Activities

A **Task** is a unit of work, the job to be performed. When marked with a central symbol it indicates a **Subprocess**, an activity that can be refined.

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

A **Call Activity** is a wrapper for a globally defined Subprocess or Task that is reused in the current process.

**Activity Markers**

- **Subprocess Marker**
- **Loop Marker**
- **Parallel**
- **Ad Hoc Marker**
- **Compensation Marker**
- **User Task**
- **Manual Task**
- **Script Task**

**Task Types**

Types specify the nature of the action to be performed:

- **Send Task**
- **Receive Task**
- **User Task**
- **Manual Task**
- **Script Task**

**Sequence Flow**

A sequence flow defines the execution order of activities.

**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.

Gateways

**Exclusive Gateway**

When splitting, it routes the sequence flow to exactly one of the outgoing branches. When merging, it waits one incoming branch to complete before triggering the outgoing sequence flow.

**Event-based Gateway**

Is always followed by catching events or receive tasks. Sequence flow is routed to the subsequent event/task which happens first.

**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**

Each occurrence of an event starts a new process instance.

**Parallel Event-based Gateway**

The occurrence of all subsequent events starts a new process instance.

Conversations

A **Communication** defines a set of logically related message exchanges. When marked with a central symbol it indicates a **Subconversation**, a compound conversation element.

A **Conversation Link** connects Communications and Participants.

A Forked Conversation Link connects Communications and multiple Participants.

**Activity Markers**

- **Subprocess Marker**
- **Parallel**
- **Ad Hoc Marker**
- **Compensation Marker**

**Markers indicate execution behavior of activities**

**Task Types**

Types specify the nature of the action to be performed:

- **Send Task**
- **Receive Task**
- **User Task**
- **Manual Task**
- **Script Task**

**Sequence Flow**

A sequence flow defines the execution order of activities.

**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.

Choreographies

A **Choreography Task** represents an Interaction (Message Exchange) between two Participants.

**Multiple Participants Marker**

Indicates a set of participants of the same kind.

A **Choreography Subprocess** contains a refined choreography with several interactions.

Collaboration Diagram

**Participants**

- **Participant A**
- **Participant B**
- **Participant C**

**Event Subprocess**

Provides a set of activities that logically belong together and might follow or integrate business rules.

**Message Flow**

- **Initiating Message**
- **Response Message**
- **Cancel Activity**

**Data Store**

- **Data Object**
- **Data Store**

Data

**A Data Input** is an external input for the entire process. It can be read by an activity.

**A Data Output** is a variable available as result of the entire process.

**A Data Object** represents information flowing through the process, such as business documents, e-mails, or letters.

**A Collection Data Object** represents a collection of information, e.g., a list of order items.

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

**A Message** is used to depict the contents of a communication between two Participants.

Swimlanes

**Pools (Participants) and Lanes**

Participants represent responsibilities for activities in a process. A pool or a lane can be an organization, a role, or a system. Lanes subdivide pools or other lanes hierarchically.

**Message Flow**

Shows the flow of information across organizational boundaries. Message flow can be attached to pools, activities, or message events.

**Order of message exchanges**

Can be specified by combining message flow and sequence flow.

Events

**Task Event**

- **Start**
- **Intermediate**
- **End**

- **Subprocess Start Event**
- **Subprocess Intermediate Event**
- **Subprocess End Event**

- **Intermediate Event**
- **Error End Event**

**Message Events**

- **Receive**
- **Send**
- **Throw**
- **Cancel**
- **Error**

**Triggering the**

- **Immediate termination of a process**

- **Cyclic timer events**

- **Catching or throwing**

- **Handling or**

- **Compensation**

- **Signalling**

- **Change**

- **Final states**

- **Interrupting**

- **Interrupting**

- **Intermediate**

- **Interruption**

- **Non interrupting**