

Server-Side Web Development

JSP

Lecture #8 2007

1 Web Services

Service Oriented Architectures

Loose Coupling

WSDL

SOAP

Related Technologies

Web Services

Today

Web Services

Service Oriented
Architectures
Loose Coupling
WSDL
SOAP
Related
Technologies

Next Time

- *Service*: A software component accessed *over a network* that provides functionality to a service requester
- *Web Service*: A service which publishes a service interface in WSDL and uses a message-driven protocol (usually via SOAP / HTTP)
- Built on a host of XML-based technologies
 - XML (data exchanged)
 - XML Schema (validation of data exchanged)
 - SOAP (XML-serialized transfer protocol)
 - WSDL (Web Service interface description, XML Schema)
- Uses a *deployment descriptor* to configure service (XML-based configuration file for the service container)

Service Oriented Architectures (SOA)

Today

Web Services

**Service Oriented
Architectures**

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

- A style of building distributed systems where functionality is provided by modular services
- Focuses on *loose coupling* between interacting services (i.e., minimizing formal knowledge between components)
- Services are *virtualized* as much as possible (i.e., focus is placed on interfaces, not implementations)
- Usually built on Web Services (today)

SOA Characteristics

Today

Web Services

**Service Oriented
Architectures**

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

- Logical view - No implementation details are revealed
- Coarse-grained - few operations, large messages
- Platform- (and language-) neutral
- Wide-spread technology base (XML, HTTP, TCP/IP)

SOA Service Characteristics

Today

Web Services

**Service Oriented
Architectures**

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

- Message-oriented - communicate by exchanging messages
 - abstract - interface defined in terms of messages
 - encapsulated - implementation details hidden
 - technology independent (platform, OS, API etc)
- Self-describing: provides machine-readable metadata (advertises capabilities, service interface, protocols etc)
- Discoverable: dynamic "on-demand" service discovery (includes service location, service interface, protocols etc)

SOA Service Characteristics

Today

Web Services

**Service Oriented
Architectures**

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

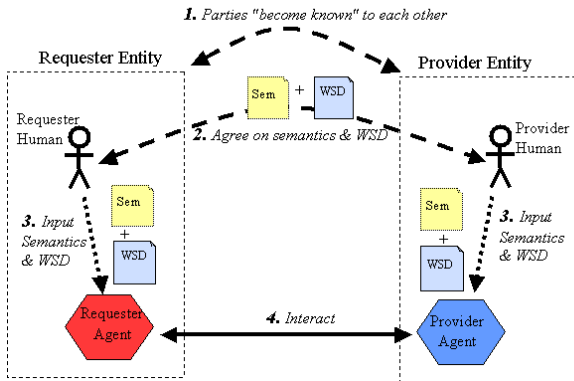
- **Modular:** solves one well-defined task
 - used individually (by different services / applications)
 - can be composed (by other services)
 - facilitates reusability
 - self-contained or dependent on other services / resources
- **Interoperable:** standardized service access
 - standardized protocols
 - standardized data formats

Interactions

Today

Web Services
Service Oriented
Architectures
Loose Coupling
WSDL
SOAP
Related
Technologies

Next Time



Loose Coupling

- Components minimize built-in knowledge of each other (focus placed on interfaces, not implementations)
- Services are dynamically discovered when needed (includes interfaces, supported protocols, location etc)
- Ideal: zero-coupling ("frictionless") (services used without providing any information)

Benefits Of Loose Coupling

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

- **Flexibility:** services can be (re)located on any server
- **Scalability:** services can be added / removed on demand (load balancing)
- **Replacability:** service implementations can be replaced (without user disruptions)
- **Fault tolerance:** upon failures, clients can query registries for alternative services offering the same functionality

Publish, Find, Bind

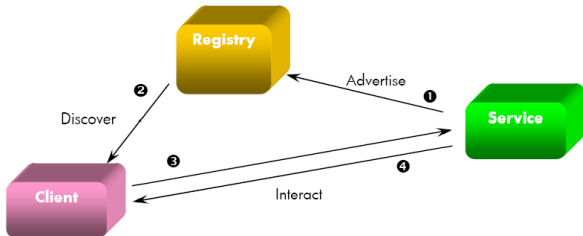
- 1 Advertisement: service publishes information in a registry
- 2 Discovery: client queries registry for services
- 3 Connection establishment: client contacts service
- 4 Interaction: client and service interact

Publish, Find, Bind

Today

Web Services
Service Oriented
Architectures
Loose Coupling
WSDL
SOAP
Related
Technologies

Next Time



WSDL

- XML Schema-based language for describing Web Services
- Completely describes the Web Service interface
- Constitutes a "contract" between the client and the service
- Can be generated from code, or vice versa
- Two major parts
 - abstract: interface (types, operations and messages)
 - physical: deployment (encodings, protocols, bindings)

Developing Web Services

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

- Two main approaches
 - generate WSDL from code
 - generate code (stubs) from WSDL
- Generated WSDL tend to be platform / tool-dependent (quick and easy, but incompatibility issues may arise)
- Generating stubs from WSDL ensures compatibility (but require more work from all parties involved)
- **GOAL: interoperability** (favor the WSDL approach)

Calling a Web Service

- 1 Locate Web Service (discovery)
- 2 Obtain WSDL description
- 3 Generate stubs from WSDL description
- 4 Use stubs to invoke Web Service methods

```
<definitions name="CounterService"
  targetNamespace="http://course.example/Counter"
  xmlns:counter="http://course.example/Counter"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>
    ...
  </types>

  <message>
    ...
  </message>

  <portType>
    <operation> ... </operation>
  </portType>

</definitions>
```


WSDL Types

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

```
<types>
  <schema targetNamespace="http://course.example/Counter"
    xmlns="http://www.w3.org/2001/XMLSchema">
    <element name="IncrementRequest">
      <complexType>
        <sequence>
          <element name="Value" type="int"
            minOccurs="1" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
    <element name="IncrementResponse">
      <complexType/>
    </element>

    <element name="GetValueRequest">
      <complexType/>
    </element>
    <element name="GetValueResponse">
      <complexType>
        <sequence>
          <element name="Value" type="int"/>
        </sequence>
      </complexType>
    </element>
  </schema>
</types>
```

WSDL Messages

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

```
<!-- Message definitions for Increment -->
<message name="IncrementRequestMessage">
  <part name="parameter" element="counter:IncrementRequest"/>
</message>
<message name="IncrementResponseMessage">
  <part name="parameter" element="counter:IncrementResponse"/>
</message>

<!-- Message definitions for GetValue -->
<message name="GetValueRequestMessage">
  <part name="parameter" element="counter:GetValueRequest"/>
</message>
<message name="GetValueResponseMessage">
  <part name="parameter" element="counter:GetValueResponse"/>
</message>
```

WSDL portTypes

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

```
<portType name="Counter">  
  <operation name="Increment">  
    <input message="counter:IncrementRequestMessage"/>  
    <output message="counter:IncrementResponseMessage"/>  
  </operation>  
  
  <operation name="GetValue">  
    <input message="counter:GetValueRequestMessage"/>  
    <output message="counter:GetValueResponseMessage"/>  
  </operation>  
</portType>
```

SOAP

- Formerly known as *Simple Object Access Protocol*
- XML-based protocol to invoke Web Services
(XML-serializes web service requests / responses)
- Usually transported via HTTP (in HTTP body)
- Can send messages
 - point-to-point (directly)
 - via intermediaries (in chains of actors)

SOAP Messages

- Outer layer (e.g., HTTP data)
- Envelope (message root element)
- Header (optional)
 - factorization
 - different recipients (actors)
- Body
 - application specific data (message payload)
 - XML elements
 - Faults (error messages)

SOAP Message

```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <soap:Body>
    <w:Greeting xmlns:w="http://www.wrox.com/helloworld/">
      <w:message>Hello world!</w:message>
    </w:Greeting>
  </soap:Body>
</soap:Envelope>
```

SOAP Faults

- Faults reported in SOAP message body
- Error messages
- Comparable to exceptions in Java
- Fault information
 - faultcode: error identifier
 - faultstring: human readable identifier
 - faultactor: origin of error
 - detail: additional fault information

SOAP Fault

Today

Web Services

Service Oriented
Architectures

Loose Coupling

WSDL

SOAP

Related
Technologies

Next Time

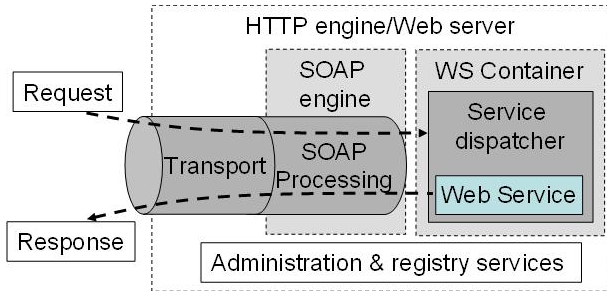
```
<soap:Envelope
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <soap:Fault>
      <faultcode>soap:Server</faultcode>
      <faultstring>Insufficient funds</faultstring>
      <detail>
        <t:TransferError xmlns:t="http://course.example/transaction">
          <sourceAccount>accountX</sourceAccount>
          <transferAmount>1000.00</transferAmount>
          <currentBalance>910.50</currentBalance>
        </t:TransferError>
      </detail>
    </soap:Fault>
  </soap:Body>
</soap:Envelope>
```


SOAP Processing

Today

Web Services
Service Oriented
Architectures
Loose Coupling
WSDL
SOAP
Related
Technologies

Next Time



Representational State Transfer (REST)

- Alternative to SOAP for invoking Web Services
- Calls conveyed directly in HTTP bodies
- No extra encoding layers
- Simpler than SOAP
- Less versatile than SOAP

Asynchronous JavaScript and XML (AJAX)

Today

Web Services
Service Oriented
Architectures
Loose Coupling
WSDL
SOAP
Related
Technologies

Next Time

- Group of techniques used to increase interactivity in web applications
- Decreases response times by performing background HTTP and Web Service requests
- Usually some form of XML-based remote procedure calls done in JavaScript
- Alleviates the response time burden in web applications
- Dynamically updated pages not available in bookmarks, browser histories & search engines

Web Service Resource Framework (WSRF)

- Framework to enable development of stateful Web Services
- Focuses on representations of state: *resources*
- Contains a whole host of specifications
- Provides
 - resource discovery
 - resource addressing
 - resource lifetime management
 - notification (publish / subscribe based state updates)
 - renewable references
 - service groups
 - base fault representations

Summary

- Web Services are
 - accessible over networks
 - technology and platform-independent
 - hosted in service containers (e.g., Apache Axis)
 - accessed through generated stubs or APIs
 - not very efficient
 - very versatile
- Service Oriented Architectures draw up guidelines for (large-scale) deployment of Web Services

Next Time

- Best practices