rs:recipe>
    </rs:writeRecipe>
  </env:Body>
</env:Envelope>
```

Flexibility of SOAP

- SOAP can be used without using **SOAP Encoding**
- SOAP can be used with other conventions for RPC than **SOAP RPC**
- SOAP can be used with other communication patterns than **SOAP RPC**

Protocol Binding

- Transmission protocols: HTTP, SMTP, ...
- Route from initial sender to ultimate receiver may involve different protocols

- RPC fits nicely into HTTP request–response

HTTP Binding in SOAP

- Message exchange patterns:
  - request–response (for RPC) ⇒ POST
  - SOAP response ⇒ GET

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset="utf-8"
Content-Length: 273

<env:Envelope
    xmlns:env="http://www.w3.org/2003/05/soap-envelope"
    xmlns:rs="http://www.brics.dk/ixwt/recipeserver">
  <env:Body>
    <rs:writeRecipeResponse env:encodingStyle=
           "http://www.w3.org/2003/05/soap-encoding"/>
  </env:Body>
</env:Envelope>
```

Summary of SOAP

- A transport neutral protocol for XML data interchange (but focusing on HTTP)

- Processing model (envelopes, intermediaries, ...)
- SOAP Encoding
- SOAP RPC
- Protocol Bindings

- Foundation of WS-*

WSDL

- **W**eb **S**ervices **D**escription **L**anguage

- Functionality? (operations, types of arguments)
- Access? (data encoding, communication protocols)
- Location?

– Necessary information for writing clients
– Automatic generation of stubs and skeletons

Structure of a WSDL Description

```
<description xmlns="http://www.w3.org/2004/08/wsdl"
             targetNamespace="..." ...>
  <types>
    <!-- XML Schema description of types being used
         in messages -->
    ...
  </types>
  <interface name="...">
    <!-- list of operations and their input and output -->
    ...
  </interface>
  <binding name="..." interface="..." type="...">
    <!-- message encodings and communication protocols -->
    ...
  </binding>
  <service name="..." interface="...">
    <!-- combination of an interface, a binding,
         and a service location -->
    ...
  </service>
</description>
```

Recipe Server with WSDL and SOAP (1/6)

```
<description xmlns="http://www.w3.org/2004/08/wsdl"
             targetNamespace="http://www.brics.dk/ixwt/recipes/wsdl"
             xmlns:x="http://www.brics.dk/ixwt/recipes/wsdl">
  <types>
    <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
               targetNamespace=
                       "http://www.brics.dk/ixwt/recipes/wsdl/types"
               xmlns:t="http://www.brics.dk/ixwt/recipes/wsdl/types">

      <xs:import namespace="http://www.brics.dk/ixwt/recipes"
                 schemaLocation="recipes.xsd"/>

      <xs:element name="lock">
        <xs:simpleType>
          <xs:restriction base="xs:string">
            <xs:length value="16"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:element>
```

Recipe Server with WSDL and SOAP (2/6)

```
      <xs:element name="lockError" type="xs:string"/>

      <xs:element name="getRecipes">
        <xs:complexType><xs:sequence/></xs:complexType>
      </xs:element>

      <xs:element name="lockRecipe">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="id" type="xs:NMTOKEN"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="lockRecipeResponse">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="t:lock"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
```

Recipe Server with WSDL and SOAP (3/6)

```
      <xs:element name="writeRecipe">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="t:recipe"/>
            <xs:element ref="t:lock"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

      <xs:element name="unlockRecipe">
        <xs:complexType>
          <xs:sequence>
            <xs:element ref="t:lock"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>

    </xs:schema>
  </types>
```

Recipe Server with WSDL and SOAP (4/6)

```
  <interface name="recipeserverInterface"
             xmlns:t="http://www.brics.dk/ixwt/recipes/wsdl/types"
             styleDefault="http://www.w3.org/2004/03/wsdl/style/rpc">

    <fault name="lockFault" element="t:lockError"/>

    <operation name="getRecipesOperation"
               pattern="http://www.w3.org/2004/03/wsdl/in-out">
      <input messageLabel="In" element="t:getRecipes"/>
      <output messageLabel="Out" element="t:collection"/>
    </operation>

    <operation name="lockRecipeOperation"
               pattern="http://www.w3.org/2004/03/wsdl/in-out">
      <input messageLabel="In" element="t:lockRecipe"/>
      <output messageLabel="Out"
              element="t:lockRecipeResponse"/>
      <outfault ref="x:lockFault" messageLabel="Out"/>
    </operation>
```

Recipe Server with WSDL and SOAP (5/6)

```
    <operation name="writeRecipeOperation"
               pattern=
                    "http://www.w3.org/2004/03/wsdl/robust-in-only">
      <input messageLabel="In" element="t:writeRecipe"/>
      <outfault ref="x:lockFault"/>
    </operation>

    <operation name="unlockRecipeOperation"
               pattern="http://www.w3.org/2004/03/wsdl/in-only">
      <input messageLabel="In" element="t:lock"/>
    </operation>

  </interface>
```

Recipe Server with WSDL and SOAP (6/6)

```
  <binding name="recipeserverSOAPBinding"
           interface="x:recipeserverInterface"
           type="http://www.w3.org/2004/08/wsdl/soap12"
           xmlns:ws="http://www.w3.org/2004/08/wsdl/soap12"
           ws:protocol="http://www.w3.org/2003/05/soap/bindings/HTTP"
           ws:mepDefault=
                "http://www.w3.org/2003/05/soap/mep/request-response"
           xmlns:soap="http://www.w3.org/2003/05/soap-envelope">
    <fault ref="x:lockFault" ws:code="soap:Sender"/>
  </binding>

  <service name="recipeserver"
           interface="x:recipeserverInterface">
    <endpoint name="recipeserverEndpoint"
              binding="x:recipeserverSOAPBinding"
              address=
                 "http://www.widget.inc/personal/jdoe/recipeserver"/>
  </service>
</description>
```

Interface Descriptions

- In-Only
- Robust In-Only
- In-Out
- In-Optional-Out
- Out-Only
- Robust Out-Only
- Out-In
- Out-Optional-In

```
<operation name="getRecipesOperation"
           pattern=
   "http://www.w3.org/2004/03/wsdl/in-out">
  <input messageLabel="In"
         element="t:getRecipes"/>
  <output messageLabel="Out"
          element="t:collection"/>
</operation>
```

RPC-Style Operations

- Can be used with In-Out, In-Only, Robust In-Only
- Input/output types must describe element sequences only
- Wrapper element in request names the operation
- ...

Binding Descriptions

- **Encodings** and **protocols** for an interface
- Predefined:
  - SOAP binding (often using SOAP’s HTTP binding)
  - HTTP binding (“raw HTTP”)

SOAP Binding

```
<binding name="recipeserverSOAPBinding"
         interface="x:recipeserverInterface"
         type="http://www.w3.org/2004/08/wsdl/soap12"
         xmlns:ws="http://www.w3.org/2004/08/wsdl/soap12"
         ws:protocol=
               "http://www.w3.org/2003/05/soap/bindings/HTTP"
         xmlns:soap="http://www.w3.org/2003/05/soap-envelope">

  <operation ref="x:getRecipesOperation"
             ws:mep=
       "http://www.w3.org/2003/05/soap/mep/request-response"/>

  ...

  <fault ref="x:lockFault" ws:code="soap:Sender"/>

</binding>
```

HTTP Binding

```
<binding name="recipeserverHTTPBinding"
         interface="x:recipeserverInterface"
         type="http://www.w3.org/2004/08/wsdl/http"
         xmlns:wh="http://www.w3.org/2004/08/wsdl/http">

  ...

  <operation ref="x:search"
             wh:method="GET"
             wh:location="search-engine/find/{q}"/>

  <fault ref="x:ServiceUnavailable"
         wh:code="503"/>
</binding>
```

Service Descriptions

```
<service name="recipeserver"
         interface="x:recipeserverInterface">
  <endpoint name="recipeserverEndpoint"
            binding="x:recipeserverSOAPBinding"
            address=
               "http://www.widget.inc/personal/jdoe/recipeserver"/>
</service>
```

Summary of WSDL

Description of **interfaces** of Web services:

- message types
- operations
- encodings and communication protocols
- location

UDDI

- **U**niversal **D**escription, **D**iscovery, and **I**ntegration
- static / dynamic discovery
- public / private registries

UDDI Descriptions

- `publisherAssertion`
  (describes relations between businesses)
- `businessEntity`
  (describes a concrete business)
- `businessService`
  (describes a Web service)
- `bindingTemplate`
  (describes invocation information)
- `tModel`
  (technical details, e.g. reference to WSDL description)

Business Entity for Recipe Server (1/2)

```xml
<businessEntity xmlns="urn:uddi-org:api_v3"
    businessKey="uddi:7398388-7F63-73K3-H314-763272DA7G41">
  <name>Widget Inc.</name>
  <contacts>
    <contact useType="Chief Executive Officer">
      <description>CEO of Widget Inc.</description>
      <personName>John Doe</personName>
     <phone useType="CEO">(202) 555-1414</phone>
     <email useType="CEO">john.doe@widget.inc</email>
    </contact>
  </contacts>

  <businessServices>
    <businessService
        serviceKey="uddi:9X65542-8JE7-8732-U893-8272634H7362"
        businessKey="uddi:7398388-7F63-73K3-H314-763272DA7G41">
      <name>Doe Personal Recipe Server</name>
      <description>
        John Doe's personal recipe service
      </description>
```

Business Entity for Recipe Server (2/2)

```xml
      <bindingTemplates>
        <bindingTemplate
          bindingKey="uddi:8H62363-K725-3345-73V5-823763FS7265"
          serviceKey="uddi:9X65542-8JE7-8732-U893-8272634H7362">
          <accessPoint URLType="http">
            http://www.widget.inc/personal/jdoe/recipeserver
          </accessPoint>
          <tModelInstanceDetails>
            <tModelInstanceInfo tModelKey=
                  "uddi:5241HY7-6252-KN72-7291-3126HJ8237A2"/>
          </tModelInstanceDetails>
        </bindingTemplate>
      </bindingTemplates>
    </businessService>
  </businessServices>
</businessEntity>
```

tModel for Recipe Server

```xml
<tModel xmlns="urn:uddi-org:api_v3"
        tModelKey="uddi:5241HY7-6252-KN72-7291-3126HJ8237A2">
  <name>Doe Personal Recipe Server</name>
  <description>John Doe's personal recipe service</description>
  <overviewDoc>
    <overviewURL>
      http://www.widget.inc/personal/jdoe/recipes.wsdl
    </overviewURL>
  </overviewDoc>
  <categoryBag>
  <keyedReference
     keyName="uddi-org:types"
     keyValue="wsdlSpec"
     tModelKey="uddi:C1ACF26D-9672-4404-9D70-39B756E62AB4"/>
  <keyedReference
     keyName="IAAWG"
     keyValue="WDG18762"
     tModelKey="uddi:82761UHS-442P-1712-KL82-8272HSH76519"/>
  </categoryBag>
</tModel>
```

UDDI Discovery

```
<find_service xmlns="urn:uddi-org:api_v3">
  <categoryBag>
    <keyedReference
        keyName="IAAWG"
        keyValue="%"
        tModelKey="uddi:82761UHS-442P-1712-KL82-8272HSH76519"/>
  <
</f
```

```
<serviceList xmlns="urn:uddi-org:api_v3">
  <serviceInfos>
    <serviceInfo
       businessKey="uddi:7398388-7F63-73K3-H314-763272DA7G41"
       serviceKey="uddi:9X65542-8JE7-8732-U893-8272634H7362">
      <name>Doe Personal Recipe Server</name>
    </serviceInfo>
    <serviceInfo
       businessKey="uddi:82736H57-HA32-P581-0021-8373H6S73443"
       serviceKey="uddi:72520X72-K23J-4X44-7W23-K82737292527">
      <name>Average Recipes on The Web</name>
    </serviceInfo>
  </serviceInfos>
</serviceList>
```

Summary

- **SOAP** – a transport neutral protocol for XML data interchange (but focusing on HTTP)
- **WSDL** – description of Web service interfaces
- **UDDI** – registries and discovery of Web services

Essential Online Resources

- SOAP:
  http://www.w3.org/TR/soap/
- WSDL:
  http://www.w3.org/2002/ws/desc/
- UDDI:
  http://www.uddi.org
- XML-RPC:
  http://www.xmlrpc.com/
- The Web Service Interoperability Organization:
  http://www.ws-i.org/