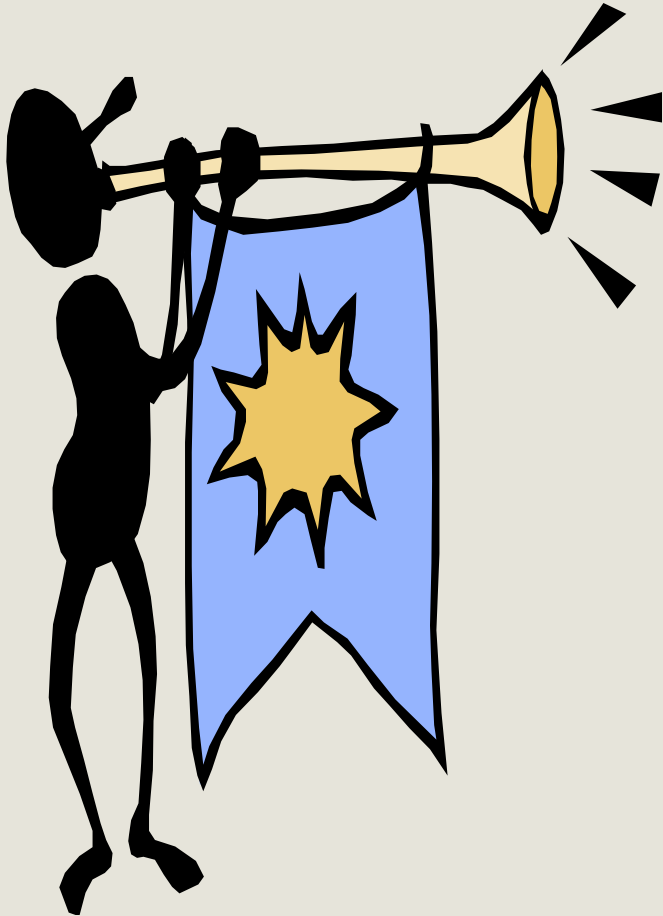


Web Services Based Architectures |

INSE 7110 – Winter 2004

**Value Added Services Engineering in Next Generation Networks
Week #9**

Outline



- Basics
- Technologies

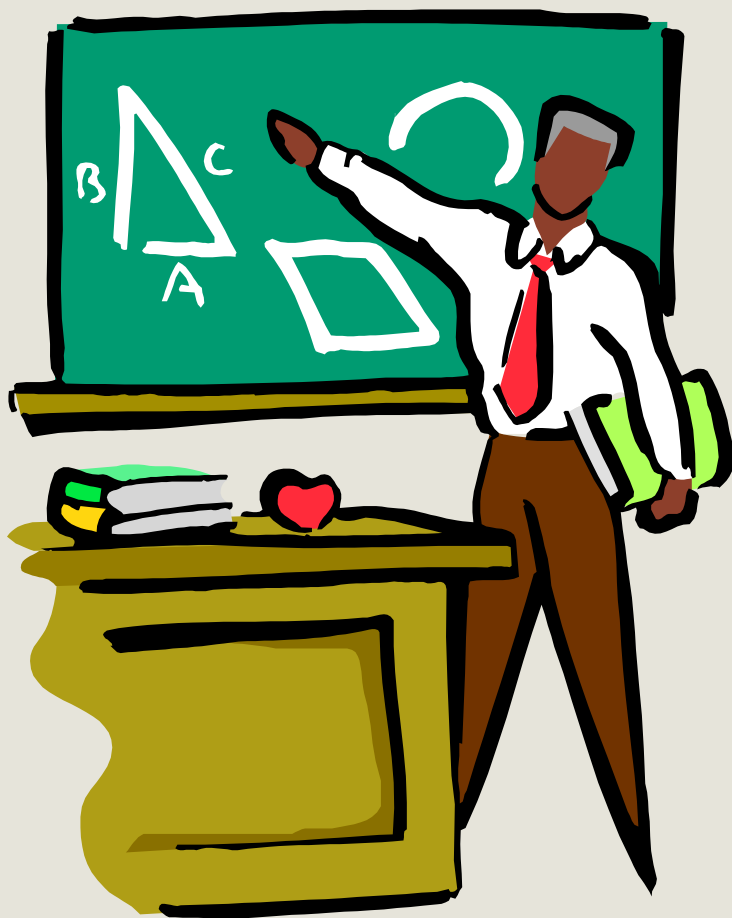
Basics



Fundamental principles

- Business model

Fundamental principles ...



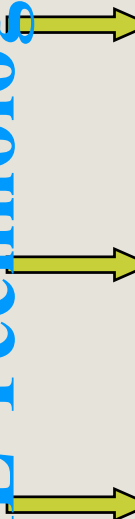
1. Evolution of the Web
2. Definitions and principles
3. Standards

Evolution of the Web

Today

- Publication of documents
- Human interaction
- Proprietary ad-hoc interfaces

XML Technology



Tomorrow

- Publication of **“reusable business logic”**
- Automated P2P interaction
- Industry standard interfaces

Definitions and principles

“The term Web Services refers to an architecture that allows applications (on the Web) to talk to each other. Period. End of statement”

Adam Bobsworth in ACM Queue, Vol1, No1

The three fundamental principles, still according to Adam Bobsworth:

- 1. Coarse grained approach (I.e. high level interface)**
- 2. Loose coupling (e.g. application A which talks to application B should not necessarily be re-written if application B is modified)**
- 3. Synchronous mode of communication, but also asynchronous mode**

Standards

Some of the involved standards bodies / Consortia

- Architectures and Technologies

- World Wide Web Consortium (W3C)
 - Interoperable technologies for the Web
- Liberty Alliance
 - Open standards for federated network identities (pertinent to Web service security)

Standards

Some of the involved standards bodies / Consortia

Application to specific areas

Telecom

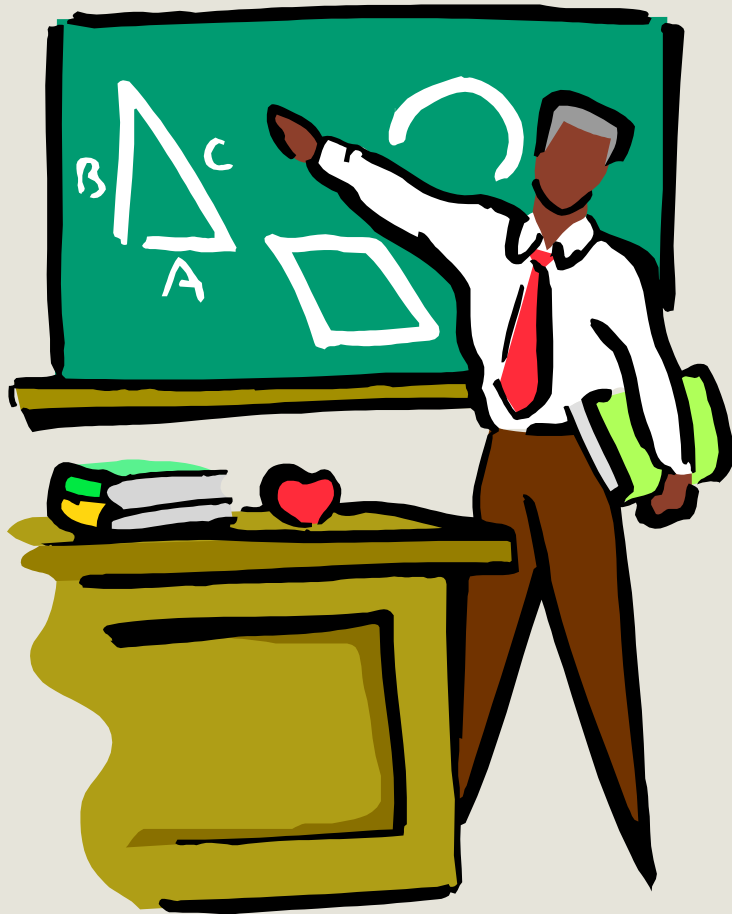
Parlay-X

Open Mobile Alliance (OMA)

Digital images

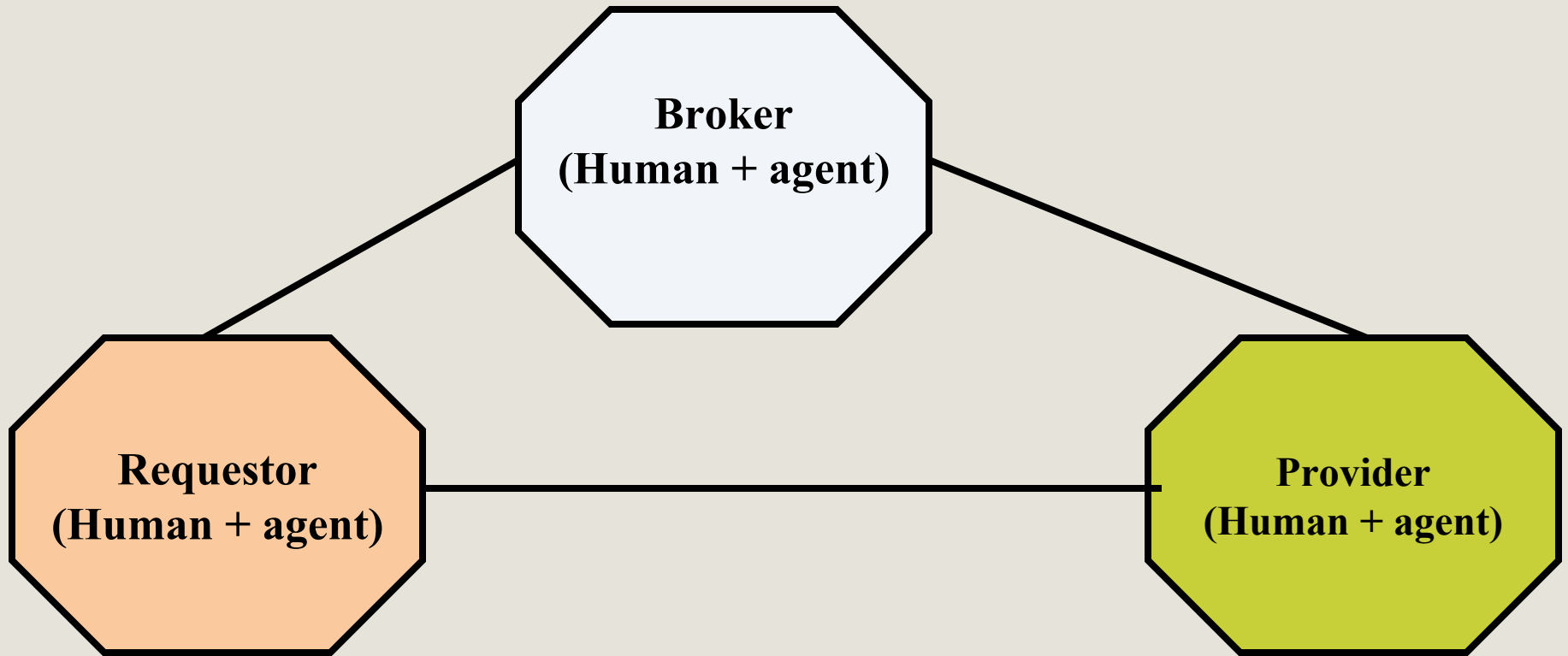
- International Imagery Association

Business model ...



1. Entities
2. Interactions

Entities



Entities

Requestor

- Person or organization that wishes to make use of a Web service.
- Uses an agent (I.e requestor agent) to exchange messages with both broker agent and provider agent.

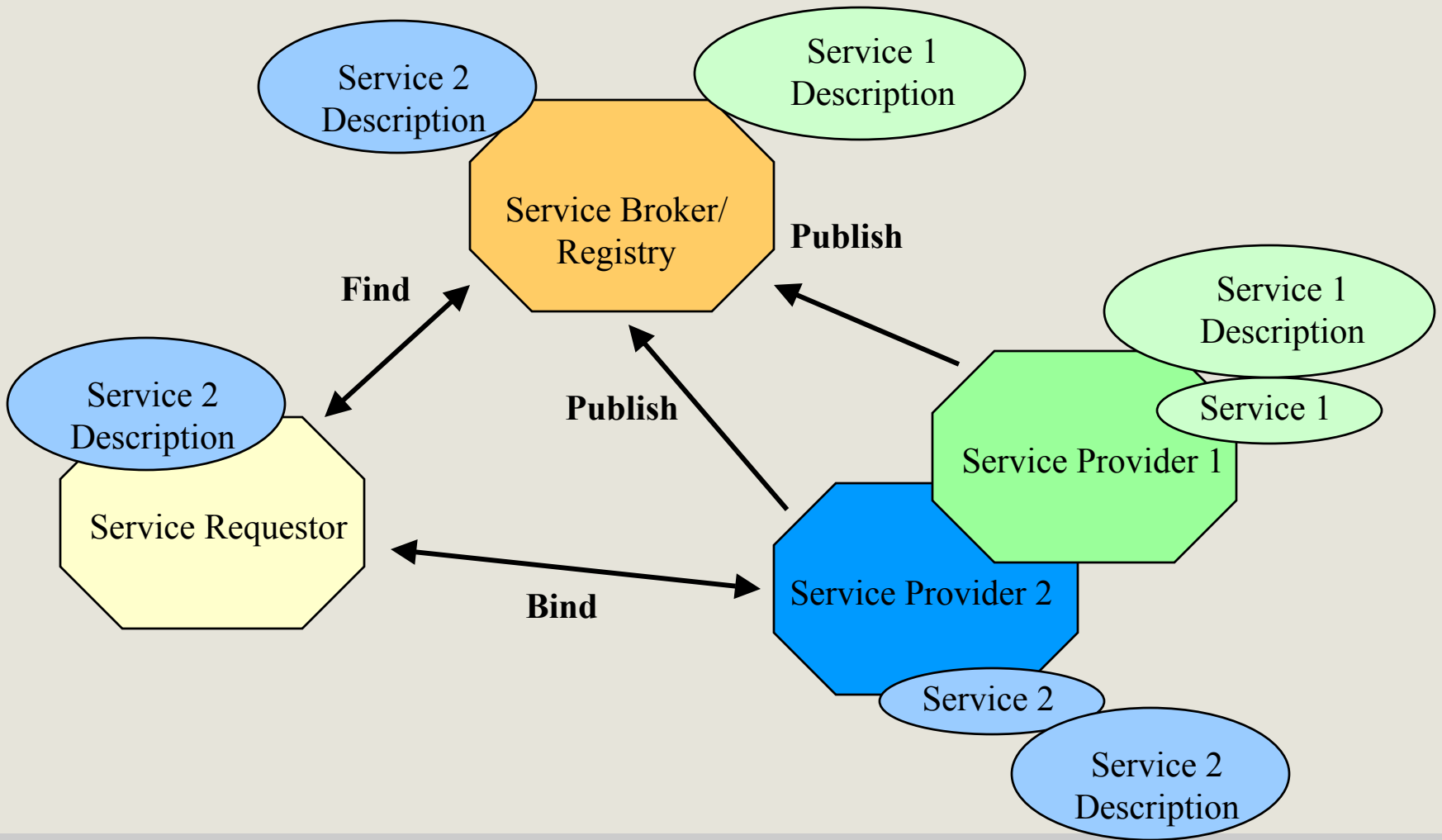
Provider

- Person or organization that owns a Web service it wants to make available for usage
- Use an agent (I.e provider agent) to exchange messages with broker agent and requestor agent.
- The provider agent is also the software piece which implements the Web service (e.g. mapping towards legacy)

Broker

- Person or organization that puts requestors and providers in contact
 - Providers use brokers to publish Web services
 - Requestors use brokers to discover Web services
- Use an agent (I.e broker agent) to exchange messages with requestor agent and provider agent

Interactions

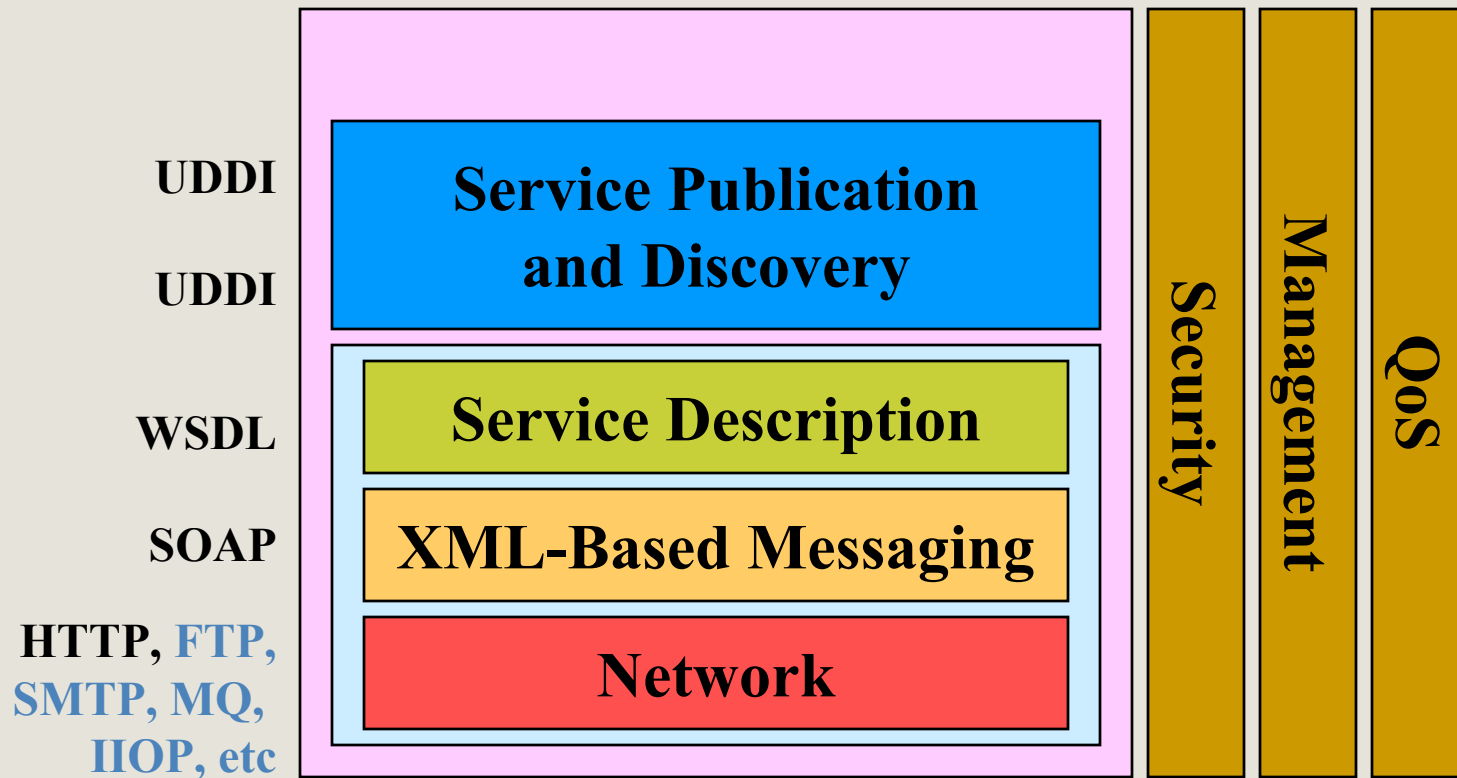


Technologies

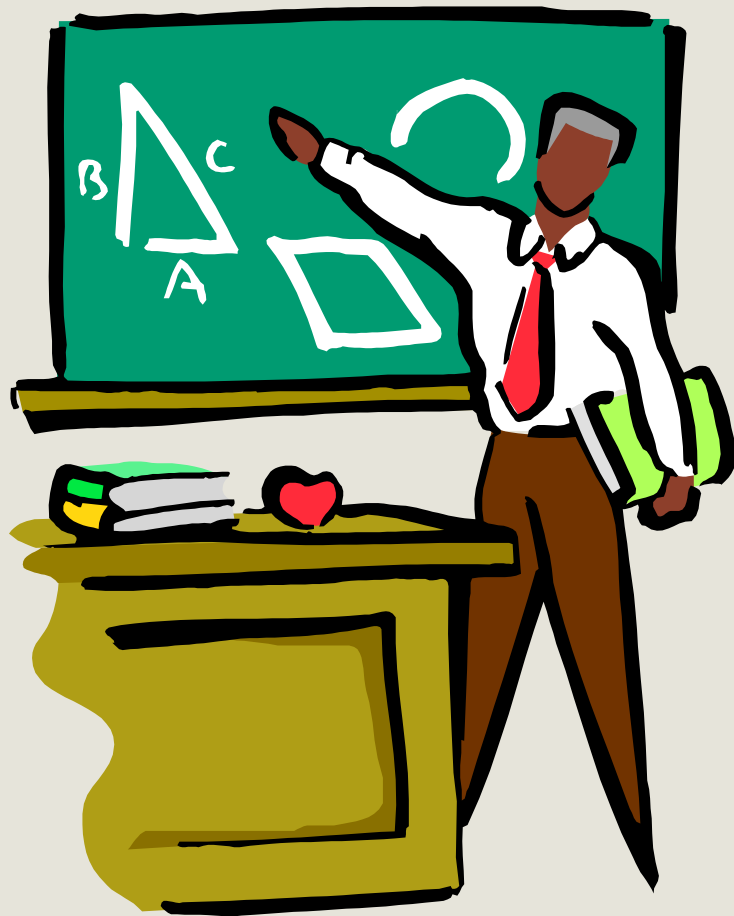


- Extensible Mark Up Language (XML)
- Web Service Description Language (WSDL)
- Simple Object Access Protocol (SOAP)
- Universal Description Discovery and Integration (UDDI)
- Putting it together

Technologies ...



XML ...



- Introduction
- Objectives
- Fundamental concepts
- Examples

Introduction

Designed to overcome the limitations of HTML

- Better support for dynamic content creation and management
 - Dynamic content management is clumsy with HTML (e.g. CGI script, servlet)
- Interactions between programs going further than browser / Web page
 - Exchange of data
 - Meaning of the data
 - What to do with the data

W3C recommendation

Objectives (As per the W3C recommendation)

1. XML shall be straightforwardly usable over the Internet.
2. XML shall support a wide variety of applications.
3. XML shall be compatible with SGML.
4. It shall be easy to write programs which process XML documents.
5. The number of optional features in XML is to be kept to the absolute minimum, ideally zero.
6. XML documents should be human-legible and reasonably clear.
7. The XML design should be prepared quickly.
8. The design of XML shall be formal and concise.
9. XML documents shall be easy to create.
10. Terseness in XML markup is of minimal importance.

Fundamental concepts

XML documents

Data objects made of elements

Schema vs. Instance

- Schema: Type (Enable the validation of instances)
 - Alternative to schema: Data Type Definition (DTD)
- Instance: actual data
- Name space
 - Scope or qualify element names

Fundamental concepts

Elements

- Structures: Enable the grouping of elements
- Mark ups: Enable the separations of elements
 - Start tag < >
 - End tag < />

Instances of documents

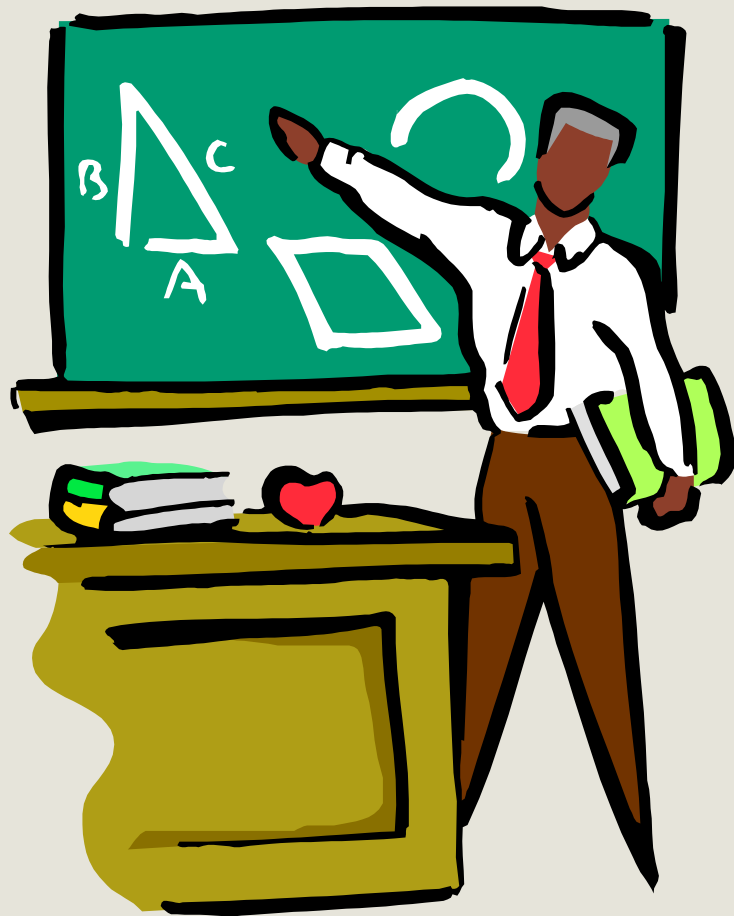
- **Characters**
 - **Character data**
 - **Name**
 - **Value**
 - **Mark ups**

Fundamental concepts

XML processor

- Read XML documents
- Provide access to the content and the structure
- Behaviour described in the XML specifications
- Most popular programming APIs
 - Document Object Model (DOM) from W3C
 - Simple API for XML (SAX) – From XML-DEV mailing list

SOAP ...



- Introduction
- Message structure
- Bindings

Introduction

Purpose: Get the XML data from one point to another point over the network

- Provider / UDDI
- Requestor / UDDI
- Provider / Requestor

W3C recommendation

- Effort initiated by IBM and IONA

Introduction

Purpose: Get the data from one point to another point over the network

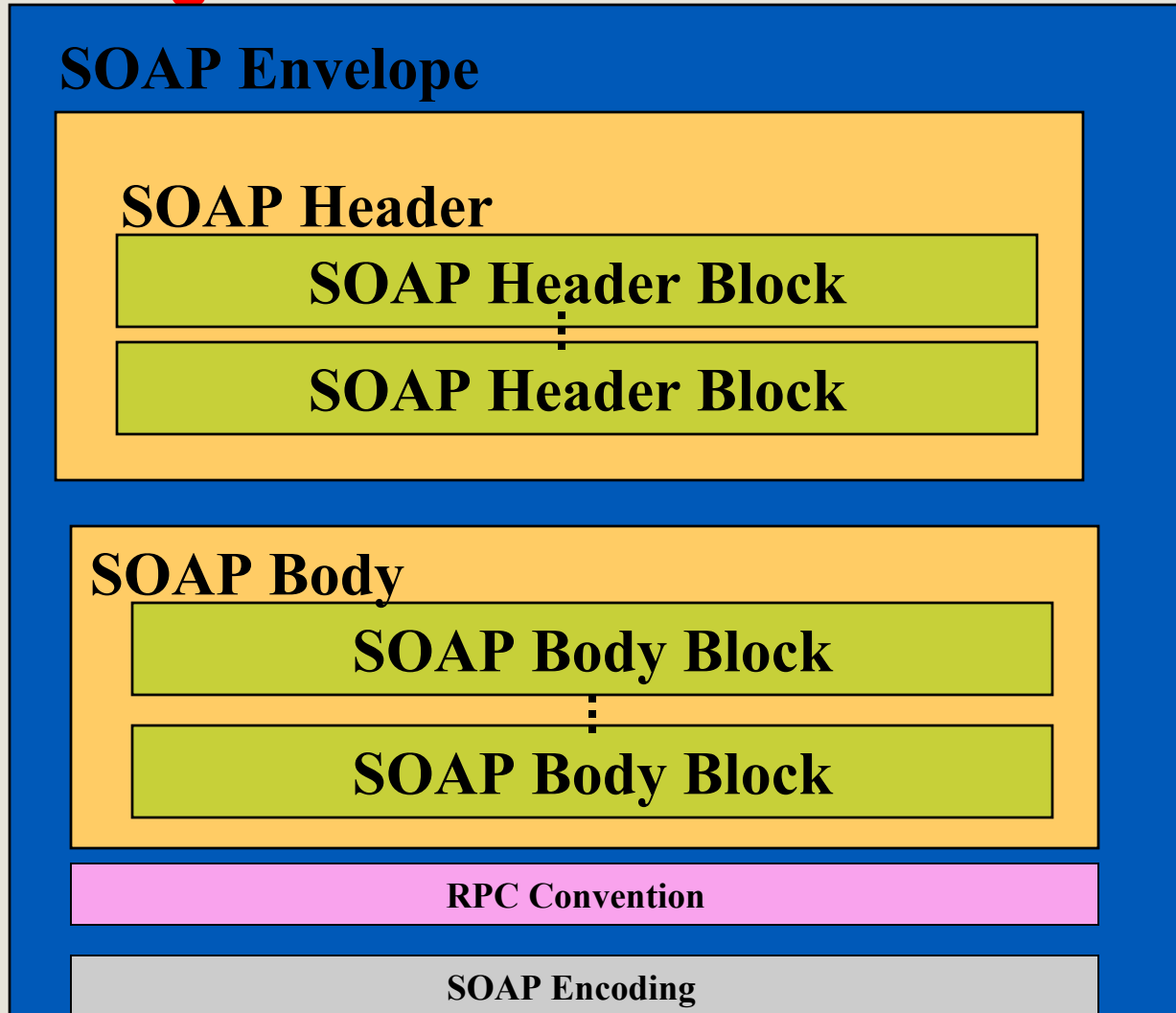
- Misleading name: Does not include an object model
- One way XML messaging protocol that can be used to build models such as
 - Request / reply
 - Asynchronous messaging
 - Event notification
- Entities
 - Sender
 - Receiver
 - Intermediary

Message structure

Several parts

- Envelope (mandatory): Start and end of message
- Header (optional): Optional attributes used in the processing
 - May be negotiated
 - Examples: transactions, priority, QoS, security
- Body (mandatory): Message being sent
 - Actual message
 - Fault codes
- Attachment (optional) : Self-explanatory
- RPC convention (optional) : Requirements for RPC mapping
 - Target URI for the SOAP node, procedure name/signature
- SOAP Encoding (optional) : How to represent data being transmitted in the message
 - Encoding scheme

Message structure



Bindings

Purpose: Specification of how SOAP messages may be passed from one node to another node using a concrete lower layer protocol

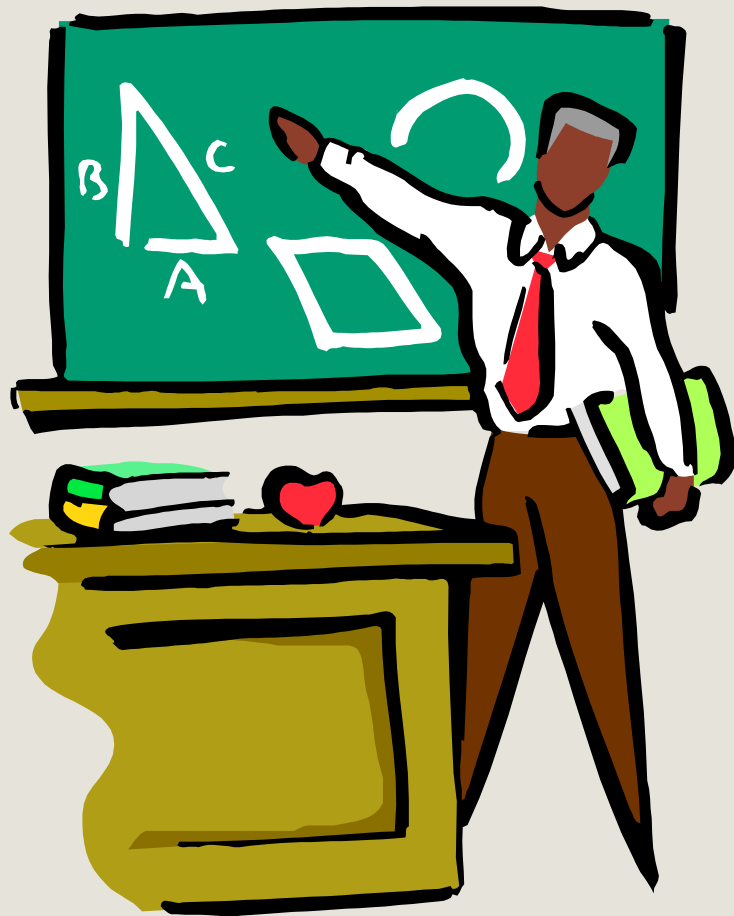
Existing bindings

- HTTP
- SOAP over email

HTTP binding

- HTTP Request URI used to identify SOAP node
- Commonly used HTTP request for carrying SOAP messages: HTTP Post

WSDL ...



- Introduction
- Elements
- Grammar

Introduction

Purpose: XML grammar for describing a Web service

- Formats and protocols
 - Input data to the Web service
 - Operations to be performed on the data
 - Binding to a transport protocol
- Initially developed by a handful of companies (e.g. IBM, Microsoft)
- Now a W3C recommendation

Elements

Types

- Data type definition using for instance XSD

Messages

- Abstract definition of what goes on the wire
 - One way
 - Request / reply
 - Solicit response
 - Notification

Operation

- Abstract definition of an action supported by the service

Port type

- Abstract set of operations supported by one or more endpoints

Elements

Types

- Data type definition using for instance

Binding

- Concrete protocol and data format specification for a particular port type

Port

- Single endpoint defined as a combination of a binding and a network address

Service

- Collection of related endpoints

Grammar

Conventions used in the specifications

- ? (0 or 1)
- * (0 or more)
- + (1 or more)

Grammar

Examples

One way messaging vs. two way messaging

```
<wsdl:definitions .... > <wsdl:portType .... > *  
  <wsdl:operation name="nmtoken">  
    <wsdl:input name="nmtoken"? message="qname"/>  
  </wsdl:operation>  
</wsdl:portType >  
</wsdl:definitions>
```

```
<wsdl:definitions .... >  
  <wsdl:portType .... > *  
    <wsdl:operation name="nmtoken" parameterOrder="nmtokens">  
      <wsdl:input name="nmtoken"? message="qname"/>  
      <wsdl:output name="nmtoken"? message="qname"/>  
      <wsdl:fault name="nmtoken" message="qname"/>*  
    </wsdl:operation>  
  </wsdl:portType >  
</wsdl:definitions>
```


Example from WSDL specification

Example 1 SOAP 1.1 Request/Response via HTTP

```
<?xml version="1.0"?>  
<definitions name="StockQuote"  
  
targetNamespace="http://example.com/stockquote.wsdl"  
  xmlns:tns="http://example.com/stockquote.wsdl"  
  xmlns:xsd1="http://example.com/stockquote.xsd"  
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"  
  xmlns="http://schemas.xmlsoap.org/wsdl/">
```

Example from WSDL specification

Example 1 SOAP 1.1 Request/Response via HTTP - Continued

```
<types>
  <schema targetNamespace="http://example.com/stockquote.xsd"
    xmlns="http://www.w3.org/2000/10/XMLSchema">
    <element name="TradePriceRequest">
      <complexType>
        <all>
          <element name="tickerSymbol" type="string"/>
        </all>
      </complexType>
    </element>
    <element name="TradePrice">
      <complexType>
        <all>
          <element name="price" type="float"/>
        </all>
      </complexType>
    </element>
  </schema>
</types>
```

Example from WSDL specification

Example 1 SOAP 1.1 Request/Response via HTTP - Continued

```
<message name="GetLastTradePriceInput">
  <part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
  <part name="body" element="xsd1:TradePrice"/>
</message>

<portType name="StockQuotePortType">
  <operation name="GetLastTradePrice">
    <input message="tns:GetLastTradePriceInput"/>
    <output message="tns:GetLastTradePriceOutput"/>
  </operation>
</portType>
```

Example from WSDL specification

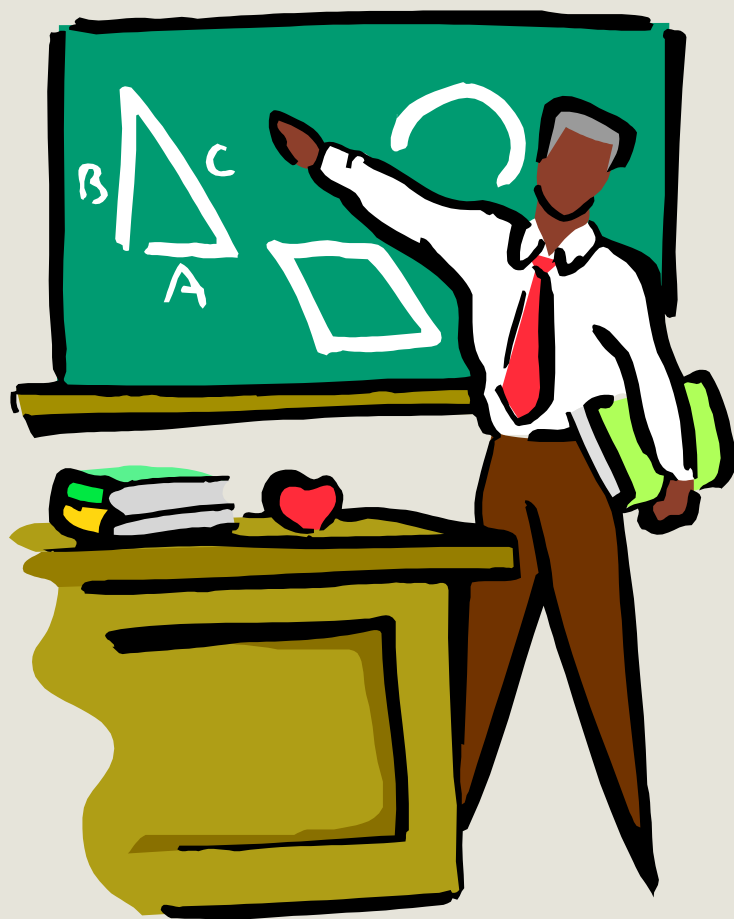
Example 1 SOAP 1.1 Request/Response via HTTP - Continued

```
<binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <operation name="GetLastTradePrice">
    <soap:operation soapAction="http://example.com/GetLastTradePrice"/>
    <input>
      <soap:body use="literal"/>
    </input>
    <output>
      <soap:body use="literal"/>
    </output>
  </operation>
</binding>

<service name="StockQuoteService">
  <documentation>My first service</documentation>
  <port name="StockQuotePort" binding="tns:StockQuoteBinding">
    <soap:address location="http://example.com/stockquote"/>
  </port>
</service>

</definitions>
```

UDDI ...



- Introduction
- Content
- Publishing APIs
- Inquiry APIs

Introduction

Purpose: Enable the publication, the discovery and the usage of Web services

- Integral part of the Web services infrastructure
 - Public
 - Semi-public (e.g. circle of trust)
 - Private (e.g. enterprise)
- Data bases accessible via SOAP APIs
 - Publishing API
 - Inquiry APIs

Introduction

UDDI.ORG

- Initiated by a handful of companies (e.g. IBM, Microsoft)
- Now open to all companies
- Produce specifications for UDDI

Initial public UDDI repository

- Operated by founders of UDDI.ORG, later joined by HP and SAP
- Synchronized data bases called operator sites (one at each site)
- Test UDDI
 - Allow requestors and providers to test their UDDI clients
- Production UDDI
 - Allow providers to actually publish Web Services and requestors to actually inquire about Web services
 - Need to register with one of the operators for publishing services (authorization)

The content ...

White pages

- Business address
- Contact person / number

Yellow pages

More info about the business

- Type of business
- Industry type
- Products / services

Green pages **NEW!**

Technical information about the services

- Service features/functionality
- Pointer to the WSDL file

The content ...

UDDI data model

Business entity

- Top level structure
- Description of the entity for which information is being registered
- Include the list of Web services provided by the entity

Business services

- Name and description of services being published
- Include binding templates

Binding templates

- Information about the services
- Include entry point for accessing the services

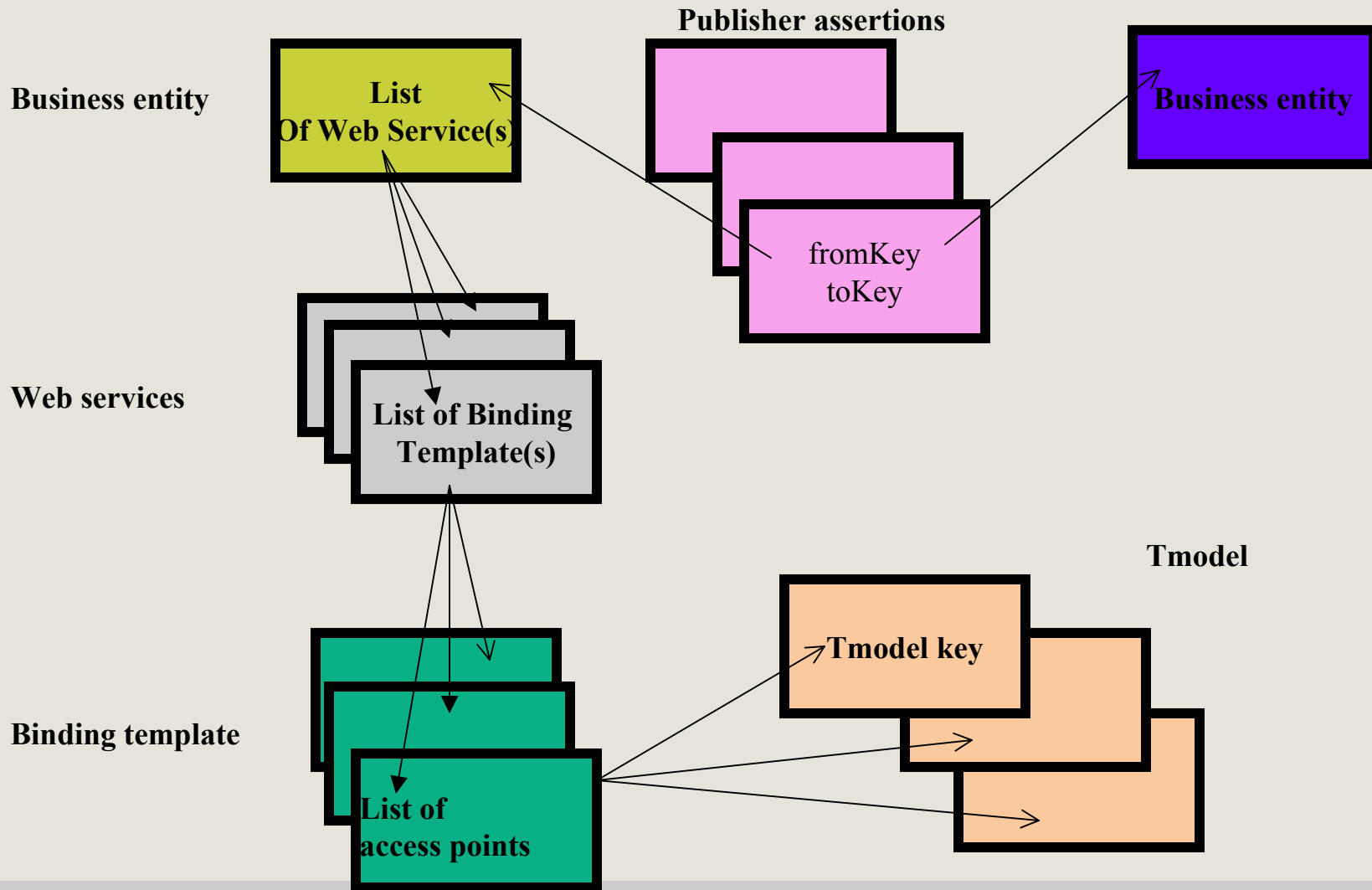
Model

- Fingerprint, collection of information that uniquely identify the service

Publisher assertion

- Business relationship between business entities (e.g. subsidiary of ..)

Data model ...



Publishing APIs

Some examples

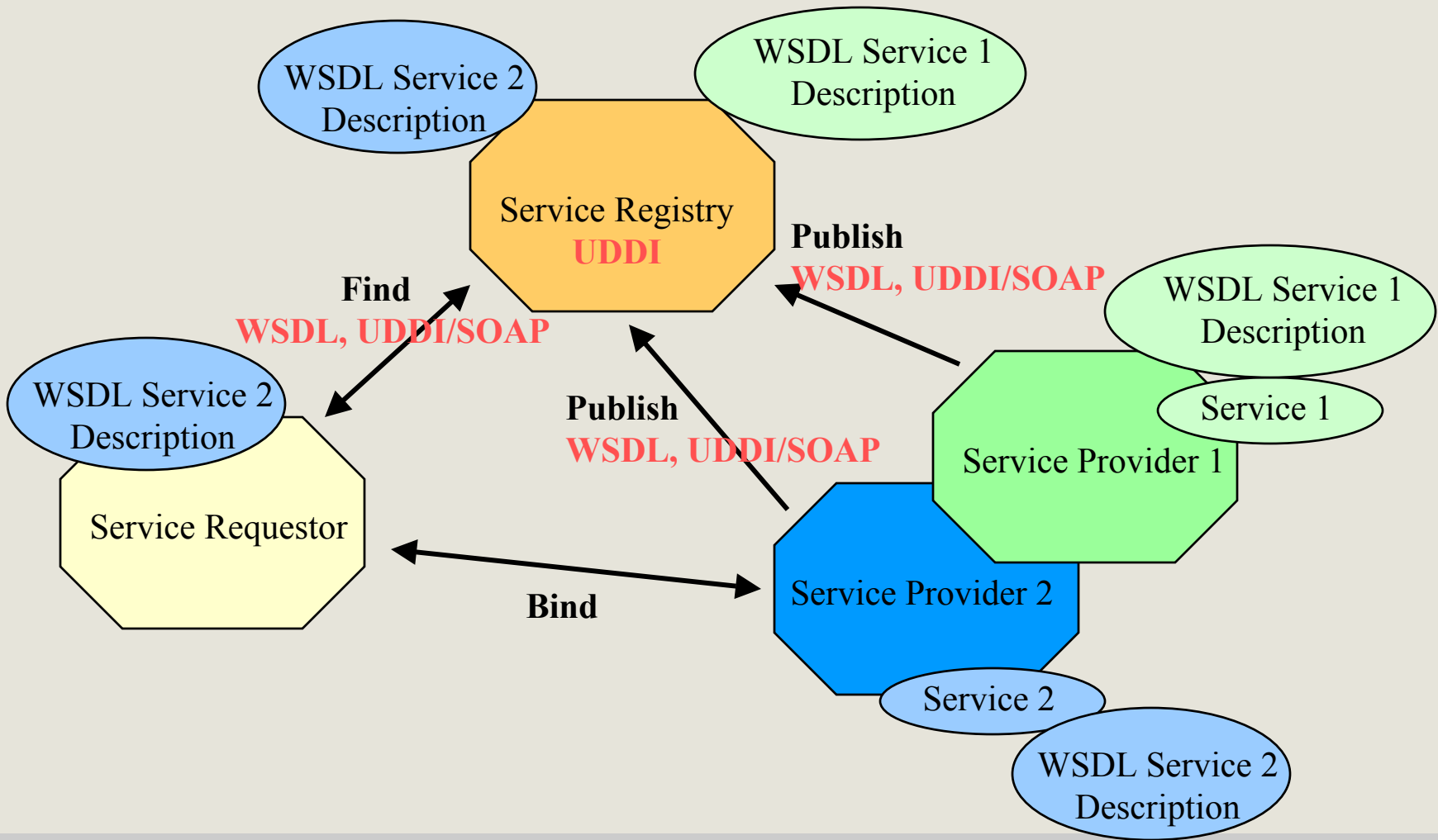
- Add Publisher Assertions
- Save/Delete binding
- Save/Delete Business
- Save/Delete Service
- Save/Delete tModel
- Set/Get Publisher assertions
- Delete_Publisher_Assertion
- Get Registered assertions
- Get Assertions status report (used by UDDI operators)

Inquiry APIs

Some examples

- Find binding
- Find business
- Find related business
- Find service
- Find tModel
- Get binding details
- Get business details
- Get tModel details

Putting it together ...



Putting it together ...

Examples of tool kits

- Apache / Axis
- BEA Weblogic
- SunOne
- .Net
- Systinet
- Get tModel details

To probe further ...

- F. Curbera et al., **Unraveling the Web services Web: An Introduction to SOAP, WSDL and UDDI**, IEEE Internet Computing, Vol. 6, No2, March-April 2002, pp. 86-93
- E. Newcomer, **Understanding Web Services: XML, WSDL, and UDDI**, Addison Wesley, 2002
- W3C specifications
- OASIS specifications (UDDI)

▪