Lezione 15 – BPMN-BPEL Mapping

Ingegneria dei Processi Aziendali

Modulo 1 - Servizi Web

Unità didattica 1 – Protocolli Web

Ernesto Damiani

Università di Milano

Agenda

BPMN BPEL Mapping and Challenges

Tool Requirements

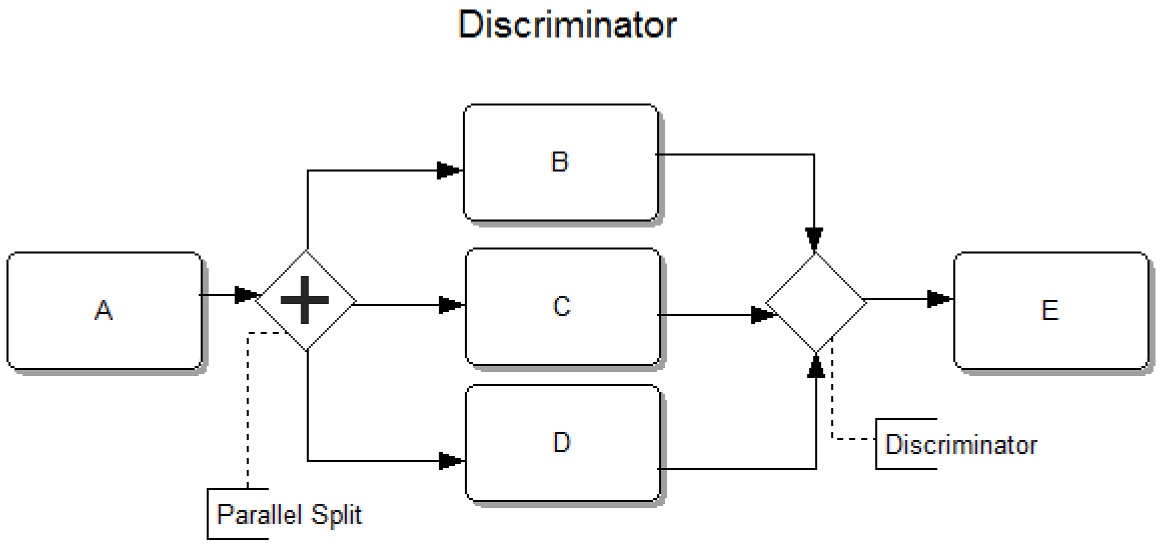
BPMN and BPELWS

BPELWS (Business Process Execution Language for Web Service)

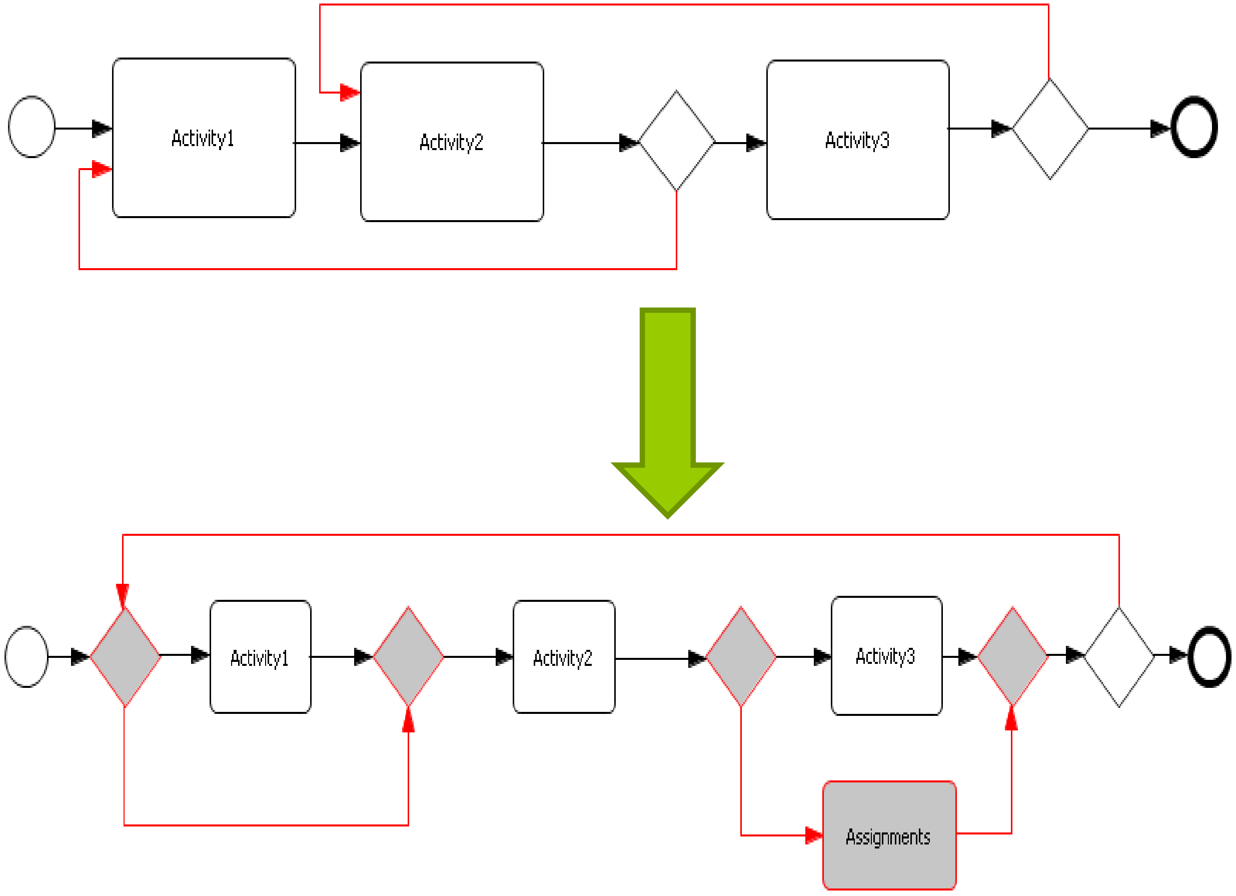
- BPELWS is a block structured language and BPMN is a graph language – BPELWS is less expressive than BPMN
- BPMN meta-model contains all objects and properties to generate deploy-ready BPELWS (with the exception of correlation set)
- You can always generate a BPMN model from a valid BPEL.
- Generating a BPEL from a valid BPMN model sometimes is not possible

Discriminator Pattern cannot be represented in BPEL

A set of parallel flows come back together at the Discrimantor. After the first Token arrives, all remaining Tokens that were generated from the Parallel Split will be blocked at the Discriminator



An intelligent tool can analyze model and redraw diagram



Cannot transform to BPEL

- Can transform to BPEL

Requirements

Help users to analyze and redraw BPMN diagram to generate a valid BPEL

Two modes of code generation – skeleton and deploy-ready BPELWS

Model/code merge

BPEL / BPMN Mapping

BPEL

Process

- namespaces
- targetNamespaces
- suppressJoinFailure
- partnerLinks
- Variables
  - variable
    - messageType
    - Type/ElementType
- FaultHandlers
  - catch/catchAll
- EventHandlers
  - OnMessage/OnAlarm
- CompensationHandler
  - Activity

BPMN

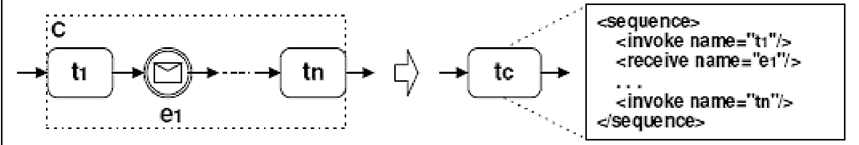
Diagram

- namespaces
- Process
  - targetNamespaces
  - suppressJoinFailure
  - Message
  - Property
- BoundaryEvent
- Intermediate Event
- BoundaryEvent
- Task

BPEL / BPMN Mapping (cont.)

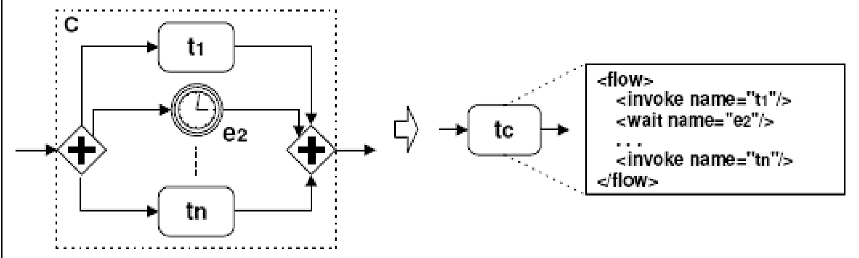
BPEL	BPMN
Activity	
Receive	StartEvent/Task
Reply	EndEvent
Invoke	Task
Throw	EndEvent
faultName	errorCode
Wait	IntermediateEvent
Sequence	
Activity	Task
Switch	Gateway
While	Subprocess
Pick	GateWay
Flow	GateWay
Activity	Task
Scope	Subprocess
FaultHandlers	
Catch/catchAll	BoundaryEvent
EventHandlers	
OnMessage/OnAlarm	Intermediate Event
Variables	
variable	
MessageType	Message
Type/ElementType	Property
Compensate	EndEvent

Mapping Cheat Sheet



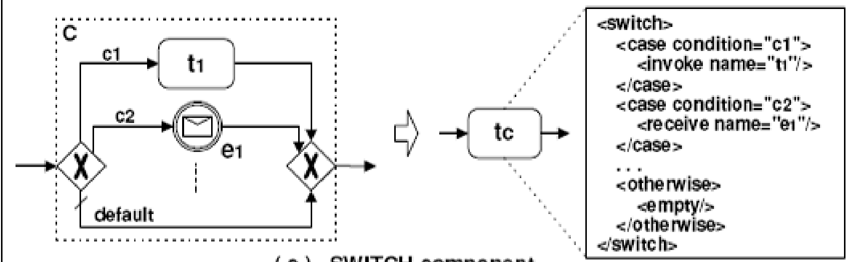
```
<sequence>
  <invoke name="t1"/>
  <receive name="e1"/>
  . . .
  <invoke name="tn"/>
</sequence>
```

(a) SEQUENCE-component



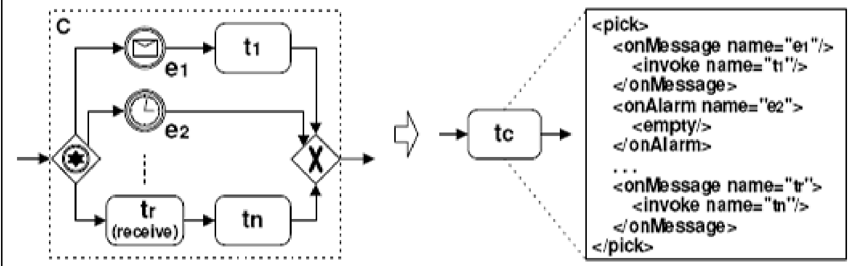
```
<flow>
  <invoke name="t1"/>
  <wait name="e2"/>
  . . .
  <invoke name="tn"/>
</flow>
```

(b) FLOW-component



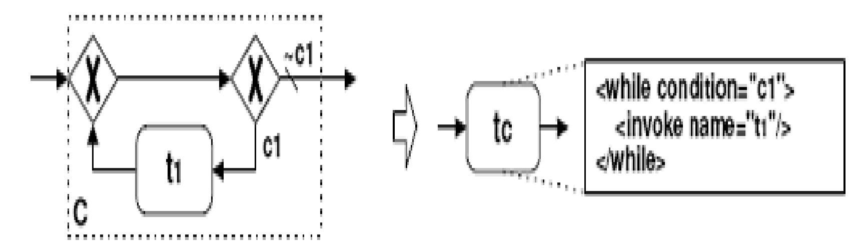
```
<switch>
  <case condition="c1">
    <invoke name="t1"/>
  </case>
  <case condition="c2">
    <receive name="e1"/>
  </case>
  . . .
  <otherwise>
    <empty/>
  </otherwise>
</switch>
```

(c) SWITCH-component



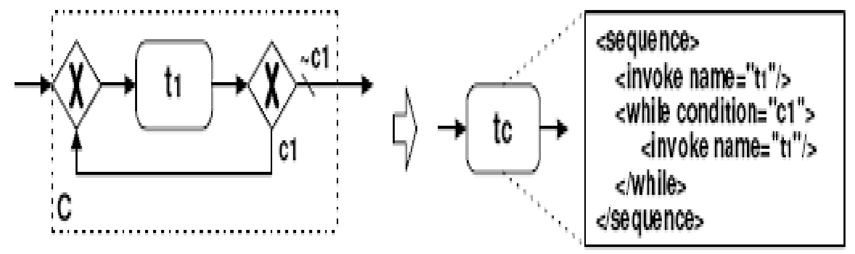
```
<pick>
  <onMessage name="e1"/>
    <invoke name="t1"/>
  </onMessage>
  <onAlarm name="e2">
    <empty/>
  </onAlarm>
  . . .
  <onMessage name="tr">
    <invoke name="tn"/>
  </onMessage>
</pick>
```

(d) PICK-component



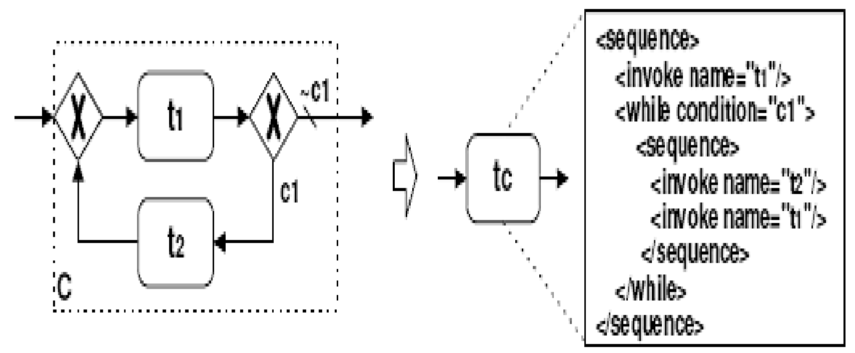
```
<while condition="c1">
  <invoke name="t1"/>
</while>
```

(e) WHILE-component



```
<sequence>
  <invoke name="t1"/>
  <while condition="c1">
    <invoke name="t1"/>
  </while>
</sequence>
```

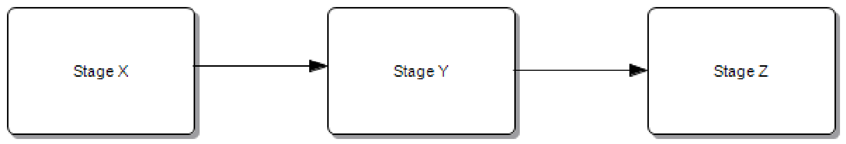
(f) REPEAT-component



```
<sequence>
  <invoke name="t1"/>
  <while condition="c1">
    <sequence>
      <invoke name="t2"/>
      <invoke name="t1"/>
    </sequence>
  </while>
</sequence>
```

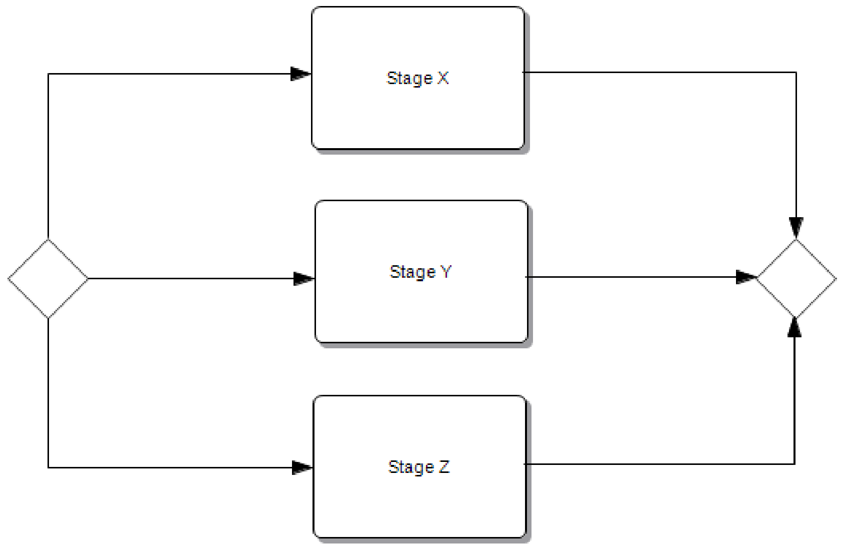
(g) REPEAT+WHILE-component

Mapping Sequential Processing



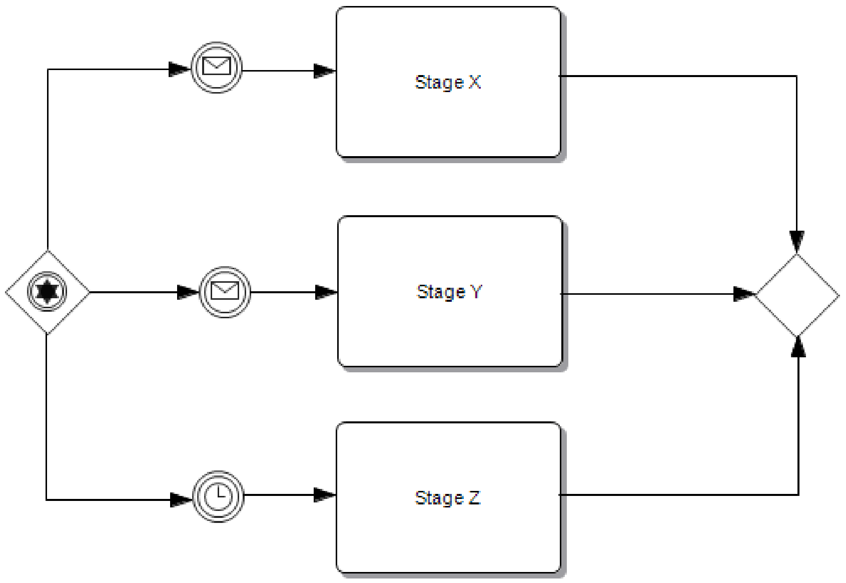
```
<?xml version="1.0" encoding="UTF-8"?>
<bpel:process xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable"
    xmlns:xsd="http://www.w3.org/2001/XM
    LSchema"
    expressionLanguage="urn:oasis:names:tc:
    wsbpel:2.0:sublang:xpath1.0"
    name="testModel"
    queryLanguage="urn:oasis:names:tc:wsbpe
    l:2.0:sublang:xpath1.0">
  <bpel:sequence>
    <bpel:invoke name="Stage_X"
      inputVariable="" outputVariable=""/>
    <bpel:invoke name="Stage_Y"
      inputVariable="" outputVariable=""/>
    <bpel:invoke name="Stage_Z"
      inputVariable="" outputVariable=""/>
  </bpel:sequence>
</bpel:process>
```

Mapping Exclusive Branching – Data-based



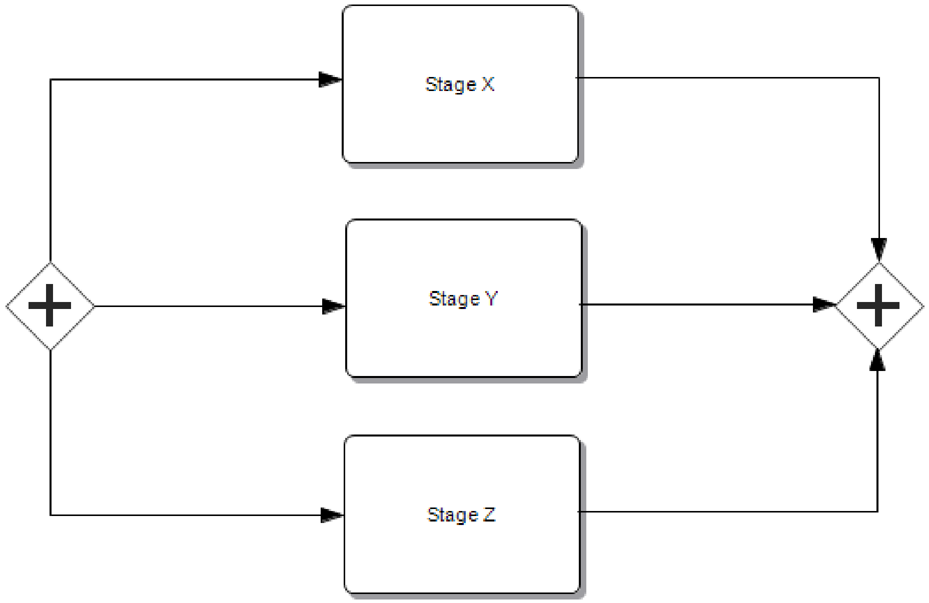
```
<?xml version="1.0" encoding="UTF-8"?>
<bpel:process xmlns:bpel="http://docs.oasis-
      open.org/wsbpel/2.0/process/executable"
      xmlns:xsd="http://www.w3.org/2001/XM
      LSchema"
      expressionLanguage="urn:oasis:names:tc:
      wsbpel:2.0:sublang:xpath1.0"
      name="testModel"
      queryLanguage="urn:oasis:names:tc:wsbpe
      l:2.0:sublang:xpath1.0">
  <bpel:if name="Gateway1">
    <bpel:condition/>
    <bpel:invoke name="Stage_X"
      inputVariable="" outputVariable=""/>
    <bpel:elseif>
      <bpel:condition/>
      <bpel:invoke name="Stage_Y"
      inputVariable="" outputVariable=""/>
    </bpel:elseif>
    <bpel:elseif>
      <bpel:condition/>
      <bpel:invoke name="Stage_Z"
      inputVariable="" outputVariable=""/>
    </bpel:elseif>
  </bpel:if>
</bpel:process>
```

Mapping Exclusive Branching – Event-based



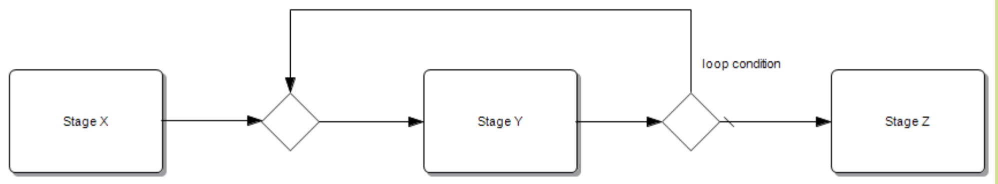
```
<?xml version="1.0" encoding="UTF-8"?>
<bpel:process xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable"
    xmlns:xsd="http://www.w3.org/2001/XML
    Schema"
    expressionLanguage="urn:oasis:names:tc:
    wsbpel:2.0:sublang:xpath1.0"
    name="testModel"
    queryLanguage="urn:oasis:names:tc:wsbpe
    l:2.0:sublang:xpath1.0">
 <bpel:pick name="Gateway1">
  <bpel:onMessage operation=""
    partnerLink="">
   <bpel:invoke name="Stage_Y"
    inputVariable="" outputVariable=""/>
  </bpel:onMessage>
  <bpel:onMessage operation=""
    partnerLink="">
   <bpel:invoke name="Stage_X"
    inputVariable="" outputVariable=""/>
  </bpel:onMessage>
  <bpel:onAlarm>
   <bpel:until>'2009-01-01T00:00:00.699-
    0800'</bpel:until>
   <bpel:invoke name="Stage_Z"
    inputVariable="" outputVariable=""/>

    </bpel:onAlarm></bpel:pick></bpel:proce
    ss>
```

Mapping Concurrent Processing



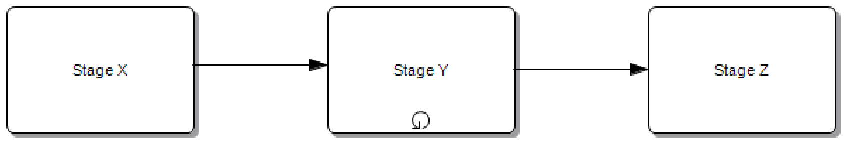
```
<?xml version="1.0" encoding="UTF-8"?>

<bpel:process xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable"
    xmlns:xsd="http://www.w3.org/2001/XM
    LSchema"
    expressionLanguage="urn:oasis:names:tc:
    wsbpel:2.0:sublang:xpath1.0"
    name="testModel"
    queryLanguage="urn:oasis:names:tc:wsbpe
    l:2.0:sublang:xpath1.0">

 <bpel:flow>

  <bpel:invoke name="Stage_X"
    inputVariable="" outputVariable=""/>

  <bpel:invoke name="Stage_Z"
    inputVariable="" outputVariable=""/>

  <bpel:invoke name="Stage_Y"
    inputVariable="" outputVariable=""/>

 </bpel:flow>

</bpel:process>
```

Mapping Looping



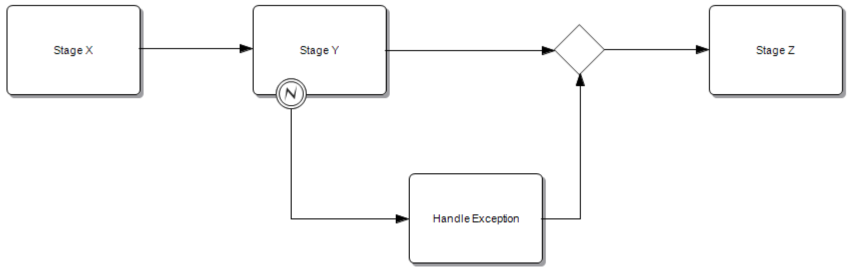
```
<?xml version="1.0" encoding="UTF-8"?>
<bpel:process xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable
    "
    xmlns:xsd="http://www.w3.org/2001/XM
    LSchema"
    expressionLanguage="urn:oasis:names:tc:
    wsbpel:2.0:sublang:xpath1.0"
    name="testModel"
    queryLanguage="urn:oasis:names:tc:wsbp
    el:2.0:sublang:xpath1.0">
 <bpel:sequence>
   <bpel:invoke name="Stage_X"
    inputVariable="" outputVariable=""/>
   <bpel:invoke name="Stage_Y"
    inputVariable="" outputVariable=""/>
   <bpel:while>
     <bpel:condition/>
     <bpel:invoke name="Stage_Y"
    inputVariable="" outputVariable=""/>
   </bpel:while>
   <bpel:invoke name="Stage_Z"
    inputVariable="" outputVariable=""/>
  </bpel:sequence>
</bpel:process>
```

Mapping Looping Example 2



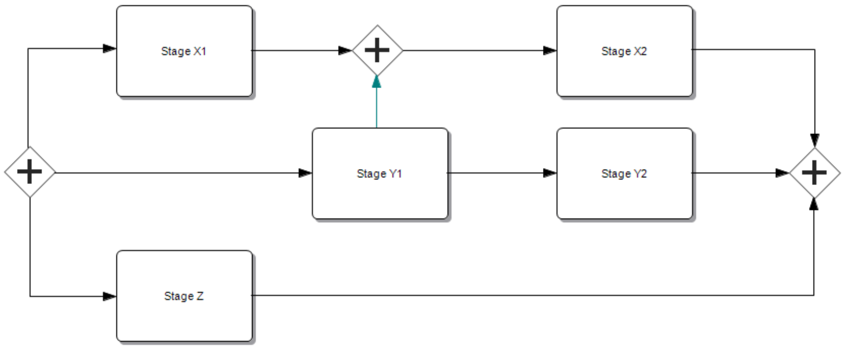
```
<?xml version="1.0" encoding="UTF-8"?>
<bpel:process xmlns:bpel="http://docs.oasis-
    open.org/wsbpel/2.0/process/executable"
    xmlns:xsd="http://www.w3.org/2001/XM
    LSchema"
    expressionLanguage="urn:oasis:names:tc:
    wsbpel:2.0:sublang:xpath1.0"
    name="testModel"
    queryLanguage="urn:oasis:names:tc:wsbpe
    l:2.0:sublang:xpath1.0">
  <bpel:sequence>
    <bpel:invoke name="Stage_X"
      inputVariable="" outputVariable=""/>
    <bpel:repeatUntil name="Stage_Y"
      suppressJoinFailure="yes">
      <bpel:invoke name="Stage_Y"
      inputVariable="" outputVariable=""/>
      <bpel:condition>not
      (true())</bpel:condition>
    </bpel:repeatUntil>
    <bpel:invoke name="Stage_Z"
      inputVariable="" outputVariable=""/>
  </bpel:sequence>
</bpel:process>
```

Mapping Fault Handling



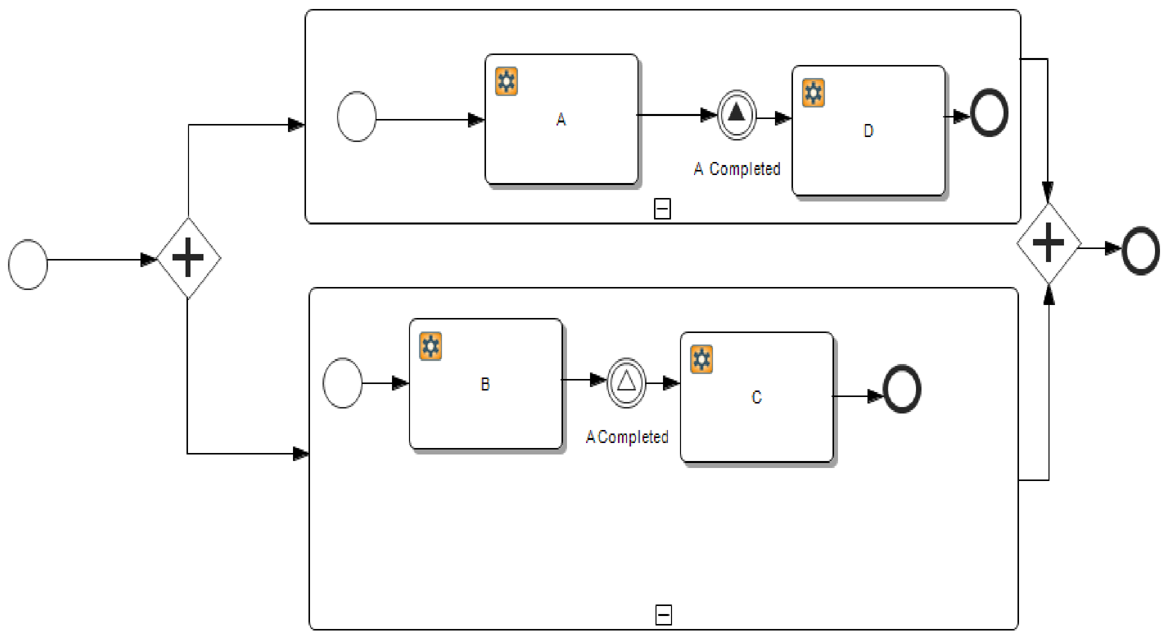
```
<?xml version="1.0" encoding="UTF-8"?>

<bpel:process xmlns:bpel="http://docs.oasis-
     open.org/wsbpel/2.0/process/executable"
     xmlns:xsd="http://www.w3.org/2001/XM
     LSchema"
     expressionLanguage="urn:oasis:names:tc:
     wsbpel:2.0:sublang:xpath1.0"
     name="testModel"
     queryLanguage="urn:oasis:names:tc:wsbpe
     l:2.0:sublang:xpath1.0">

  <bpel:sequence>

    <bpel:invoke name="Stage_X"
     inputVariable="" outputVariable=""/>

    <bpel:invoke name="Stage_Y"
     inputVariable="" outputVariable="">

      <bpel:catchAll>

        <bpel:invoke name="Handle_Exception"
     inputVariable="" outputVariable=""/>

      </bpel:catchAll>

    </bpel:invoke>

    <bpel:invoke name="Stage_Z"
     inputVariable="" outputVariable=""/>

  </bpel:sequence>

</bpel:process>
```

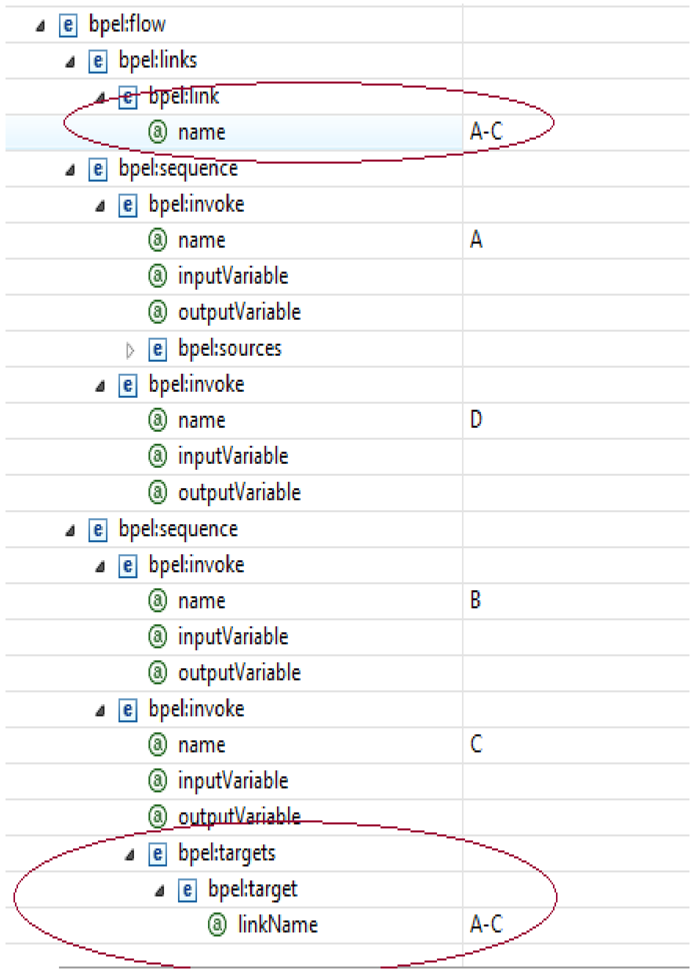
Mapping Synchronization in Concurrent Processing



```
<?xml version="1.0" encoding="UTF-8"?>

<bpel:process xmlns:bpel="http://docs.oasis-
      open.org/wsbpel/2.0/process/executable"
      xmlns:xsd="http://www.w3.org/2001/XMLSchema"
      expressionLanguage="urn:oasis:names:tc:wsbpel:2.0:sublan
      g:xpath1.0" name="testModel"
      queryLanguage="urn:oasis:names:tc:wsbpel:2.0:sublang:xpa
      th1.0">

  <bpel:flow>

    <bpel:links><bpel:link name="Stage_Y1-Stage_X2"/>
      </bpel:links>

    <bpel:sequence>

      <bpel:invoke name="Stage_X1" inputVariable=""
        outputVariable=""/>

      <bpel:invoke name="Stage_X2" inputVariable=""
        outputVariable="">

        <bpel:targets> <bpel:target linkName="Stage_Y1-
        Stage_X2"/>   </bpel:targets>

      </bpel:invoke>

    </bpel:sequence>

    <bpel:sequence>

      <bpel:invoke name="Stage_Y1" inputVariable=""
        outputVariable="">

        <bpel:sources><bpel:source linkName="Stage_Y1-
        Stage_X2"/></bpel:sources>

      </bpel:invoke>

      <bpel:invoke name="Stage_Y2" inputVariable=""
        outputVariable=""/>

    </bpel:sequence>

    <bpel:invoke name="Stage_Z" inputVariable=""
      outputVariable=""/>

  </bpel:flow></bpel:process>
```

Mapping Signal to BPEL flow link





bpel:flow	
bpel:links	
bpel:link	
name	A-C
bpel:sequence	
bpel:invoke	
name	A
inputVariable	
outputVariable	
bpel:sources	
bpel:invoke	
name	D
inputVariable	
outputVariable	
bpel:sequence	
bpel:invoke	
name	B
inputVariable	
outputVariable	
bpel:invoke	
name	C
inputVariable	
outputVariable	
bpel:targets	
bpel:target	
linkName	A-C

FINE