

Mobile Web Services

Course ID: 582496

31 October 2005 - 08 December 2005

Monday & Thursday : 16:00-18:00

WST: Web Service Technology Overview

21st November 2005

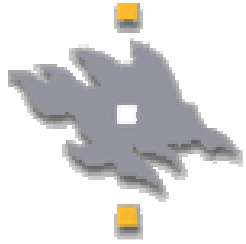
Suresh Chande
Department of Computer Science
email: chande@cs.helsinki.fi



Web Services Technology (WST) Map

- Provide a complete landscape of Web Services technologies and its development towards enabling services oriented architectures.
- Main purpose of a specific Web Services technology being proposed
- Understand the relation of each other and the manner it affects the Web Services architecture

NOTE: The Technologies we study today are only for overview of ongoing Activities, they are all not necessarily are part of Web Services Standards
Some of them might disappear or get merged into others



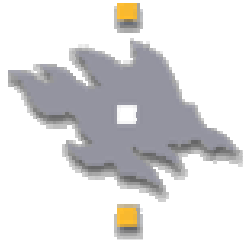
WST Map

Web
Services
Administration

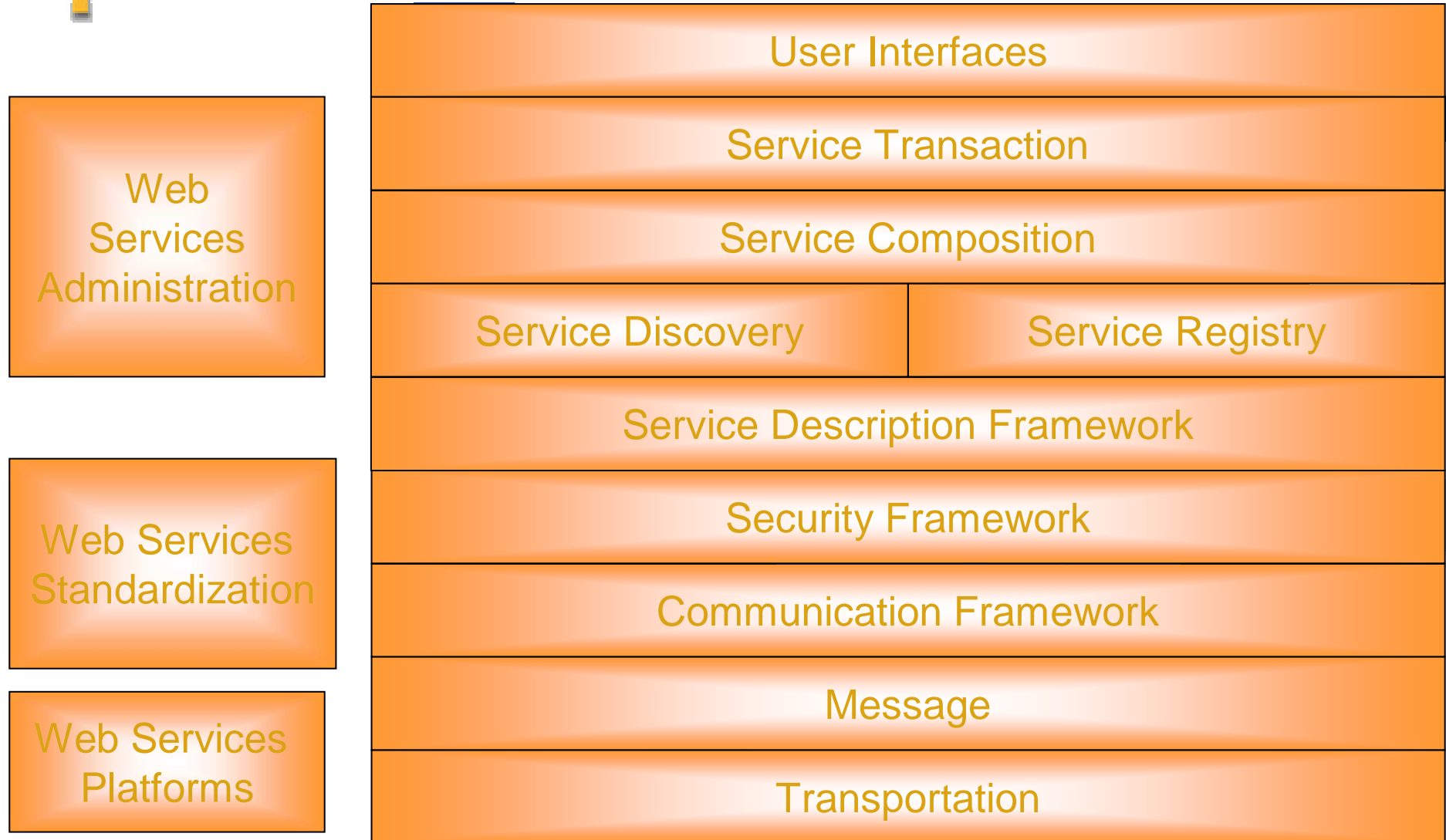
Web Services
Standardization

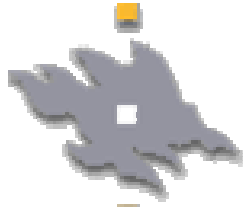
Web Services
Platforms





Web Services Technology WST-Map

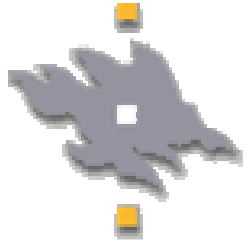




WST-Transportation

- Provides a Means of transporting XML between Web Services end points / nodes
- On the Web/Internet world HTTP Bindings today are only widely applied.
- There could several different transportation support for Web Services protocols in the future as new technologies become available:
 - Mobile Environments.
 - Proximity (RF, IRda)
 - Local communication network (Bluetooth, RFiD)
 - Image processing based: Barcodes, 2D barcodes, etc..



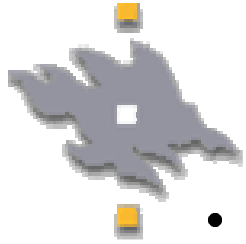


WST- Transportation

- TCP – Transmission Control Protocol
- UDP – User Datagram Protocol

- FTP – File Transfer Protocol
- SMTP – Simple Mail Transfer Protocol
- HTTP – HyperText Transfer Protocol

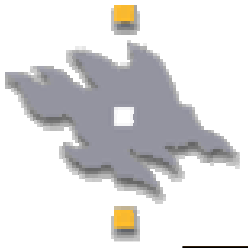




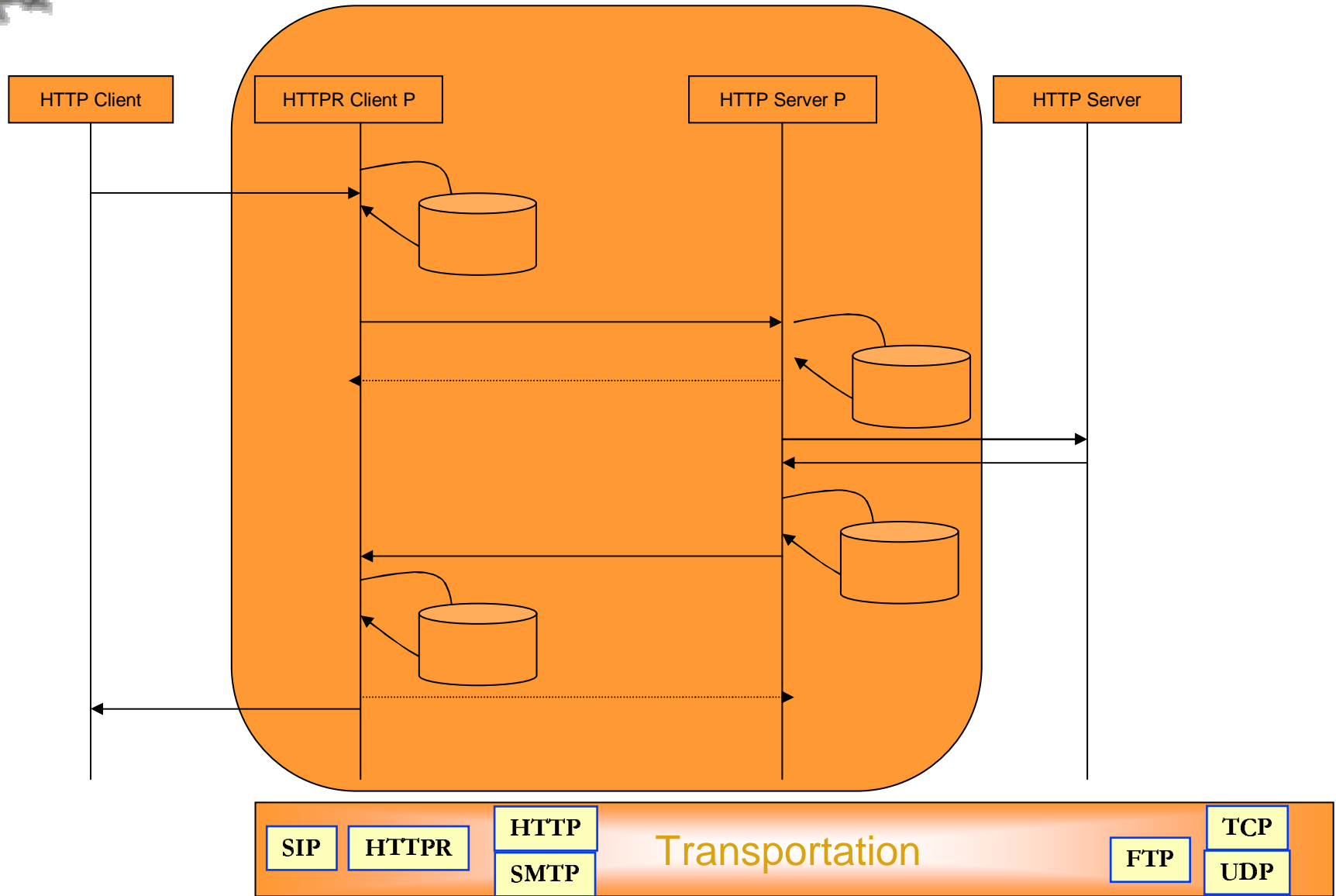
Transportation - HTTPR

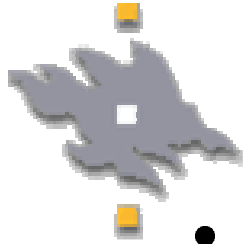
- Purpose:
 - HTTP provides reliable delivery of HTTP packets as long as failure does not occur. If there is a failure there is not means to recover or trace the cause of error.
- HTTPR solves this by:
 - Providing a protocol that provides reliable delivery of HTTP packets between the server and client
 - Provides rules that make it possible to ensure that all messages are delivered to their destination in their exact form and only once.
- HTTPR:
 - Defines how metadata and application messages are encapsulated
 - Message is delivered to its destination application exactly once or then a failure is notified to the requester/responder correspondingly
 - Emphasises the storage of the messages prior to successful exchange of the messages between client & Servers
- HTTPR is proposed by IBM and no more supported and Technology these days to provide this functionality is : WS-ReliableMessaging
- Further reading : <http://www-106.ibm.com/developerworks/webservices/library/ws-phtt/>





Transportation - HTTPR

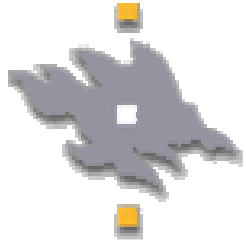




Transportation - SIP

- SIP - Session Initiation Protocol
- SIP is a text-based protocol, similar to HTTP and SMTP, for initiating interactive communication sessions between users.
- SIP can be used in the Web Services world to set up a connection between two Web Services end points.
- SIP works on many different types of transportation protocols such as TCP and UDP.
- **Conclusion:** Web Services provide open standards based services over the Web and SIP could allow them to be presented within integrated communication applications that include Voice and multi-media services.
- Further reading:
 - SIP Specifications: <http://www.ietf.org/html.charters/sip-charter.html>
 - SIP & SOAP: http://www.sipcenter.com/files/Ubiquity_SIP_and_SOAP.pdf

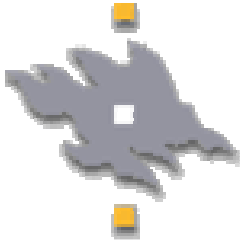




WST-Message

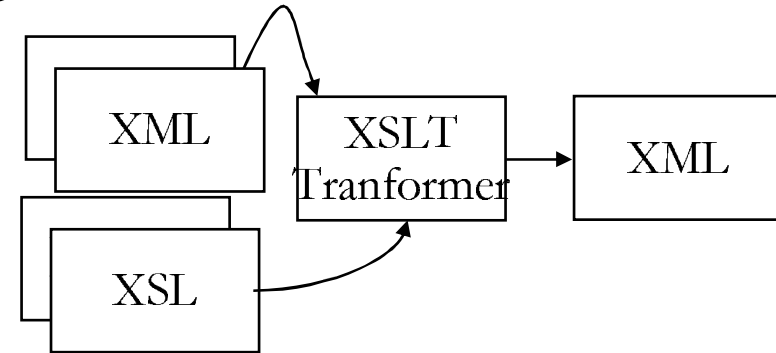
- A Framework layer to contain Message details:
 - Definition
 - Containment
 - Exchange &
 - Communication
 - Transformation
 - Adaptation
 - Conversational aspects

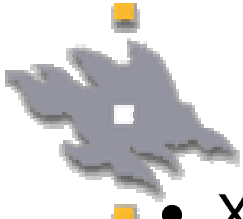




Message - XSLT

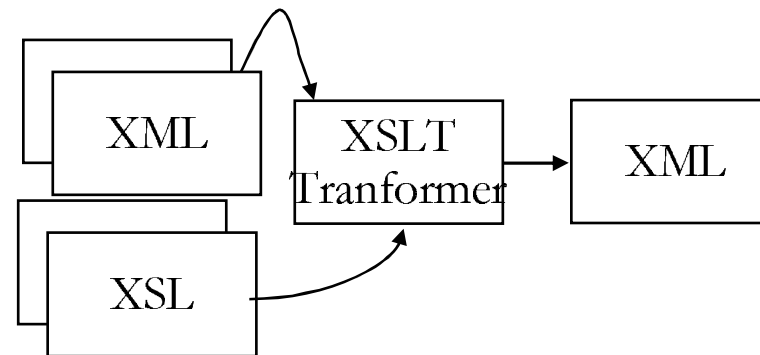
- XSLT is a language for transforming XML documents into other XML documents
- It is usually used as part of the XSL which defines the formatting styles for XML
- This language specification defines the syntac and semantics of transforming XML documents. This specifications is referred to be the namespace : <http://www.w3.org/1999/XSL/Transform>
- **Conclusion:** XSLT can be utilised in Web Services for message adaptation and transformation of messages to handle message incompatibilities and to some extent also define Stylesheet for presentation purposes

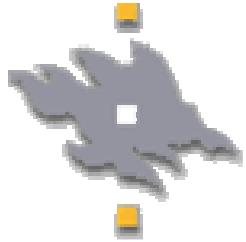




Message - XSLT

- XSLT describes rules for transforming a source XML document into a target XML document
- The rules consists of associated patterns with templates.
 - A pattern is matched against elements in the source XML Document using the XPATH.
 - A template is instantiated to create part of the target XML Documents.
- The elements from the source XML document are used to alter in the transformation process and included as part of the target document

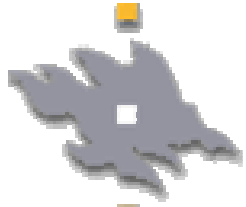




Message – XML Schema

- XML Schema provides a method to define the structure, content and semantics of XML documents.
- An XML document which complies to a Scheme defined by an XML Schema is called an Instance Document.
- XML Schemas are Namespace aware and hence can leverage reuse of schema definitions
- XML schema is defined in terms of:
 - **Simple Types:** A Simple XML elements which do not have subelements/attributes: simple data types, enumeration, lists, restricted formatted values(patterns/regular expression evaluated)
 - **Complex Types:** An XML Element which could contain 1 or more subelements and attributes.
 - This also defines the sequence, Choice of occurrence, number of occurrence (min/max), default values for attributes
 - **Global / Local Element types:** Scope of the element declarations
- XML Schema Namespace : <http://www.w3.org/2001/XMLSchema>
- **Further reading:** <http://www.w3.org/XML/Schema>

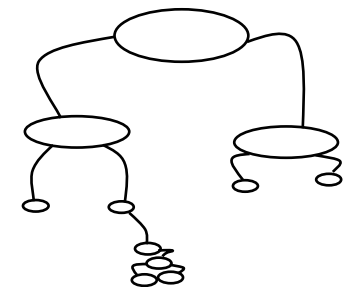


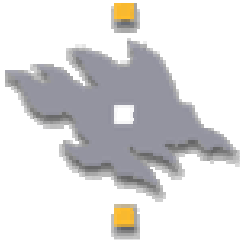


Message – XPath

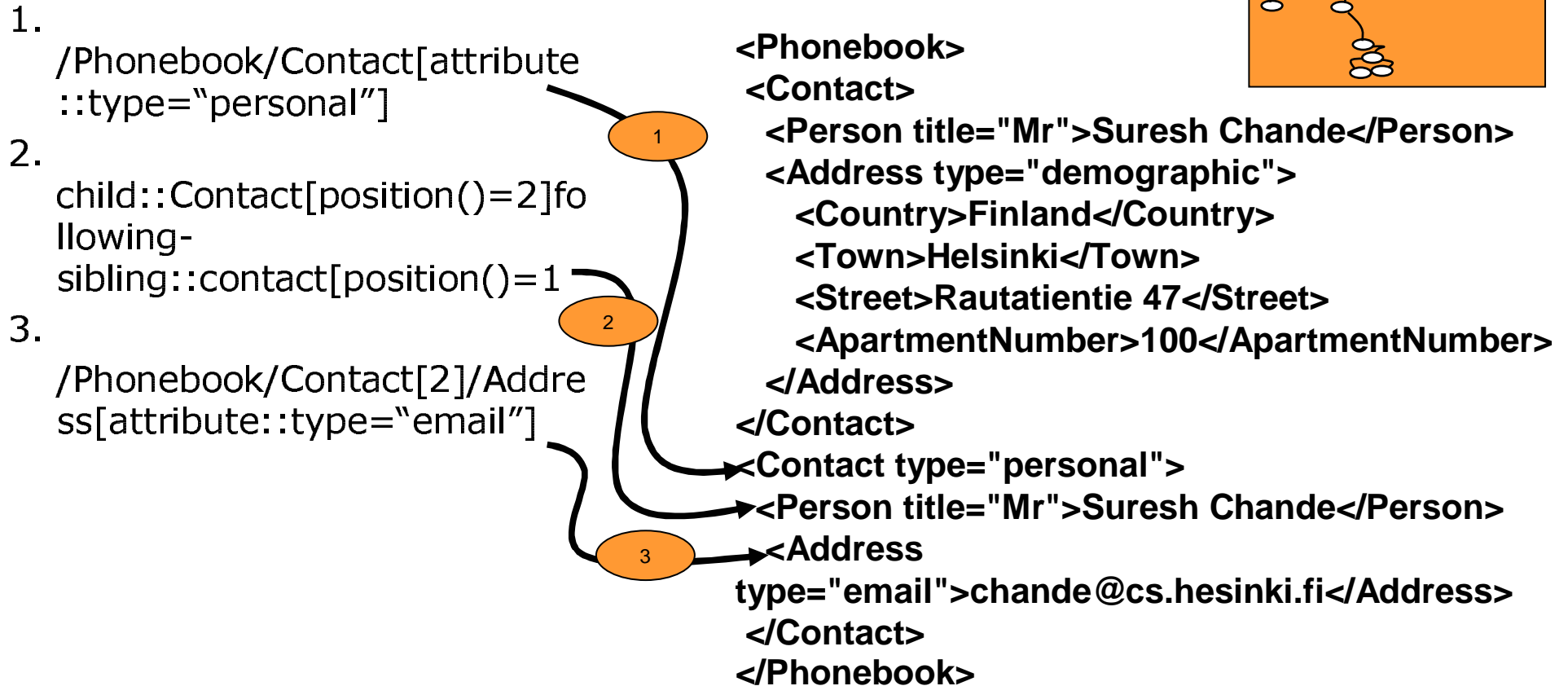
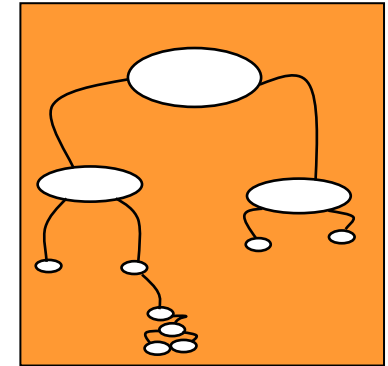
- This is a language for addressing parts of an XML document and matching nodes to patterns
- XPATH represents the XML Document as a tree of node elements, the node being : Elements, Attributes, Text, Process Instructions, Comments, etc..
- XPATH uses an non-XML syntax to address and to specify the expressions referring to nodes/node element, so as to be utilised as value of attributes.
- The Expressions consists of an addresssing mechanism with specific functions over the value/name/structure of the tree elements
- The result of the expression could lead to : node-set, boolean, number, string

Further Reading : <http://www.w3.org/TR/xpath>



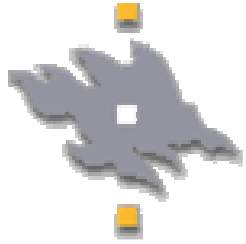


Message - XPath



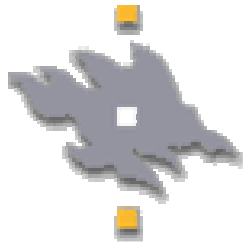
Further Reading : <http://www.w3.org/TR/xpath>





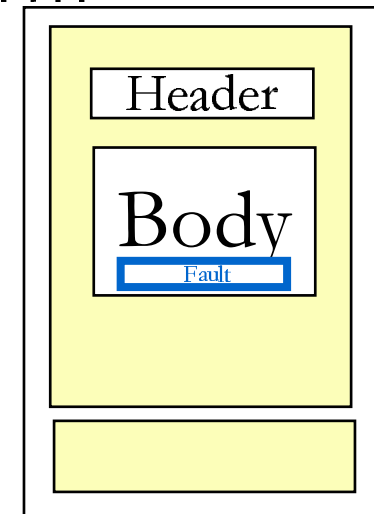
Message Containment & Communication - SOAP

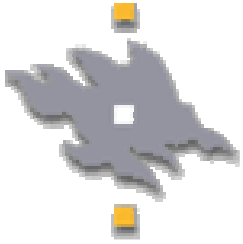
- Simple and flexible messaging framework for transferring information specified in the form of an XML infoset between an initial SOAP sender and an ultimate SOAP receiver. SOAP **does not define** application semantics but **defines** a mechanism to express application semantics.
- "SOAP defines a simple and lightweight mechanism for exchanging structured and typed information between peers over the web in a decentralized, distributed environment using XML."



Message Containment & Communication - SOAP

- Specifies :
 - Message Parts: Headers, Body, Fault and Attachments
 - Rigid rules for construction of SOAP messages
 - Message processing of SOAP messages : originary Intermediaries, ultimate recipient
 - Transportation of SOAP Messages over HTTP, SMTP as means of message exchanges
 - Message Exchange patterns :
 - Request-Response
 - RPC
 - Further reading:
<http://www.w3.org/2000/xml/Group/>



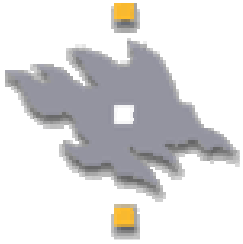


Message-MessageData

- This specification is proposed by BEA, Yaron Goland, Mark Nottingham, David Orchard February 2003
- This is a SOAP header extension to carry Meta-data about the SOAP Message so that other specifications can reuse the same header "MessageData" and the elements included there in into their specifications
- The "MessageData" SOAP header specification contains two types of meta data, namely MessageId and RefToMessageId
 - To represent a message with a unique ID represented by a URI
 - To represent a relationship to already existing Message.

Reference: http://dev2dev.bea.com/webservices/WS-MessageData-0_9.html

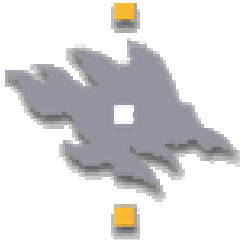




Example- MessageData

```
<?xml version="1.0" encoding="UTF-8"?>
  <Envelope xmlns:s=http://schemas.xmlsoap.org/soap/envelope/
    xmlns:wsmd="http://www.openuri.org/2003/02/soap/messagedata/">
    <Header>
      <wsmd:MessageData>
        <wsmd:MessageId>http://a.org/messageID/f81d4fae-7dec-11d0-a765-
          00a0c91e6bf6 </wsmd:MessageId>
        <wsmd:RefToMessageId> Unique ID(UUDI)</wsmd:RefToMessageId>
      </wsmd:MessageData>
    </Header>
    <s:Body>
      -----
    </s:Body>
  </Envelope>
```



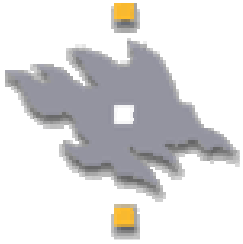


SOAP-Conversation

- This specification is made by BEA, [David Bau](#), [David Orchard](#) 13/June/2002
- This a protocol specifying SOAP headers to represent Asynchronous and Conversational Messaging.
- This protocol has a counter part WSDL Description.
- Basic SOAP Header extension are :
 - StartHeader (Sender->Receiver)
 - conversationID
 - callbackLocation
 - CallbackHeader (Receiver ->Sender)
 - conversationID
 - callbackLocation
 - ContinueHeader (Receiver <->Sender)
 - conversationID

Reference: <http://dev2dev.bea.com/pub/a/2002/06/SOAPConversation.html>





SOAP-Conversation

1. Sender-> Receiver

```
<StartHeader xmlns="http://www.openuri.org/2002/04/soap/conversation/">  
  <conversationID>UNIQUE_ID_1001</conversationID>  
  <callbackLocation>http://www.cs.helsinki.fi/u/Chande/cs/MWS</callbackLocation>  
</StartHeader>
```

(HTTP POSTd to: <http://www.cs.helsinki.fi/u/Chande/cs/MWS>)

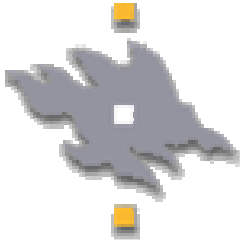
2. Reciever -> Sender

```
<ContinueHeader xmlns="http://www.openuri.org/2002/04/soap/conversation/">  
  <conversationID>UNIQUE_ID_1001</conversationID>  
</ContinueHeader>
```

3. Consecutive Messages from Sender -> Receiver

```
<ContinueHeader xmlns="http://www.openuri.org/2002/04/soap/conversation/">  
  <conversationID>UNIQUE_ID_1001</conversationID>  
</ContinueHeader>
```



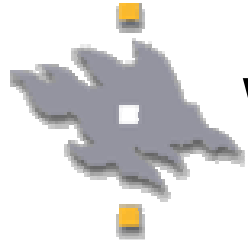


SOAP-Conversation

4. CallbackHeader will be sent by the receiver to the Sender

```
<CallbackHeader xmlns="http://www.openuri.org/2002/04/soap/conversation/">  
<conversationID>UNIQUE_ID_1001</conversationID>  
<callbackLocation>http://www.mobileservices.org/MWS</callbackLocation>  
</CallbackHeader>
```





WST- Communication Framework

- This layer provides a framework enabling SOAP based services into communication infrastructure



Communication Framework – Routing

- This is a simple and light weight SOAP header based protocol to define the paths of message transportation independent from the transportation protocol.
- The purpose of this being that SOAP can be bound to any lower transportation protocol and maintain the simple and easy routing mechanism without a need to specify the transport level routing
- This specifications addresses both the forward and return path of the Message paths
 - Forward path : is the route through which the message is transported from the initial sender via zero or more intermediaries and finally to the ultimate recipient
 - Reverse : Is the route of the message transportation from the ultimate recipient via zero or more intermediaries to the initial message originator
 - Message co-relations: the inter-relations between the messages exchanged across the routing.
- Routing is specified using the SOAP header entries: ***action, to, from, via, fwd, rev, id, relatesTo, fault***
- WS Routing Namespace : <http://schemas.xmlsoap.org/rp/>

Further Reading :

<http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnglobspec/html/ws-routing.asp>

Communication Framework – Address

- Addressing in Web Services provides transport-neutral (for ex: HTTP/SMTP) mechanisms to address Web services and messages
- This specification defines XML elements to identify Web service endpoints and to secure end-to-end endpoint identification in messages
- This specification is defined based on SOAP Header entries containing:
 - **Message ID, RelatesTo, To, Action, From, ReplyTo, FaultTo, Recipient, EndPointReference(Address, ReferenceProperties, PortType, ServiceName, Policy?)**
- **WS Addressing Namespace:** "http://schemas.xmlsoap.org/ws/2003/03/addressing"
- **Specified by:** IBM, BEA, Microsoft - March 2003
- **Further reading:** <http://www-106.ibm.com/developerworks/webservices/library/ws-add/>



Communication Framework – Events

- This specification defines a protocol that allows Web services to subscribe to or accept subscriptions for event notification messages.
- **Provides an ability:**
 - For one Web Services to Subscribe to Events generated by another Web Services
 - Allow the subscriber to relate the events being notified to the subscription
 - Leasing and Expiry of Subscription
- WS Addressing is utilised for addressing Web Services as part of the SOAP header entries
- The protocol uses the SOAP body to propagate the subscription requests and event responses.
- The elements used in the protocol are : ***Subscribe, SubscribeResponse, Id, NotifyTo, EndTo, Expires, Filter, Renew, RenewResponse, UnSubscribe, SubscriptionEnd, Code, Reason***
- **Specified by: Microsoft, IBM, TIBCO Software, Sun Microsystems, BEA Systems, August 2004**
- Further reading: <http://ftpna2.bea.com/pub/downloads/WS-Eventing.pdf>

WS-Notification

WS-Addressing

WS-Eventing

WS-Transfer

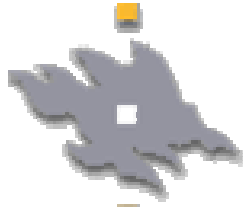
Communication Framework

WS-Routing

WS-Callback

WS-Resource

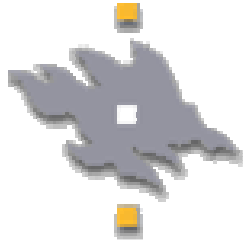
WS-Polling



Communication Framework – Notification

- A protocols based on SOAP and WSI for Services to notify about events or messages to other services as an act of response to a subscription
- This is based on a topic – oriented publish / subscribe Messaging exchange pattern, which contains standard means for service providers, notification brokers to publish messages including the operational requirements for Publishers and subscribers based on XML model of topics
- The features considered by this Specifications are:
 - Independence from the message transportation protocol
 - Ability to include the MOM as the implementation platform
 - Notification Service provider/broker
 - Message Topic transformation and aggregation
 - Metadata about subscribable Message topics





Communication Framework – Notification

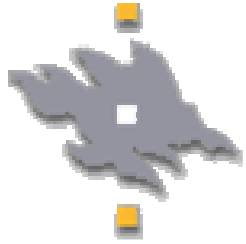
- Notification provides roles for multiple players: Publisher, Subscriber, NotificationBroker, SubscriptionManager,
- WS-Notification Namespace :
<http://www.ibm.com/xmlns/stdwip/web-services/WS-Notification>
- This is a specification proposed by: IBM, Sonic Software, Tibco, HP, Akamai, SAP, Globus, Argonne National laboratory (**20 January 2004**)
- **Further Reading:** <http://www-106.ibm.com/developerworks/library/ws-resource/ws-notification.pdf>



Communication Framework – Polling

- This specification defines a mechanism to deliver messages destined to an unreachable endpoint by allowing the destination to 'poll' the source for messages targeted for it
- When a Client – Server can not open connections to each other freely:
 - Behind Firewall
 - Client does not have a listener when the server wants to send a message Asynchronously.
- Specifies both SOAP header & Body Elements
 - GetMessage
 - StatusRequested <-> Status
- Utilizes: WS-Addressing
- Specification is a submission to W3C for discussion and is proposed by:IBM on 26 October
- **Further Reading:** <http://www.w3.org/Submission/ws-polling/>

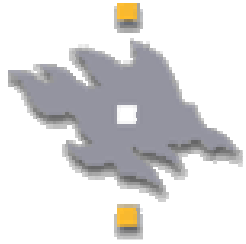




Communication Framework – Resource

- This is WS Specification developed by: IBM, Computer Associates, Oracle, webMethods, Argonne National Laboratory, Fujitsu Laboratories of Europe, Hewlett-Packard, dated 1 January 2004.
- Is a set of specifications to access stateful resources of Web Services.
- Web Services can expose resources available over the web and provide corresponding interfaces and define message structures to:
 - Manage and control the resource properties
 - Query, Lease and Expire the validity of the existence of web resources
- This is supported by two specification proposals, namely :
- **Resource properties:**
 - Managing the properties of a Web Resource:
 - Accessing properties of the resource
 - Updating the properties of the resources.
 - Inserting / Deleting resource properties
 - Query resource properties
 - Subscribe for changes of resource properties



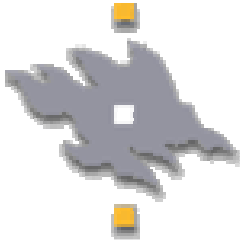


Communication Framework – Resource

- Resource Time Span:
 - Managing the expiry and validity of the Resource
 - Any client of a WS-Resource may establish and extend the scheduled termination time of a WS-Resource.
 - WS-Resource can destroy itself without the need for an explicit destroy request message from a client.
 - the WS-Resource can be destroyed at the very instant of time when a request is scheduled or can be scheduled to be destroyed.
 - Lease and renew the termination time of a WS-Resource
- Is specified as a set of specifications: WS-ResourceProperties, WS-ResourceLifetime, WS-BaseFaults, and WS-ServiceGroup

Further reading: <http://www-128.ibm.com/developerworks/library/specification/ws-resource/>

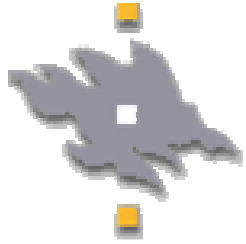




Communication Framework – Transfer

- This Specification provides a SOAP based protocol to access the XML representation of the WS based resource
- Used to Access XML based representation of two types of entities with corresponding operations:
 - Resources: Sending(put) & Receiving(get)
 - Resource Factories: Creating & Deleting
- Utilises WS-Addressing SOAP Headers with a URI to refer to specific operations (Action SOAP Header),
- This is a specification proposed by: September 2004, Systinet, Microsoft, Sonic Software, BEA, Computer Associates
- **Further Reading:**
<http://ftpna2.bea.com/pub/downloads/webservices/ws-transfer.pdf>





Communication Framework – Transfer

<Envelope

xmlns:wsa="http://schemas..../addressing">

<header>

Examples:

<wsa:Action> <http://schemas..../transfer/Get></wsa:Action> or

<wsa:Action> <http://schemas..../transfer/Put></wsa:Action> or

<wsa:Action> <http://schemas..../transfer/Delete></wsa:Action> or

<wsa:Action> <http://schemas..../transfer/Create></wsa:Action> or

...

</header>

<body/>

</Envelope>

WS-Notification

WS-Addressing

WS-Eventing

WS-Transfer

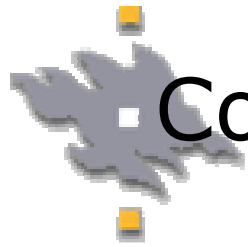
Communication Framework

WS-Routing

WS-Callback

WS-Resource

WS-Polling



Communication Framework – Callback

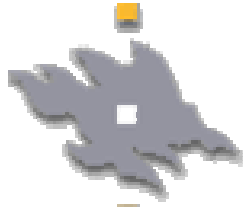
This specification details where to send an asynchronous responses dynamically to SOAP requests

SOAP Header: Callback, callbackLocation(URI)

This Specification is proposed by: Yaron Goland, Mark Nottingham, David Orchard, BEA System

Reference: http://dev2dev.bea.com/webservices/WS-Callback-0_9.html





Web Services User Interfaces

- Web Services are initially been applied for service to service integration.
- A need to address direct user interactions with user interface support is considered.

Technologies addresssing User interaction mechanism and presentation of Web Services messages are handled by :

WSRP


WSXL and

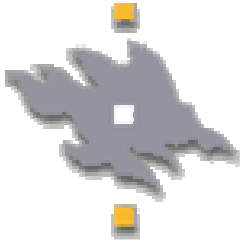
xForms are emerging to address this space .

User Interfaces

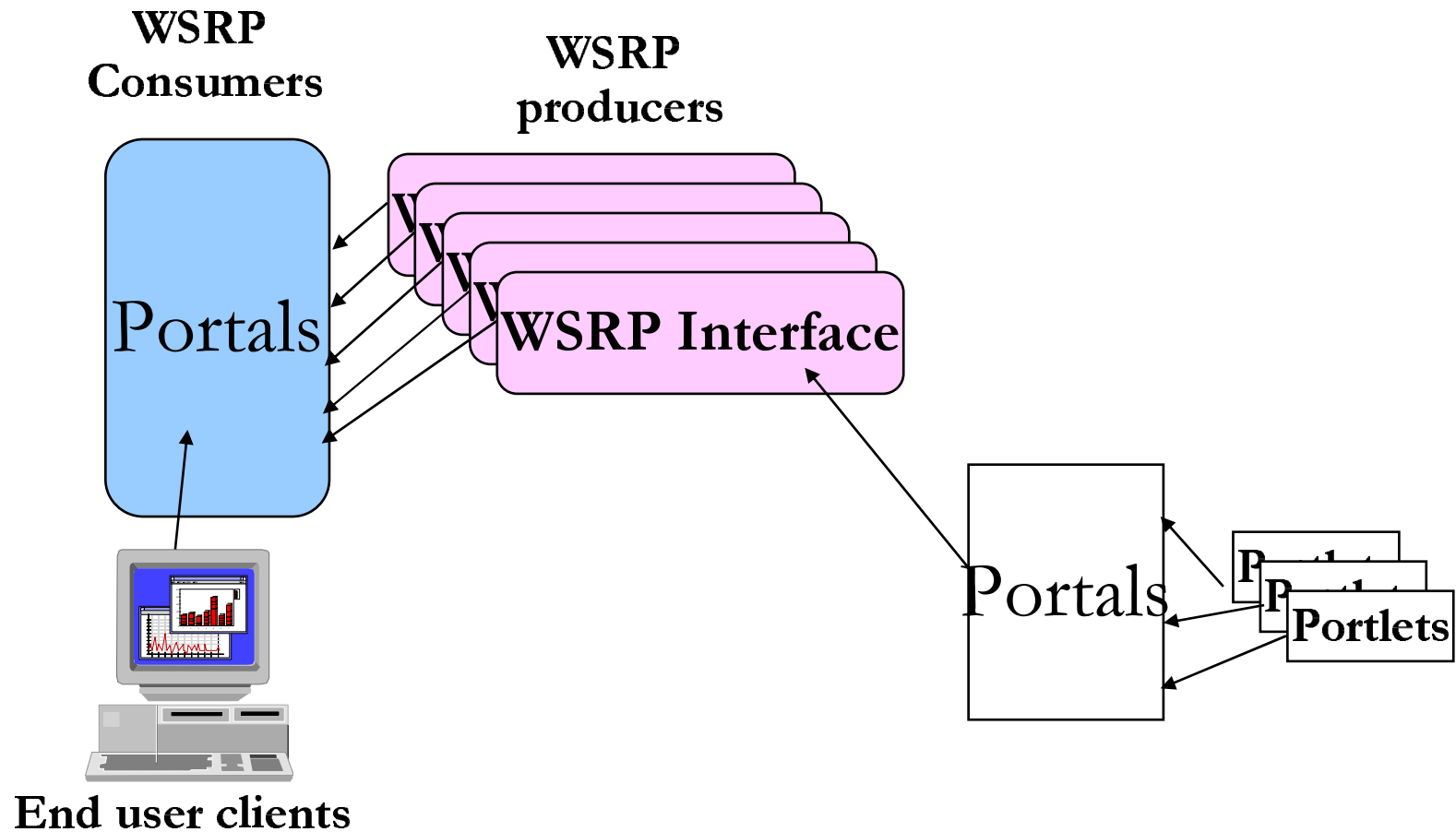


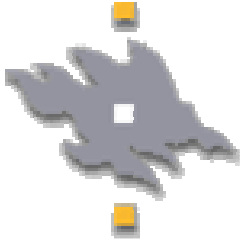
WSRP: Web Services for Remote Portlets

- This specification is joint efforts of 2 technical Committees
WSIA: Web Services for interactive Applications & WSRP:
Web Services for Remote Portals in OASIS
 - This specification integrates information from a wide set of content providers and application with as minimum efforts in order to define presentation-oriented interactive web services
 - WSRP is specified based on the WSDL.
 - The Services are usually data oriented which requires additional application logic to convert the unique interfaces based data results into presentation and interaction logic
 - The WSRP defines a presentation oriented web service interfaces which includes both the interaction logic and as well as the presentation logic and hence allowing a very integration of presentation oriented Web Services into portals.
- 



WSRP Actors

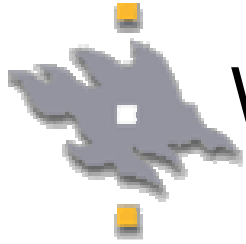




WSRP Interfaces

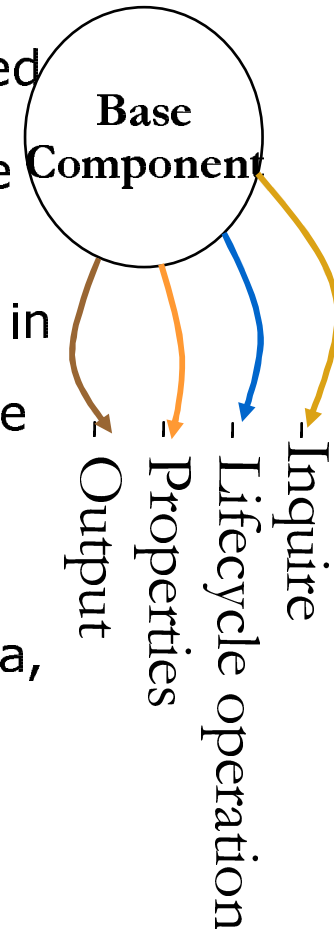
- Life-cycle operations
- Processing Actions and getting markup
- Interaction protocols such as order of operation invocations & caching

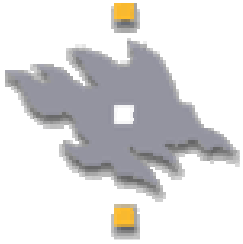




WSXL Web Services Experience Language

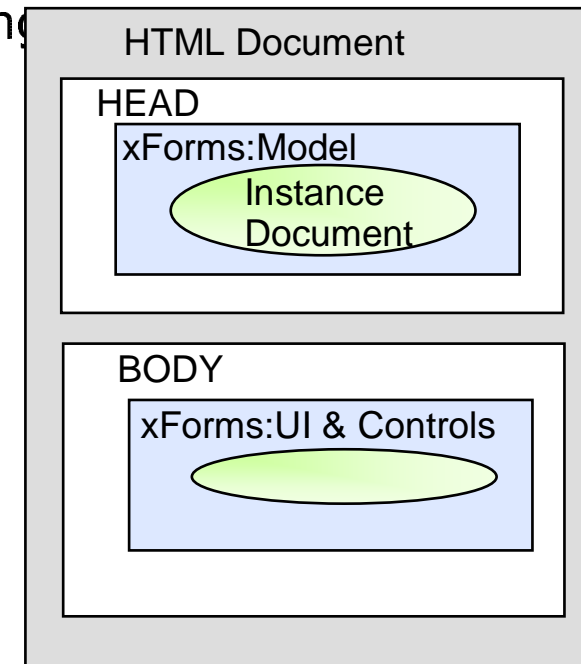
- Web Services Experience Language (WSXL) is defined based on the WSDL.
- WSXL provides specific portTypes / interfaces to handle life cycle management, accepting user input, and producing presentation markup
- It enables specification of Web Services based application in a Model-View- Controller architecture. Thus enabling re-assembly of multiple alternative versions of the web service components in order to meet the requirements of separate channels, users, and tasks
- WSXL's core is Base Component which provides interfaces(portTypes) to Inquire, Lifecycle operation and output generation. The rest of the components namely data, presentation and control implement this component.
- WSXL includes and extensible Adaption Description Language Further reading: <http://www-106.ibm.com/developerworks/library/ws-wsxl/>

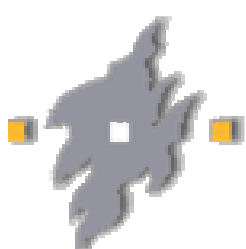


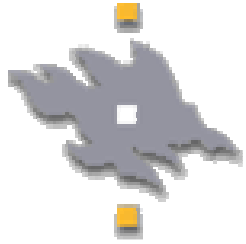


xForms a Brief Outline

- Forms :
 - “A Section of a document containing normal content, markup, special element controls (checkboxes, radio buttons, menus, etc) and labels on those controls. Users generally complete a form by modifying its controls (entering text, selecting menu items, etc.), before submitting the form to an Agent for processing (e.g, to a web server, to a mail server. Etc.)” [W3C]
- xForms:
 - Well formed XML based document representing a Form
 - Contains a Form Processing Model
 - Data-typing & Validation
 - Event handling
 - Support XML Schema
 - Submits XML to Services/Applications
 - Clear Separation of concerns (MVC Design Pattern)
 - Data Model
 - UI
 - Controls

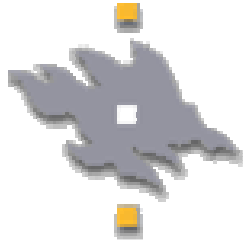






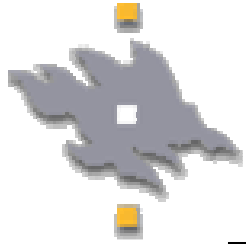
Service Middleware Features – Transaction

- This ensures that all activities in Web Services, meaning a set of message exchanges including the processing of the messages have been successfully executed or none are allowed to succeed.
- The set of service invocations together are considered to belong to one consistent behaviour and the success of their execution is guaranteed by the success of all other activities
- The effects of such set of executions is committed to the system only once all the activities have confirmed their execution



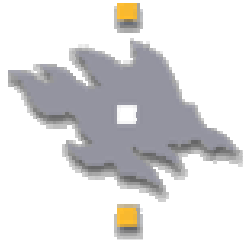
Service Middleware Features – Transaction [Contd]

- Transaction in Web Services are considered at two levels:
 - **Atomic Transactions:** is used to coordinate activities having a short duration and executed within limited trust domains. The transaction propagates the properties of “all or nothing”. In the case of atomic transaction there is a possibility to hold the resources until the commit as the transactions are short lived
 - **Business level transactions:** This is long lived transactions as the transactions happen across different business boundaries, i.e. in different business partners boundaries. The transactional aspects are handled differently in these environments as the resources can be held until completion of the transaction(due to long live). Instead a loosely coupled isolation of concerns and instead of locking compensational effects are carried out to roll back in case of failures.



Service Middleware Features – Reliability

- Two or more Web Services nodes or end points can communicate over different transportation protocols relying on the fact that the messages exchanged between them are Guaranteed and done so in predictable manner, even with the assumption that there could be network failures.
- There are two related approaches for ensuring Web Services Reliability:
 - **WS-Reliability:** Driven by OASIS (EbXML): 1.0 - 6th January 2004
 - **WS Reliable Messaging:** Driven by IBM, Bea, Microsoft, Tibco, - 13th March 2003
- These specifications provide a SOAP based protocols to ensure reliable messaging, guaranteed delivery with right message ordering, no duplicates



Service Middleware Features – Reliability [Contd..]

- The reliability aspects covered in these specifications span in addressing the following features:
 - Uniquely being able to identify the message, discard duplicates
 - Message order sequencing, Message persistence for recovery
 - Reliability applied to a group of message communications
 - Acknowledgements – on receipt & delivery of messages
 - Failure recovery, using levels of persistence as needed
 - Timeouts & Expiry
 - Re-transmissions
 - Provides HTTP based binding, but the value lies in not restricting the need for HTTP