BPMN 2.0 Tutorial

Daniel Brookshier
Distinguished Fellow
No Magic Inc.
About the Tutorial

- Generated from MagicDraw UML
- Based on current BPMN 2.0 for UML reference implementation.
- Developed by Daniel Brookshier, Distinguished Fellow, No Magic Inc. danielb@nomagic.com
TRACEABILITY

The best way to reduce risk is tracing requirements to the things that can bite you.
What is BPMN?

- Business Process Modeling Notation
- Developed by Business Process Management Initiative (BPMI), and is currently maintained by the Object Management Group since the two organizations merged in 2005
- Supports business process management for technical and business users
- Bridge communication gap between business process design and implementation
What?

- BPMN is simple
- Process diagrams business people like
- Less complex (business likes that too)
- Under the covers, technical enough for techies
What Does BPMN Not Do?

- State transitions
- Functional decomposition
- Organizational hierarchies
- Data modeling
What is BPMN Like?

- Similar to flowcharts and UML Activity diagrams
- Flow of activities with various messaging and data
- Can be used for service orchestration in SOA
Primary Components
Why BPMN?

- Standard notation
- Model concepts and/or implementation of business process
- Models high-level process concepts
- Notation is not complex
Issues With BPMN

- Limited complexity
- Process/conversation oriented
- Very high level
- Cannot see details of tasks or data
Solving BPMN Issues Via UML

- BPMN as an extension to UML
- Enhanced ability to implement complexity
- Link implementation with orchestration
- Greater tool support
- Fill in gaps with details state, decomposition, data, implementations

BPMN for UML specification in progress at OMG
Executable Verses Non-executable

- Process flows can be executable or non-executable
- Executable process follows specific rules
- Formal condition expressions are typically not included in non-executable form
- Executable does not have Manual, Abstract, and other non-execution elements
BPMN Diagram Zoo

- **Process** – Flow of activity, decisions, data and events
- **Collaboration** – Conversations and interactions (also process)
- **Choreography** – Tasks performed by participants and how participants coordinate interactions via messages.
Simple BPMN Process Diagram
Process with Pools and Lanes
Simple Collaboration

- Librarian
- Reader
- Book Reservation
- Conversation Link
- Participant

Conversation that represents multiple messages between participants connected by conversation links
Simple Collaboration
Understanding Collaborations
BPMN Choreography

- Sequence of interactions between Participants.
- Choreographies exist outside of or in between Pools.
- A Choreography Task is an atomic Activity in a Choreography Process.
- The task represents an Interaction, which is one or two Message exchanges between two Participants.
- Helps to show who initiates the activity and the first message.
Simple Choreography Task With Messages

- **Message One**
  - **Initiating participant (light)**
  - **Participant A**
    - **9999 Choreography Task**
      - **Participant B**
    - **Message Two**
  - **Message that initiates the choreography task (light).**
  - **Name and ID of task.**
  - **Non-initiating message (dark).**

- **Participant (dark)**
Setting Messages on Choreography

Message Flows from Process are used by Choreography Task

Select Message Flows

Show Message Flows related to Participants

OK  Cancel
Example BPMN Diagrams
Book Loan and Reservation (Librarian Perspective)
Loan a book (main scenario)
Loan a Book
Process Symbols
Activities

- Work that is performed within a Business Process.
- Activity can be atomic or non-atomic (compound)
- High-level, so does not describe the activity detail (not the job of BPMN)
- Three types: Task, Sub-process, and Transaction
**Activiy Symbols**

An Activity is work that is performed within a Business Process. An Activity can be atomic or non-atomic (compound).

- **Task**: Single unit of work that is not or cannot be broken down to a further level of business process detail without diagramming the steps in a procedure (not the purpose of BPMN).

- **Subprocess**: Has its own self-contained start and end events, and sequence flows from the parent process must not cross the boundary.

  - Hide or reveal additional levels of business process detail - when collapsed a sub-process is indicated by a plus sign against the bottom line of the rectangle; when expanded the rounded rectangle expands to show all flow objects, connecting objects, and artifacts.

- **Transaction**: A form of sub-process in which all contained activities must be treated as a whole, i.e., they must all be completed to meet an objective, and if any one of them fails they must all be compensated (undone). Transactions are differentiated from expanded sub-processes by being surrounded by a tramline border.
Task Symbols (1 of 2)

Generic task. Single unit of work that is not or cannot be broken down to a further level of business process detail without diagramming the steps in a procedure.

A Service Task is a Task that uses some sort of service, which could be a Web service or an automated application.

A Send Task is a simple Task that is designed to send a Message to an external Participant (relative to the Process). Once the Message has been sent, the Task is completed.

A Receive Task is a simple Task that is designed to wait for a Message to arrive from an external Participant (relative to the Process). Once the Message has been received, the Task is completed.
Task Symbols (2 of 2)

User Task

A User Task is a typical “workflow” Task where a human performer performs the Task with the assistance of a software application and is scheduled through a task list manager of some sort.

Manual Task

A Manual Task is a Task that is expected to be performed without the aid of any business process execution engine or any application. An example of this could be a telephone technician installing a telephone at a customer location.

Business Rule Task

A Business Rule Task provides a mechanism for the Process to provide input to a Business Rules Engine and to get the output of calculations that the Business Rules Engine might provide. The InputOutputSpecification of the Task will allow the Process to send data to and receive data from the Business Rules Engine.

Script Task

A Script Task is executed by a business process engine. The modeler or implementer defines a script in a language that the engine can interpret. When the Task is ready to start, the engine will execute the script. When the script is completed, the Task will also be completed.
What are Sub Processes?

- Is in the context of the process and can access the contextual data.

- Can be expanded or collapsed to show detail of the sub-process or to hide the detail.

- Sub process MUST define an internal process with a start and end event.

- A sub process is only reusable within the parent process (i.e. it is not reusable in the overall design).
Sub Process Symbols

- **Expanded Sub Process**
  - task

- **Collapsed Sub Process**
  - Note that the event annotates collapsed process.

- **Expanded Event Sub Process**
  - task

- **Expanded Ad Hoc Sub Process**
  - task

- **Collapsed Event**
  - +

- **Collapse Ad Hoc**
  - ∼ +
Gateways

- Gateways are used to control how Sequence Flows interact as they converge and diverge within a Process.
- Capable of consuming or generating additional tokens.
- Define decisions/branching (exclusive, inclusive, and complex), merging, forking, and joining.
Gateway Symbols

Gateways are used to control how Sequence Flows interact as they converge and diverge within a Process.

- **Exclusive**
  - A diverging Exclusive Gateway (Decision) is used to create alternative paths within a Process flow. This is basically the “diversion point in the road” for a Process. For a given instance of the Process, only one of the paths can be taken.

- **Inclusive**
  - Inclusive Gateway (Inclusive Decision) can be used to create alternative but also parallel paths within a Process flow. The true evaluation of one condition expression does not exclude the evaluation of other condition expressions. All Sequence Flows with a true evaluation will be traversed by a token.

- **Complex**
  - The Complex Gateway can be used to model complex synchronization behavior. An Expression activationCondition is used to describe the precise behavior.

- **Event Based**
  - The Event-Based Gateway represents a branching point in the Process where the alternative paths that follow the Gateway are based on Events that occur, rather than the evaluation of Expressions using Process data (as with an Exclusive or Inclusive Gateway). A specific Event, usually the receipt of a Message, determines the path that will be taken.

- **Parallel**
  - A Parallel Gateway is used to synchronize (combine) parallel flows and to create parallel flows.
About Data Symbols

- Data Objects provide information about what Activities require and/or what they produce.
- Represent a singular object or a collection of objects.
- Data Input and Data Output provide the same information for Processes.
Data

- **Data Object**: Primary construct for modeling data within the Process flow. DataObject has a well-defined lifecycle, with resulting access constraints.

- **Data Store**: Provides a mechanism for Activities to retrieve or update stored information that will persist beyond the scope of the Process.

- **Data Input**: Data Input is a declaration that a particular kind of data will be used as input of the InputOutputSpecification.

- **Data Output**: Data Output is a declaration that a particular kind of data can be produced as output of the InputOutputSpecification.
An Event is something that “happens” during the course of a Process or a Choreography. These Events affect the flow of the model and usually have a cause (Trigger) or an impact (Result). Events are circles with open centers to allow internal markers to differentiate different Triggers or Results. There are three types of Events, based on when they affect the flow: Start, Intermediate, and End.

- **Start Event** indicates where a particular Process or Choreography will start.

- **Intermediate Events** occur between a Start Event and an End Event. They will affect the flow of the Process or Choreography, but will not start or (directly) terminate the Process.

- **End Event** indicates where a Process or Choreography will end.
Start Events

- None
- Multiple
- Conditional
- Message
- Timer
- Signal
- Error
- Escalation
- Copression
Intermediate Events

- None Catching
- Signal Catching
- Message Catching
- Conditional Catching
- Multiple Catching
- Link Catching
- Parallel Multiple Catching
- Timer Catching
End Events

- End Event indicates where a Process will end.
- In terms of Sequence Flows, the End Event ends the flow of the Process, and thus, will not have any outgoing Sequence Flows.
- No Sequence Flow can connect from an End Event.
- Depending on the type, other rules are enforced (like error handling and/or how the process is terminated).
End Event Symbols

None (not defined)

Escalation

Signal

Message

Cancel

Terminate

Error

Compensation

Multiple
End Events (1 of 3)

- **None (not defined)**
  - The None End Event does not have a defined result.

- **Message**
  - A Message is sent to a Participant at the conclusion of the Process.

- **Error**
  - A named Error should be generated. All currently active threads in the particular Sub-Process are terminated as a result. The Error will be caught by a Catch Error Intermediate Event with the same errorCode or no errorCode which is on the boundary of the nearest enclosing parent Activity (hierarchically).
End Events (2 of 3)

An Escalation is to be triggered. Other active threads are not affected by this and continue to be executed. The Escalation will be caught by a Catch Escalation Intermediate Event with the same escalationCode or no escalationCode which is on the boundary of the nearest enclosing parent Activity (hierarchically).

Indicates that the Transaction should be canceled and will trigger a Cancel Intermediate Event attached to the Sub-Process boundary. In addition, it will indicate that a TransactionProtocol Cancel Message should be sent to any Entities involved in the Transaction.

Compensation

indicates that compensation is necessary. If an Activity is identified, and it was successfully completed, then that Activity will be compensated.
End Events (3 of 3)

- **Signal**: A Signal will be broadcasted when the End has been reached. Note that the Signal, which is broadcast to any Process that can receive the Signal, can be sent across Process levels or Pools, but is not a Message (that has a specific source and target).

- **Terminate**: All Activities in the Process should be immediately ended. This includes all instances of multi-instances. The Process is ended without compensation or event handling.

- **Multiple**: There are multiple consequences of ending the Process. All of them will occur (e.g., there might be multiple Messages sent).
Pools

- A Pool is the graphical representation of a Participant in a Collaboration.
- It also acts as a “swimlane” and a graphical container for partitioning a set of Activities from other Pools.
- Usually in the context of B2B situations.
- Pool MAY have internal details, in the form of the Process that will be executed.
- Pool MAY have no internal details, i.e., it can be a "black box."
Lanes

- Lane (Swimlane) is a sub-partition within a Process
- Extend the entire length of the Process
- Lanes are used to organize and categorize Activities
Pool and Lanes
A Sequence Flow can have a condition Expression that are evaluated at runtime to determine whether or not the Sequence Flow will be traversed. If the conditional flow is outgoing from an Activity, then the Sequence Flow will have a mini-diamond at the beginning of the connector.

If the conditional flow is outgoing from a Gateway, then the line will not have a mini-diamond.
Default Sequence Flow

For Data-Based Exclusive Gateways or Inclusive Gateways, one type of flow is the Default condition flow (see page 97). This flow will be used only if all the other outgoing conditional flow is not true at runtime. These Sequence Flows will have a diagonal slash will be added to the beginning of the connector (see the figure to the right).
Exception Flow

Exception flow occurs outside the normal flow of the Process and is based upon an Intermediate Event attached to the boundary of an Activity that occurs during the performance of the Process.
Data Flow
Compensation Association occurs outside the normal flow of the Process and is based upon a Compensation Intermediate Event that is triggered through the failure of a transaction or a throw Compensation Event. The target of the Association MUST be marked as a Compensation Activity.
Message Flows

- Flow of Messages between two Participants that are prepared to send and receive them.
- A Message Flow MUST connect two separate Pools.
- They connect either to the Pool boundary or to Flow Objects within the Pool boundary.
- They MUST NOT connect two objects within the same Pool.
A Message Flow is used to show the flow of Messages between two Participants that are prepared to send and receive them. In BPMN, two separate Pools in a Collaboration Diagram will represent the two Participants (e.g., PartnerEntities and/or PartnerRoles).

Can be shown with a message or without a message (implied message).
Message Flow Between Lanes

Lane A
- Receive Message
- Task to Task1
- Boundary to Boundary1
- Process Message

Lane B
- Send Message
- TASK
- OUT
- TASK1
- Throw Message
- Throw Message X
- Throw to Boundary1
- Throw to Catch1
- Catch Message X

Lane to Lane Message Flow Examples
Data Associations

- Data Associations are used to move data between Data Objects, Properties, and inputs and outputs of Activities, Processes, and GlobalTasks.

- Tokens do not flow along a Data Association, and as a result they have no direct effect on the flow of the Process.

- Used for retrieving data from Data Objects or Process Data Inputs to fill the Activities inputs and push the output values from the execution of the Activity back into Data Objects or Process Data Outputs.
Data Association Example
INCOMPETENCE

THERE ARE THINGS WE KNOW, DON'T KNOW AND THE UNKNOWN.
INCOMPETENCE IS FROM THE THINGS WE DON'T KNOW WE DON'T KNOW.
Thank You

Daniel Brookshier
DanielB@NoMagic.com