

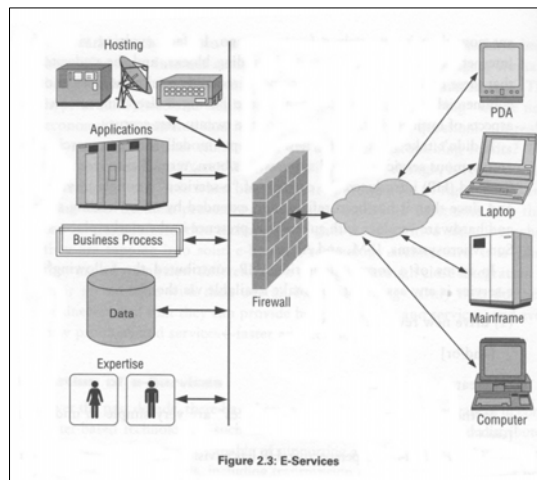
Web Services Overview

MSIS 531 – Spring 2006

Providing e-Services

2

- Business services provided through firewall
- Usual "W's": who, what, when, where, why
- Need differentiation along many dimensions: customer, product, etc.
- Important: data must flow transparently through corporate gateways, firewalls



Business Rules

- Business rules: logical rules behind business decisions
 - Consistent, high-quality, accessible
 - Guide collective organizational behavior
 - Rules may drive automated or manual processes
- Benefits:
 - Faster automation, easier change management
 - Connects business to its “reasoning base”
 - Repository of corporate memory/intelligence
- Implementing business rules provides a nice fit w/web services

Business Rules->Web Services

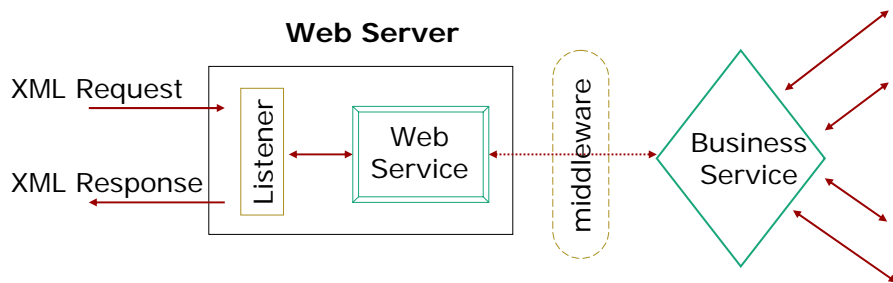
- So how do we make our business rules visible?
 - One popular method: web services
- Platform-neutral technologies to allow delivery of network services via intranet or internet
- Standards-based, service oriented architecture
 - Supported by all the major hardware/software players
- Goal is interoperability: across programming languages, operating systems, application servers, databases, etc.

Web Services: Key Components

- Six key components of a web service:
 - self-contained ←
 - self-describing ←
 - modular applications ←
 - published ←
 - located ←
 - invoked across the web ←
- object-oriented
WSDL
UDDI
SOAP or XML-RPC
- An obvious fit for Java & HTTP, but equally for .NET and HTTP
 - Note use of HTTP resolves many firewall issues

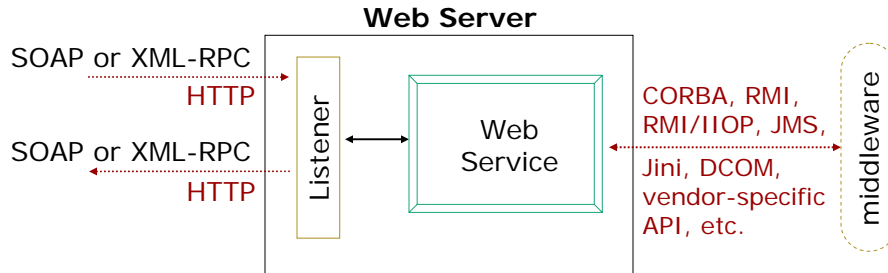
Web Services Architecture

- A web service is essentially a standards-based wrapper for accessing non-standardized middleware components



Architectural Details

7



- SOAP and XML-RPC have become accepted standards for XML-based messaging. - the “wire format”
- HTTP is the defacto protocol for the Internet
- The Web Service parses the request and either fulfills it directly, or invokes one or more business services via a middleware API

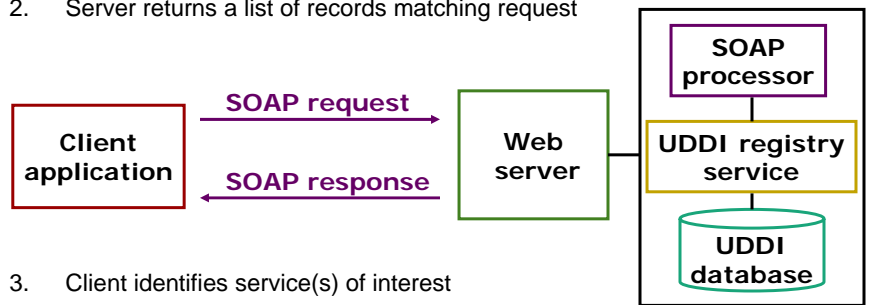
© UW Business School, University of Washington 2006

UW BUSINESS SCHOOL

Invoking a Web Service

8

1. Client queries a UDDI registry node
2. Server returns a list of records matching request



3. Client identifies service(s) of interest
4. Server binds client to the service so client can invoke service methods

UDDI registry node

© UW Business School, University of Washington 2006

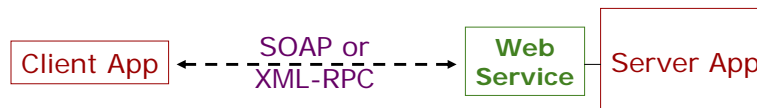
UW BUSINESS SCHOOL

Web Services in a Nutshell

- Loosely Coupled
 - Agreement on two things:
 - wire format (HTML, XML, WML, etc.)
 - protocol (HTTP, SMTP, TCP/IP, etc.)
 - All other Implementation details are hidden
- Coarse-Grained
 - Networks maximize efficiency by reducing round-trips
 - Expose interfaces that may hide other services
- Asynchronous
 - Service-oriented requests cannot always be immediately performed
 - Ability to keep track of a “conversation”

Implementing Web Services

- Web Services *enable* new kinds of applications, but they *are separate from* these applications
 - Still necessary to use a programming language to design the client, server pieces of the communication
 - Java, C/C++, ASP, Perl, Ruby, etc.
 - Web services serve as the glue between the client and server applications



SOAP

11

- Simple Object Access Protocol
 - Protocol (format) for executing code remotely
 - Expresses request as XML
 - SOAP message is transmitted over HTTP

```
<?xml version="1.0" encoding="UTF-8" ?>
- <soapenv:Envelope xmlns:soapenv="http://
  xmlns:xsd="http://www.w3.org/2001/x
- <soapenv:Body>
  - <listResponse soapenv:encodingStyle="h
    - <listReturn xsi:type="soapenc:Array"
      xmlns:soapenc="http://schemas.x
        <item>accept:image/gif, image/x
          application/vnd.ms-excel, appl
        <item>referer:http://infosys.badr
        <item>accept-language:en-us</ite
        <item>accept-encoding:gzip, defle
        <item>user-agent:Mozilla/4.0 (co
        <item>host:infosys.badm.washing
        <item>connection:Keep-Alive</ite
      </listReturn>
    </listResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

WSDL

12

- Web Services Description Language (WSDL)
 - Draft specification released March 2001 by Ariba, IBM, and Microsoft
 - XML-formatted language that defines ways a web service can be contacted and describes the message formats used
- WSDL is the language that defines web services
 - Neutral declaration of the existence of one or more services
 - Description of the way a client can interact with those services

WSDL Elements

- A WSDL service definition applies several layers of abstraction
- There are six layers:
 - Data types that the service operates on
 - Message types the service can receive/send
 - How these messages combine into operations
 - What protocols the service will bind to
 - WSDL ports (operation + protocol)
 - Which services bind on which ports

UDDI

- Universal Description, Discovery, and Integration
 - Initial spec released Sept. 2000 by Ariba, IBM, Microsoft
 - partnership among industry leaders – almost 100 companies have signed up to support UDDI.org
- UDDI functions:
 - Service Provider
 - provides and publishes e-business web services
 - Service Registry
 - a SOAP-activated web service yellow pages
 - Service Requester
 - finds required services and binds client to those services

UDDI Elements

- Core UDDI elements:
 - Business Services
 - a collection of web services that constitute a logical service grouping (“purchasing”, “tech support”)
 - two essential attributes: businessKey & serviceKey
 - Binding Templates
 - describes location of web service and how to access
 - provides information on protocol and access to interface definition (WSDL) or other API
 - Tmodels
 - descriptive information about data type
 - type of business, type of service, interface type used

J2EE APIs for Web Services

- **JAX-RPC** (Java API for XML-based RPC)
 - Allows remote procedures to be executed
 - Client side API communicates with web service
 - Server endpoints (JAX-RPC or EJB) receive/process request
- **SAAJ** (SOAP with Attachments API for Java)
 - Build, read SOAP messages, metadata (SOAP headers)
- **JAXR** (Java API for XML Registries)
 - Access UDDI registries, search for web service endpoints
- **JAXP** (Java API for XML Processing)
 - Read, write, modify XML documents
 - DOM (Document Object Model), SAX (Simple API for XML)

JAXWS-2.0 – Sun’s Latest and Greatest

- Attempt by Sun to make the creation of web services as easy as in .NET
 - We’ll see what you think
- We’ll try an example, building a web service, a unit test, and then test accessing everyone else’s service
- I’ll give you a URL with a link to the process, but I’ll fill in some of the steps
 - Hint: everything works in online examples except what doesn’t...
- Side note: JDK 1.5 feature (annotations) used to simplify code generation process