CMER

Centre for Mobile Education and Research

Introduction to Web Services

Week I

Overview

- Introduction
- Definitions
- XML
- WS Architecture
- Styles
- Profiles
- Specifications
- Similar Efforts

- Legacy Systems
- Contract Development
- Platforms
- Advantages
- Disadvantages

Introduction

- **Web Services - “a software system designed to support interoperable Machine to Machine interaction over a network” – W3C**
- **Public methods or APIs that can be accessed over a private network or a public network such as the Internet.**
- **Can enable applications to publish their functionality to select groups or even the entire world.**

Definitions

- **"Web Services provide a simplified mechanism to connect applications regardless of the technology or devices they use, or their location. They are based on industry standard protocols with universal vendor support that can leverage the internet for low cost communications, as well as other transport mechanisms. The loosely coupled messaging approach supports multiple connectivity and information sharing scenarios via services that are self describing and can be automatically discovered." - http://roadmap.cbdiforum.com/reports/roi/**

More Definitions

- "Web services are a new way of connecting businesses. Web services are platform-neutral and vendor-independent protocols that enable any form of distributed processing to be performed using XML and Web-based technologies." - www.dmreview.com/rg/resources/glossary.cfm

- "Web services are self-contained business functions that operate over the internet." - http://www.studiodog.com/glossary-w.html

XML

- Web service technologies are based on XML
- XML facilitates the sharing of structured data across different information systems (i.e Internet)

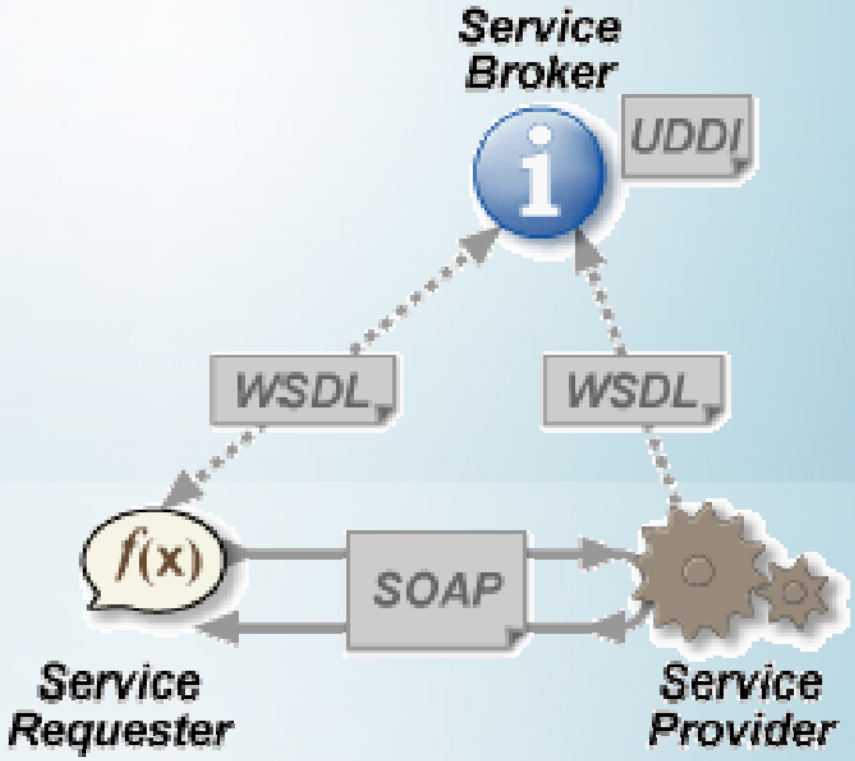
XML Advantages

- Text-based
- Human and machine readable
- Strict syntax and parsing make it reliable International standards
- Validation
- Hierarchical structure is suitable for most types of data
- Platform independant

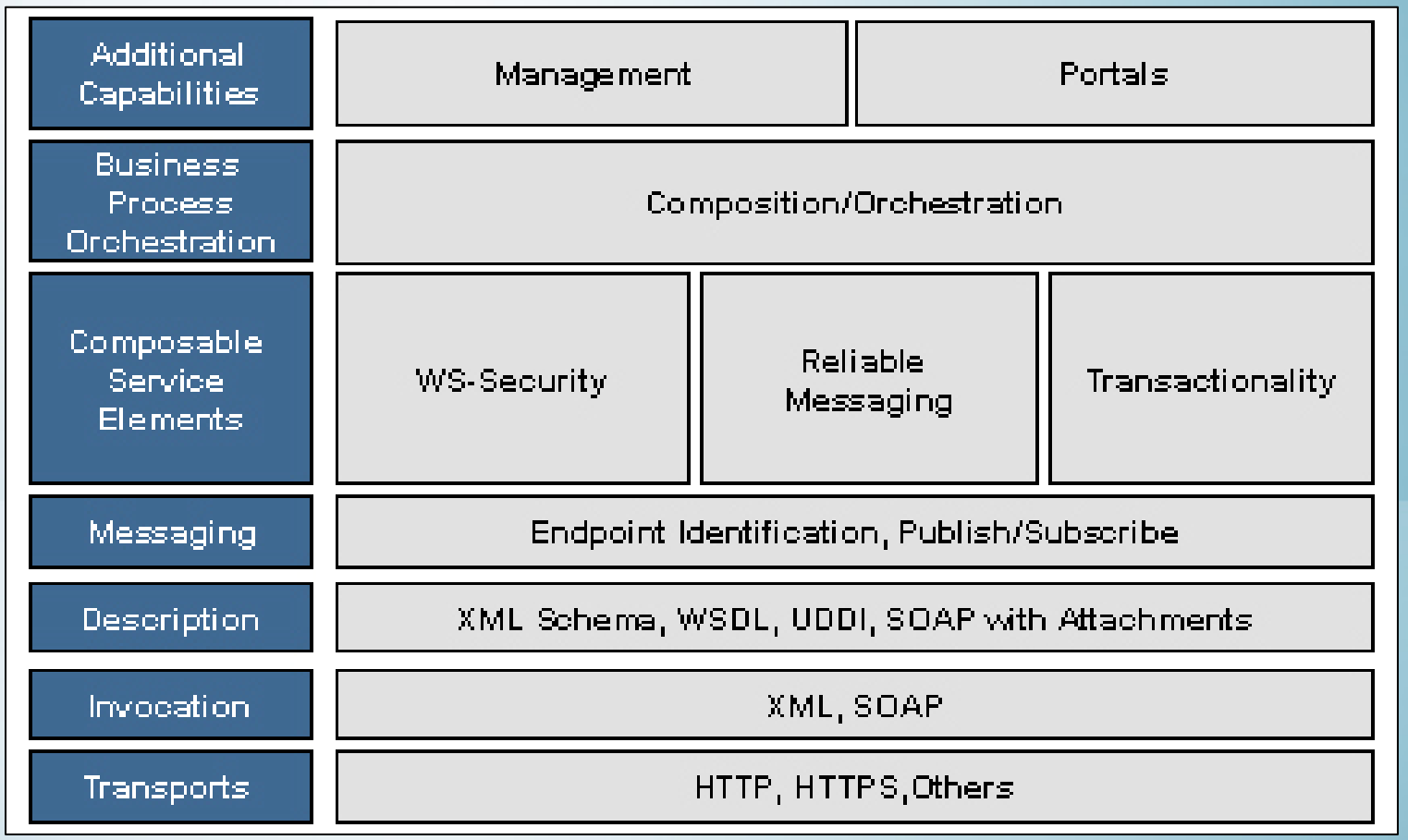
XML Disadvantages

- Redundant with large amounts of data.
- Data redundancy storage, transmission and processing costs.
- Syntax is verbose
- XML namespaces are problematic and/or tricky to use.
- The distinction between content and attributes in XML seems unnatural to some and makes designing XML data structures harder.

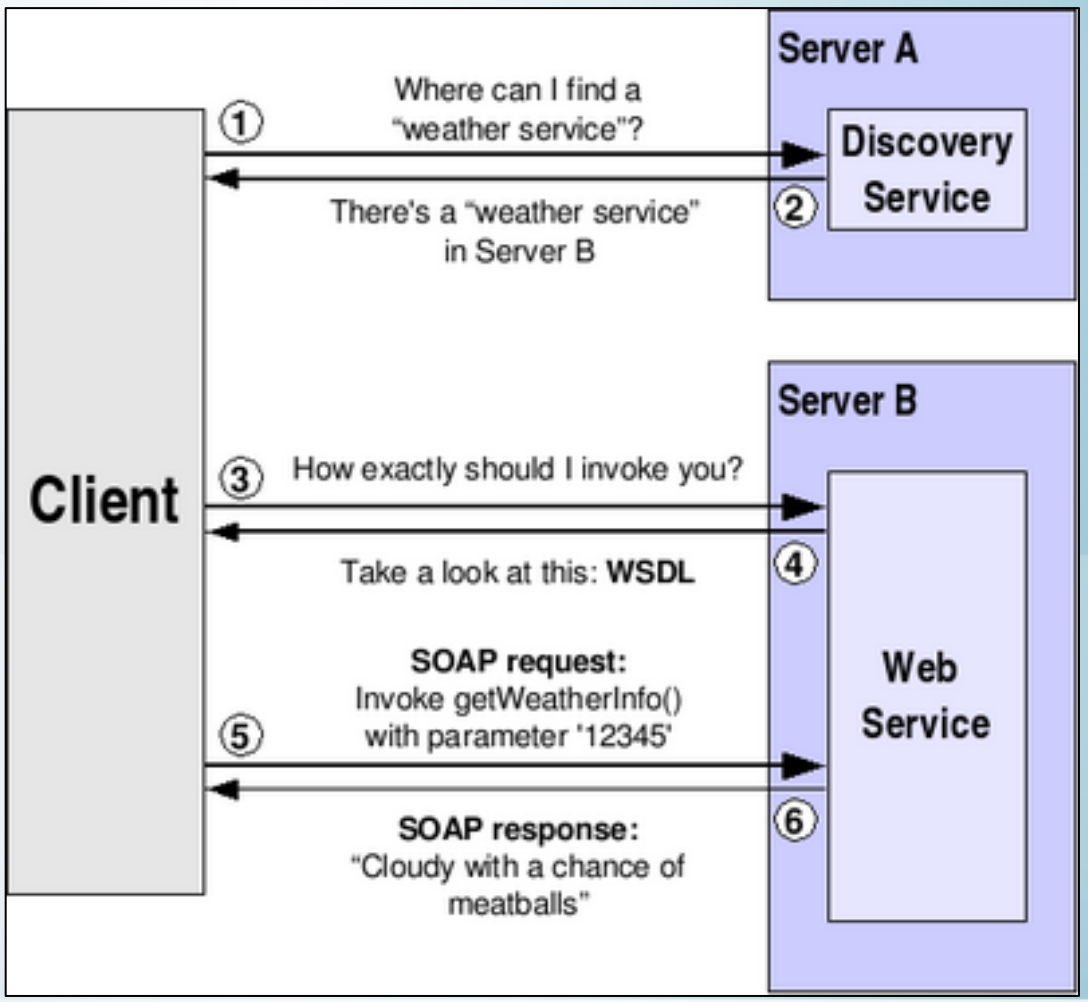
Architecture



Web Services Stack



Invocation


Server A
Where can I find a "weather service"?
1
Discovery Service
There's a "weather service" in Server B
2
Server B
Client
3
How exactly should I invoke you?
Take a look at this: WSDL
4
SOAP request:
Invoke getWeatherInfo() with parameter '12345'
5
Web Service
SOAP response:
"Cloudy with a chance of meatballs"
6

UDDI

- Universal Description, Discovery & Integration
- XML-based registry
- Stores information about businesses and Web services
- Enables people/businesses to:
  - Publish their service information
  - Define how their services should interact
  - Discover other services

UDDI (Cont.)

- Meant to be interrogated by SOAP messages
- Provides access to WSDL documents

UDDI (Cont.)

Consists of 3 components:

- White pages
  - Address, contact and known identifiers of business
- Yellow pages
  - Industrial categorization based on standard taxonomies
- Green pages
  - Technical information about the services

WSDL

- Web Service Description Language
- Pronounced “wiz-del”
- XML-based language for describing Web service interfaces
  - Functions that service provides
  - Parameters the function requires
  - Results the function returns
- Specification is divided into 6 major elements…

WSDL Elements

- "definitions" element
  - Root element of WSDL document
  - Defines name of Web service
  - Declares namespaces used in the document
  - Contains all other elements

WSDL Elements (Cont.)

- "types" element
  - Describes all data types to be used between the client and server
    - Does not include XML Schema built-in simple types (i.e string, integers)
  - If WSDL type is not specified then defaults to XML Schema type

WSDL Elements (Cont.)

- **“message” element**
  - **Describes a one-way message (request or response)**
    - **Message name**
    - **Message parameters**
    - **Message return values**

WSDL Elements (Cont.)

- **“portType” element**
  - **Combines multiple message elements to form a complete “round trip” operation**
    - **i.e Can combine 1 request message and 1 response message into a single request/response operation**
  - **Can define multiple operation**

WSDL Elements (Cont.)

- "binding" element
  - Specifics of how service will be implemented on the wire
- "service" element
  - Specified address for invoking service
  - Most often a URL

SOAP

- XML-based messaging protocol
- Can work over HTTP
  - Communicates through proxies and firewalls
- Once stood for “Simple Object Access Protocol” but acronym has since been dropped as it was considered to be misleading

SOAP Elements

- **“envelope” element**
  - **specifies that the XML document is a SOAP message; encloses the message itself.**
- **“header” element (optional)**
  - **contains information relevant to the message, e.g., the date the message was sent, authentication data, etc.**

SOAP Elements (Cont.)

- **“body” element**
  - **includes the message payload.**
- **“fault” element (optional)**
  - **carries information about a client or server error within a SOAP message.**

SOAP Request

```
POST /EndorsementSearch HTTP/1.1
Host: www.snowboard-info.com
Content-Type: text/xml; charset="utf-8"
Content-Length: 261
SOAPAction: "http://www.snowboard-info.com/EndorsementSearch"
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-
     ENV:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
  <SOAP-ENV:Body>
    <m:GetEndorsingBoarder xmlns:m="http://namespaces.snowboard-
      info.com">
      <manufacturer>K2</manufacturer>
      <model>Fatbob</model>
    </m:GetEndorsingBoarder>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

SOAP Response

```
<SOAP-ENV:Envelope
  xmlns:SOAP-
    ENV="http://schemas.xmlsoap.org/soap/envelope/"
  SOAP-
    ENV:encodingStyle="http://schemas.xmlsoap.org/soap/
    encoding/">
  <SOAP-ENV:Body>
    <m:GetEndorsingBoarderResponse
    xmlns:m="http://namespaces.snowboard-info.com">
      <endorsingBoarder>Chris
    Englesmann</endorsingBoarder>
    </m:GetEndorsingBoarderResponse>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Styles

1. Remote Procedure Call (RPC)
2. Service-Oriented Architecture (SOA)
3. Representational State Transfer (REST)

RPC

- Has familiar function/method call interface
- Interacts through WSDL
- First Web services focused on RPC
- Most widely used style
- Criticized for not being loosely coupled
  - **Requires mapping XML to other languages**

SOA

- Web services built on SOA concepts
- Basic unit of communication is the message (not operation)
  - **Message oriented services**
- Supported by most major software vendors
- Loose coupling is more prominent than in RPC

REST

- Emulates the HTTP protocol by restricting interface operations
  - GET, PUT, DELETE
- Interaction with stateful resources rather than messages or operations
- WSDL 2.0 supports binding for all HTTP require methods

Profiles

- A set of specifications with versions:
  - **i.e SOAP 1.1, WSDL 1.1 UDDI 2.0, XML Schema 1.0 (Basic profile)**
- Published by the WS-I
- Improves the interoperability of Web services

Specifications

- Extend Web service capabilities
- Generally referred to as WS-*
  - WS-Security
  - WS-Reliability
  - WS-ReliableMessaging
  - WS-Addressing
  - WS-Transaction
  - Etc.(Many more)

Previous Efforts

1. RMI
2. CORBA
3. DCOM

RPC

- Remote Procedure Call
- Sun Microsystems developed the Open Network Computing RPC (1987) – communication mechanism for NFS
- A remote function could be invoked as if it were a local one
- Open Software Foundation' DCE (Distributed Computing Environment) 1989
- Microsoft RPC initiative based on DCE/RPC (1990)

Java RMI

- Remote Method Invocation
- A mechanism that is part of the Java programming language
- Allows Java objects to invoke methods on objects from another JVM
- Object equivalent of RPC
- Core package of Java 1.1+

CORBA

- Common Object Request Broker Architecture
- A standard defined by the Object Management Group (OMG, 1991)
- Enables software components written in multiple computer languages and/or running on multiple computers to work together

DCOM

- Distributed Component Object Model
- Proprietary Microsoft technology
- Extends Microsoft's COM
- Deprecated in favor of Microsoft .NET

Legacy Systems

- Web services can port legacy systems to communicate with the rest of the business
- Could be expensive but so is building a new system
- It’s a workaround, but it works.

Contract First vs. Contract Last

- Relates to how and when the WSDL file is generated
- The client relies on the WSDL to understand the Web service interface
- The client is dependant on the Web service not to change

Contract First

- The advised approach
- Generate the WSDL file before you create the Web service
- Most likely you will already know the method definitions before implementing the services
- Prevents the WSDL from changing if the Web service changes

Contract Last

- Not advised
- But most convenient
- Create the Web services first
- Use a utility that automatically generates the WSDL for you
- If you make changes to the service then it may change the WSDL
  - **Unexpected change negatively affect the client**

Popular Supported Platforms

- C/C++
- .Net
- Java
- PHP
- Python
- Ruby (on Rails)

Some Related Efforts

- BPEL
- ebXML
- DPWS
- WSDM

BPEL

- Business Process Execution Language
- OASIS Standard
- A language for specifying business-to-business process behavior based on Web Services
- Builds on Web services standards

ebXML

- Electronic Business using eXtensible Markup Language (e-business XML)
- Pronounced (ee-bee-ex-em-el)
- A family of XML based standards for:
  - Promoting an open, XML-based infrastructure
  - Enabling the global use of electronic business information
  - Ensuring interoperability, security, and consistency for all trading partners

DPWS

- Devices Profile for Web Services
- Defines a minimal set of implementation constraints to enable secure Web service:
  - Messaging
  - Discovery
  - Description
  - Eventing on resource constrained devices
- A Web service comparison to Universal Plug and Play (UPnP)

WSDM

- Web Services Distributed Management
- Pronounced (*wisdom*)
- A standard for managing and monitoring the status of Web services
- A Web service comparison to Simple Network Management Protocol (SNMP)

Advantages

- **Interoperable**
  - **Vendor, platform, and language independent XML technologies**
  - **Ubiquitous HTTP for transport**
  - **Client only needs WSDL**
  - **Service implementation does not matter to client**

Advantages (Cont.)

- **Code Re-use**
  - **"write once, use anywhere" programming**
  - **Reduces development time**
- **Cost Savings**
  - **Reduces development time**
  - **Reduces operating costs**

Advantages (Cont.)

- **Versatile**
  - **Can be accessed by humans via web-based interfaces**
  - **Can be accessed by autonomous applications or other We services**
  - **Web services can be combined to create new services (Service Composition)**
  - **The implementation behind the service does not change the service itself**

Disadvantages

- **Overhead**
  - **Transmitting large amounts of data in XML could be costly**
  - **Portability vs. efficiency**
  - **Not yet suitable for critical real-time applications**

Disadvantages (Cont.)

- **Quality of Service**
  - **Reliability of connection**
  - **Availability of service**
  - **Performance of the service**
- **Reliability**
  - **HTTP is not a reliable protocol in that it doesn't guarantee delivery or a response.**
  - **Avoid changes to service interface that customers won’t expect**

References

- http://gdp.globus.org/gt4-tutorial/multiplehtml/ch01s02.html
- http://en.wikipedia.org/wiki/SOAP
- http://sharat.wordpress.com/2007/04/17/what-are-disadvantageschallenges-in-web-services/