BPMN 2.0 – Business Process Model and Notation

Activities

Task

A **task** is a unit of work (the job to be performed). When marked with a [+] symbol, it indicates a collapsed **sub-process**, an activity that can be refined.

Transaction

A **transaction** is a set of activities that logically belong together; it might follow a specified transaction protocol.

Event Subprocess

An **event subprocess** is placed into a process or subprocess. It is activated when its start event gets triggered and can interrupt the higher level process context or run in parallel (noninterrupting), depending on the start event.

Call Activity

A **call activity** is a wrapper for a globally defined task or process reused in the current process. A call to a process is marked with a [+] symbol.

Activity Markers
describe the execution behavior of activities:

- Subprocess Marker
- Loop Marker
- Parallel MI Marker
- Sequential MI Marker
- Ad Hoc Marker
- Compensation Marker

Task Types
specify the nature of the action to be performed:

- Send Task
- Receive Task
- User Task
- Manual Task
- Business Rule Task
- Service Task
- Script Task

Sequence Flow
defines the execution order of the process elements.

Default Flow
is the default branch to be chosen if all other conditions evaluate to false.

Conditional Flow
has a condition assigned that defines whether or not the flow is used.

Events

	Start			Intermediate				End
	Standard	Event Subprocess Interrupting	Event Subprocess Non-Interrupting	Catching	Boundary Interrupting	Boundary Non-Interrupting	Throwing	Standard
None: Untyped events, indicate start point, state changes or final states.	○						○	○
Message: Receiving and sending messages.	✉	✉	✉	✉	✉	✉	✉	✉
Timer: Cyclic timer events, points in time, time spans or timeouts.	🕒	🕒	🕒	🕒	🕒	🕒		
Escalation: Escalating to a higher level of responsibility.		⮝	⮝		⮝	⮝	⮝	⮝
Conditional: Reacting to changed conditions.								
Link: Two corresponding link events represent a sequence flow.				⇨			➡	
Error: Catching or throwing named errors.		⚡			⚡			⚡
Cancel: Reacting to cancelled transactions or triggering cancellation.					✕			✕
Compensation: Handling or triggering compensation.		◁◁			◁◁		◀◀	◀◀
Signal: Signalling across different processes. A signal thrown can be caught multiple times.	△	△	△	△	△	△	▲	▲
Multiple: Catching one out of a set of events. Throwing all events defined.	⬠	⬠	⬠	⬠	⬠	⬠	⬟	⬟
Parallel Multiple: Catching all out of a set of parallel events.	✚	✚	✚	✚	✚	✚		
Terminate: Triggering the immediate termination of a process or sub-process.								●

Gateways

Exclusive Gateway
It merges the incoming sequence flows while directly passing on incoming tokens. When splitting, exactly one branch is activated.

Event-based Gateway
Is always followed by catching events or receive tasks. The branch whose event/task is triggered first is activated.

Parallel Gateway
When used to split the sequence flow, all outgoing branches are activated simultaneously. When merging parallel branches, it waits for all incoming branches to complete before triggering the outgoing flow.

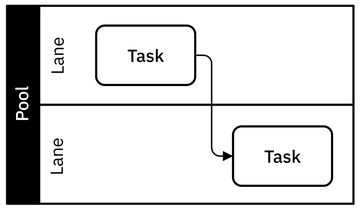
Inclusive Gateway
When splitting, one or more branches are activated. All active incoming branches must complete before merging.

Complex Gateway
Complex merging and branching behavior that is not captured by other gateways.

Exclusive Event-based Gateway (instantiate)
Each occurence of a subsequent event starts a new process instance.

Parallel Event-based Gateway (instantiate)
The occurence of all subsequent events starts a new process instance.

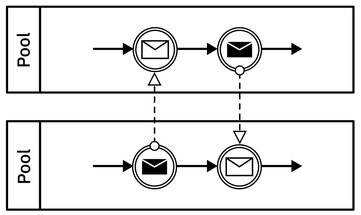
Participants



Pools and **lanes** represent responsibility for activities.

A pool or a lane can be an organization, a role, or a system.

Message flow
Message flow symbolizes the message exchange. Message flows can be attached to pools, sub-processes, activities and message events.



Message Exchange/Collaboration
The collaboration of multiple processes can be defined through a combination of message and sequence flows.

Data

A **data object** represents information flowing through the process, such as business documents, emails, or letters.

A **collection data object** represents a collection of information, e.g., a list of order items.

A **data input** is an external input for the entire process. A kind of input parameter.

Input

A **data output** is a data result of the entire process. A kind of output parameter.

Output

A **data association** is used to associate data elements with flow objects such as activities, events and sub-processes.

A **data store** is a place where the process can read or write data, e.g. a database or a filing cabinet. It persists beyond the lifetime of the process instance.

Data Store

Artifacts

Text Annotation

Groupings
visually combine elements

(directed) Association

links artifacts and flow objects

Individual Symbols

can be used as artifacts

CAMUNDA

camunda.com

© 2024