

# Middleware and Web Services

## Lecture 10: Service Orchestration

**doc. Ing. Tomáš Vitvar, Ph.D.**

tomas@vitvar.com • @TomasVitvar • <http://vitvar.com>



Czech Technical University in Prague

Faculty of Information Technologies • Software and Web Engineering • <http://vitvar.com/courses/mdw>



Modified: Sun Jan 04 2015, 19:20:16  
Humla v0.3

# Overview

- **Business Processes in SOA**
  - *Process Execution Models*
  - *Key Mapping*
- **Business Process Management Notation**

# Overview

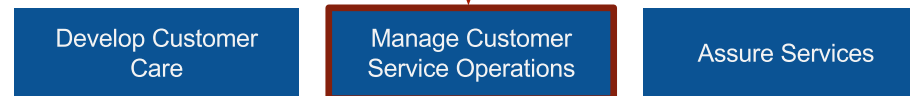
- Service Orchestration
  - *How a functionality of a service is realized by interactions with other services.*
  - *Business process is one possible form of a service orchestration that uses a business process language to describe the orchestration.*
- Business Process
  - *A set of activities and transitions between activities performed by one or more persons or systems..*
  - *A process usually spans multiple departments/divisions and applications*
- Business Process Management
  - *Management of business processes that involves:*
    - *Methodology to design processes*
    - *Tools to design, execute, monitor processes*
- Business process standardization
  - *APQC – American Productivity & Quality Center*
  - *TMForum – eTOM – Enhanced Telekom Operations Map (Business Process Framework)*

# Process Classification Framework: Recall

**Level 0**  
**Business functions**  
*Business functions of the enterprise*



**Level 1**  
**Process Groups**  
*Defines groups of processes*



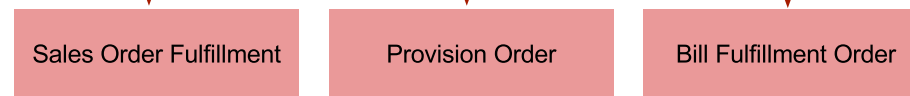
**Level 2**  
**Core Business Process**  
*Defines roles and high-level functions*



**Level 3**  
**Business Activity**  
*Defines activities within a process, multiple roles*



**Level 4**  
**Business Task**  
*Tasks are executed by a single actor (human or system)*



**Level 5**  
**Business Step**



APQC  
 Process Classification  
 Framework (PCF)

Organization's  
 Implementation specific

# Level 5 Business Process

- Level 5 Details
  - *The lowest level of a process classification framework*
  - *Level 5 business process involves technical parts that run in ESB*



# Languages

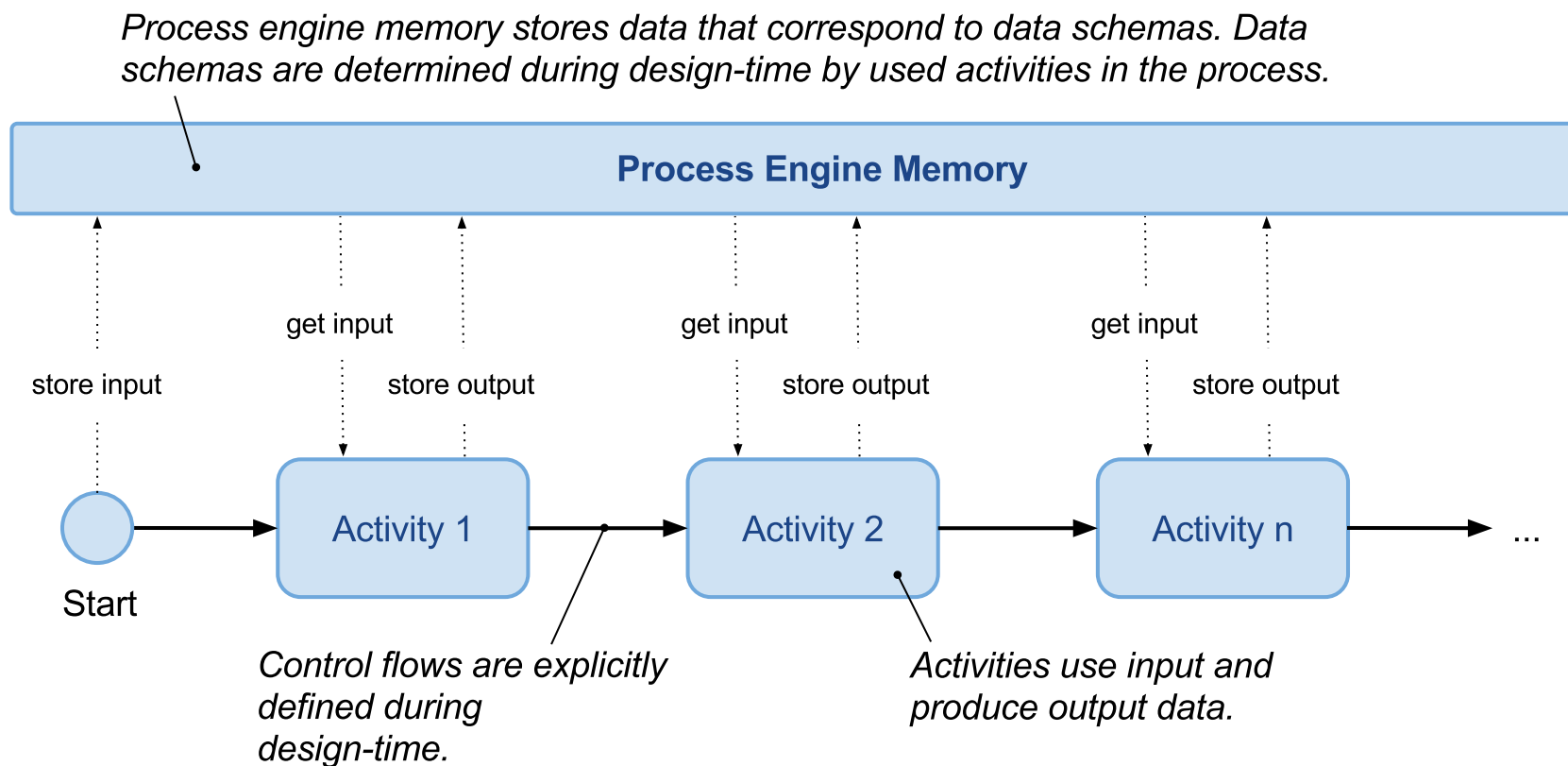
- Business Process Management Notation (BPMN)
  - *Easy to understand*
  - *Originally only a language for people to model processes*
  - *Now also possible to describe execution ready processes*
    - *Seamless transition from a business view to a technical view*
- Business Process Execution Language (BPEL)
  - *Only execution language for ESB, not suitable for modeling higher levels of business processes*
  - *Architects need to translate from BPMN (or other language) to BPEL*
    - *Part of technical SOA design*

# Overview

- Business Processes in SOA
  - *Process Execution Models*
  - *Key Mapping*
- Business Process Management Notation

# Static Process Execution

- Explicitly defined process using control flows





# Dynamic Process Execution

- Implicitly defined process using rules



- Forward-chaining algorithm (inference)
  1. A user defines a goal – described as a state (memory content)
  2. Rule engine finds a rule which condition matches content in the memory
  3. Rule's action is executed; the result may add or modify data in the memory
  4. When the memory content matches the goal, execution stops otherwise go to 2

# Dynamic Process Execution Example

- Rule Set

- *A: if a customer is a premium customer, give them 10% discount*  
`if customer.type=='premium' then order.discount=0.1`
- *B: if a customer is a gold customer, give them 5% discount*  
`if customer.type=='gold' then order.discount=0.05`
- *C: if customer spends 1,000 or more, make them a premium customer*  
`if customer.revenue>1000 then customer.type='premium'`
- *Note: actions are memory modifications but can be results of service calls*

- Execution

- Option 1

*MEM:* `customer.type=='gold', customer.revenue==1500`

*Rules:* C, A

*MEM:* `customer.type=='premium', customer.revenue==1500,  
order.discount==0.1`

- Option 2

*MEM:* `customer.type=='gold', customer.revenue==200`

*Rules:* B

*MEM:* `customer.type=='gold', customer.revenue==200, order.discount==0.05`

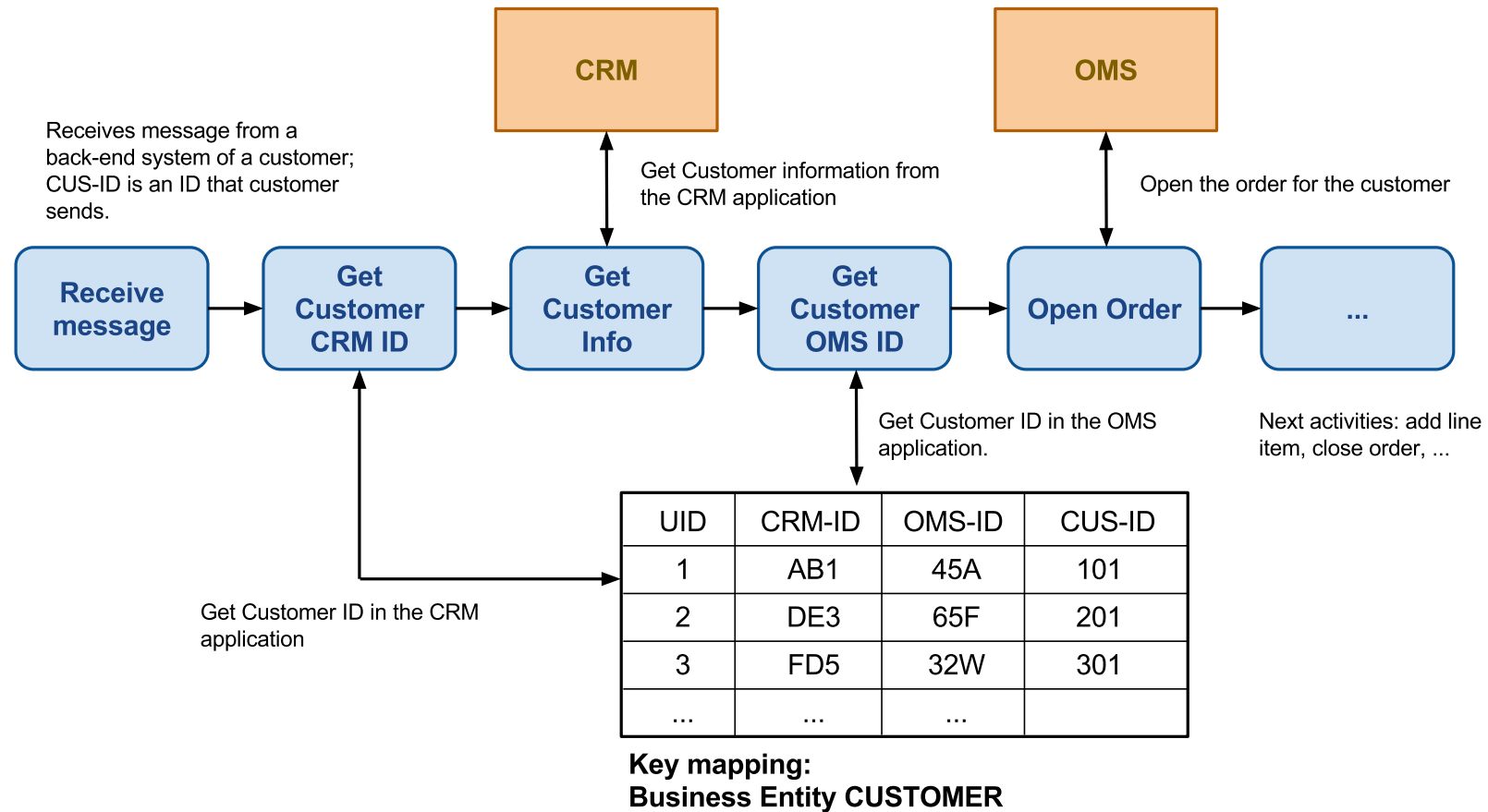
# Overview

- Business Processes in SOA
  - *Process Execution Models*
  - *Key Mapping*
- Business Process Management Notation

# Overview

- What is key mapping
  - *Key = identifier of an entity in a back-end application*
  - *Key Mapping = a mapping of an ID of an entity in one system to an ID of the same entity in another system.*
  - *Key mapping is realized using universal IDs (UID)*
- Example
  - *A customer MOON exists in CRM and OMS systems*
  - *In CRM system, MOON has an CRM-ID=AB1*
  - *In OMS system, MOON has an CRM-ID=45A*
  - *Key mapping allows to map the CRM-ID AB1 to the OMS-ID 45A*
  - *Key mapping is a table*
    - CRM-ID → UID → OMS-ID**

# Key Mapping Example



# Overview

- Business Processes in SOA
- Business Process Management Notation
  - *Patterns*
  - *Conversation and Correlation*

# BPMN Constructs – Activities and Gateways

- Activities (Tasks)



- *Service Task – synchronous service invocation*
- *Send and receive task – asynchronous service invocation*
- *User task – human step, managed by workflow engine*
- *Manual task – human step, not managed by workflow engine*
- *Script task – automatic activity, assigning variables, modify variables' values*
- *Call activity – to call a another process (modularisation, process chaining)*

- Gateways



- *Exclusive OR (XOR)*
- *Inclusive OR (OR)*
- *Parallel fork and join (AND)*
- *Complex gateway*

# BPMN Constructs – Catch Events

- Catch Events

- *They start a process or continue a process from a sub-process when event occurs.*



- *Catch Event Types*

- *Start Message – starts a process by a new message*

- *Start Timer – starts a process by a timer*

- *Start Signal – starts a process by a new event*

- *Catch Error – catches error during execution of an activity (boundary event)*

- *Catch Timer – a timeout when during execution of an activity (boundary event)*

- *Catch Signal – catches a signal during execution of an activity (boundary event)*



# BPMN Constructs – Throw Events

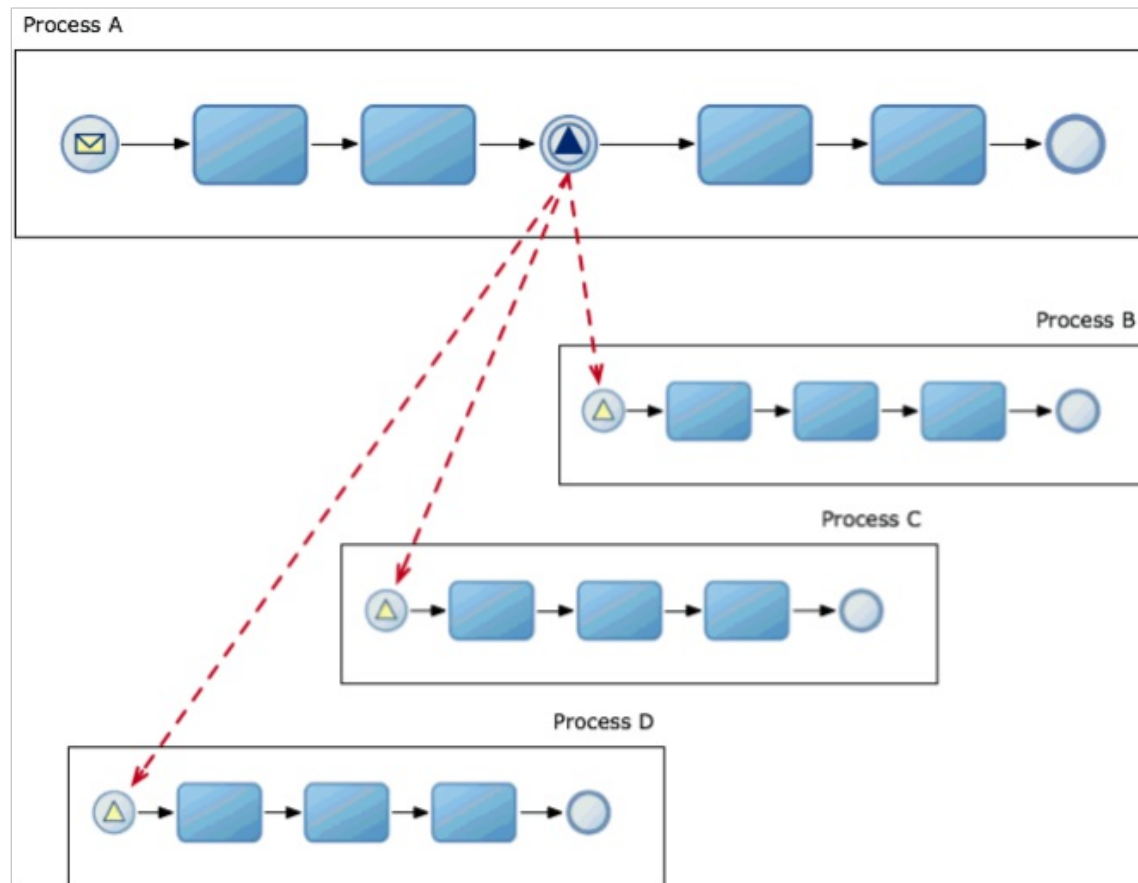
- Throw Events
  - *They end a process or a sub-process*



- *Throw Event Types*
  - *End Message – ends a process by sending a message back to client*
  - *End Terminate – ends a process silently*
  - *End Signal – ends a process by signalling an event*
  - *End Error – ends a process by throwing an error (fault)*
  - *Throw Message – throws a message during process execution*
  - *Throw Signal – throws an event during process execution*

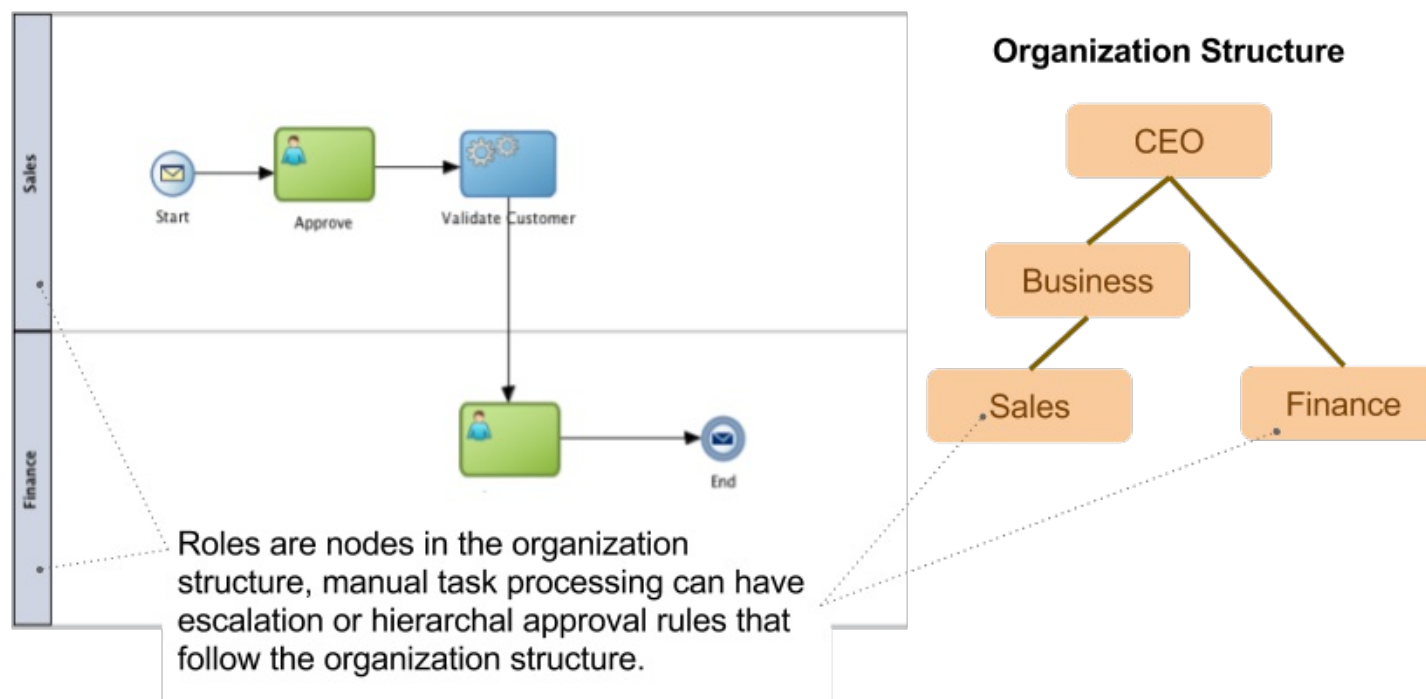
# Throw and Catch Events

- Broadcasting a signal
  - *Process A throws a signal, processes B, C, and D catch the signal and start*



# Swim Lanes

- Swim Lanes
  - Group of activities in a process
  - Each swim lane is associated with a role; a user in the role performs activities
  - A special role "Automated" or "System" defines automated activities or activities performed by the system, not humans.
- Example

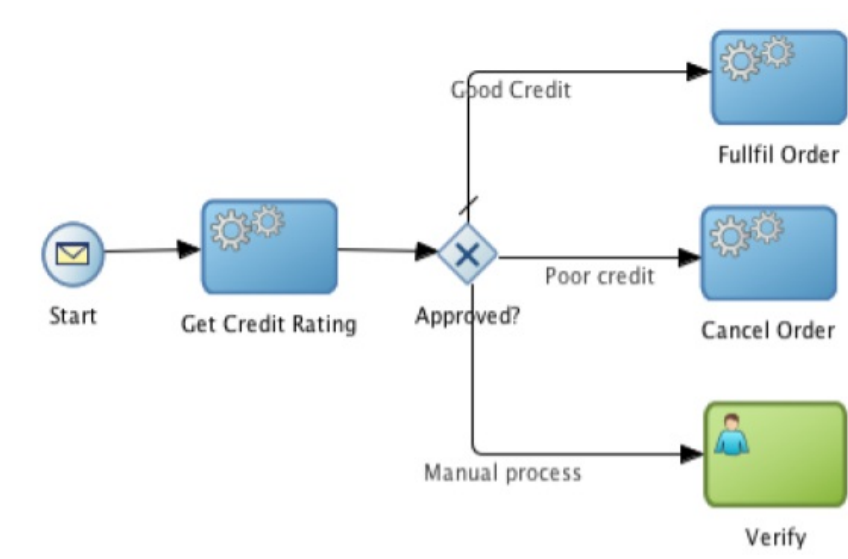


# Overview

- Business Processes in SOA
- Business Process Management Notation
  - *Patterns*
  - *Conversation and Correlation*

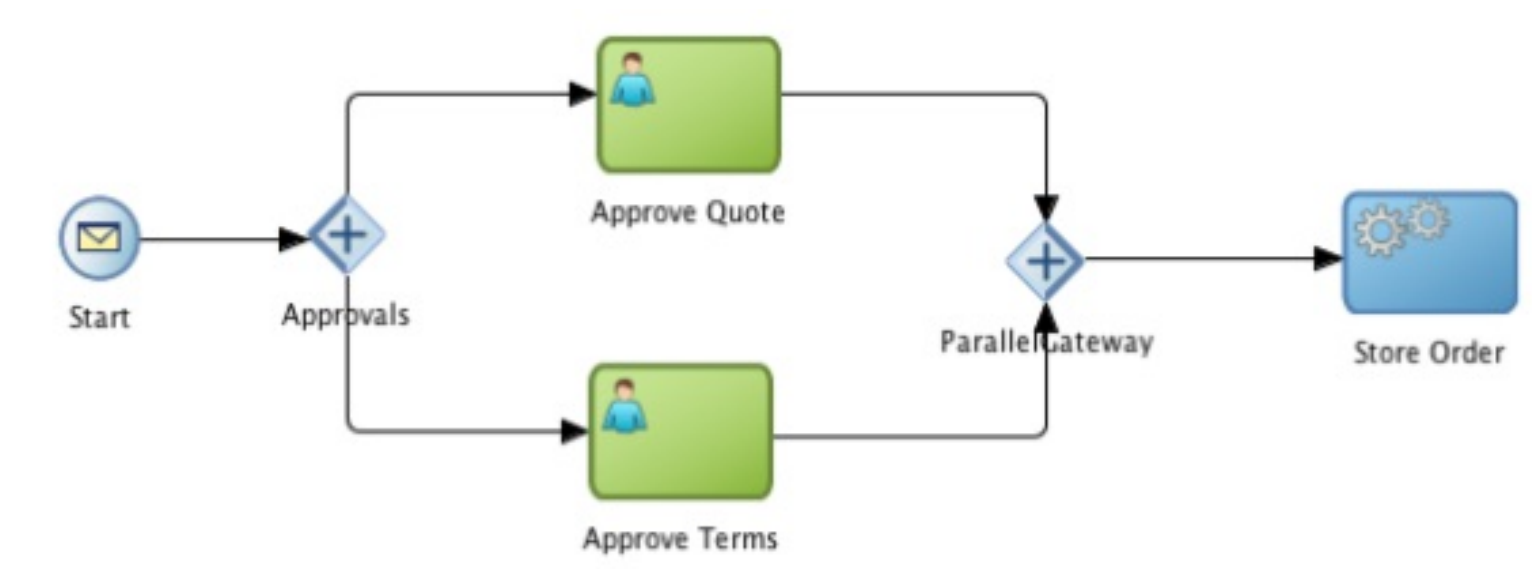
# Exclusive OR

- Definition
  - *XOR defines multiple alternative paths*
  - *One and only one path can result from an evaluation of conditions on each branch*
  - *There must be a **default path***
    - *Path that will be used when no conditions on other paths will evaluate to true*
- Example



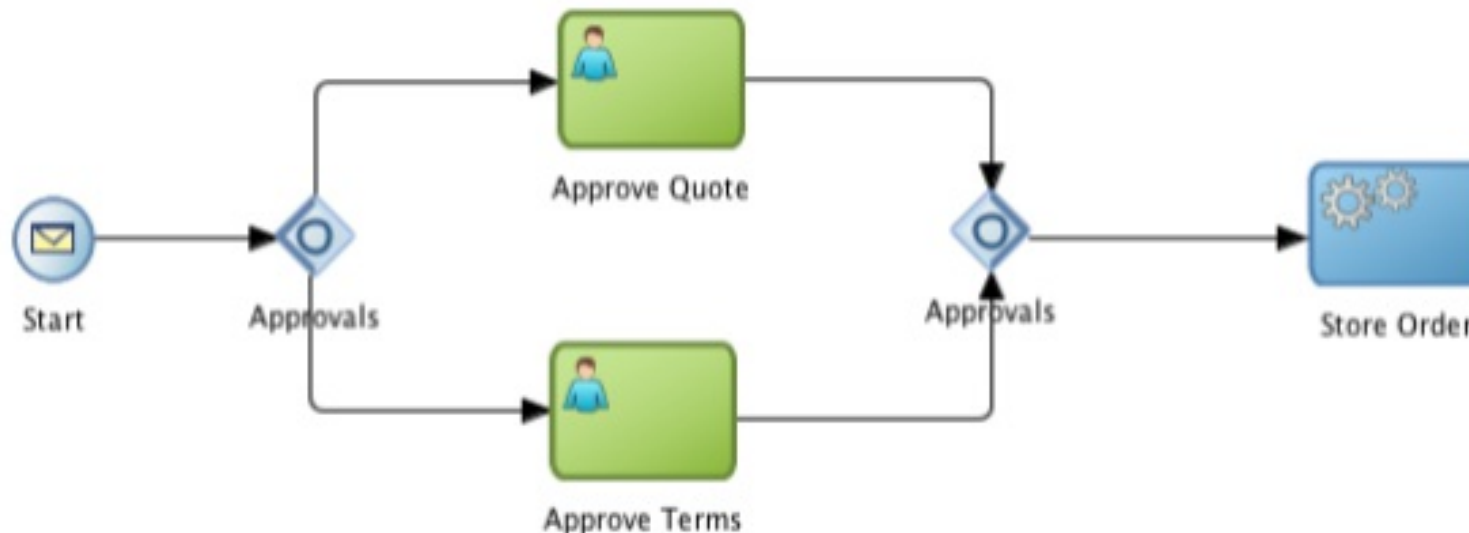
# Parallel Fork and Join

- Definition
  - *AND Gateway* – defines multiple paths
  - *All paths are processed*
  - *All paths must be either joined by AND Gateway or each path may end with a separated end events*
- Example



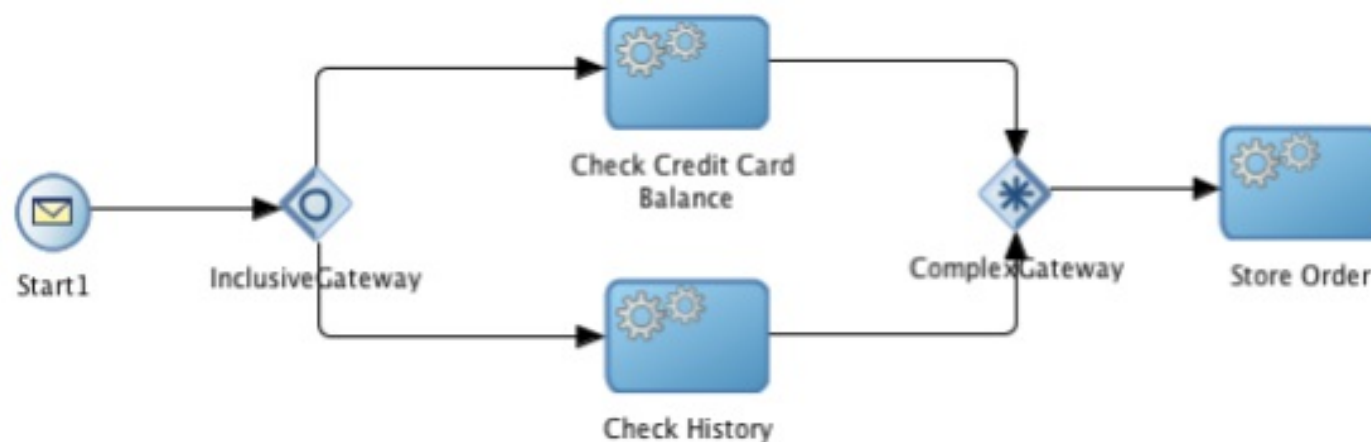
# Inclusive OR

- Definition
  - *OR Gateway – defines multiple paths*
  - *One or more paths are processed in parallel*
    - *There can be more than one possible outcomes*
  - *There can be a default path*
  - *All paths must be either joined by OR Gateway or each path may end with a separated end events*
- Example



# Complex Gateway

- Definition
  - *Evaluates conditions to decide when the process should continue*
  - *Used only when merging process flows (OR is used for splitting)*
- Example



– *Complex gateway condition*

1 | `credit-card/balance/text() > 1000 OR customer/history/num-purchases/text() > 10`

→ *Only one condition needs to be satisfied for the process to continue*

→ *Other parallel flows are canceled*



# Loops

- Definition
  - *Loops can be created with XOR Gateways by connecting a flow to a previous step*
  - *XOR checks for conditions*
    - *logical expression or counters*
- Example



- *Example condition on "YES" path*

1 | `quote/price/text() > 0`

# Error Handling

- Definition
  - *Mechanism to handle errors during process executions*
  - *Throwing errors*
    - *On evaluation of conditions*
    - *A result of external service calls*
  - *Catching Errors*
    - *Take actions to resolve errors, continue process execution*
    - *Throw errors as a result of process execution*
- Implementation
  - *Use boundary catch error events to catch errors on activities or sub-processes*
  - *Implement paths in process flows that resolve errors*

# Overview

- Business Processes in SOA
- Business Process Management Notation
  - *Patterns*
  - *Conversation and Correlation*

# Conversation: Recall

- Definition
  - *A group of client-service interactions that logically and technically belong together, e.g. `submitOrder`, `getOrderStatus`, `cancelOrder`*
  - *There is typically one conversation per process but can be more*
- Implementation
  - *Single conversation corresponds to a single WSDL interface definition (`portType` in WSDL 1.1)*
  - *Two conversations will have own interface definition each*
    - *hence you can define separated binding for each*
  - *To relate interactions in a single conversation, you should have:*
    - *a **main process**, that is a main operation/event that triggers the process and creates a conversation, e.g. `submitOrder`*
    - *a **sub-process** to the main process, e.g. `checkStatus`*
  - *WS-Addressing defines `MessageID` element that is a conversation ID*
  - *See details in lecture 8*

# Correlation

- Definition
  - *Correlation associates a client interaction with a running process instance by using correlation keys and data in an input message.*
  - *A process may have one or more correlations defined*
- Correlation keys and properties
  - *Correlations are defined by so called **correlation keys***
  - *A key may have one or more properties*
    - *Each property value is defined by xpath on a message structure*
- Example
  - *A correlation with two properties*
    - *Key: **order\_correlation***
    - *Property 1: **customer\_id***
    - *Property 2: **order\_id***

# Correlation Example

- Correlation key for Order message
  - *Order message*

```
1  <s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
2    <s:Body>
3      <ns1:submitOrder xmlns:ns1="http://xmlns.oracle.com/bpmn/bpmnProcess/Process"
4        xmlns:ns2="http://www.example.org/order">
5        <ns2:Order>
6          <ns2:OrderId>566</ns2:OrderId>
7          <ns2:CustomerId>4354</ns2:CustomerId>
8          <ns2:Address>Thakourova 6</ns2:Address>
9          <ns2:Phone>+42024537765</ns2:Phone>
10         <ns2:LineItems>
11           <ns2:item>
12             <ns2:name>HP ProBook 4540s</ns2:name>
13             <ns2:price>900</ns2:price>
14           </ns2:item>
15         </ns2:LineItems>
16       </ns2:Order>
17     </ns1:submitOrder>
18   </s:Body>
19 </s:Envelope>
```

- *Correlation key **order\_correlation**, xpath definition*

```
1  customer_id property: /ns1:submitOrder/ns2:Order/ns2:CustomerId
2  order_id property:    /ns1:submitOrder/ns2:Order/ns2:OrderId
```