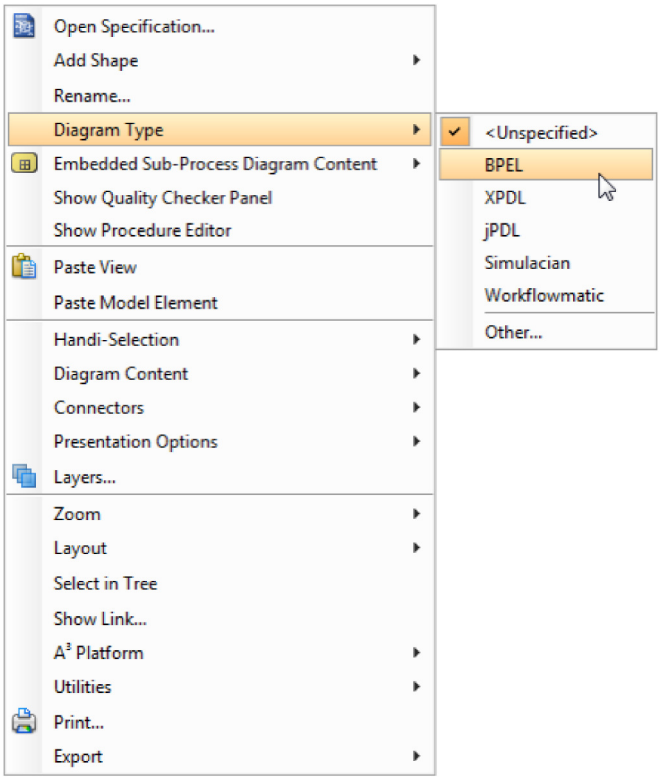
Writing WSDL

WSDL, short for Web Service Description Language, is an XML based language for describing the interface of web services. You can write WSDL definition in VP-UML with WSDL diagram. By combining with business process model, a full set of WSDL and BPEL can be generated.

Alternative way to create WSDL diagram - from BPMN pool to WSDL

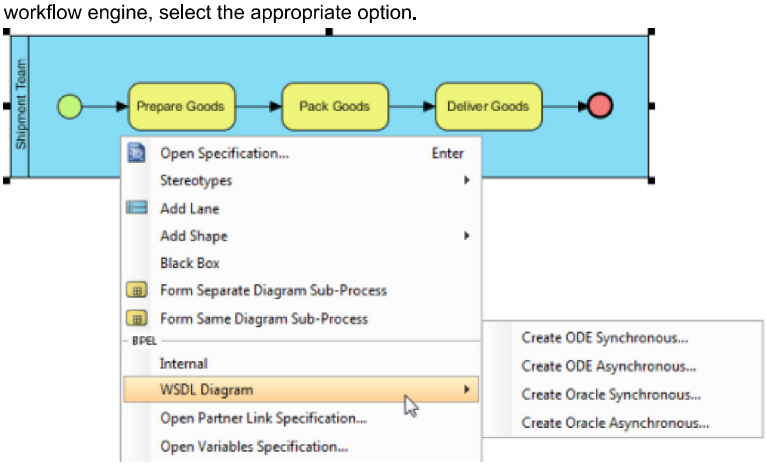
A process is the interactions between collaborators. Other than creating a WSDL diagram from diagram navigator, an alternative way is to create a from a pool (i.e. the collaborator) that initiate the process.

1. Define the business process diagram of where the pool reside as BPEL diagram by right clicking on the background of diagram and selecting **Diagram Type > BPEL** from the popup menu. If you have done this before, ignore this step and move to step 2.



Set diagram type to be BPEL

2. Right click on the pool which initiate the process and select **WSDL Diagram** from the popup menu. Base on the type of process, and the type of workflow engine, select the appropriate option.



Create WSDL diagram

Option	Description
Create ODE Synchronous	Apache ODE (Orchestration Director Engine) is a kind of workflow engine supported by VP-UML. Select this option to create a WSDL diagram for a synchronous process that can be executed by ODE. Once a synchronous process is being invoked, its operations had to be completed before the invoking process move on.
Create ODE Asynchronous	Apache ODE (Orchestration Director Engine) is a kind of workflow engine supported by VP-UML. Select this option to create a WSDL diagram for an asynchronous process that can be executed by ODE. Once an asynchronous process is being invoked, the invoking process proceed with the asynchronous process in parallel without waiting it to complete.
Create Oracle Synchronous	Oracle workflow engine is a kind of workflow engine supported byVP-UML . Select this option to create a WSDL diagram for a synchronous process that can be executed by Oracle. Once a synchronous process is being invoked, its operations had to be completed before the invoking process move on.
Create Oracle Asynchronous	Apache ODE (Orchestration Director Engine) is a kind of workflow engine supported by VP-UML. Select this option to create a WSDL diagram for an asynchronous process that can be executed

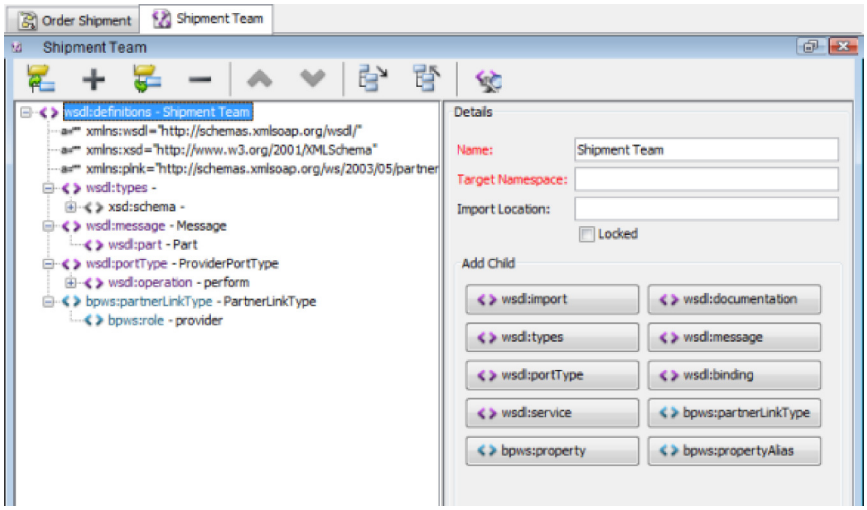
by Oracle. Once an asynchronous process is being invoked, the invoking process proceed with the asynchronous process in parallel without waiting it to complete.

Description of options for creating WSDL diagram

Starting from a blank WSDL document...

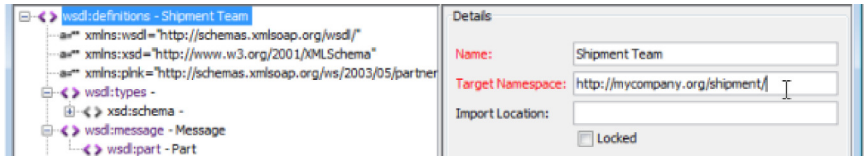
If you create a blank WSDL diagram (document) there are some points that you need to pay attention to in order to create an executable definition.

1. Take **Oracle Synchronous** as an example. Once selected, a WSDL diagram will be created, with a tree on the left hand side listing the WSDL file structure, and the **Details** pane on the right, where you can edit the chosen node in tree.



WSDL diagram is created

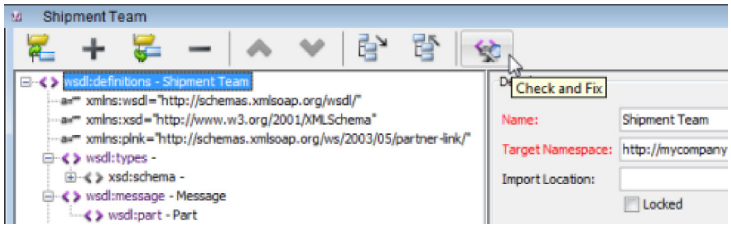
2. Those fields labelled red in **Details** pane are fields that you must enter in order to export BPEL. One of them is the **Target Namespace** field of the root wsdl:definition node. Things that you will name in the WSDL definition (e.g. a message, portType) will automatically becomes part of the target namespace.



Defining target namespace

3. Navigate through the tree and edit the nodes in it. Some of the properties are preset, but you can change them. Again, do remember that the red fields are compulsory. For instance, target namespace for xsd:schema.

4. At the end, click the button **Check and Fix** in toolbar, which helps you make appropriate changes to the WSDL definition in further (if you haven't done), such as to add property *xmlns* in wsdl:definitions element which refers to the target namespace specified.



Check and fix

Summary of WSDL elements

wsdl:definition

The WSDL definition element is the root element which may contain elements like wsdl:import, wsdl:documentation, wsdl:types, wsdl:message, wsdl:portType, wsdl:binding and wsdl:service. In wsdl:definition, the target namespace (attribute) must be specified. Elements contained by wsdl:definition will be part of the target namespace specified. Here is an example of WSDL definitions, with target namespace and some other fundamental attributes defined.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
  targetNamespace="http://mycompany.org/shipment/"
  xmlns="http://mycompany.org/shipment/"
  xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
  xmlns:ns1="http://mycompany.org/shipment/"
  xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
```

```
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 ...
 <wsdl:types>...</wsdl:types>
 <wsdl:message>...</wsdl:message>
 <wsdl:portType>...</wsdl:portType>
 ...

</wsdl:definitions>
```

wsdl:types

The WSDL types element is responsible for describing data types used by the operation(s) within the web service. To be clear, the type definition is to be used as input, output and/or fault types of operations. Most often data types are specified using XML schema. Here is an example of WSDL types element that defines an input type, an output type and a fault type.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
 targetNamespace="http://mycompany.org/shipment/"
 xmlns="http://mycompany.org/shipment/"
 xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
 xmlns:ns1="http://mycompany.org/shipment/"
 xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">

 ...

 <wsdl:types>
  <xsd:schema targetNamespace="http://mycompany.org/shipment/">
  <xsd:element name="requestReceived" type="typeRequestReceived"/>
  <xsd:complexType name="typeRequestReceived">
   <xsd:sequence>
    <xsd:element name="receiveDate" type="xsd:date"/>
   </xsd:sequence>
  </xsd:complexType>
  <xsd:element name="requestBody" type="xsd:string"/>
   <xsd:element name="invalidRequest" type="xsd:string"/>
  </xsd:schema>
 </wsdl:types>
 ...
</wsdl:definitions>
```

wsdl:message

The WSDL message defines the data for input or output of an operation. Each WSDL message can include one or more WSDL parts element. A part serves as a parameter of WSDL operation.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
 targetNamespace="http://mycompany.org/shipment/"
 xmlns="http://mycompany.org/shipment/"
 xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
 xmlns:ns1="http://mycompany.org/shipment/"
 xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 ...

 <wsdl:message name="getOrderRequest">
  <wsdl:part element="requestReceived" name="num" type="xsd:string"/>
  <wsdl:part name="createDate" type="xsd:date"/>
 </wsdl:message>

 ...

</wsdl:definitions>
```

wsdl:portType (Interface)

The WSDL portType, also known as interface, defines operations in a web service. It encloses the operations that can be performed, and each operation contains the input and output which refer to the messages (wsdl:message) defined. You may also add a fault element in addition to input and output for defining the message to send to client when a fault happen.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
 targetNamespace="http://mycompany.org/shipment/"
 xmlns="http://mycompany.org/shipment/"
```

```
 xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
 xmlns:ns1="http://mycompany.org/shipment/"
 xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 ...

 <wsdl:portType name="shipment">
  <wsdl:operation name="shipOrder">
   <wsdl:input message="getOrderRequest" name="getOrderRequest"/>
   <wsdl:output message="getOrderRequest" name="getOrderResponse"/>
  </wsdl:operation>
 </wsdl:portType>

 ...

</wsdl:definitions>
```

wsdl:binding

For ODE, in order to make your web service accessible by others, you must defining the WSDL binding to describe the how your web service is bound to a network protocol.

```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
 targetNamespace="http://mycompany.org/shipment/"
 xmlns="http://mycompany.org/shipment/"
 xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
 xmlns:ns1="http://mycompany.org/shipment/"
 xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 ...

 <wsdl:binding name="mybinding" type="shipment">
  <soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
  <wsdl:operation name="shipOrder">
   <soap:operation soapAction="" style="document"/>
   <wsdl:input name="order">
    <soap:body namespace="http://mycompany.org/shipment/" use="literal"/>
   </wsdl:input>
   <wsdl:output name="order">
    <soap:body namespace="http://mycompany.org/shipment/" use="literal"/>
   </wsdl:output>
  </wsdl:operation>
 </wsdl:binding>

 ...

</wsdl:definitions>
```

wsdl:service

For ODE, the WSDL service element defines the end points (i.e. address) of web service. It contains one or more port elements which references the binding element.

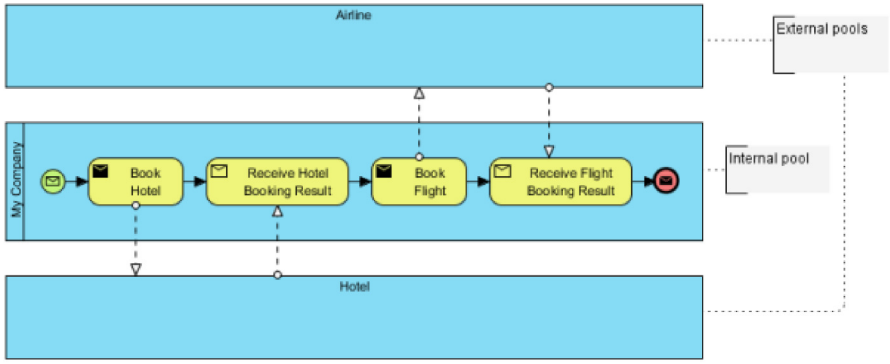
```
<?xml version="1.0" encoding="UTF-8"?>
<wsdl:definitions name="Shipment Team"
 targetNamespace="http://mycompany.org/shipment/"
 xmlns="http://mycompany.org/shipment/"
 xmlns:bpws="http://docs.oasis-open.org/wsbpel/2.0/process/executable"
 xmlns:ns1="http://mycompany.org/shipment/"
 xmlns:plnk="http://docs.oasis-open.org/wsbpel/2.0/plnktype"
 xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema">
 ...

 <wsdl:service name="OrderShipment">
  <wsdl:port binding="mybinding" name="shipmentEndPoint">
   <soap:address location="http://localhost:8080/ode/processes/ProviderPortService"/>
  </wsdl:port>
 </wsdl:service>
 ...

</wsdl:definitions>
```

Pool

In terms of business process modeling, participants of a process, such as companies or departments, are modeled by pools. When dealing with the mapping from BPMN to BPEL, each pool represents a business process.

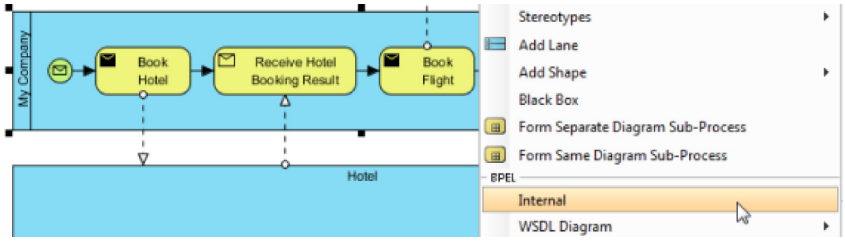
Internal and external pool

Internal pool refers to the pool that contains the flow objects of the execution process. When drawing a business process diagram for mapping with BPEL, there must be one and only one internal pool. You can create external pools which represent external partners or external process that interacts with the internal pool. Usually, external pools (or participants in business perspectively) are out of the interest for internal process, and hence presented as black box.



Internal and external pools

To define an internal pool, right click on the pool to select **Internal** from the popup menu.

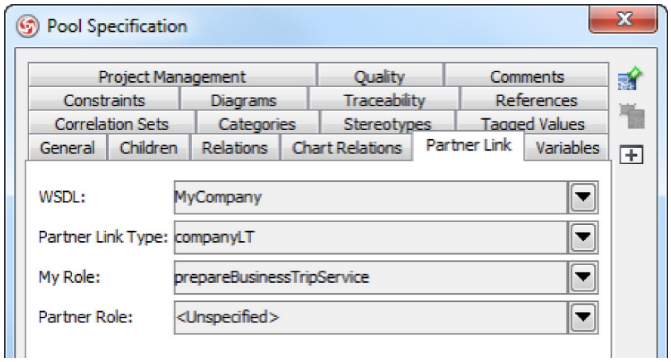


Set a pool to be internal pool

NOTE: There **MUST** be one and **ONLY** one internal pool per each business process diagram you need to export BPEL.

Partner link types

In every process, partner link types need to be defined. Partner link types define the interaction between a process and the parties. To edit partner link types, right click on the pool and select **Open Partner Link Specification...** from the popup menu. In the specification dialog box there are four fields you can fill in.



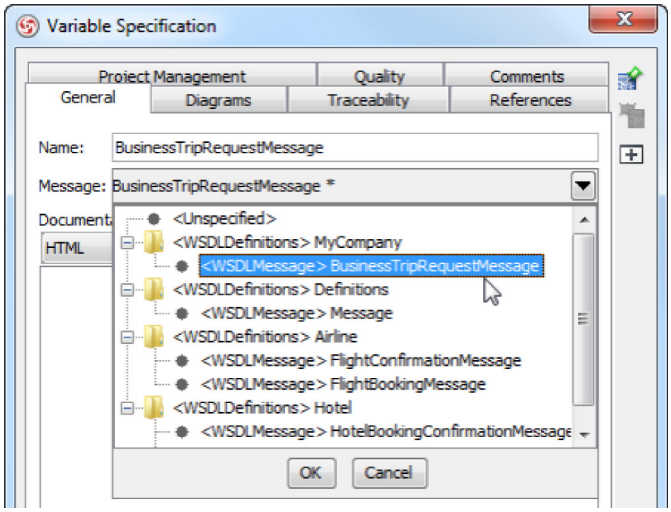
Partner link of pool

Field	Description
WSDL	The WSDL definition where the service needed was defined.
Partner Link Type	The interaction between the process and pool.
My Role	For internal pool, specify provider to be my role. For external pool, specify requester to be my role.
Partner Role	For internal pool, specify requester to be partner role. For external pool, specify provider to be partner role.

Variables

A variable correspond to a message that send to/from partners. It also can be defined for internal logic usage. You need to add variable(s) to internal pool and select the WSDL message type. To add a variable:

1. Right click on a pool and select **Open Variables Specification...** from the popup menu.
2. In the specification dialog box, click **Add...** at the bottom of dialog box.
3. Give a name to the variable.
4. Click on the **Message** drop down menu and select the WSDL message.

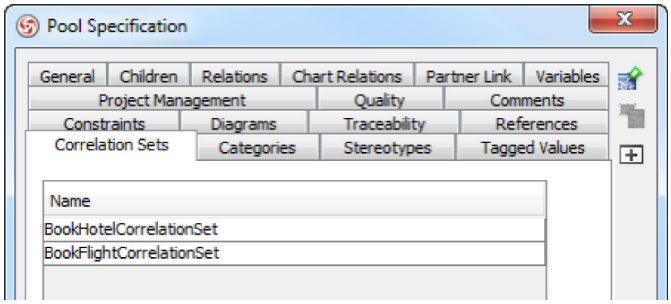


Selecting a WSDL message for variable

Correlation sets

When an asynchronous BPEL process flows, it will invoke an external process, and wait for the external process's call back before continuing. When the workflow engine receives a message, it need to know which instance the message should pass to. In this case, some values within the received message can be used as id for identifying the instance the message should pass to. Such ID is represented by a correlation set. To add a correlation set:

1. Right click on a pool and select **Open Correlation Sets Specification...** from the popup menu.
2. In the specification dialog box, click **Add...** at the bottom of dialog box.
3. Give a name to the set and click **OK** to confirm.



Correlation Sets added

Related Resources

The following resources may help you learn more about the topic discussed in this page.

- New to VP-UML? We have a lot of UML tutorials written to help you get started with VP-UML
- Visual Paradigm on YouTube
- Visual Paradigm Know-How - Tips and tricks, Q&A, solutions to users' problems
- Contact us if you need any help or have any suggestion

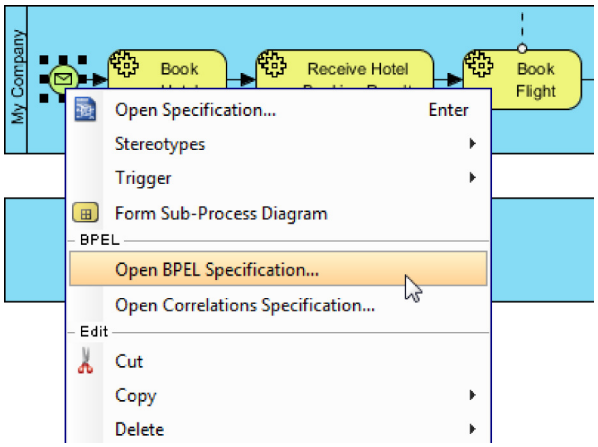
Start Event (Receive)

To represent the initiation of a BPEL process, you need a start event, which correspond to, in BPEL a receive element with attribute createInstance set to yes. You can specify the partnerLink, portType and operation to be invoked by the start event.

NOTE: An alternative way to start a process is to start by a receive task. Create a receive task, open its specification and check **Instantiate** in its **General** tab.

Editing BPEL properties

To edit BPEL properties for a start event, right click on the start event and select **Open BPEL Specification...** from the popup menu. In the specification dialog box you can specify partner link, operation and variable.



Open BPEL specification

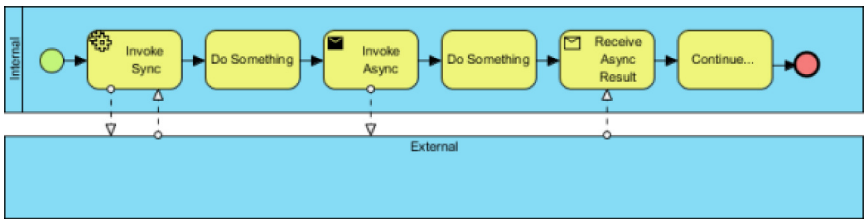
Related Resources

The following resources may help you learn more about the topic discussed in this page.

- New to VP-UML? We have a lot of UML tutorials written to help you get started with VP-UML
- Visual Paradigm on YouTube
- Visual Paradigm Know-How - Tips and tricks, Q&A, solutions to users' problems
- Contact us if you need any help or have any suggestion

Invoke (Task)

In BPEL, an invoke activity (i.e. an <invoke> element) is used by a process to invoke Web service provided by partner. An invocation can be either synchronous (request and response) or asynchronous (one-way). For a synchronous invocation, only one task is drew. For an asynchronous invocation, two tasks are drew, with one connecting to external pool and another connecting from external pool.



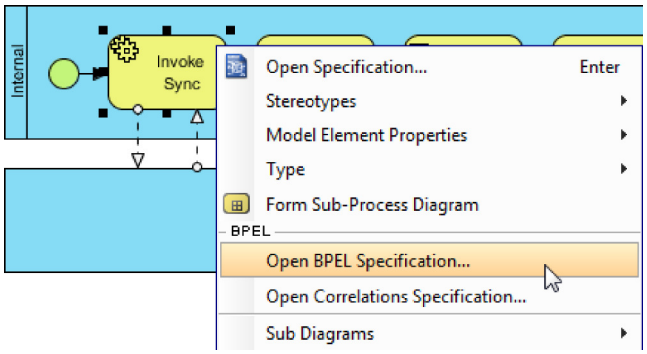
A sample BPD with tasks

Note that not all types of task are supported by BPEL. Below is a list of supported type. The different in type support different kinds of BPEL properties.

- Service
- Send
- Receive
- User
- Manual
- Script
- Typeless (Without type)

Editing BPEL properties

To edit BPEL properties for a task, right click on the task and select **Open BPEL Specification...** from the popup menu. In the specification dialog box you can specify partner link, operation and/or variable.



To open BPEL specification

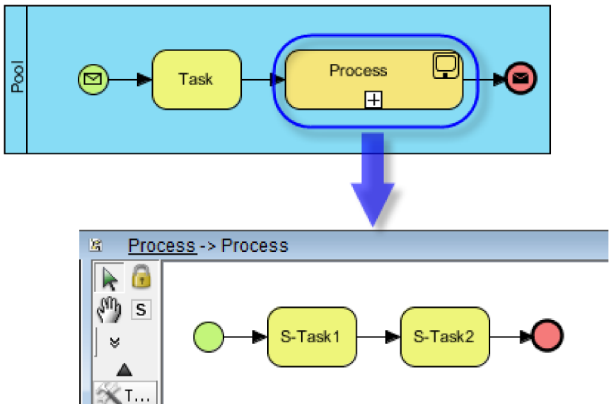
Related Resources

The following resources may help you learn more about the topic discussed in this page.

- New to VP-UML? We have a lot of UML tutorials written to help you get started with VP-UML
- Visual Paradigm on YouTube
- Visual Paradigm Know-How - Tips and tricks, Q&A, solutions to users' problems
- Contact us if you need any help or have any suggestion

Sub-process

In BPMN, there are two kinds of activity - task and sub-process. To represent either an empty BPEL activity or an invoke of BPEL activity, you use a task with proper type set. Sub-process, on the other hand acts as a placeholder of a set of tasks. You can draw a sub-process, expand it and draw the tasks in the sub-process diagram. Note that only sub-process with type Embedded is considered in BPEL generation. The other types will be ignored.



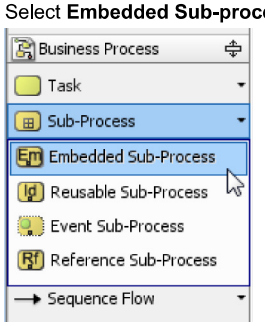
A sub-process and its sub-process diagram

When you generate BPEL, the flow modeled in sub-process diagram will be merged to the ordinary flow. The following code fragment shows a BPEL file generated from the above BPDs. The empty activities STask1 and STask2 were modeled in the sub-process diagram, and they follow the empty activty Task in the main flow.

```
<?xml version="1.0" encoding="UTF-8"?>
<process name="BusinessProcessDiagram1" targetNamespace="http://BusinessProcessDiagram1" xmlns="http://schemas.xmlsoap.org/ws/2003/03/
business-process/" xmlns:Pool="http://a" xmlns:tns="http://BusinessProcessDiagram1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <partnerLinks>
      <partnerLink myRole="provider" name="Pool" partnerLinkType="Pool:PartnerLinkType"/>
   </partnerLinks>
   <variables>
      <variable messageType="Pool:Message" name="Variable"/>
   </variables>
   <sequence>
      <receive createInstance="yes" operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
      <empty name="Task"/>
      <empty name="STask1"/>
      <empty name="STask2"/>
      <reply operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
   </sequence>
</process>
```

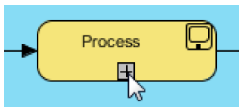
Creating an embedded sub-process

1. Select **Embedded Sub-process** in diagram toolbar.



Selecting Embedded Sub-Process from diagram toolbar

2. Click inside the pool to create an embedded sub-process. If you tend to create a sub-process through the resource-centric interface of the previous flow object, you can change the type by right clicking on the sub-process and selecting **Type > Embedded Sub-Process** from the pop-up menu.
3. Click on the + icon at the bottom of embedded sub-process. You can now model the sub-process in the diagram created.



To expand a sub-process

Gateway with WS-BPEL

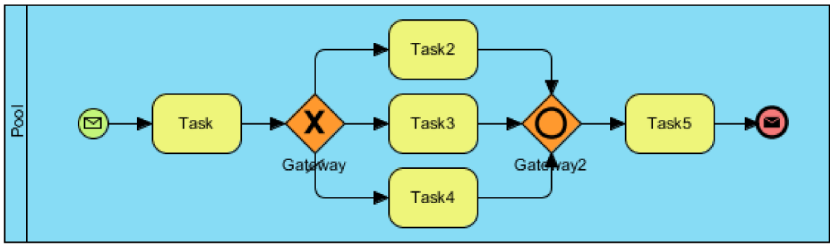
Different types of gateway modeling

Although gateway does not come along with any BPEL property, the place you put it in a flow does influence the flow of BPEL process, hence the content of BPEL. Below are several types of gateway modeling, with respect to the generated BPEL content.

Scenario 1 - As switch

By drawing a pair of gateway, with the beginning one typed as XOR and the other one typeed as OR, this stands for a selection (switch in terms of BPEL) of the task exhibited from the XOR gateway. To set the type of gateway, right click on the gateway and select **Type**, then the type from the popup menu.

Note also that you can set up the default flow (otherwise in terms of BPEL) by specifying the condition type of the sequence flow as Default. To set the condition type of a sequence flow, right click on the flow and select **Condition Type**, then the type from the popup menu.

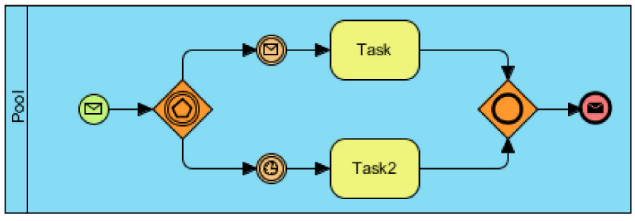


Drawing in this way will cause switch to be generated

```
<?xml version="1.0" encoding="UTF-8"?>
<process name="BusinessProcessDiagram1" targetNamespace="http://BusinessProcessDiagram1" xmlns="http://schemas.xmlsoap.org/ws/2003/03/
business-process/" xmlns:Pool="http://mypool" xmlns:tns="http://BusinessProcessDiagram1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <partnerLinks>
      <partnerLink myRole="provider" name="Pool" partnerLinkType="Pool:PartnerLinkType"/>
   </partnerLinks>
   <variables>
      <variable messageType="Pool:Message" name="Variable"/>
   </variables>
   <sequence>
      <receive createInstance="yes" operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
      <empty name="Task"/>
      <switch name="Gateway">
         <case condition="true()">
            <sequence>
               <empty name="Task2"/>
            </sequence>
         </case>
         <case condition="false()">
            <sequence>
               <empty name="Task3"/>
            </sequence>
         </case>
         <otherwise>
            <sequence>
               <empty name="Task4"/>
            </sequence>
         </otherwise>
      </switch>
      <empty name="Task5"/>
      <reply operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
   </sequence>
</process>
```

Scenario 2 - As pick

By drawing an event-driven XOR gateway, follwed by different events, this indicates the selection of path base on a condition.



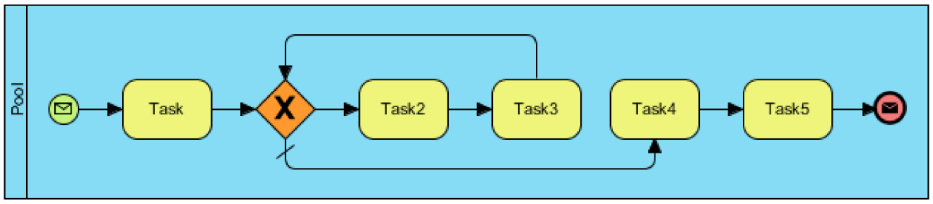
Drawing in this way will cause pick to be generated

```
<?xml version="1.0" encoding="UTF-8"?>
<process name="BusinessProcessDiagram1" targetNamespace="http://BusinessProcessDiagram1" xmlns="http://schemas.xmlsoap.org/ws/2003/03/
business-process/" xmlns:Pool="http://b" xmlns:tns="http://BusinessProcessDiagram1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
```

```
  <partnerLinks>
    <partnerLink myRole="provider" name="Pool" partnerLinkType="Pool:PartnerLinkType" partnerRole="requester"/>
  </partnerLinks>
  <variables>
    <variable messageType="Pool:Message" name="Variable"/>
  </variables>
  <sequence>
    <receive createInstance="yes" operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
    <pick createInstance="no">
      <onMessage operation="continue" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable">
        <sequence>
          <empty name="Task"/>
        </sequence>
      </onMessage>
      <onAlarm for="'PT5S'">
        <sequence>
          <empty name="Task2"/>
        </sequence>
      </onAlarm>
    </pick>
    <invoke inputVariable="Variable" operation="performCallback" partnerLink="Pool" portType="Pool:RequesterPortType"/>
  </sequence>
</process>
```

Scenario 3 - As looping (while)

By having a sequence flow back into a gateway, forming a loop, this indicates the need of repeating a flow as long as a condition satisfy.



Drawing in this way will cause while to be generated

```
<?xml version="1.0" encoding="UTF-8"?>
<process name="BusinessProcessDiagram1" targetNamespace="http://BusinessProcessDiagram1" xmlns="http://schemas.xmlsoap.org/ws/2003/03/
business-process/" xmlns:Pool="http://mypool" xmlns:tns="http://BusinessProcessDiagram1" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <partnerLinks>
    <partnerLink myRole="provider" name="Pool" partnerLinkType="Pool:PartnerLinkType"/>
  </partnerLinks>
  <variables>
    <variable messageType="Pool:Message" name="Variable"/>
  </variables>
  <sequence>
    <receive createInstance="yes" operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
    <empty name="Task"/>
    <while condition="false()">
      <sequence>
        <empty name="Task2"/>
        <empty name="Task3"/>
      </sequence>
    </while>
    <empty name="Task4"/>
    <empty name="Task5"/>
    <reply operation="perform" partnerLink="Pool" portType="Pool:ProviderPortType" variable="Variable"/>
  </sequence>
</process>
```

Related Resources

The following resources may help you learn more about the topic discussed in this page.

- New to VP-UML? We have a lot of UML tutorials written to help you get started with VP-UML
- Visual Paradigm on YouTube
- Visual Paradigm Know-How - Tips and tricks, Q&A, solutions to users' problems
- Contact us if you need any help or have any suggestion