Web Services

1

How Do Web Services Relate to Course?

- Relatively novel XML-based distributed/network computing paradigm
- Supported by all major computing companies (Microsoft, IBM, Sun, Oracle, HP, …)
- Technical concepts are similar to other paradigms already discussed in the course
- The main difference is in “business issues”

2

Service-Based Software Systems

- "Software is a service" business model
  - Software is not a product any more, but a service available over the network
  - Application Service Providers (ASPs)
- Service-Oriented Architecture (SOA)
  - Applications decomposed into distributed services
  - On-line access to libraries of composable components

3

W3C Definition of a Web Service

- A software application that:
  - has a unique Uniform Resource Identifier (URI),
  - can be defined, described, and discovered using XML (Extensible Markup Language),
  - supports exchange of XML messages via Internet-based protocols.

4

Microsoft .NET Definition of a Web Service

- A black-box programmable application logic that can be reused without worrying about how the service is implemented
- Defined strictly in terms of the messages the Web Service accepts and generates
- Simply and easily composed over the network, across domain boundaries
- Accessible using standard and ubiquitous Internet protocols and data formats (XML)

5

Some Goals of the Work on Web Services

- Faster development and deployment of IT (Information Technology) systems
- Dynamic and ad hoc A2A (Application-to-Application) and B2B (Business-to-Business) integration
- Interoperability across programming languages and operation platforms
- Leveraging and compatibility with existing Internet/Web infrastructure

6

Web Services vs. Distributed Objects

- Ad hoc & temporary vs. planned & stable integration
- “Loose coupling” vs. “tight coupling”
- Synchronous or asynchronous vs. synchronous communication
- RPC (Remote Procedure Call) or document-based vs. RPC interactions
- B2B or inter-enterprise vs. inter-enterprise integration

7

Who Is Working in This Area?

- All major computing companies
  - Microsoft .NET, IBM Dynamic e-business, Sun ONE (Open Net Environment), Oracle, HP, …
- Several industrial “standardization” bodies
  - W3C (World Wide Web Consortium)
  - OASIS (Organization for the Advancement of Structured Information Standards)
  - WS-I (Web Services - Interoperability)
  - …

8

Basic Technologies

- XML (Extensible Markup Language) - For data representations
- SOAP [a.k.a. Simple Object Access Protocol, XML Protocol] - An XML packaging protocol, defines message formats and encoding rules
- WSDL (Web Services Description Language) - An XML service description language

9

Some Popular Technologies

- UDDI (Universal Description, Discovery, and Integration) - For discovering service providers if you do not know their URLs
- WS-Inspection - For discovering services located on a particular Web site (you need to know its URL)
- BPEL4WS (Business Process Execution Language for Web Services) - For describing Web Service compositions (orchestrations)

10

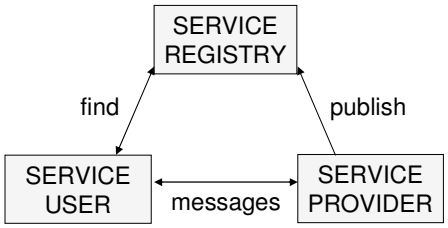
And Many Other Technologies …

- WS-Policy - For specifying policies (e.g., security)
- WSLA (Web Service Level Agreements) - A way to to specify quality of service (QoS) guarantees and service level agreements
- Many WS-??? and WS?L technologies …

11

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-AtomicTransaction, WS-CAF
  - ...

UNDER DEVELOPMENT!

12

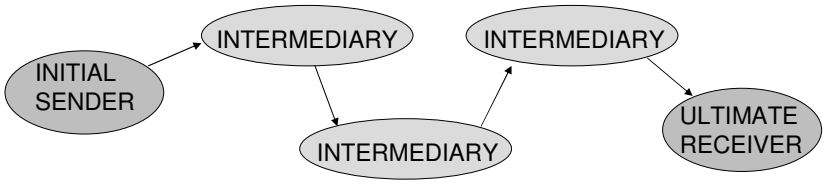
SOAP

- 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)

13

The SOAP Processing Model



SOAP Envelope:

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

14

Structure of a SOAP Message

SOAP Envelope

SOAP Header (optional)
- various management information

SOAP Body (mandatory)
- the transferred information in XML

15

Envelope Headers

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

16

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>
```

17

Special SOAP Header Attributes

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

18

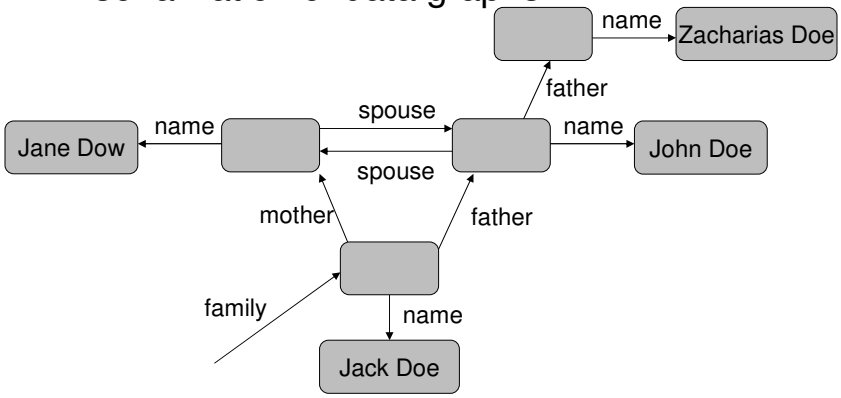
Another Example

```
<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>
```

19

SOAP Encoding

XML serialization of data graphs



20

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>
```

21

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>
```

22

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**

23

Protocol Binding

- Transmission protocols: HTTP, SMTP, ...
- Route from initial sender to ultimate receiver may involve different protocols
- RPC fits nicely into HTTP request–response

24

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>
```

25

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-*

26

WSDL: Web Service Description Language

- Standard for describing Web services
  - Abstract interface for defining operations and their messages
    - Messages contain either document-oriented or procedure-oriented information
  - Bindings to message formats and protocols
  - Defines how to locate the endpoint for the service
    - Example: URLs for HTTP
  - Extensible (SOAP and HTTP extensions are defined)
  - Written in XML, leverages XML schema
- WSDL V1.1 Specification
  - http://www.w3.org/TR/wsdl

27

Usage Scenarios

- As IDL (Interface Definition Language)
  - Allows tools to generate client access code for a service
  - Examples: IBM WebSphere Studio Application Developer, IBM Web Services Toolkit
- Standardized service interface descriptions
  - Allows advertisement and dynamic discovery of services
  - Enables dynamic binding to service
  - Complements UDDI registry

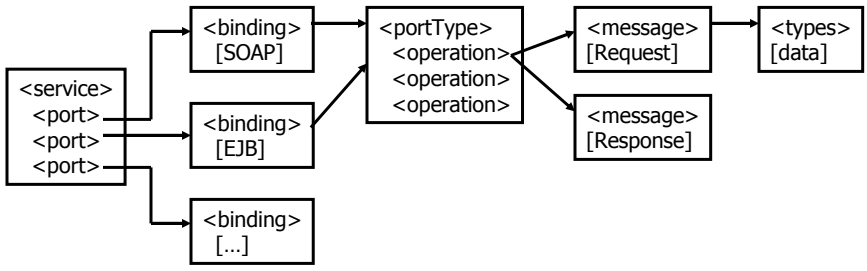
28

Document Content

- Abstract Definitions
  - <types> data type definitions
  - <message> operation parameters
  - <portType> operation definitions
- Concrete Definitions
  - <binding> operation bindings
  - <service> location/address for each binding
- Also:
  - <import>used to reference other XML documents

29

Document Structure



Service(s) Supported Protocol(s) Java class method method method SOAP Request/ response

30

SOAP Binding - RPC Style

- Indicates that the Web service is accessed using SOAP V1.1 protocol
- Use style="rpc" attribute on SOAP binding element
- Example SOAP service method signature:
  - public float getQuote (String symbol)

SOAP	Java	Description
Operation	getQuote	Method name
Input Message	String symbol	Input parameter
Output Message	float	Return type

31

Example: Stock Quote Service [1]

```
<definitions name="StockQuoteService"
   targetNamespace="http://tempuri.org/StockQuoteService"
   xmlns:tns="http://tempuri.org/StockQuoteService" ...>

 <message name="SymbolRequest">
   <part name="symbol" type="xsd:string" />
 </message>
 <message name="QuoteResponse">
   <part name="quote" type="xsd:float" />
 </message>
 <portType name="StockQuoteService">
   <operation name="getQuote">
     <input message="tns:SymbolRequest" />
     <output message="tns:QuoteResponse" />
   </operation>
 </portType>
 ...
```

```
public float getQuote (String symbol)
```

```
public float getQuote (String symbol)
```

```
public float getQuote (String symbol)
```

32

Example: Stock Quote Service [2]

```
...
<binding name="SoapBinding" type="tns:StockQuoteService">
  <soap:binding style="rpc"
     transport="http://schemas.xmlsoap.org/soap/http" />
  <operation name="getQuote">
    <soap:operation soapAction="http://tempuri.org/GetQuote" />
    <input>
      <soap:body use="encoded"
        namespace="http://tempuri.org/live-stock-quotes"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"/>
    </input>
    <output>
      <soap:body use="encoded" namespace="..." encodingStyle="..."/>
    </output>
  </operation>
</binding>
...
```

33

Example: Stock Quote Service [3]

```
...
<service name="StockQuoteService">
  <documentation>Stock Quote Service</documentation> -
  <port name="Demo"  binding="tns:SoapBinding">
    <soap:address
      location="http://tempuri.org/services/StockQuoteService" />
  </port>
</service>
</definitions>
```

34

WSDL4J

- WSDL Java API
  - WSDL object model
  - Parse contents of a WSDL document
  - Programmatically create new WSDL documents
- Open source project sponsored by IBM and HP
  - http://sourceforge.net/projects/wsdl4j/
- Is the reference implementation for JSR 110
  - Primarily a set of Java interfaces that can be implemented by anyone
  - Java package name: javax.wsdl

35

WSDL4J Example - Find Port

```
// Get WSDLReader
WSDLReader wsdlReader = WSDLFactory.newInstance().newWSDLReader();

// Read WSDL service implementation document
Definition wsdlDefinition = wsdlReader.readWSDL(null, wsdlURL);

// Get the service elements
Map services = definition.getServices();

// Get an iterator for the list of services
Iterator serviceIterator = services.values().iterator();

boolean bPortFound = false;
while ((serviceIterator.hasNext()) && !(bPortFound)) {
  // Get next service element
  Service service = (Service) serviceIterator.next();

  // Determine if this service element contains the specified port
  if ((port = service.getPort(portName)) != null)
     bPortFound = true;
}
```

36

Summary of WSDL

Description of **interfaces** of Web services:

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

37

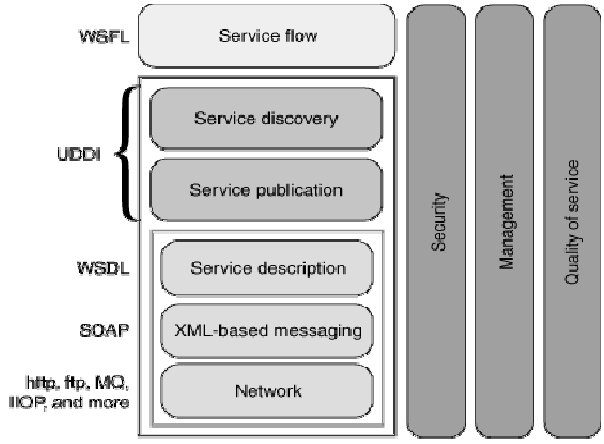
UDDI

- **U**niversal **D**escription, **D**iscovery, and **I**ntegration

- static / dynamic discovery
- public / private registries

38

Web Services Protocol Stack



39

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)

40

UDDI Operations

Inquiry Operations:
Find
- find_business
- find_service
- find_binding
- find_tModel

Get details
- get_businessDetail
- get_serviceDetail
- get_bindingDetail
- get_tModelDetail
- get_registeredInfo

Publishing Operations:
Save
- save_business
- save_service
- save_binding
- save_tModel

Delete
- delete_business
- delete_service
- delete_binding
- delete_tModel

Security
- get_authToken
- discard_authToken

41

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

42

Semantic Web Services: the ultimate vision for the WWW

The Vision

- 500 million users
- more than 3 billion pages

Static

WWW
URI, HTML, HTTP

The Vision

Serious Problems in

- information finding,
- information extracting,
- information representing,
- information interpreting and
- and information maintaining.

Static

WWW
URI, HTML, HTTP

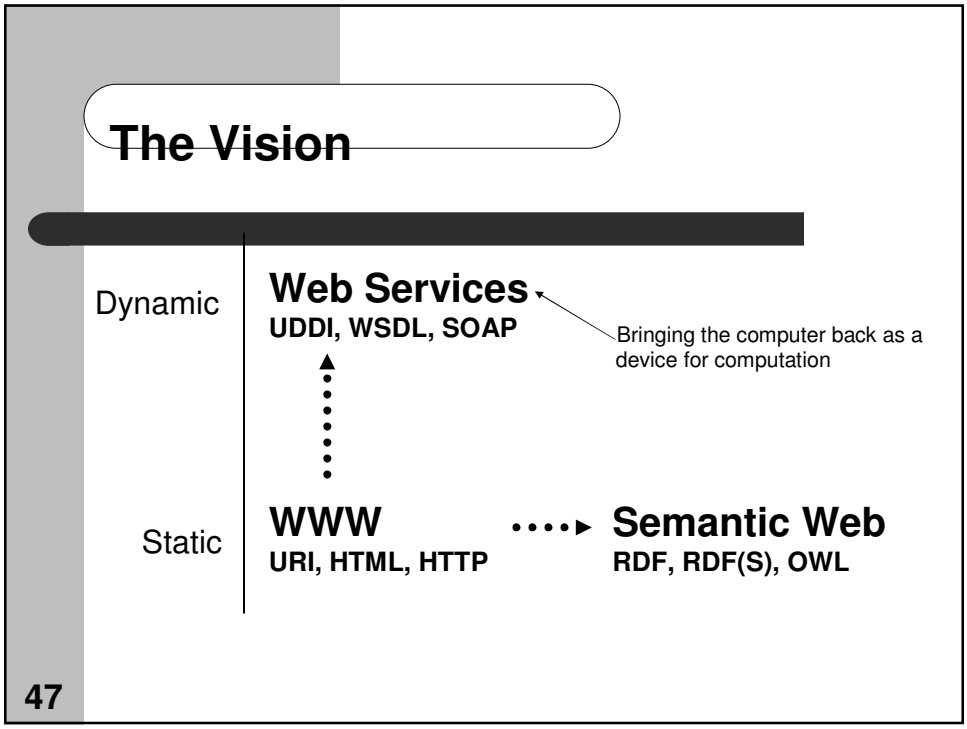
Semantic Web
RDF, RDF(S), OWL

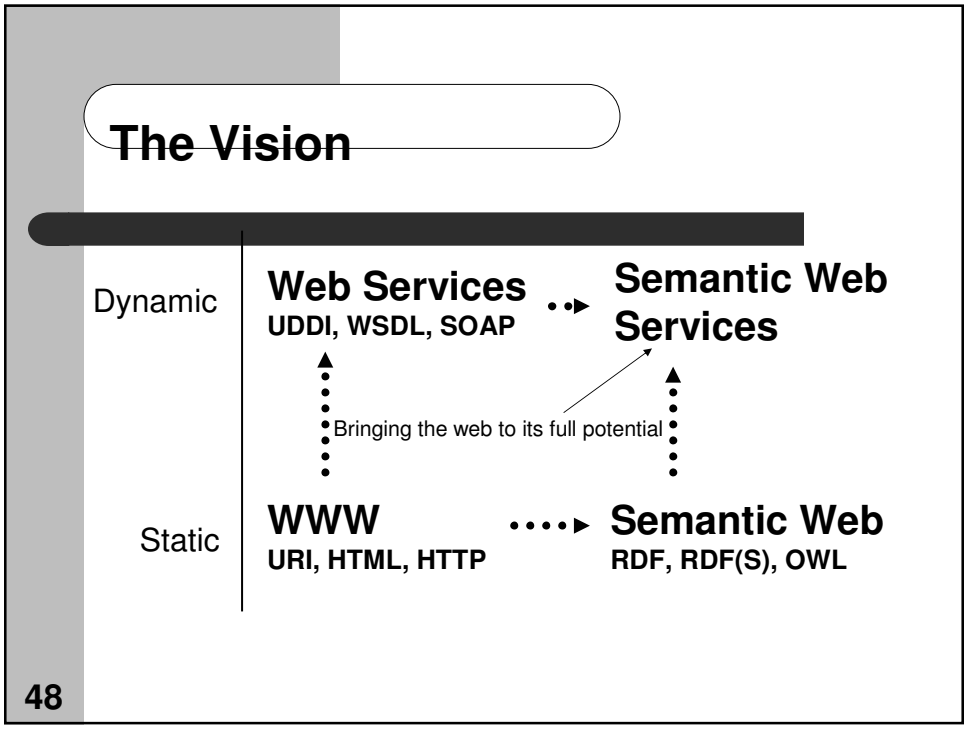
45

Why RDF/OWL, why not simply XML and related technologies

- XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- XML Schema is a language for restricting the structure of XML documents and also extends XML with datatypes.
- RDF is a datamodel for objects ("resources") and relations between them, provides a simple semantics for this datamodel, and these datamodels can be represented in an XML syntax.
- RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes.
- OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjointness), cardinality (e.g. "exactly one"), equality, richer typing of properties, characteristics of properties (e.g. symmetry), and enumerated classes.

46


The Vision
Dynamic
Web Services
UDDI, WSDL, SOAP
Bringing the computer back as a device for computation
Static
WWW
URI, HTML, HTTP
Semantic Web
RDF, RDF(S), OWL
47

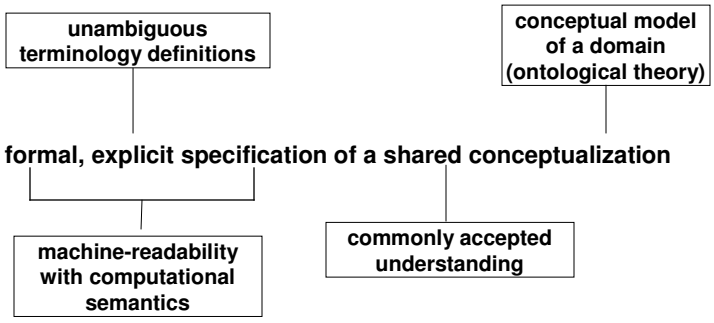

The Vision
Dynamic
Web Services
UDDI, WSDL, SOAP
Semantic Web Services
Bringing the web to its full potential
Static
WWW
URI, HTML, HTTP
Semantic Web
RDF, RDF(S), OWL
48

The Semantic Web

- the next generation of the WWW
- information has machine-processable and machine-understandable semantics
- not a separate Web but an augmentation of the current one
- Ontologies as basic building block

49

Ontology Definition



50

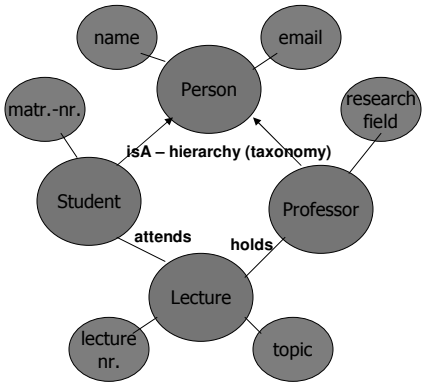
Ontology Example

Concept
conceptual entity of the domain

Property
attribute describing a concept

Relation
relationship between concepts or properties

Axiom
coherency description between Concepts / Properties / Relations via logical expressions



holds(Professor, Lecture) =>
Lecture.topic = Professor.researchField

51

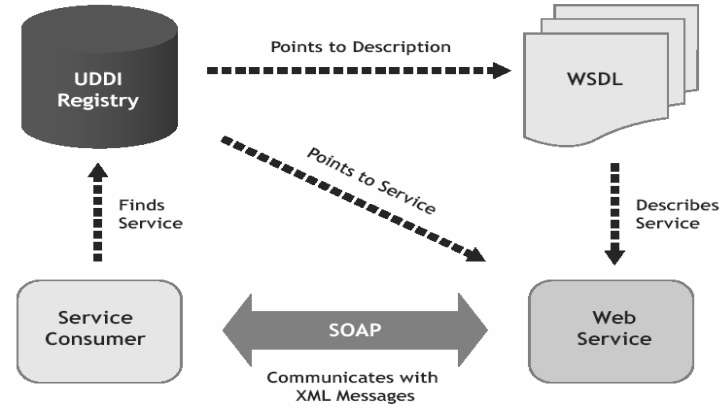
Web Services

- loosely coupled, reusable components
- encapsulate discrete functionality
- distributed
- programmatically accessible over standard internet protocols
- add new level of functionality on top of the current web

52

The Promise of Web Services

web-based SOA as new system design paradigm



53

Deficiencies of WS Technology

- current technologies allow usage of Web Services
- but:
  - only syntactical information descriptions
  - syntactic support for discovery, composition and execution
  - => Web Service usability, usage, and integration needs to be inspected manually
  - no semantically marked up content / services
  - no support for the Semantic Web

=> current Web Service Technology Stack failed to realize the promise of Web Services

54

Semantic Web Services

Semantic Web Technology

- allow machine supported data interpretation
- ontologies as data model

+

Web Service Technology

automated discovery, selection, composition,
and web-based execution of services

=> Semantic Web Services as integrated solution for realizing the vision of the next generation of the Web

55