

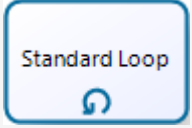
Palette

Activities Overview

Activities represent work or tasks carried out by members of the organization. They stand for manual or automatic tasks performed by an external system or user. Activities can be atomic or non-atomic (compound) and they are classified into tasks and sub-processes.

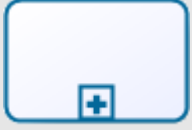




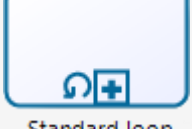

Tasks


ELEMENT	DESCRIPTION	NOTATION
Task	Is an atomic Activity within a Process flow. It is used when the work in the Process cannot be broken down to a finer level of detail.	
User Task	Is a typical workflow Task where a person performs the Task with the assistance of a software application.	
Service Task	Is a Task that uses some sort of service that could be a Web service or an automated application.	
Receive Task	Is a Task designed to wait for a message to arrive from an external participant (relative to the Process).	
Send Task	Is a Task designed to send a message to an external participant (relative to the Process).	
Script Task	Is a Task that is executed by a Business Process Engine. The Modeler defines a script in a language that the engine can interpret.	
Manual Task	Is a Task that is expected to be performed without the aid of any business process execution or any application.	
Business Rule Task	Offers a mechanism for the Process to provide input to a Business Rule Engine and get the output of calculations that the engine might provide.	
Multi-Instance Loop	Tasks may be repeated sequentially, behaving like a loop. The Multi-instance Loop iterates a predetermined number of times. The iterations occur sequentially or in parallel (simultaneously).	

Standard Loop	Tasks may be repeated sequentially, behaving like a loop. This feature defines a looping behavior based on a Boolean condition. The Activity will loop as long as the Boolean condition is true.	
---------------	--	---

Sub-process

A [sub-process](#) is a compound Activity that is included within a Process. Compound means that it can be broken down into lower levels, that is, it includes shapes and elements within it.

ELEMENT	DESCRIPTION	NOTATION
Sub-process	Is an Activity which internal details have been modeled using activities, gateways, Events, and sequence flows. The elements has a thin border.	 Subprocess
Reusable Sub-process	Identifies a point in the Process where a predefined Process is used. A reusable Sub-process is called a Call Activity in BPMN. The element has a thick border.	 Reusable Subprocess
Event Sub-process	A Sub-process is defined as an Event Sub-process when it is triggered by an Event. An Event Sub-Process is not part of the normal flow of its parent Process - there are no incoming or outgoing Sequence Flows.	 Event Subprocess
Transaction	Is a Sub-process whose behavior is controlled through a transaction protocol. It includes the three basic outcomes of a transaction: Successful Completion, Failed Completion and Cancel Intermediate Event.	 Transaction
Ad-Hoc Sub-process	Is a group of activities that has no REQUIRED sequence relationships. A set of activities can be defined, but the sequence and number of performances for the activities is determined by the resources of the activities.	 Ad-Hoc Subprocess
Standard loop	Sub-processes may be repeated sequentially, behaving like a loop. This feature defines a looping behavior based on a Boolean condition. The activity will loop as long as the Boolean condition is true.	 Standard loop
Multi-Instance loop	Sub-processes may be repeated sequentially, behaving like a loop. The Multi-instance Loop iterates a predetermined number of times. The iterations occur sequentially or in parallel (simultaneously).	 Multi-Instance sequential loop








		 Multi-Instance parallel loop
--	--	--

Events Overview










An Event is something that happens during the course of the Process, affecting the Process flow and normally has a trigger or result.






To make an event a throw or a catch event, right click on it and select *Is Throw*. This option will enable or disable its behavior (applies for certain events described below) .

Start Events





ELEMENT	DESCRIPTION	NOTATION
Start Event	Indicates where a particular Process starts. It does not have any particular behavior.	 Start Event
Message Start Event	Is used when a message arrives from a participant and triggers the start of the Process.	 Message
Timer Start Event	Is used when the start of a Process occurs on a specific date or cycle time (e.g., every Friday)	 Timer
Conditional Start Event	This type of Event triggers the start of a Process when a condition becomes true.	 Conditional
Signal Start Event	The start of the Process is triggered by the arrival of a signal that has been broadcast from another Process. Note that the signal is not a message; messages have specific targets, signals do not.	 Signal
Parallel Multiple Start Event	Indicates that there are multiple triggers required to start the Process. ALL triggers must be triggered before the Process is instantiated.	 Parallel Multiple
Multiple Start Event	This means that there are multiple ways of triggering the Process. Only one of them is required.	 Multiple








Intermediate events

ELEMENT	DESCRIPTION	NOTATION
Intermediate Event	Indicates where something happens somewhere between the start and end of a Process. It will affect the flow of the Process, but will not start or (directly) terminate the Process.	 Intermediate Event
Message Event	<p>Indicates that a message can be sent or received.</p> <p>If a Process is waiting for a message and it is caught the Process will continue its flow.</p> <p>A catch Message Event waits for a message to arrive and once the message has been received, the Process will continue. The Event marker in this instance will be unfilled.</p> <p>A throw Message Event sends a message to an external participant. The filled Event marker is allocated to the throw message.</p>	 Message Throw  Message Catch
Timer Event	Indicates a delay within the Process. This type of Event can be used within the sequential flow indicating a waiting time between activities.	 Timer
Escalation Event	The Event indicates an escalation through the Process.	 Escalation
Compensation Event	Enables the handling of compensations. When used within the sequential flow of a Process they indicate that compensation is necessary.	 Compensate
Conditional Event	This Event is triggered when a condition becomes true.	 Conditional
Link Event	<p>This Event is used to connect two sections of the Process.</p> <p>Link Events can be used to create looping situations or to avoid long Sequence Flow lines.</p> <p>If there are two link events on a process (one catch and one throw) the Modeler will understand they are linked together. If there is one catch and two throw, the Modeler will understand both throws are received by the catch. If there are several catch and throw events the name of the 'pairs' must match for the Modeler to understand which throw belongs to which catch.</p>	 Link Throw  Link Catch

Signal Event	These Events are used to send or receive signals within or across the Process. A signal is similar to a signal flare that is shot into the sky for anyone who might be interested to notice and then react.	 Signal Throw
	If the Event is used to throw the signal, the signal Event marker will be filled. Alternatively, the unfilled Event marker is allocated to catch the message.	 Signal Catch
Multiple Event	This means that there are multiple triggers assigned to the Event.	 Multiple Throw
	When used to catch the trigger, only one of the assigned triggers is required and the Event marker will be unfilled.	 Multiple Catch
Parallel multiple Event	This means that there are multiple triggers assigned to the Event. Unlike the normal Multiple Intermediate Event, ALL of the assigned triggers are required for the Event to be triggered.	 Parallel Multiple





Intermediate Events Attached to an Activity Boundary






ELEMENT	DESCRIPTION	NOTATION
Message Event	If a message Event is attached to the boundary of an activity, it will change the normal flow into an exception flow when a message is received.	 Interrupting
	If the Event interrupts the activity to which it is attached, the boundary of the Event is solid, if not it is dashed.	 Non interrupting
Timer Event	If a Timer Event is attached to the boundary of an activity, it will change the normal flow into an exception flow when a cycle time is completed or a specific time-date is reached.	 Interrupting
	If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.	 Non interrupting

<p>Escalation Event</p>	<p>If attached to the boundary of an Activity, the Intermediate Event catches an Escalation.</p> <p>If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.</p>	 <p>Interrupting</p>  <p>Non interrupting</p>
<p>Error Event</p>	<p>A catch Intermediate Error Event can only be attached to the boundary of an Activity.</p> <p>It reacts to (catches) a named Error, or to any Error if a name is not specified.</p> <p>An Error Event always interrupts the Activity to which it is attached, i.e., there is not a non-interrupting version of this Event. Thus the boundary of the Event is always solid.</p>	 <p>Error</p>
<p>Cancel Event</p>	<p>This Event is used within a Transaction Sub-Process and must be attached to the boundary of one.</p> <p>It shall be triggered if a Cancel End Event is reached within the Transaction Sub-Process. It also shall be triggered if a Transaction Protocol Cancel Message has been received while the transaction is being performed.</p> <p>A Cancel Event always interrupts the Activity to which it is attached, i.e., there is not a non-interrupting version of this Event. Thus the boundary of the Event is always solid.</p>	 <p>Cancel</p>
<p>Compensation Event</p>	<p>When attached to the boundary of an Activity, this Event is used to catch the Compensation Event. When it occurs, the compensation activity will be performed.</p> <p>Interrupting a non-interrupting aspect of other Events does not apply in the case of a Compensation Event, thus the boundary of the Event is always solid.</p>	 <p>Compensate</p>
<p>Conditional Event</p>	<p>If a Conditional Event is attached to the boundary of an Activity, it will change the normal flow into an exception flow when a business condition is fulfilled.</p> <p>If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.</p>	 <p>Interrupting</p>  <p>Non interrupting</p>

Signal Event	If a Signal Event is attached to the boundary of an Activity, it will change the normal flow into an exception flow when a signal is received.	 Interrupting
	If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.	 Non interrupting
Multiple Event	When attached to the boundary of an Activity, it will change the normal flow into an exception flow when one of the assigned triggers is caught.	 Interrupting
	If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.	 Non interrupting
Parallel multiple Event	Unlike the Multiple Event, when attached to the boundary of an Activity, it will change the normal flow into an exception flow when ALL of the assigned triggers are caught.	 Interrupting
	If the Event interrupts the Activity to which it is attached, the boundary of the Event is solid, if not it is dashed.	 Non interrupting





End Events

ELEMENT	DESCRIPTION	NOTATION
End Event	Indicates when the Process ends.	 End
Message End	Indicates that a message is sent when the flow has ended.	 Message
Escalation End	Indicates that an Escalation is necessary when the flow ends.	 Escalation
Error End	Indicates that a named Error should be generated. All currently active threads of the Process are terminated.	 Error

	The Error will be caught by a Catch Error Intermediate Event.	
Cancel End	Is used within a Transaction Sub-Process. It indicates that the Transaction should be canceled and an alternative flow can be performed.	 Cancel
Compensation End	Handles compensations. If an activity is identified, and it was successfully completed, the activity will be compensated.	 Compensation
Signal End	Indicates that a signal is sent when the flow has ended.	 Signal
Multiple End	This means that there are multiple consequences of ending the flow. All of them will occur.	 Multiple
Terminate End	Ends the Process and all its activities immediately.	 Terminate



Gateways Overview

Gateways are used to control the divergence and convergence of sequence flows. They determine ramifications, bifurcations, combinations and merges in the Process. The term "Gateway" implies that there is a gating mechanism that either allows or disallows passage through the Gateway.

ELEMENT	DESCRIPTION	NOTATION
Exclusive Gateway	As Divergence: It is used to create alternative paths within the Process, but only one is chosen. As Convergence: It is used to merge alternative paths.	  Exclusive gateway Exclusive gateway
Event Based Gateway	Represents a branching point in the Process where the alternative paths that follow the Gateway are based on Events that occur. When the first Event is triggered, the path that follows that Event will be used. All the remaining paths will no longer be valid.	 Event Based gateway
Exclusive Event Based Gateway	Is a variation of the Event based gateway and it is only used to instantiate Processes. One of the Events of the Gateway configuration must be triggered in order to create a Process instance. It must have NO incoming transitions.	 Exclusive Event Based gateway

Parallel Event Based Gateway	Unlike the exclusive Event based Gateway, ALL the Events of the Gateway configuration must be triggered in order to create a Process instance. It must have NO incoming transitions.	 Parallel Event Based gateway
Parallel Gateway	As Divergence: is used to create alternative paths without checking any conditions. As Convergence: is used to merge alternative paths, the gateway waits for all incoming flows before it continues.	 Parallel gateway
Complex Gateway	As Divergence: is used to control complex decision points in the Process. It creates alternative paths within the Process using expressions. As Convergence: Allow continuing to the next point of the Process when a business condition becomes true.	 Complex gateway
Inclusive Gateway	As Divergence: represents a branching point where alternatives are based on conditional expressions. The TRUE evaluation of one condition does not exclude the evaluation of the other conditions. All evaluations of a TRUE condition will be traversed by a token. As Convergence: is used to merge a combination of alternative and parallel paths.	 Inclusive gateway

Data

ELEMENT	DESCRIPTION	NOTATION
Data Objects	Provides information about how documents, data and other objects are used and updated during the Process.	
Data Store	Provides a mechanism for activities to retrieve or update stored information that will exist beyond the scope of the Process.	

Artifacts Overview

[Please click for further information about Artifacts](#)










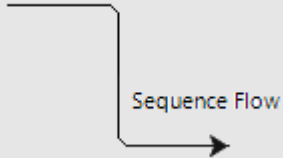

ELEMENT	DESCRIPTION	NOTATION
Group	Is an Artifact that provides a visual mechanism to group elements of a diagram informally.	
Annotation	Is a mechanism for a modeler to provide additional information for the reader of a BPMN Diagram.	

Image	Enables an image stored on your computer to be inserted into the diagram.	
Header	Displays the diagram properties (author, version, description), and it is updated automatically with the information contained in those properties. To edit its information, it is only needed to edit the diagram's properties.	
Formatted Text	This Artifact enables rich text to be inserted into the diagram to provide additional information.	
Custom Artifacts	Helps to define and use your own Artifacts. Artifacts provide the capability of showing additional information about the Process that is not directly related to the flow.	

Swimlanes

ELEMENT	DESCRIPTION	NOTATION
Pool	A Pool is a container of a single Process (contains the sequence flows between activities).	
Lane	Is a sub-partition within the Process. Lanes are used to differentiate elements as internal roles, position, department, etc. They represent functional areas that may be responsible for tasks.	
Milestone	Is a sub-partition within the Process. It can indicate different stages during the Process.	



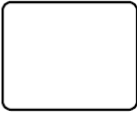
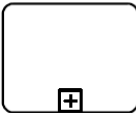





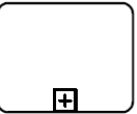


Connectors

ELEMENT	DESCRIPTION	NOTATION
Sequence Flow	A Sequence Flow is used to show the order that Activities will be performed in the Process.	
Association	It used to associate information and Artifacts with Flow Objects. It also shows the activities used to compensate for an activity.	

<p>Message Flow</p>	<p>Is used to show the flow of messages between two entities that are prepared to send and receive them.</p>	
---------------------	--	---

The following is a table that reflects all the possible connections using Message flow. The arrow pointing upwards shows what CAN be connected. Anything else outside the table should not be connected using a Message flow.

Table 7.4 – Message Flow Connection Rules

From\To						
						
	^	↗	↗	↗	↗	
	^	↗	↗	↗	↗	
	^	↗	↗	↗	↗	
	^	↗	↗	↗	↗	
	^	↗	↗	↗	↗	