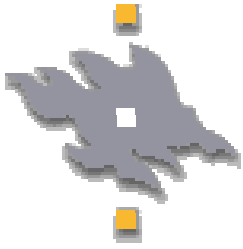




Lecture #7: 23rd February 2004

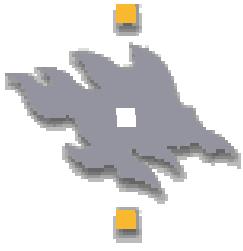
WSDL Continuation UDDI Web Services Workflow

Suresh Chande



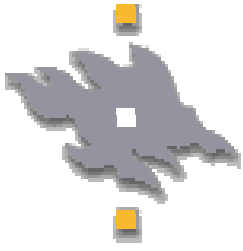
WSDL Continuation..

Software Engineering principles applied
to Web Services
Purpose: Seeds for research



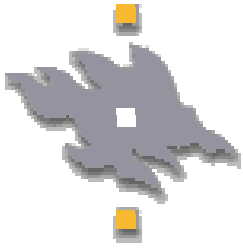
Software Engineering Principles

- Software Engineer Principles to modelling XML Schema based structure
- Object oriented design approaches to design Web Service Interfaces and implementation(MDA)
- UML based modelling of Service interfaces
- Sequence diagrams for defining the interaction with Web Service Interfaces
- Data Flow Diagrams to define the flow of data between multiple service interfaces



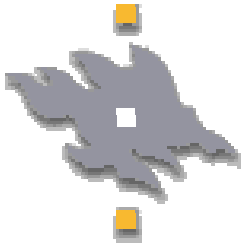
Software Engineering Principles

- Developing Design Patterns for Web Services deployment and specification
- Mapping of WSI to CORBA infrastructure
- Aspect oriented computing paradigm applied to Web Services composition ?
- Could there be Web Services Aspects based programming language to introduce aspects such as: routing, security, logging, etc?

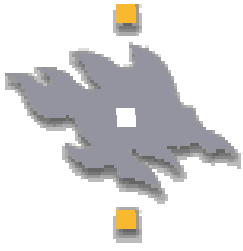


Future WSDL Applications

- Automation based on WSDL declarations
- Rich Service Discovery (Context, Service Mapping)
- Services 2 Service communication models(analogous to Peer 2 Peer models)
- How about application of WSDL based service discovery in proximity regions over bluetooth ?



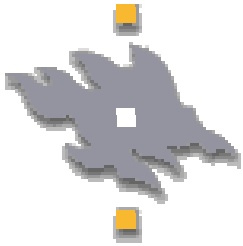
WSDL 2



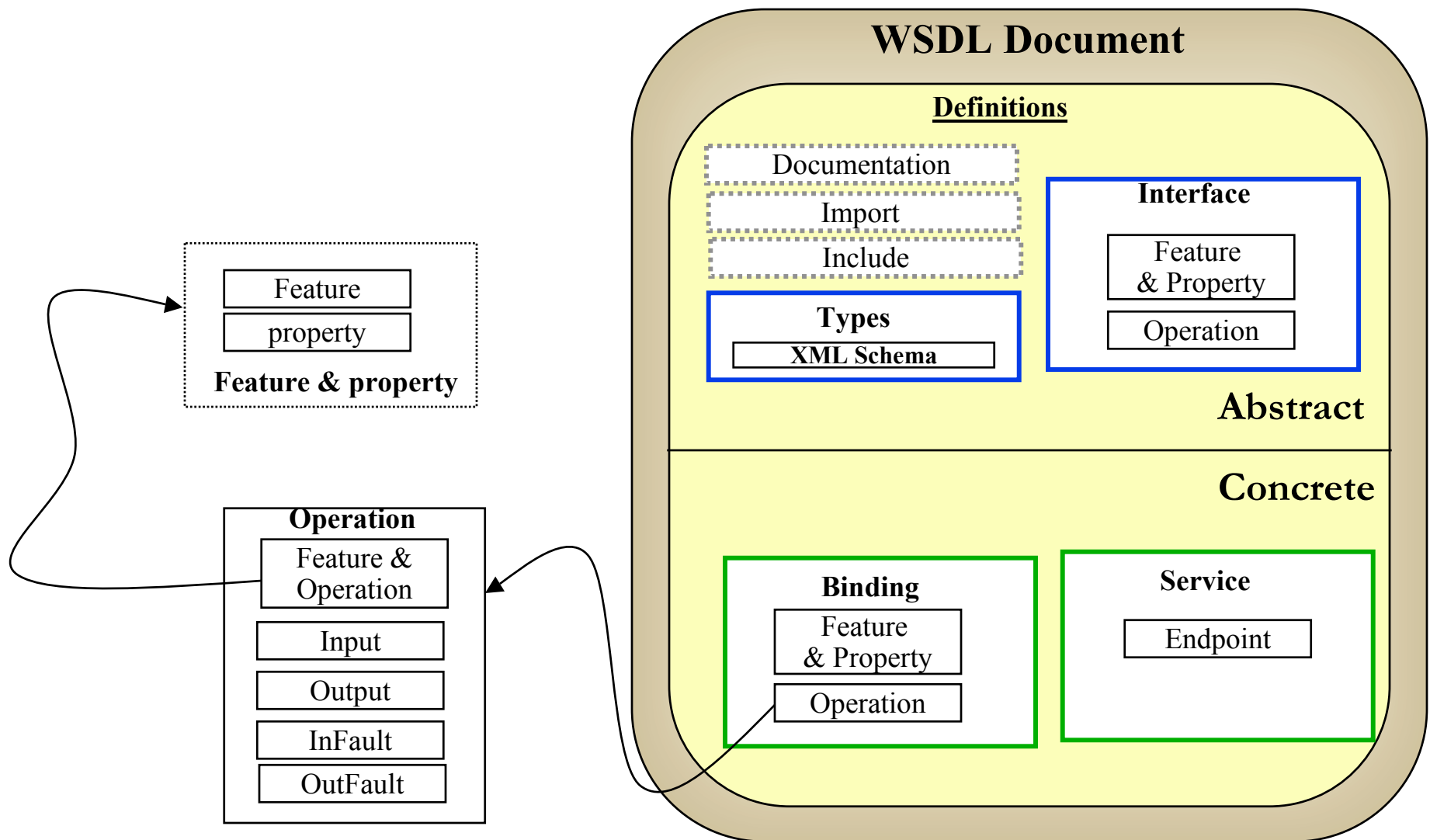
What's new with WSDL 2.0?

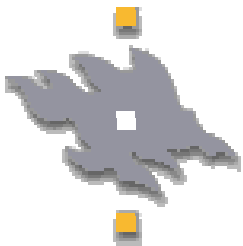
- New Namespaces
- portType changed to Interface
- **Operation has new elements:** Feature, Property, InFault and outFault
- Interfaces can extend from another interface, basically inheriting all previously defined operations and features set
- Message referencing
- Separation of Faults: inFault and outFault

- **Operation Styles:** RPC, Get-Attribute, Set-Attribute
- **Multiple Message Exchange Patterns:** In-Only, Robust In-Only, In-Out, In-Multi-Out?, Out-Only, Robust-OutOnly, Out-In, Asynchronous Out-In, Out-Multi-In?

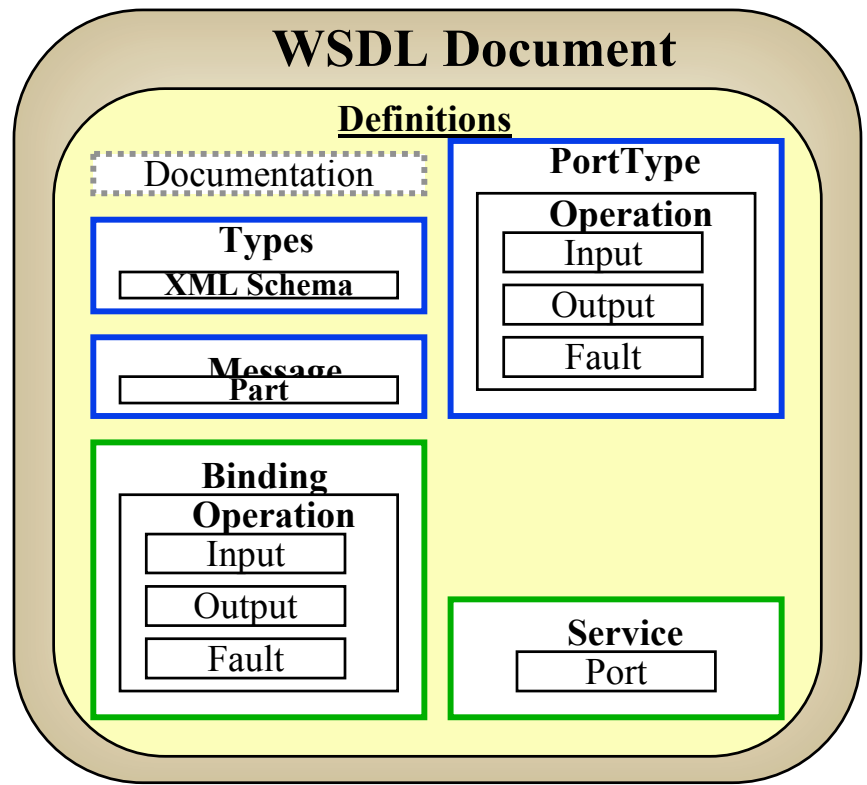
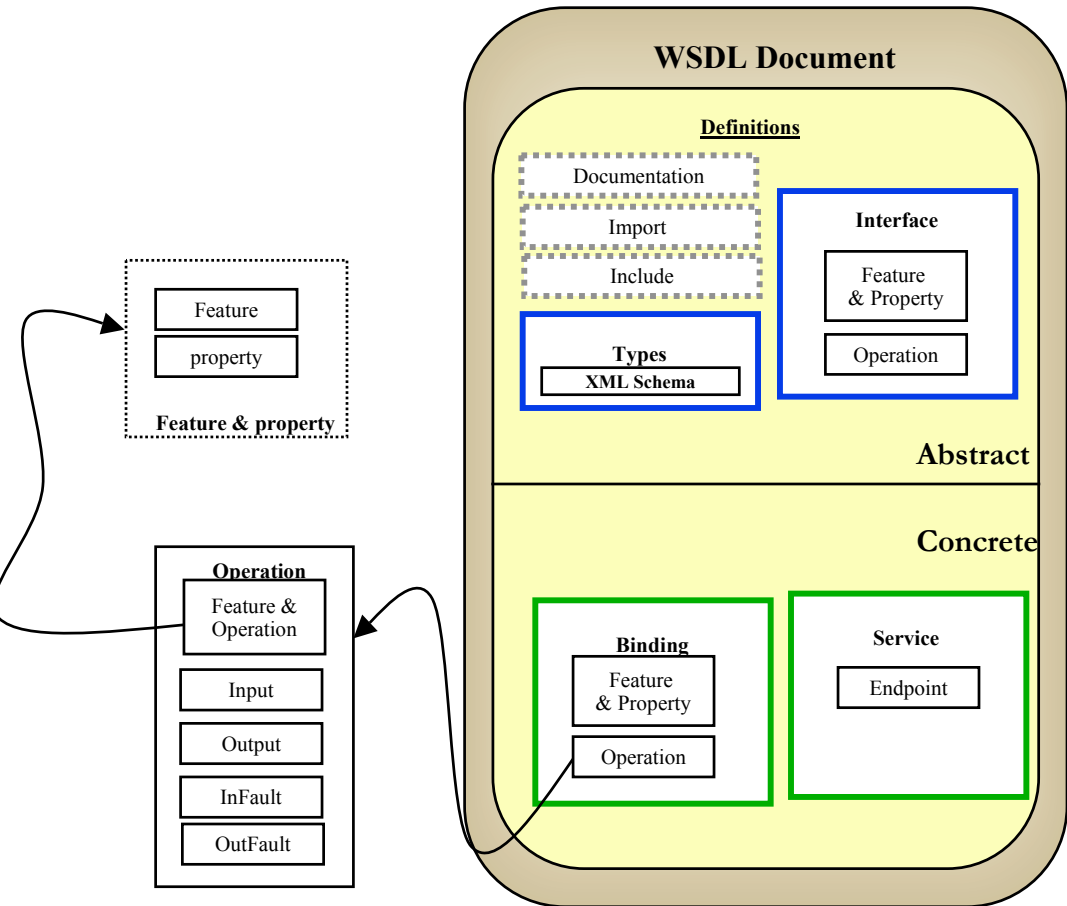


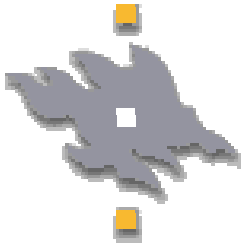
WSDL Document Structure





WSDL2 document Structures in comparison to WSDL1.1





WSDL namespaces used

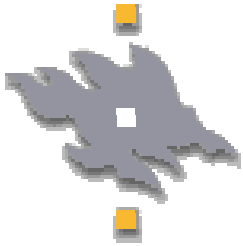
- WSDL Namespace: <http://www.w3.org/2003/11/wsd>
- Binding namespaces:
 - SOAP1.2 binding namespace:
<http://www.w3.org/2003/11/wsd/soap12>
 - HTTP binding namespace:
<http://www.w3.org/2003/11/wsd/http>
 - MIME Binding:
<http://www.w3.org/2003/11/wsd/mime>



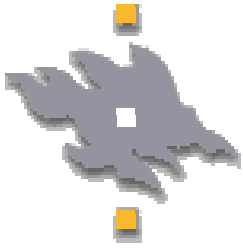
A look into the WSDL2.0 XML Schema



Wsd12.xml



UDDI



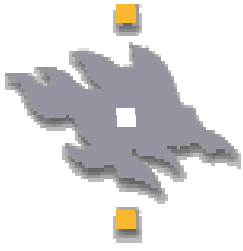
UDDI

- UDDI Stands for **U**niversal **D**escription, **D**iscovery & **I**ntegration: It enables businesses to quickly, easily, and dynamically find and transact with one another
- Web Services are being deployed today and UDDI today provides the best way to publish and discover these webservices, by providing a business registry.
- UDDI provides means to:
 - A). **Describe** a business and its services
 - B). **Discover** other businesses offering desired services
 - C). **Integrate** with these other businesses.



3 Components of UDDI business registry

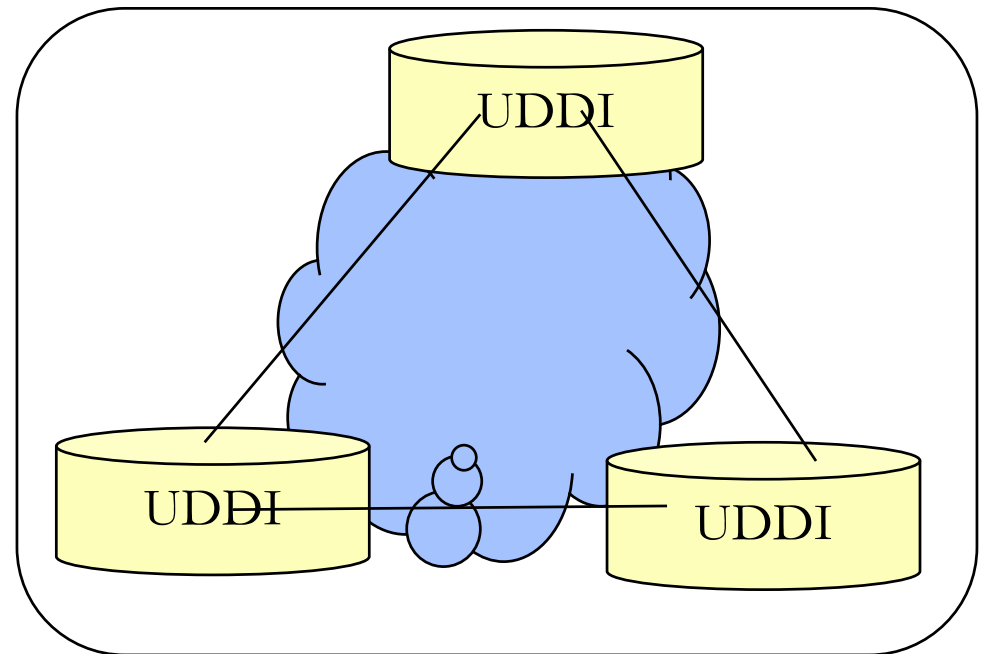
- **White Pages:** Company Contact information(name, contacts, human readable description and identifiers)
- **Yellow Pages:** Categorization of businesses using standard taxonomies (Industry codes, Geographic indexes)
- **Green Pages:** technical information about the services that are published(Service description, business rules, application invocations, data binding).

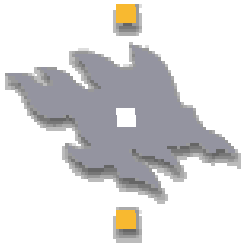


Logical Centralized Business Registry

UDDI registries are physically distributed, but replicate data amongst a set of partnered registries on a regular basis

Publishing in any one registry will be automatically replicate across multiple registries , hence providing a view of **Logically a single business registry**





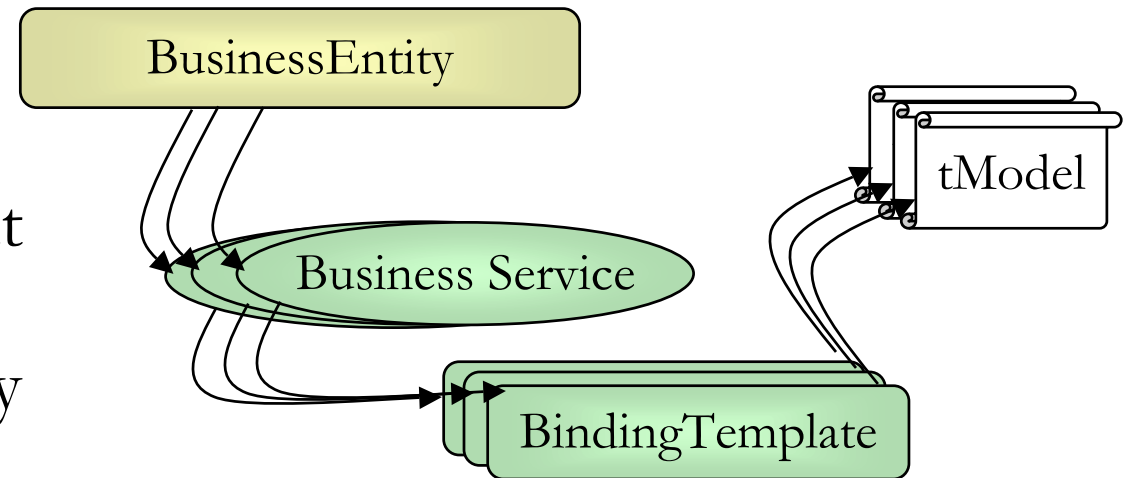
Four Basic Elements of UDDI Schema

UDDI utilizes XML Schema to describe information about services.

The namespace is depicted by URN: **urn:uddi-org:api_v2**

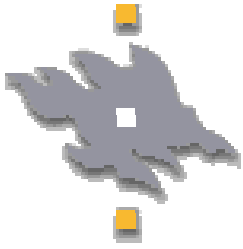
The Schema defining a specific Business services can be found here

http://uddi.org/schema/uddi_v2.xsd



The Schema consists of four major components:

- BusinessEntity
- BusinessServices
- BindingTempate
- tModel

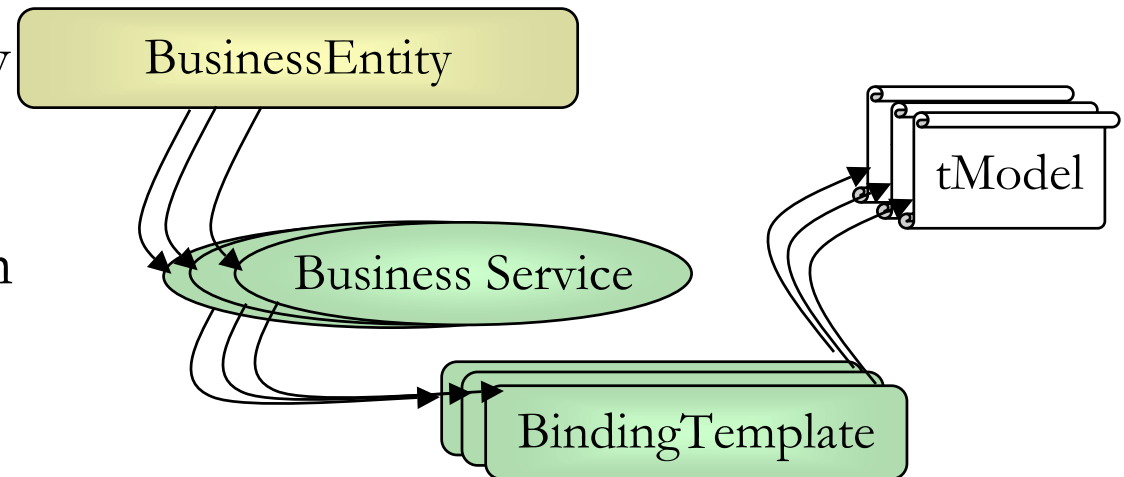


Four Basic Elements of UDDI Schema

BusinessEntity: Defines the yellow pages containing business name, identifiers, categorization, contacts. This forms the top level information for all details related to a particular business

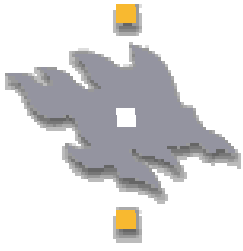
BusinessServices: A container in order to group a set of web services and categorized according to product, industry, service and geographical locations

BindingTemplate: Contains information to invoke a service. This contains references to



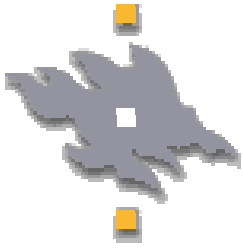
specifications the service complies to.

tModels: Metadata about specifications, which includes the name, publisher details and URL to the actual specification



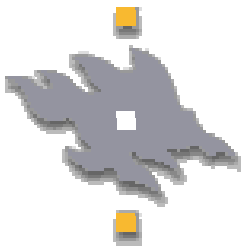
UDDI's provided APIs

- UDDI provides a SOAP based API to the business registry.
- UDDI provides two types of APIs, namely:
 - **Inquiry APIs** : Browse, Drilldown & Invocation patterns
 - Find & Get details of : binding, business, relatedBusiness, service, toModel
 - **Publish APIs** : Add, Get , Delete & Save(BusinessEntity, businessService, bindingTemplate and tModel), : publisherAssertions, Authtokens,

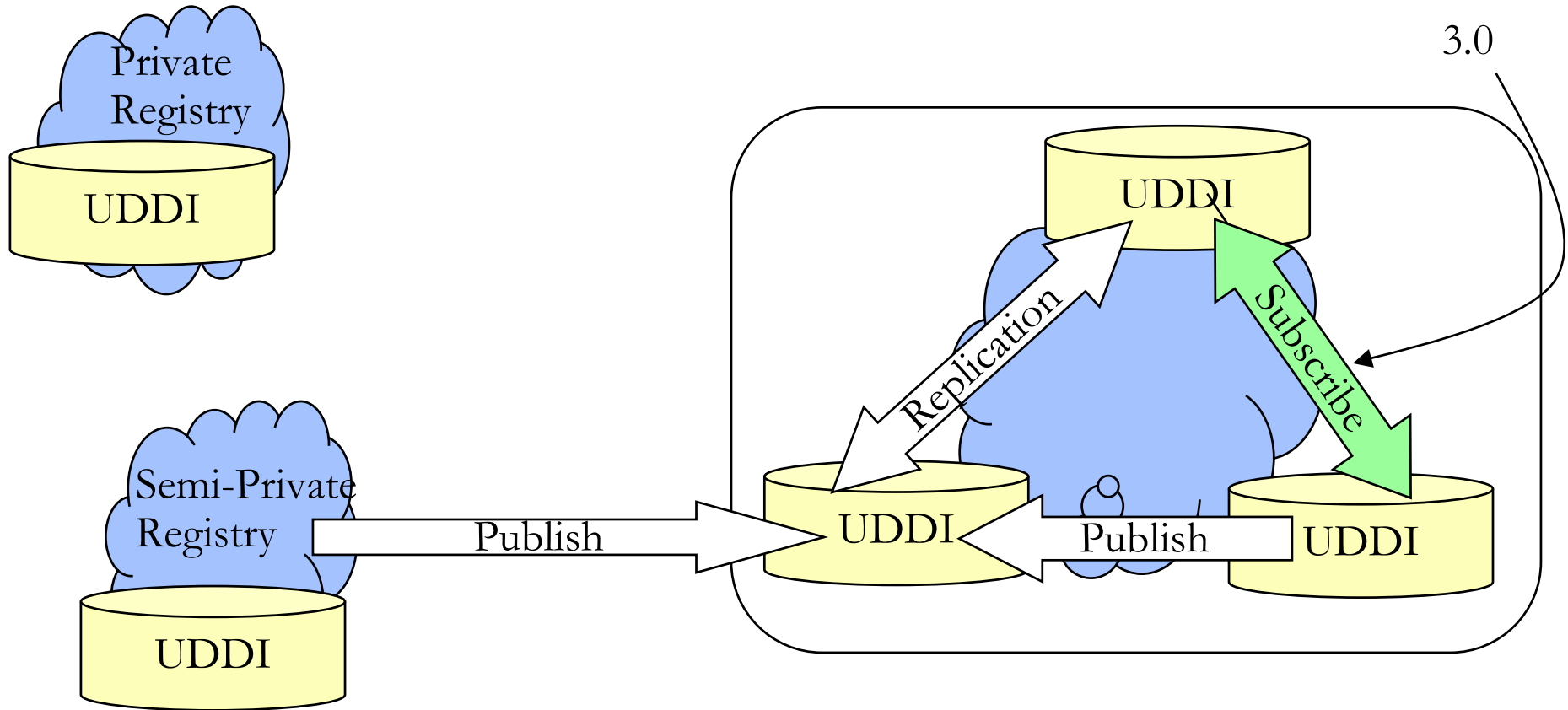


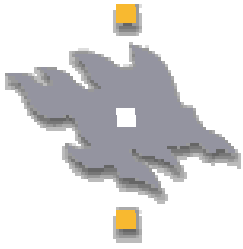
UDDI Security

- Publisher API are accessible to authenticated users
- Inquiry API are accessible to any user
- Changes to published information is only allowed to the creator of the original information.
- Each UDDI registry is allowed to create its own end user authentication mechanism



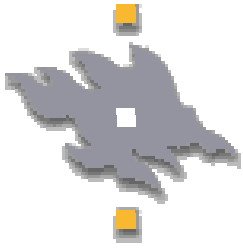
UDDI Deployment Models





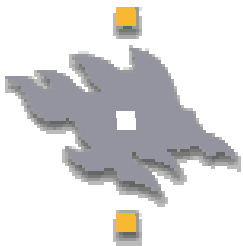
UDDI Evolution

- Microsoft Ariba and IBM released version 1.0
September 2000
- June 2001 UDDI version 2.0 released
- July 2002 UDDI version 3.0 released
- Universal Description, Discovery & Integration
originally developed by 3 industry players, IBM,
Microsoft & Ariba was submitted to OASIS in
June 2002 along with UDDI version 2.0 and 3.0
for standardization

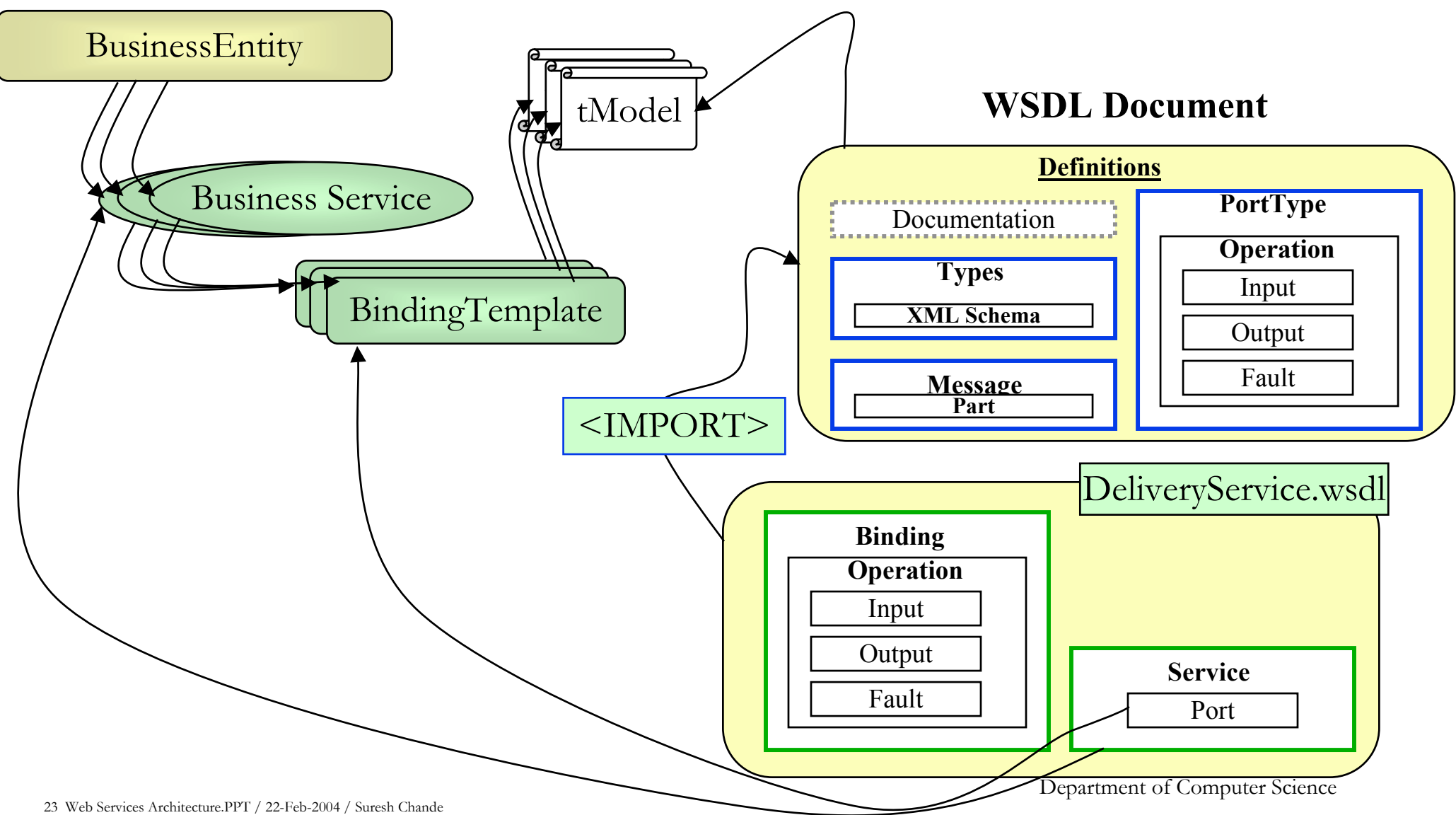


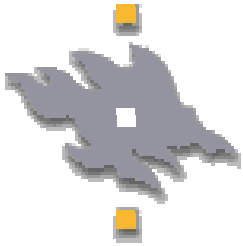
WSDL mapping to UDDI

- Considering the fact that the WSDL consists of two basic parts
 - The abstract reusable service interface definition
 - A Concrete binding of the abstract to a particular instance of the service interface
- In context with the UDDI only the abstract part of the WSDL is of immediate interest



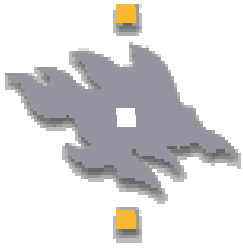
UDDI Mapping to WSDLs





Tmodel Example

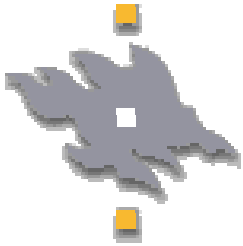
```
<tModel authorizedName="..." operator="..." tModelKey="...">
  <name>Notification Service</name>
  <description xml:lang="en">
    This is the WSDL description for a Delivery Web Service Interface
  </description>
  <overviewDoc>
    <description xml:lang="en">
      WSDL source document if needed
    </description>
    <overviewURL>
      http://www.mobile.services.org/services/notification.wsdl
    </overviewURL>
  </overviewDoc>
  <categoryBag>
    <keyedReference tModelKey=" uuid:q9Kfg46S-7639-7834-9838-48S2479Q93NE6 "
      keyName="uddi-org:types"
      keyValue="wsdlSpec"/>
  </categoryBag>
</tModel>
```



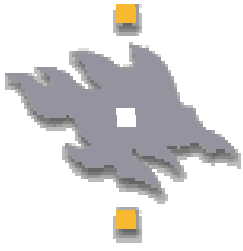
Business Service

```
<businessService businessKey="..." serviceKey="...">
  <name>NotificationService</name>
  <description> (...) </description>
  <bindingTemplates>
    <bindingTemplate>
      (...)
      <accessPoint urlType="http">
        http://www.mobile.services.org/services/notification
      </accessPoint>
      <tModelInstanceDetails>
        <tModelInstanceInfo tModelKey="...">

          </tModelInstanceInfo>
        <tModelInstanceDetails>
          </bindingTemplate>
        </bindingTemplates>
      </businessService>
```



Web Services Workflows



Workflows - Definitions

- **Business Process**

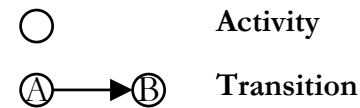
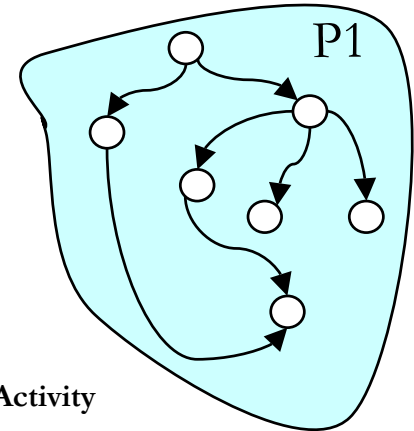
"A set of one or more linked procedures or activities which collectively realize a business objective or policy goal, normally within the context of an organization structure defining functional roles and relationships" [WfMC]

- **Workflow**

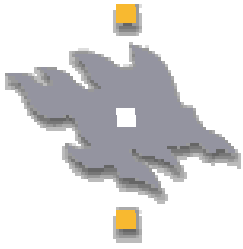
*"The automation of a **Business Process** in whole or a **part**, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules " [WfMC]*

Workflows - Definitions

- **Process Definition** is the description of all the steps involved in a single process.
- It consists of the *Activities* depicted as the nodes in the graph
- The connecting lines between the *Activities* are the *Transitions* from one activity to another
- The directed edge of the graph depicts the direction of the transition
- The transition of the execution from one *Activity* to another is guarded by a certain *TransitionCondition* being satisfied during the phase of execution of a particular process step

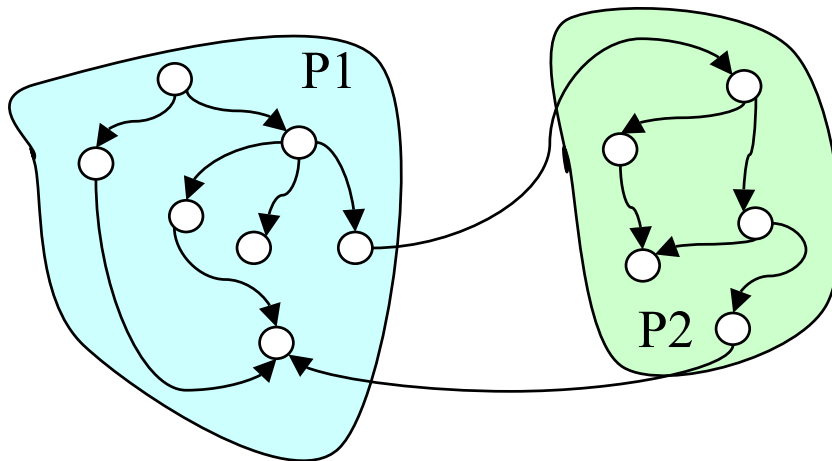


- The *Process* is normally represented by an acyclic graph
- If there are requirements for cyclic graph it is dealt with very carefully so as to have clear exit condition

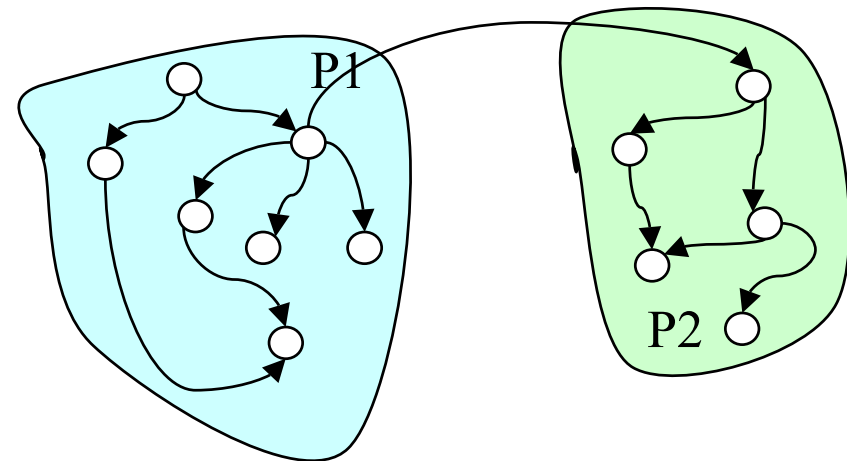


Workflows - Definitions

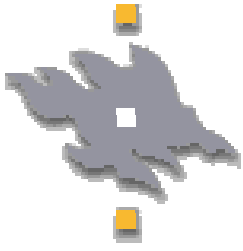
- A Process should be able to also define Sub Processes
- Two types of subProcess,
 - Blocking: Makes the parent process wait until the completion of the SubProcesses
 - Non blocking: where either of the process can complete independent of each other.



Blocking



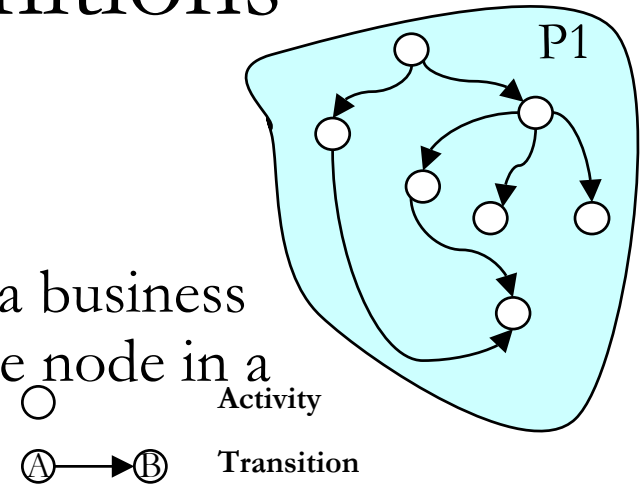
Non Blocking

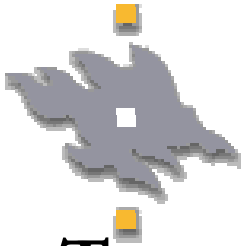


Workflows - Definitions

Activity:

- An *Activity* is a single step of execution within a business process definition and is usually denoted by the node in a graph.
- Two Types:
 - Atomic Activity can not be broken down further into smaller tasks
 - Complex Activity can contain few other Activities or a subProcess as depicted in the above figure
- Activity's implementation can be mapped to a single operation defined in a single WSDL portType description.

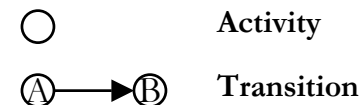
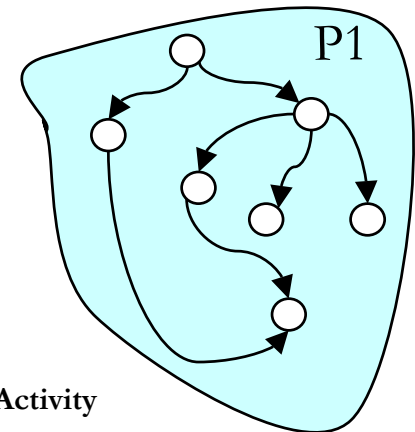


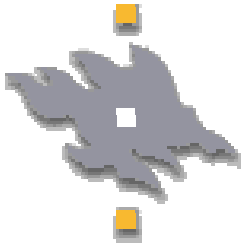


Workflows - Definitions

Transition:

- Transition is the flow of execution and data from one Activity to another.
- Transition is usually guarded by a condition and this is typically a logical condition which when satisfied will advance the process execution from one activity to another
- Transition condition is evaluated on various states of the process:
 - Once an Activity has completed its execution.
 - Once all the joining activities have finished and reached this part of the graph.
 - Once a set number of joining activities has completed their execution.

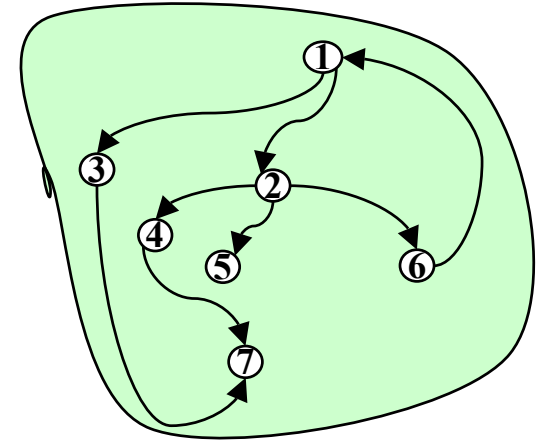


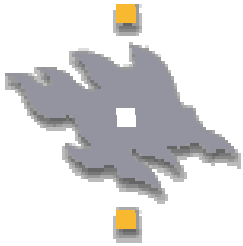


Workflows - Definitions

Transition:

- Sequential: Execution is transitioned to only one Activity after the currently executing Activity is completed, for e.g. 1->2->5
- Parallel: There could more than one activity simultaneously executed within the process execution, for e.g. Activity 1 is to Activities 2 & 3
- Conditional: Decides which activity executed is on the next transition based on pre-determined conditions, for e.g. After Activity 1 either Activity 2 or then Activity 3 is transitioned to.
- Loop: Execution to Activities is iteratively , with the Loop guarded by an Exit Condition, for e.g : 1->2->6->1->2 ->...

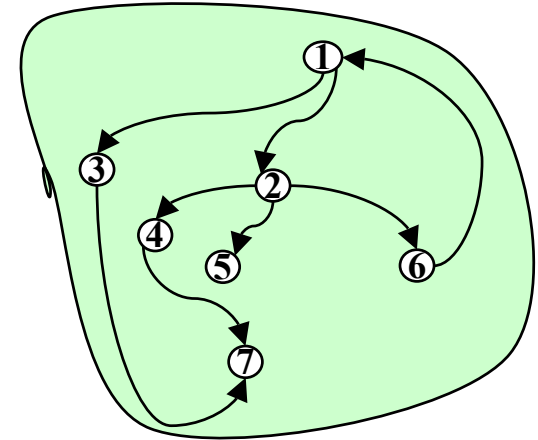


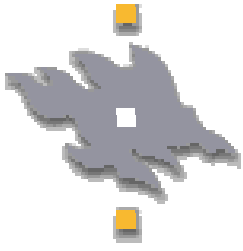


Workflows - Definitions

Exceptions:

- Handle the possible error or fault conditions that may occur during the execution of the complete process
- Exception handling could themselves be the execution of further activities to overcome the exception that has occurred :
 - Retry the similar activities
 - Perform a clean up operation to revert the state of the system so as to undo the affect of the Activities processed so far in the complete process
 - Notify the corresponding recipient of the failure to continue the process execution

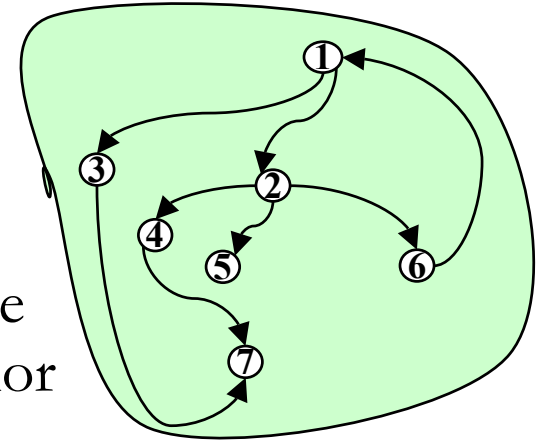




Workflows - Definitions

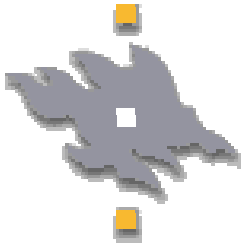
Transactions:

- Limited to a certain set of considered activities as a single unit of work and if either of the Activity fails the Transaction is to return the systems back to a state prior to the start of the execution of this unit of work.



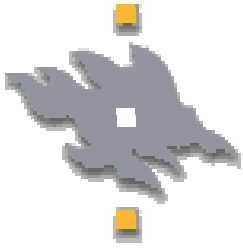
Compensations:

- Contains a set of activities, which will perform part of task, which will try to rollback the system as best possible to the original state prior to the start of the Transactions.



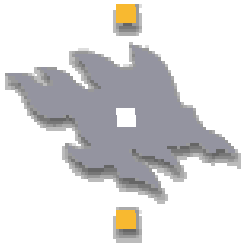
Workflows - Definitions

- **Property:** This functionality allows alter and refer the values of the property in context of the execution of the Business process
- **Transactions:** Provides the transactional support within a Process
- **Exception Handling:** Handling of error conditions during the execution of the Business Process Steps
- **Compensation Behaviour:** Enables compensate the affects of transaction once it has as successfully executed.
- **Fault Triggers:** Allows to raise a fault condition during the execution of the Business Process



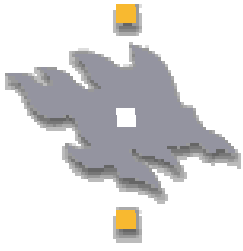
Workflows - Definitions

- **Multiple Service Interactions:** Enables the definition of the interconnection between the different Services (operations within as well)
- **Service Lookup:** Facilitates dynamic lookup to identify the next activity within the execution of the Business Process



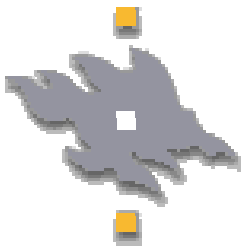
Limitations in Web Services

- Inter-relation between the Web Services The Service interfaces are exposed as end points for a set of services. There is no method to relate these interfaces to each other.
- Exception handling, Can there be re-tries?
- Can we describe compensations to be executed as a failure recovery; these can not be done with the current standards. They have to be explicitly coded within the client implementation. Here we are moving the complete logic to the client side implementations.
- The interfaces exposed to the client are too granular when we are considering SOAP/RPC oriented WS Exposure. How can we reduce this? Can we not provide a Composite Service, which will allow batch processing sort of functionality? This can not be done with the help of current standards.

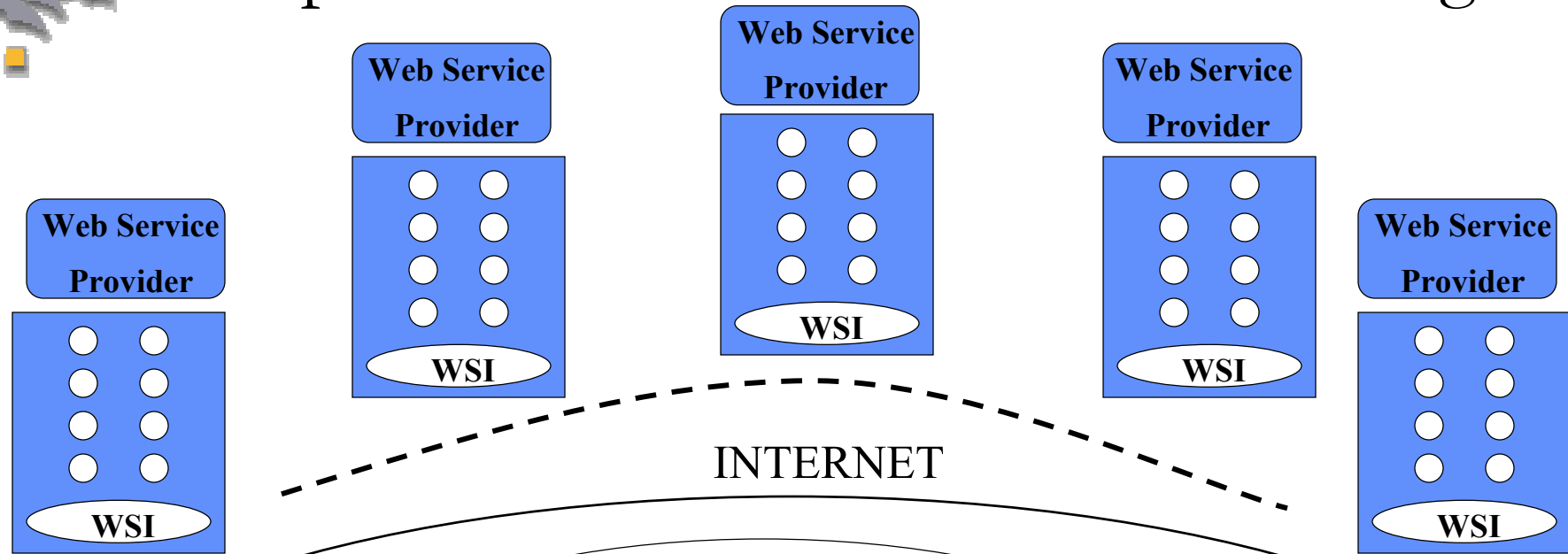


Limitations in Web Services

- Transactions, is not emphasized in the basic WS standards. There are proposals (BTP / WS-Transaction), but we see that they are being applied more in context to the Work flow system functionality.
- How can we provide value-added services on top of our current simple WS interfaces? There is not much provided by the current WS standards, the functionality such as Reliability / Guarantees can be built on top of this by providing a Workflow based Engine which can persist the communication within a context of service invocations and maintain the level of guarantees.



Gaps in the Core Web Service technologies



- Order of
- Mes
- In
- Au
- Status

Not all issues are required to be addressed for the non business critical services, Some are addressed by :

- Developer Guides
- Message Level Conventions
- Some are missing
- Some not required

Usage Model ?

(Site Services)

text management



Benefits Choreography adds to Web Services

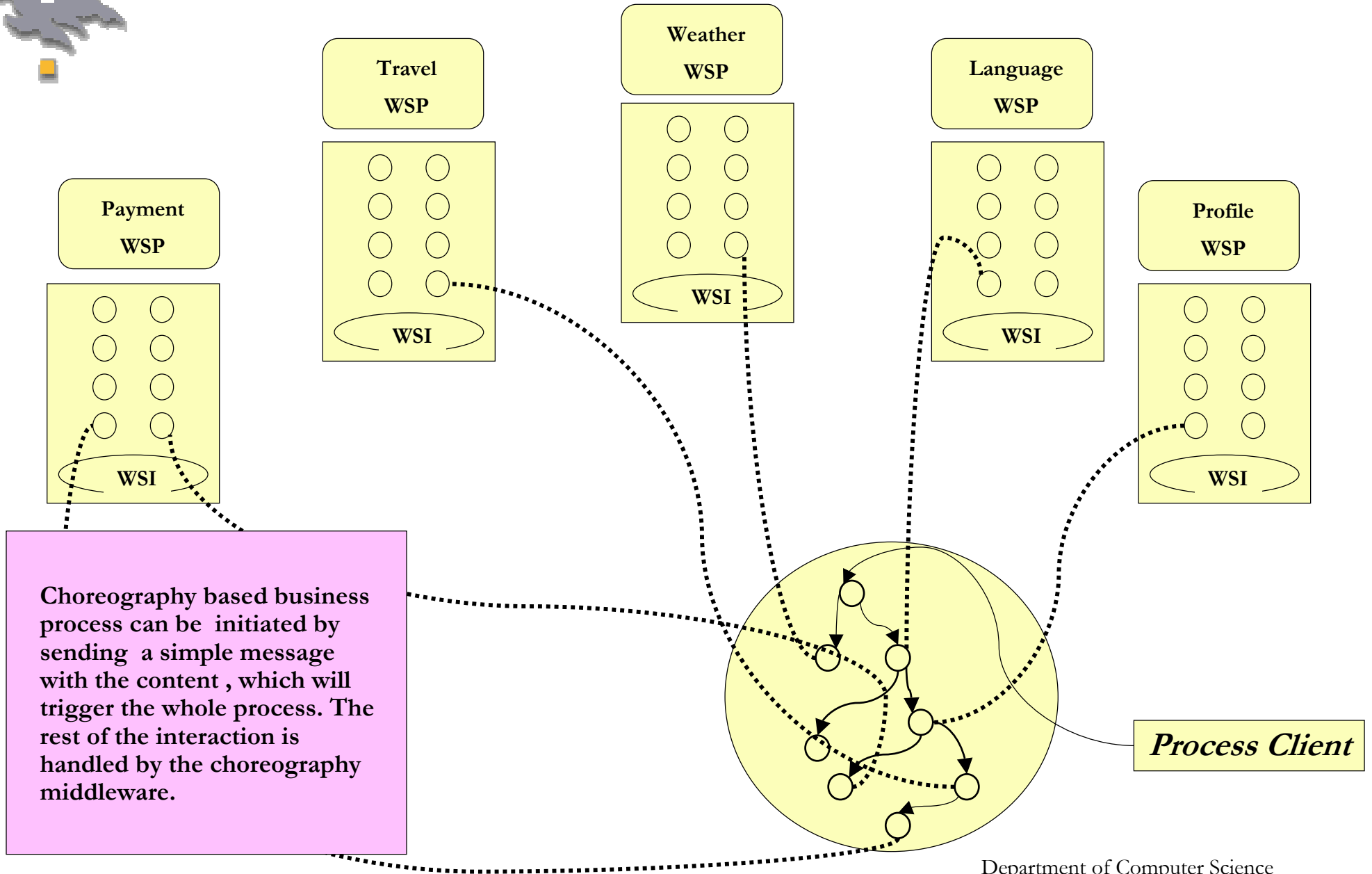
- Enables the exposure of a process oriented model to Web Service Interfaces
- Provide a simplified interface to a set of interfaces, Facilitate simplified packaging of a set of interfaces.
- WS with choreography defined can be better managed when compared to current WSIs, specially when there exists inter-dependent set of WSIs.
- Provides a clear usage model for a set of WSIs.
- Allows creation of composite services
- Enables specify design patterns depicting the best practices of using a set of WSI's
- Allows Provisioning of an execution environment which can guarantee execution of the complex/composite service invocation with compensations etc



Benefits Choreography adds to Web Services

- Provides a good environment for Long Running Middleware infrastructure. Where there can be pre-defined set of operations that need to be performed over a period of time (even scheduled). This will simplify the client interactions with the WSIs, but inturn needs to interact with the process to manage and monitor the status of such execution.
- *The most important factor is also that it helps define a coarse grained interfaces to the existing granular interfaces (RPC / Document style), reducing the interactions between the WSC and Choreography provider.*
- *Helps monitor that status of the request and even manage the lifecycle of the process*

An Example: Mobile Web Services Choreography





Workflows background - Standards

- **Workflow Management Coalition (WfMC)**
(285 Member consortium)
- **Business Process Management Language (BPML)**
(BPMI - Intalio)
- **Web Services Conversation Language (WSCL)**
(Hewlett Packard)
- **XLANG - used in BIZTalk (Obsolete by BPEL4WS)**
(Microsoft) - Used in BizTalk
- **Web Services Flow Language (WSFL) (Obsolete by BPEL4WS)**
(IBM) - Used in Web Sphere MQ Workflow Product
- **Business process Specification Schema (BPSS)**
(ebXML / OASIS)
- **Extended Distributed Object Computing (EDOC)**
(OMG)
- **Web Services Choreography Interface (WSCI)** ([BEA Systems](#), Intalio, SAP & Sun Microsystems)
- **§ (IBM, Microsoft, Seibel Systems, SAP & [BEA Systems](#))**
- OASIS WSBPEL TC
- **WS - Conversation Protocol**
(IBM) - JCA Implementation to work with J2EE servers
- **DAMLS / OWL-S - Defense Advanced Research Projects Agency (DARPA- Agent Markup Language, DAML+OIL, W3C,)**

Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS
Top Level Composition	<conversation>	<behavior>	<interface>	<package>	<definitions>	<process>	Process
High Level composition of the complete description.						This is separate document from the basic WSDLs	
Process	No	No	Yes	Yes	(Not directly)	Yes	Yes
A construct is available to represent a Business process as a whole	<conversation interactions> defines all the interactions in a conversation	<body> defines all the actions involved in a service	<process>	<Process>	Included into Definitions + flow model + global model	<process>	Process
<u>Activity</u>							
Atomic Activity	Yes	Yes	Yes	Yes	Yes	Yes	Yes
	<Interaction>	<action>			<Activity>	<invoke>	More than one element is considered as atomic.

Basic Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS ¹
Composite Activity	No	Yes	Yes	Yes	Not Directly <flowmodel> / <globalmodel >	Yes	
<u>TRANSITIONS</u>							
Sequential This functionality allows sequential execution of each of the steps involved within a process.	Yes <Transition> defines the next interaction from the source interaction	Yes <sequence>	Yes <call> / <sequence>	Yes <call> / <sequence>	<In-directly> Through the condition attribute in <controlLink> &t <dataLink>	<Sequence>	Sequence
Conditional This functionality allows a conditional execution of a set of steps involved within a process	Yes A condition guards the <Transition>	Yes Switch / Pick (Message Event Handler)	Yes <choice> (triggering events->message / timeout/ fault)/ <switch>	Yes <choice> / <switch>	<In-directly> Through the condition attribute in <controlLink> &t <dataLink>	<Switch>, <pick>,	Switch, pick, if-then-else, condition,
Parallel This functionality allows parallel execution of process or sub process within a process	No	No	Yes <spawn>/<join>	Yes <spawn>/<join>	Yes Through Life Operations <export> &t Join	<flow>	Split, split+join

Basic Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS ¹
<p>Loop</p> <p>This functionality allows iterative execution of each of the steps involved within a process</p>	No	Yes <while>	Yes <forEach> / <while> / <until>	Yes <forEach> / <while> / <until>	In a restricted fashion of only one activity. Later version will provide more features.	<While>	Repeat-While, repeat-until,
<p>Runtime Execution Functionality</p>							
<p>Context</p> <p>This functionality provides an execution context related each unique conversation between the client and the Business process</p>	No	Yes <context> Contains - correlation sets+port references	Yes <context>	Yes <context>	<Limited Support> Through the FlowSource & FlowSink	<scope>	No directr construct for this , but can maintain properties for a context.
<p>Transactions</p> <p>This functionality provides the transactional support within a Business Process</p>	No	Yes <transaction>	Yes <transaction>	Yes <transaction>	NO	REF- WS-Transaction	No

Basic Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS
<p>Exception Handling</p> <p>This functionality allows handling of error conditions during the execution of the Business Process Steps</p>	No	Yes	Yes	Yes	No	<throw>	Yes
		<exception>	<exception> onMessage, onTimeout, onFault	<exception> onMessage, onTimeout, onFault			
<p>Fault Triggers</p> <p>This functionality allows to raise a fault condition during the execution of the Business Process</p>	No	No	Yes	Yes	No	<FaultHandlers>	NO
			<fault>	<fault>	The lifecycle operations have a means to report fault conditions using <fault>		
<p>Compensation Behaviour</p> <p>This functionality enables compensate the affects of transaction once it has as successfully executed.</p>	No	Yes	Yes	Yes	No	<compensationHandler>	Only --> Indirectly
		<compensation>	<compensation> /<compensate>	<compensation> /<compensate>			

Basic Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS
<p>Properties</p> <p>This functionality allows to maintain properties within context of execution of a Business Process</p>	No	Yes <PropertyDef>		<property>	No Partial functionality by FlowSource \ FlowSink	Not defined as properties but we have Data Containers to hold data for the process manipulation. <containers> <property>	Yes
<p>Correlation</p> <p>This functionality allows correlation between the messages exchanged within the execution of a business process</p>	No	Yes <correlation>	Yes <correlation> /<correlate>	No	No	<correlations > <coorelation >	NO, Can be enabled indirectly
<p>Life Cycle Operations</p> <p>This functionality allows monitor and manage the execution of the Business Process</p>	No	No	No	Yes XPath Functions	Yes <Operation> under <flowmodel>	<wait>, <terminate>, <receive> IN FUTURE-> suspend,resu me,Query,	Yes Monitor, control

Basic Functionality	WSCL	XLANG	WSCI	BPML	WSFL	BPEL4WS	DAMLS
Dynamic Functionality							
<p>Mapping of Data sets</p> <p>This functionality enables mapping of the message construct exchanged between the services operations.</p>	No	No	No	No	Yes <map>		NO
<p>Multiple Service Interactions</p> <p>This functionality enables the definition of the interconnection between the different Services (operations within as well)</p>	No But can be achieved Indirectly (statically defined)	No <Contract> provides static definition of the interconnection between services	Yes <connector>	Yes (via WSCI) <connect>	Yes <plugLink>	Service link <serviceLinkType> <partners>	Yes
<p>Service Lookup</p> <p>This functionality dynamically lookup to identify the next activity within the execution of the Business Process</p>	No	No	Yes <locate> / <locator>	Yes (Via WSCI) <locator>	Yes Both static & dynamic using <locator>		No But can be done indirectly using the external systems.
<p>Property Settings</p> <p>This functionality allows alter the values of the property in context of the execution of the Business process</p>	No	No	Yes Selector - XPATH	Yes <Assign>	No	Yes <assign>, <copy>	Not directly..