

# IS660G - Lecture 3

Prof. Burns

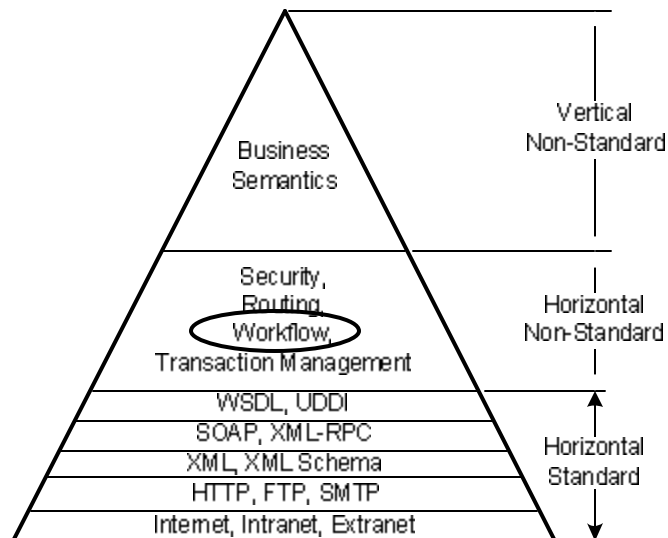
## Agenda

- Web Service flow languages, execution languages, orchestration languages, web-enabled workflow languages
  - Chapter 18 – WSFL
  - Others
- Team Project

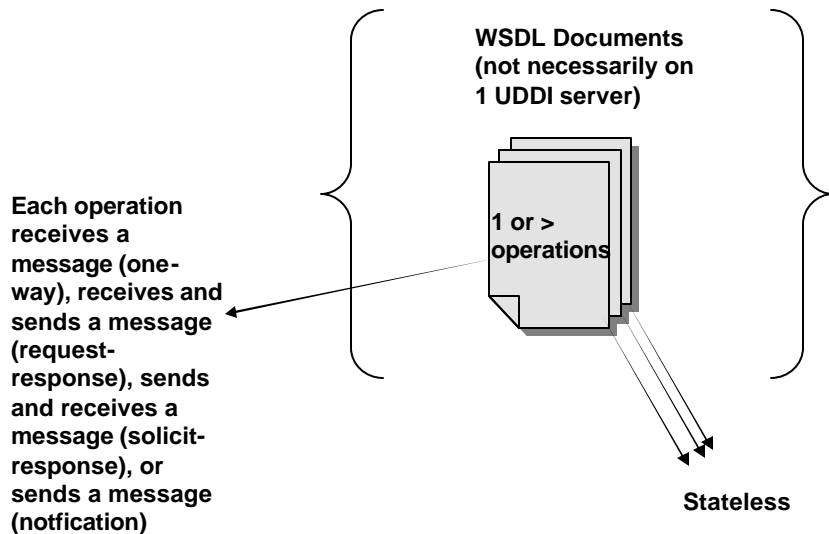
## Why are these languages compelling?

- There are 2 trends converging in E-Commerce that are creating opportunities and pressures to automate business processes:
  - XML-based standards and the Internet
  - Need to improve efficiency of processes across enterprise boundaries
- The goal of web services is to exploit XML technology and the internet to integrate applications that can be published, located and invoked over the web.
- i.e. Galileo system connects > 42,000 travel agencies to 37 car rental companies, 47,000 hotels, and 350 tour operators.
  - Requires long-running interactions that are driven by an explicit model

## The Web-Services Pyramid



Taken as a whole ...these are composed services which may/usually require long-lived transactions



## Web Services Interactions

- Simple – stateless – one peer acts as the server and the other acts as the client
  - WSDL and SOAP suffice
- Complex – peer to peer interactions involving multiple invocations flowing between 2 or more services to achieve a desired business goal.
  - Flows – order of operations
    - Public
    - Private
  - Orchestration – single point of control

# Flow Models

Describes a usage pattern of a collection of web services so that composition of those services provides the functionality needed to achieve a business goal → “Business process”

- Steps in the flow (business tasks)
- Order (control flow)
- Conditional logic
- Data flow

Figure 18.1 on page 480

WSFL Flow  
Engine

In WSFL:

- Activities
- Control Links
- Data Links

More on workflow patterns can be found at  
<http://tmitwww.tm.tue.nl/research/patterns/>

## WSFL Activities

- An activity is a single step in the flow
- Describes “how” an operation (the actual implementation) fits into the flow
  - Operation can involve a human or a piece of code.
- WSFL code on page 481

## WSFL Control Links

- Describes the execution order of individual activities/tasks and the conditional logic (logical dependency between 2 tasks) specifying whether they should be performed or not.
  - $A_1$  must be completed before  $A_2$  can start

## Activities + Control Links

- Directed, acyclic (chain structure) graph
- Figure 18.2
- Conditions
  - Transition
  - Join

## WSFL Data Links

- How the output message of a given business task is used by (input) one or more subsequent business tasks
- $A_2$  uses the result data of  $A_1$
- Transformation (map) may be necessary if the input message type is different from the output message type
- Code snippet on page 484
  - Note: Data flow is parallel to the Control Flow

***Now let's discuss the realization of  
the flow's activities .... as  
Compositions of Web Services***

# Service Providers

The service providers with which the flow interacts is separated from the binding of those service providers to the actual business partners implementing those services. See Figure 18.4 page 486

- serviceProvider declaration in the flow itself
- serviceProviderType attribute links to the portType element
  - portType
    - operation/s

Why? Based on a UDDI lookup and some established criteria (i.e. quality of service), we might want to bind to the most desirable service provider, from a list of potential providers, dynamically rather than statically. Static implies using the same business partner all of the time.

- locator elements – locating the actual business partner - see page 488.
  - Type local
  - Type mobile

***Now let's discuss exposing the flow as a service provider itself so others can use it ...***

## Flow Model as a Service Provider

- Figure 18.5, page 490 – interface to which the flow model's clients connect; 4 operations (1 inbound, 3 outbound) depicted as arrows
  - Setup portType and serviceProviderType
  - Use external instead of internal

## Lifecycle Operations, Flow I/O

- Many instances of a given flow can execute at the same time
  - Call – WSDL request-response operation with the input of the flow as the input message and the output of the flow as the output message
  - Spawn – WSDL one-way operation with the input of the flow as the input message
- Additional operations: suspend, resume, inquire, terminate



## Recursive Composition

- Recall – a flow defines a composition of services by specifying a usage pattern of its service providers' functionality
- A flow model can become a service itself, offering a set of WSDL interfaces to possible clients; therefore the flow could become part of yet another service composition itself.
- The result is a mechanism by which services can be recursively composed following a **flow composition model**.
- The executing engine is the single point of control

## Global Models

- Used to specify multiparty transactions
- Plug Links
- Figure 18.7 – 2 parties
- Figure 18.8 – 3 parties

## ***Beyond WSFL ...***

### **IBM's WSFL**

MQ Series

**BPML** -  
*<http://www.bpmi.org/>*

**ebXML's  
BPSS(OASIS and  
UN/CEFACT)**

### **MicroSoft's XLang**

Biztalk

**WSCI** (from  
*BEA, Intalio,  
SAP, and Sun*).

**XPDL (WfMC)**

**BPEL4WS** (not a standard, yet)

*Do the proposed standards support a variety of complex, stateful, asynchronous, etc., business processes interactions? We'll hear more on this next class from Team 2.*

*Is there a consensus on the workflow constructs that need to be supported and their semantics? i.e. Many interpretations of Joint constructs.*

## Evaluation based on competing standards

<http://tmitwww.tm.tue.nl/research/patterns/standards.htm>

## Team Projects

Team 1 – Bldg an application – Ardian, Lisa, Andrew, Arvinder

Team 2 - Transactional Web Services – Ian, Simon, Zheng, Pratima

Team 3 – Bldg an application – Muhammad, Patricia, Larry

Team 4 – Bldg an application – Piotr, Julian, Smitha, Igor, Rosa (one too many someone switch to Team 3)

Team 5 – Web Security – Prashant, Michelle, Michael