Lesson 8 – Process languages (Part I)

Service Oriented Architectures Security

Module 1 - Basic technologies

Unit 5 – BPEL

Ernesto Damiani

Università di Milano

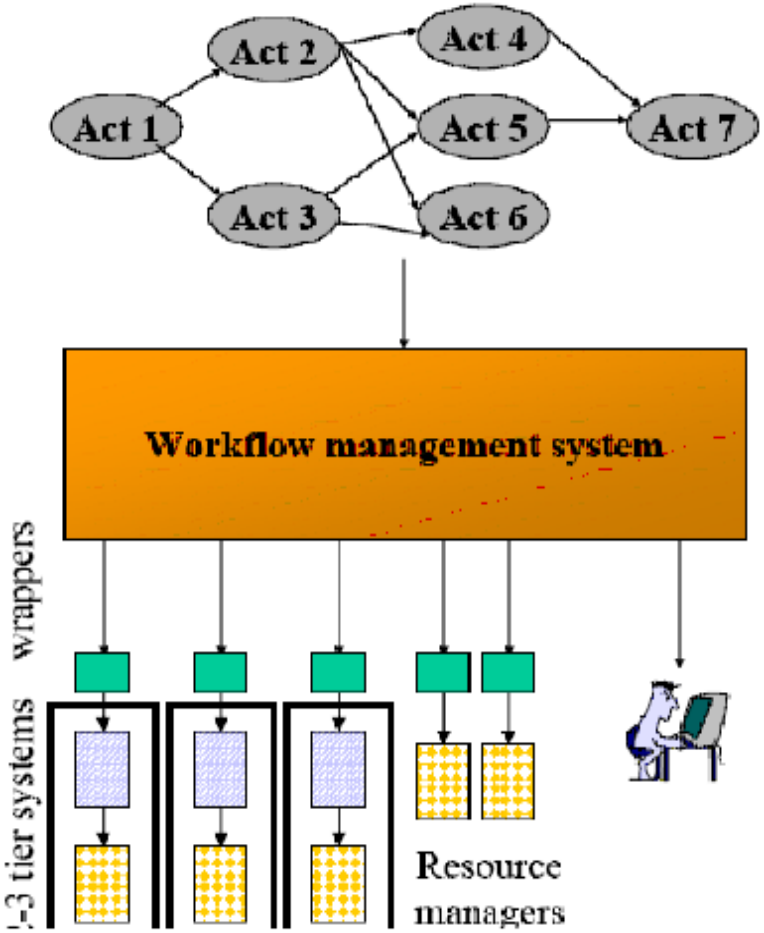
Business processes (1)

- A business process describes key procedures within an organization and these procedures involve:
  - multiple steps
  - numerous persons
  - large amounts of resources

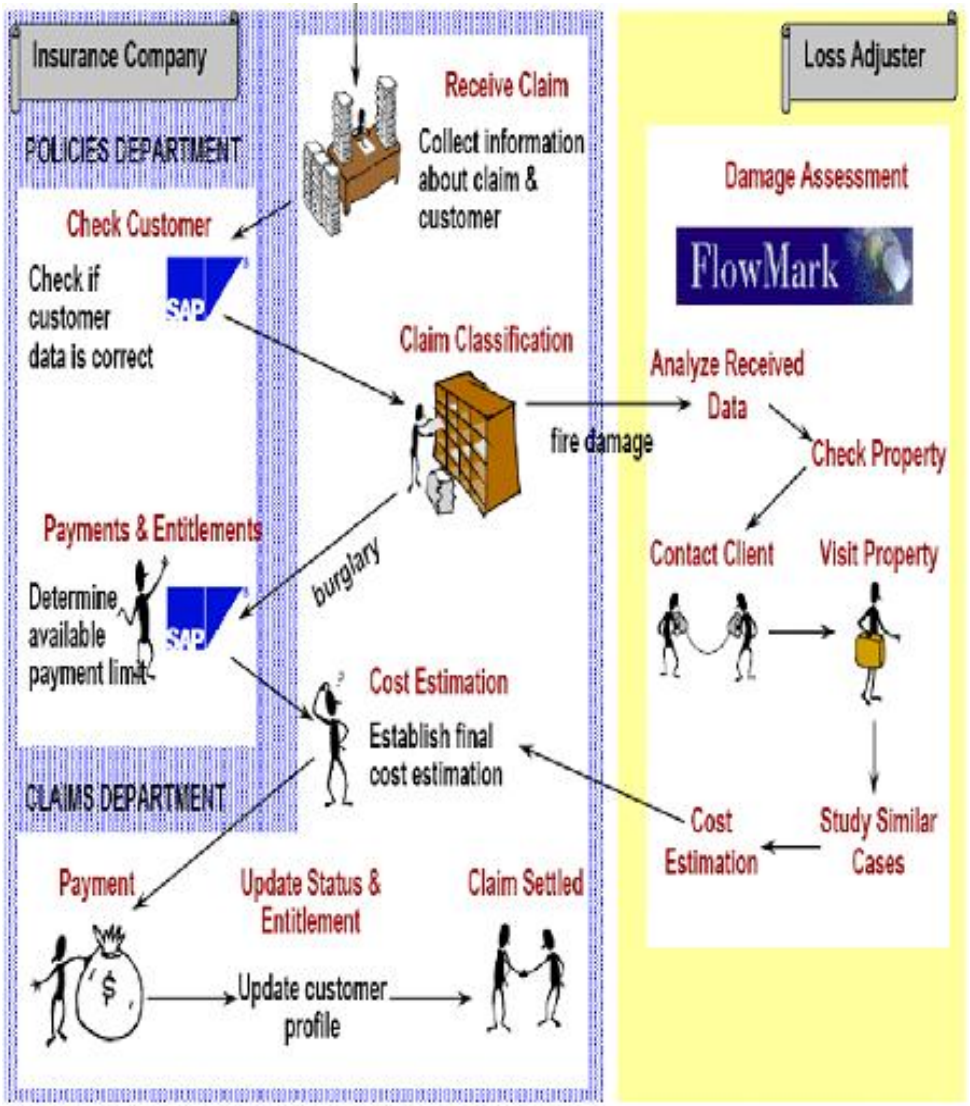
Business processes (2)

- In large corporations there are many factors increasing the complexity of the business processes:
  - multiple people lack information
  - conformance to rules not guaranteed
  - processes are not well documented
  - company lacks monitoring tools
  - steps, people and resources are not properly coordinated
- Workflow Management Systems try to address these problems by automating the coordination aspects of a business process, that is, who has to do what, when, and with which tools

Business processes (3)



An example


Insurance Company
POLICIES DEPARTMENT
Receive Claim
Collect information about claim & customer
Check Customer
Check if customer data is correct
SAP
Claim Classification
fire damage
burglary
Payments & Entitlements
Determine available payment limit
SAP
Cost Estimation
Establish final cost estimation
CLAIMS DEPARTMENT
Payment
Update Status & Entitlement
Claim Settled
Update customer profile
Loss Adjuster
Damage Assessment
FlowMark
Analyze Received Data
Check Property
Contact Client
Visit Property
Study Similar Cases
Cost Estimation

Goals of workflow (1)

• The complexity of many business processes makes them a key element in a company's ability to evolve. Business Process Reengineering is a way to define, better understand, and optimize these processes:

- enhancing processing capabilities (1.5 million/invoices year)
- reducing overhead
- decreasing processing time (7 days to 4 hours) and cutting processing cost

Goals of workflow (2)

- Processes that are already implemented with traditional middleware tools are very difficult to understand, let alone modify. A process can be seen as a program: it needs to be well documented, in a suitable format, and have a consistent life cycle
- For processes that are not implemented in software, the reengineering effort often leads to attempts to implement them. This is where workflow technology plays an important role

Goals of workflow (3)

- Today, many of the individual activities within a business process are performed directly or indirectly with computers
- As we rely more and more on computers, every PC and workstation in which individual activities take place become an isolated information repository where part of the business process resides. Each of those computers is an island of information

Goals of workflow (4)

- But those computers cannot easily talk to each other and it is not straightforward to gather all the information stored in each island
- Under these circumstances, it is very difficult to get a global view of what is taking place: monitoring, auditing, data gathering, etc.
- All the information islands must be connected together to build a single system in which business processes exist as physical concepts and not as abstract entities

Goals of B2B (1)

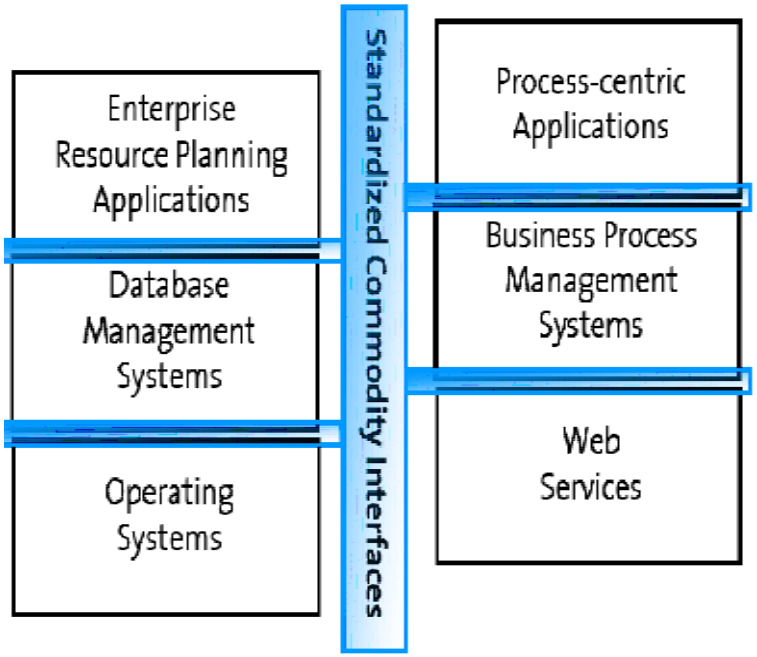
- The goals of B2B or e-commerce are similar to those of workflow:
  - reduce the cost of transactions
  - provide a high level view of the IT operations
  - map IT to business processes
  - reduce overhead and time in performing operations

Goals of B2B (2)

- The biggest obstacle for workflow was heterogeneity in the applications
  - This obstacle goes away to a great extent through the use of web services
- Workflows, or processes, seem to be the best language for specifying at a high level how a collection of services should interact
- Workflows are often seen as the language of Service Oriented Architectures

Why standardize composition?

- Portability and Interoperability of business process models defining executable compositions and abstract coordination protocols



Value added services (1)

- Once basic services begin to be published on the Web, the natural thing to do is to find and combine them into more complex, value added services
- Assuming that the necessary basic services are available, it is possible to define the business logic of an application out of their composition. Different applications can be built by composing the same services in different ways
  - Composite services are still services, i.e., services that can be composed: composition is recursive

Value added services (2)

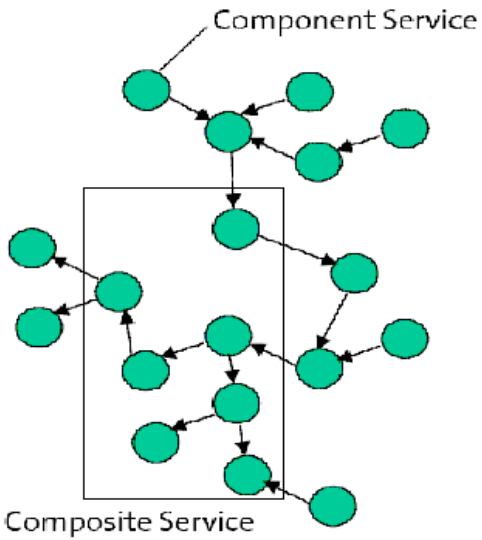
- Web service composition provides abstractions for high level modeling of compositions and the infrastructure for executing them efficiently
  - Examples:
    - Look for the cheapest price for a book/movie/song on all Internet e-stores and buy it, ensuring that it will be shipped before your birthday
    - Search from different Web search engines, merge all results by removing duplicates
    - Plan for a vacation: order plane tickets, reserve hotels and cars for a set of destinations
    - Convert stock quote prices to a different currency
    - Get weather information and snow & avalanches reports for the nearest ski resorts
    - Gather bids from suppliers, select the winner after a deadline and notify all participants of the outcome

Composition and coordination

- Composition and coordination can be seen as two faces of the same problem
- Composition models the internal structure and implementation of a service
- Coordination protocols focus on the external interactions of a set of services

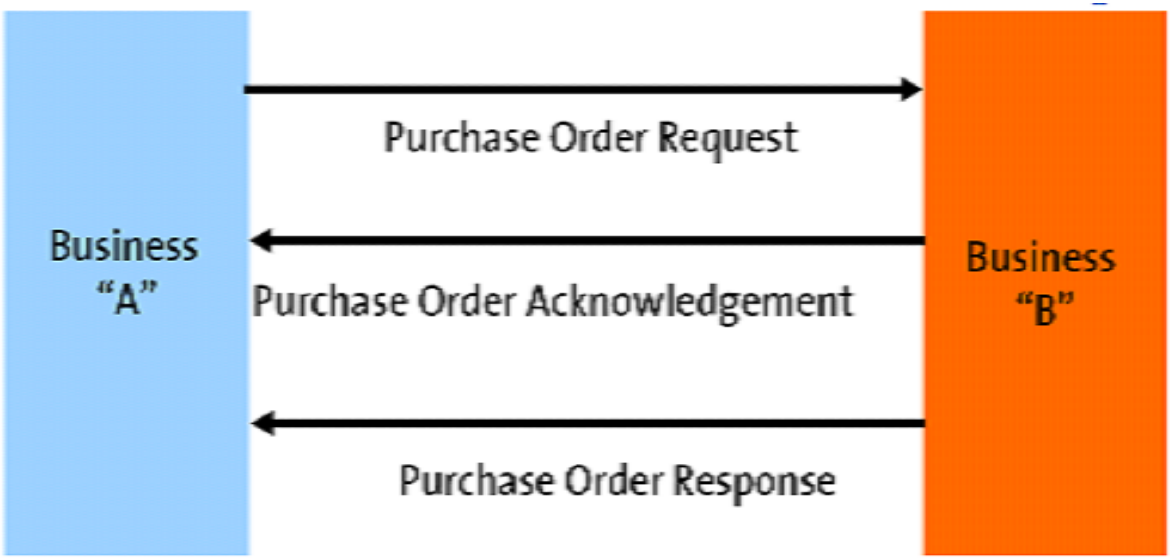
Composition and coordination (2)

- The composition must follow the coordination protocol of all of its component services
- The coordination protocol for the composite service can be inferred by looking at its internal structure



Example (1)

- This simple example defines a protocol to exchange a purchase order between two Web services of two different companies



Example (2)

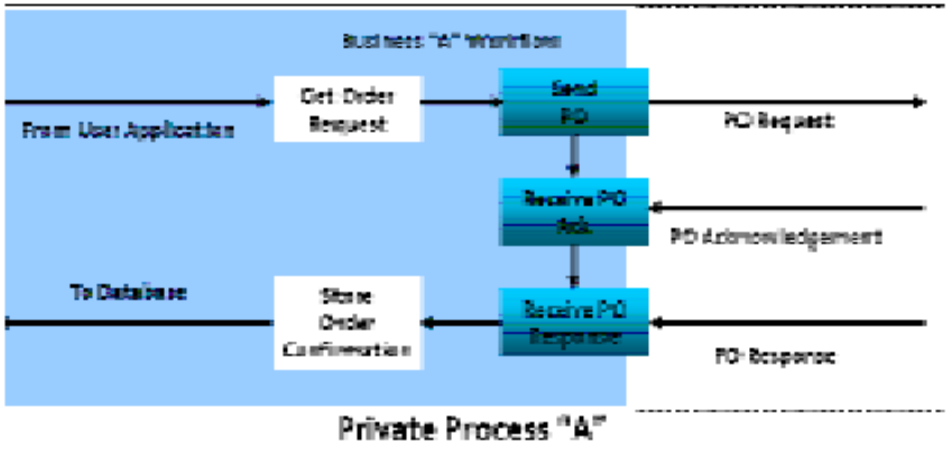
- A Purchase Order request is sent, followed by an immediate acknowledgement and a confirmation response later on

Coordination view

- The coordination view over this interaction defines the observable exchange of messages among the Web services
- This is described by the business protocol (or public process) linking the two parties

Composition view

- The composition view over this interaction defines the internal behavior of each of the Web services involved
- This is described by the orchestration (or private process) of each of the parties involved

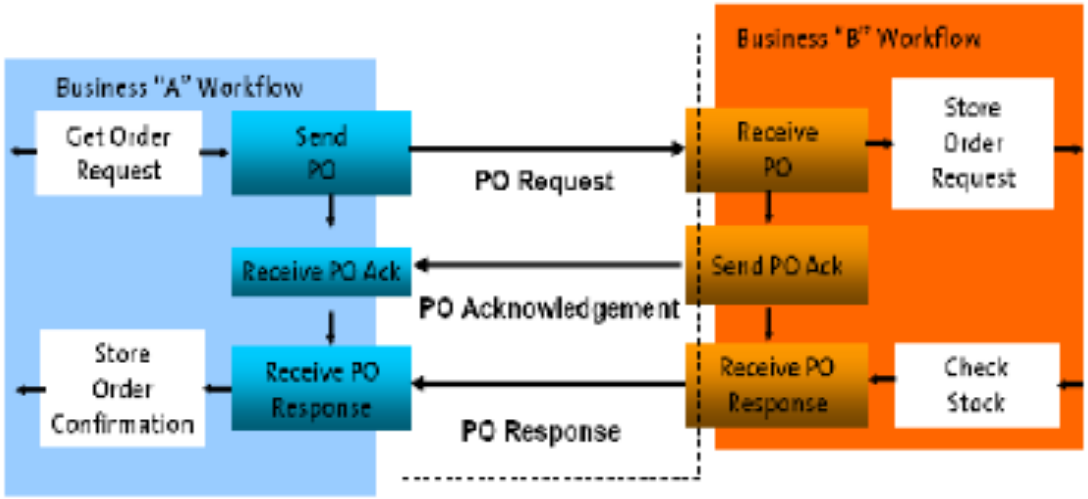


Example: complete view (1)

- The public and private processes of each of the Web services must be consistent in order for the interaction to work
- The public process (business protocol) that coordinates how the Web services interact must be agreed upon

Example: complete view (2)

- The private process that orchestrates the message exchange with the internal systems is usually kept confidential



Aspects of composition

- Component Model Web services (Synchronous and Asynchronous)
- Composition Model Formal: Statecharts, Petri Nets, π-Calculus
- XML-Oriented: Activity Hierarchies
- Workflow Based: WS-BPEL (XML), JOpera (Visual)
- Others: Rule Based, UML Activity Diagrams, Data Driven
- Service Selection Model
- Data Transfer Model
- Exception Handling
- Transactions

Component model

- A component model defines the assumptions about the components that should be composed
- In general, composing homogeneous components is easier than composing heterogeneous ones as the corresponding infrastructure is less complex
- In one case, all components could be Web services (accessed with the SOAP/HTTP protocols and described by a WSDL document)
- On the other extreme, the components are just “services” that can be accessed using a variety of invocation mechanisms

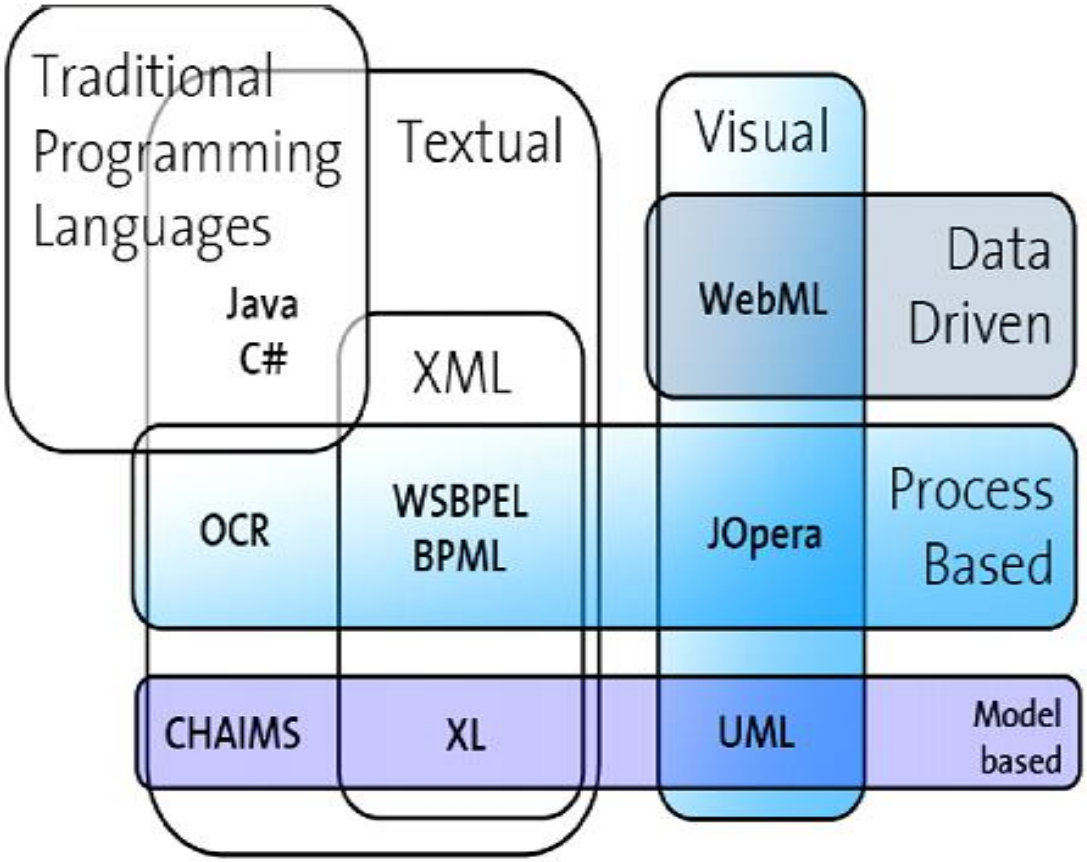
Composition model (1)

- Composition (or orchestration) models define how to build a coherent system out of a collection of services; they define:
  - what is the order in which the services are invoked and what are the conditions under which a certain service may or may not be invoked (control flow)
  - how the services exchange data with one another (data flow) and how they interact
  - some form of exception handling, i.e., what happens if a service is not available
- There are many different languages for modeling compositions

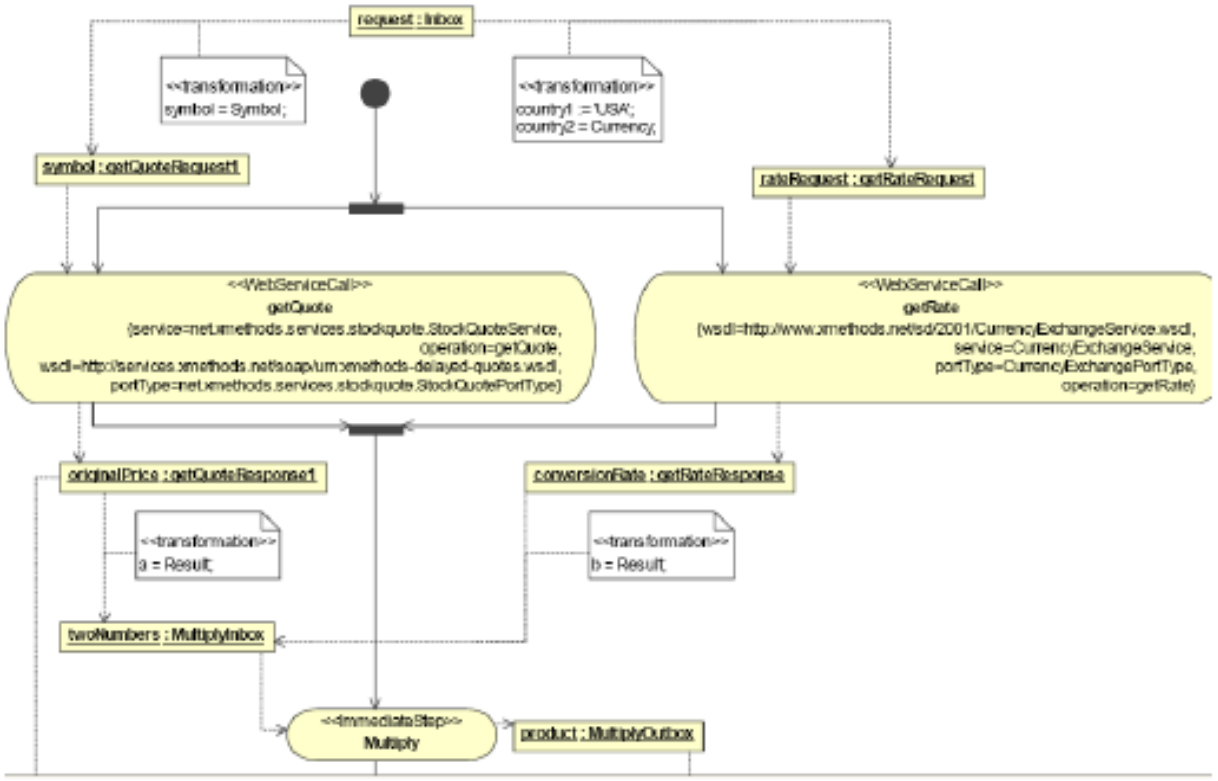
Composition model (2)

- Composition can be defined along the spatial or temporal dimension
- ADLs (Architectural Description Languages) describe the spatial architecture of a system in terms of components and connectors
- Workflow definition languages use the notion of process to define how components interact in time
- Traditional programming languages do also fulfill the requirements for composition languages

Languages for composition



Extended UML Activity diagrams



WS-BPEL (XML based)

```
<process name="shippingService"
                    targetNamespace="http://acme.com/shipping"
                    xmlns="http://schemas.xmlsoap.org/ws/2004/03/business-process/"
                    xmlns:sns="http://ship.org/wsdl/shipping"
                    xmlns:props="http://example.com/shipProps/"
                    abstractProcess="yes">

<partnerLinks>
   <partnerLink name="customer"
                    partnerLinkType="sns:shippingLT"
                    partnerRole="shippingServiceCustomer"
                    myRole="shippingService"/>
</partnerLinks>

<variables>
  <variable name="shipRequest"
                    messageType="sns:shippingRequestMsg"/>
  <variable name="shipNotice"
                    messageType="sns:shippingNoticeMsg"/>
  <variable name="itemsShipped"
                    type="props:itemCountType"/>
</variables>

<correlationSets>
  <correlationSet name="shipOrder"
                    properties="props:shipOrderID"/>
</correlationSets>

<sequence>
   <receive partnerLink="customer"
                    portType="sns:shippingServicePT"
                    operation="shippingRequest"
                    variable="shipRequest">
```

Service Selection Model (1)

- In addition to specifying what are the messages to be exchanged, a composition model should also define how to select the services that should receive or send such messages
- In order to enhance the reusability of a composition, such services are usually not hardcoded into the composition, but bound into it at different times
  - Composition time
  - Compilation and deployment time
  - Startup time
  - Invocation time

Service Selection Model (2)

- The service selection model defines how services are bound into a composition
  - Static binding (URL of service endpoint is hardcoded)
  - Dynamic binding by reference: (Service URL is computed and stored into a variable)
  - Dynamic binding by lookup (before each service invocation a query is sent to a registry to locate a suitable implementation)
  - Dynamic operation selection (no assumptions are made about the signature of the arbitrary service to be invoked)

Data Model (1)

- Services typically interact by exchanging some data
- Not all composition languages include an explicit model of the data flow
  - Data is not always treated in the same way: composition-level data is modeled at a finer level as it is used to control the composition (control flow branches) and has data types associated with it
  - Application-specific data is seen as an opaque pointer (URL) which is forwa

Data Model (2)