2. BPMN

2.1 Modellieren mit BPMN

Workflow Language Genealogy

BPM Lifecycle: Where BPMN Fits

Process Modeling is Layered

History of BPMN

- First draft proposed by bpmi.org in March 2004
  - bpmi.org was a consortium that attempted to standardize aspects of business processes
  - It ceased to exist because of the success of BPEL
- Adopted as by OMG in February 2006
- Version 1.1 adopted in January 2008
  - Fixed bugs in version 1.0
- Version 2.0 is on its way...
  - Main weaknesses will be fixed
    - Clear(er) operational semantics & metamodel
    - New features (choreographies, correlations, exchange format,...)
- This lecture is based on the BPMN specification version 2.0. If not stated explicitly, BPMN thus means BPMN 2.0
BPMN 2.0 vs. BPMN 1.1

- BPMN 2.0 is based on BPMN 1.1 but additionally contains:
  - An XML-based interchange format for BPMN models
  - Clean operational semantics and metamodel (connecting BPMN to OMG's MDA efforts)
    - The metamodel is one of the most important changes, hence the new name (see below)
- The graphical notation is mostly unchanged, except for some new event types
- New BPMN “Level”: executable BPMN

BPMN in 1.1: “Business Process Modeling Notation”
BPMN in 2.0: “Business Process Model and Notation”

Basic Order Process

- Thin circle represent a start event
- Thick circle at the end is called and end event
- Rounded rectangles represent activities
- Activities represent actions, specific work performed
  - distinct from functions (e.g. Order Handling) or state (e.g. Order Received)
  - Typically, activities are named in the form Verb-Noun

Same process with Swimlanes

- Swimlanes indicate who a certain activity performs
- Swimlanes (or lanes) typically represent roles or organizational units
- They are drawn as subdivisions of the rectangle containing the process, the so called pool (e.g. “Order Process” in the example above)
Subprocesses

- This is an expanded view of the “Fulfill Order” activity in the previous slide. It represents a subprocess, i.e. an activity that contains sub-parts of the flow that are again expressed by a flow.
- A task in contrast is an activity that cannot contain sub-parts.
- Note how the gateway following the expanded sub process is used to continue the flow in the parent process based on the end event that was reached.

Parallel Split and Join

- Pick Stock and Arrange Shipment are parallel tasks.
- This is indicated by the parallel gateway, sometimes also called AND-gateway.
- A parallel gateway split must be joined by a parallel gateway join again.
- Note that the default gateway (e.g. “In stock?” above) denotes an exclusive decision and thus cannot be used to fork or join parallel activities (deadlock!)

Collaborations

- External participants (e.g. the “Customer”) to a process are modeled as a pool.
- These interact with the process by exchanging messages.
- Interactions between participants (modeled by lanes) are indicated by message flows (e.g. “Failure Notice”).
- Message flows are drawn as dashed lines with unfilled arrowheads between two pools.
- Note that the regular sequence flow is drawn as a solid line within a pool.
- The diagram in the previous slide is called a “Collaboration Diagram”, since additionally to the flow within a process, it shows how the process interacts with external participants.
- **Caveat:** In BPMN 1.1 the pattern of message flows was called “Choreography.”
1. Activities

**Activity**

- **Activity** is a generic term for work that has to be performed
- An activity can be atomic or non-atomic (compound)
- The types of activities that are a part of a Process Model are:
  - Process
  - Subprocess
  - Task
- Tasks and subprocesses are rounded rectangles
- Processes are either unbounded or contained within a pool

**Operational Semantics**

- BPMN operational semantics is based on passing tokens
- An activity or gateway (see later...) is performed when "required" tokens arrived at the activity or gateway
- "Required" depends on...
  - the gateway governing the activity (see later)
  - the specified start quantity defined
  - tokens still on their way (global semantics! – see later!)
  - ...
- At completion, an activity or gateway produces tokens on all of its outbound sequence flows (aka control connectors – see later...)
  - Conditions that evaluate to false consume these tokens immediately

**Operational Semantics (cont.)**

- A start event generates a token for each of its leaving sequence flows
- These tokens must eventually be consumed at end events
- If parallel sequence flow target an end event, then the tokens will be consumed as they arrive
- All tokens that were generated within the process must eventually be consumed by an end event before the process is completed
- If the process is a subprocess, it can be stopped prior to normal completion through interrupting intermediate events (see later)
  - In this situation the tokens will be consumed by an intermediate event attached to the boundary of the subprocess

**BPMN Token Semantics: Example**

- Message arrived: Start event generates a token
- Activity A gets activated
- Activity A completes
- Token is generated
- Parallel gateway gets activated
• Parallel gateway produces tokens for each leaving sequence flow
• Activities B and C may be performed

• Activity B completed and produced a token
• Exclusive gateway gets activated
  • It needs a single token only for activation
• Activity C has not yet been activated
  • E.g. because user is still busy

• Exclusive gateway produces a token
• Activity C has still not yet been activated

• Activity D completed and produced a token
• Activity C completed and produced a token
• End event does not consume token (that would terminate the process) because “upstream” token may reach it later on

• Exclusive gateway produced a token
• End event does not consume token

• Activity D consumed the token, completed and produced a token
2. Tasks

Task

- A Task is an atomic activity that is included within a process
  - This is unusual terminology – most often, tasks are activities to be performed by human beings
  - The term activity is usually used in other standards and products
- A Task is used when the work in the Process is not broken down to a finer level of Process Model detail
- Generally, an end-user and/or an application is used to perform the Task when it is executed

Data Objects

- ...provide information about what activities require to be performed and/or what they produce
- ...are related via Associations (“dashed lines”) to tasks, sequence flows

Tasks and Data Objects

- Data Objects may be shown as input/output of a process or a task
- Directionality added to the Association will show whether the Data Object is an input or an output
- State attribute of the Data Object can change to show the impact of the Process/Task on the Data Object

Service Task

- Service Task: A Task that uses some sort of service to provide an action
  - ...Web service or an automated application
User Task

- **User Task**: A typical “workflow” task where a human being performs the task possibly with the assistance of a software application
  - Often, it is scheduled through a task list manager of some sort

Script Task

- **Script Task**: ..is executed by a business process engine
  - The modeler or implementer defines a script in a language that the engine understands how to execute
  - When the Task is ready to start, the engine will execute the script
  - When the script is completed, the Task will also be completed

Manual Task

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

Send Task

- **Send Task**: A simple Task that is designed to send a message to an external participant
  - Once the message has been sent, the Task is completed

  **Note**: A send task (left) is same as the throwing message intermediate event (right)

Receive Task

- **Receive Task**: A simple Task that is designed to wait for a message to arrive
  - Once the message has been received, the Task is completed
  - A Receive Task is often used to start a Process

  **Note**: A receive task (left) is same as the catching message intermediate event (in the center, see later). Use the event gateway (right) to model a timeout for the wait
BPMN Style: Level 1

- In the next couple of slides, we will introduce
  - BPMN principles of composition
    - e.g. when and how to use black-box pools, labels, message flows, ...
  - Elementary rules of usage
    - Rules for start- and end events, pools, sequence flow, ...

---

**Principle #1**

- Make the process logic clear from the printed diagram

- The process logic, various end-states, etc. should all be clear from the printed diagram itself
  - That means, you should be labeling essentially everything: subprocesses, events, gateways, message flows, ...
  - Adhere to naming conventions

**Principle #2**

- Make models hierarchical, fitting each process level on one page

- The top level diagram should both capture the end-to-end process on one page and show its external participants.
  - Use subprocesses for hierarchical nesting

**Principle #3**

- Use black-box pools for external participants

- Customer and other external participants should be modeled as black-box pools (see the diagram on the right)

**Principle #4**

- Begin customer-facing processes with a message start event receiving a message flow from the customer pool
Principle #5

- Model internal process participants (activity performers) as lanes within a single process pool, not as separate pools. Label lanes with the role or organizational unit that performs its contained activities.
- Representing various organizational units that perform process activities as separate pools (left) is usually incorrect. This would imply that each unit’s process is independent of the others, and can stand alone.
- Instead, use lanes to indicate organizational units (right).

Principle #6

- Label white-box pools with the name of a process; label black-box pools with a participant role or business entity.

Principle #7

- Show message flows between the process and all external pools in a top-level diagram, and show message flows consistently in parent and child-level diagrams.
- See next 2 slides for an example.

Example for #7: Top Level Diagram

Example for #7: Expansion (“Order Car from...”)

Principle #8

- Label activities VERB-NOUN.
- Examples:
  - “Check credit” (action), not “Credit check” (function) or “Credit ok” (state)
  - “Approve Loan” (action), not “Loan approval” (function) or “Loan rejected” (state)
  - “Receive report” (action), not “Report received” (state)
Principle #9

If possible, label exclusive decision gateways with a “yes” / “no” question, and label the outgoing sequence flows “yes” and “no”.

Principle #10

Label message start events “Receive X”, where X is the object triggering the process.

Principle #11

Label timer start events to indicate the process schedule.

Principle #12

Indicate success and failed end states of a process with separate end events, and label them to indicate the end state (see below). If multiple paths lead to the same effective end state, route them all to a single end event (see next slide).

Principle #13

Label message flow with the name of the message. Use a message linked to the message flow to indicate additional detail, if necessary.

Rule #14

Use start events in a process or subprocess. Use end events to represent the end of each path of a process or subprocess.
Rule #15
- All activities, gateways, and events must be connected via a continuous chain of sequence flows leading from a start event to an end event.

Rule #16
- A sequence flow (or an equivalent link event pair) must not cross a pool boundary. Use message flow to link pools.

Rule #17
- Sequence flow (or equivalent link event pair) must not cross subprocess boundaries from child level to parent level.

Rule #18
- Message flow cannot connect points in the same pool.

Rule #19
- Message flow cannot connect a gateway.

2. BPMN

2.3 Stilfragen

Ende
Kapitel 2
BPMN

2.4 Weitere Beispiele

Abläufe zur Durchführung einer Aufgabe mit ihren einzelnen Schritten definieren und darstellen

Im Vordergrund: Nachrichtenaustausch der Prozessbeteiligten

stärkere Verbindung zu den Prozessdiagrammen, Betonung der Aufgaben der einzelnen Prozessbeteiligten

Im Vordergrund: Wie Choreographie-Diagramm, Nachrichtenaustausch der Prozessbeteiligten

- einzelne Nachrichten und deren Austausch nach inhaltlichen Gesichtspunkten zusammenfassen
Prozessdiagramme-Elemente

**Task (Aktivität): Grundelement**

Tasks können mit einem *sequence flow* verbunden sein.

**Sequence flow**

Abbildung 6 Sequence Flow

**Event**

Drei Sorten von Events:

- start ~
- Intermediate ~
- end ~

Abbildung 7 Events

**Gateway**

Abbildung 8 Gateways

**Association, Kommentierung**

Abbildung 9 Kommentierungen

**Daten, Datenfluss**

Abbildung 10 Daten und Datenfluss
Subprozess

Abbildung 1 Subprozess

Wird verfeinert

Aufruf eines global definierten Tasks ; mit einer dickeren Außenlinie dargestellt.

Call-Aktivität

Abbildung 12 Call-Aktivität

Details 1

<table>
<thead>
<tr>
<th>Bezeichnung</th>
<th>Erklärung</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abstract Task</td>
<td>Definiert die allgemeinste Form einer Task. Für die Ausführung ist es eine Taskdefinition ohne weitere Angaben.</td>
<td>Aktivist Task</td>
</tr>
<tr>
<td>Service Task</td>
<td>Symbolisiert einen synchronen Aufruf einer WebService (default) oder den Aufruf einer anderen Anwendung. Er wird über die Anwendung von einem Benutzer gestartet.</td>
<td>Server Task</td>
</tr>
<tr>
<td>Send Task</td>
<td>Symbolisiert das Auslösen einer Nachricht an eine andere Anwendung oder eine andere Anwendung. Er wird über die Anwendung von einem Benutzer gestartet.</td>
<td>Send Task</td>
</tr>
<tr>
<td>Receive Task</td>
<td>Symbolisiert das Warten auf eine Nachricht (beispielsweise von einer WebService). Der Prozess wartet solange, bis eine Nachricht von einer anderen Anwendung ausgelöst wird.</td>
<td>Receive Task</td>
</tr>
<tr>
<td>Script Task</td>
<td>Symbolisiert die Ausführung eines externen Skripts.</td>
<td>Script Task</td>
</tr>
</tbody>
</table>

Details 2

<table>
<thead>
<tr>
<th>Bezeichnung</th>
<th>Erklärung</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>Loop</td>
<td>Ein Task mit einem Loop werden erstellt, wenn eine vorgeschriebene Bedingung nicht mehr gilt.</td>
<td>Loop Task</td>
</tr>
<tr>
<td>Multi-Instance (Parallel)</td>
<td>Es werden mehrere Instanzen eines Subprozesses erstellt und parallel ausgeführt.</td>
<td>Multi-Instance Subprozess Parallel</td>
</tr>
<tr>
<td>Multi-Instance (Sequential)</td>
<td>Es werden mehrere Instanzen eines Subprozesses erstellt und sequentiell ausgeführt.</td>
<td>Multi-Instance Subprozess Sequential</td>
</tr>
<tr>
<td>Compensation</td>
<td>Symbolisiert die Schritte, die zu einer Veränderung der Startzeit, der bei einem anderen Task oder Subprozess notwendig sind.</td>
<td>Compensation Subprozess</td>
</tr>
</tbody>
</table>

Details 3

Event-Subprozess

Abbildung 13 Beispiel für einen Event-Subprozess

Ebene, wo das Ereignis entsteht

Ereignis, das den Subprozess auslöst
**Subprozess mit Transaktion**

Transaktion:
Subprozess, dessen Ausführung durch ein Transaktionsprotokoll aus WS-Transaction

Abbildung 14 Subprozess mit Transaktion

---

**Spezielle Start-Events**

**Spezielle Intermediate events**

**Spezielle End-Events**

---

**Nicht unterbr. Boundary Event**

Abbildung 15 Beispiel für ein nicht unterbrechendes Boundary Message Event

---

**Unterbr. Boundary Event**

Abbildung 16 Beispiel für ein unterbrechendes Boundary Event
**Intermediate Boundary Event**

**Exclusive gateway**

**Event based gateway**

**Inclusive gateway**

**complex gateway**

**parallel gateway**
Pool

Pool mit swimlanes

Abbildung 22 Darstellung eines Pools

Abbildung 23 Lanes in einem Pool

Prozess mit zwei pools

Abbildung 24 Beispiel eines Prozesses mit zwei Pools

Abbildung 25 Öffentlicher Prozess

Choreographiediagramm-Elemente

Fig. 1.1 An online transaction for an electronic bookstore in BPMN notation
Kollaborationsdiagramm-Elemente
Konversationsdiagramm-Elemente

Abbildung 37 Kollaborationsdiagramm mit Choreographie und öffentlichem Prozess

Abbildung 38 Conversation Node

Abbildung 39 Beispielhafter Nachrichtenaustausch

Abbildung 40 Verwendung einer Conversation Node

Abbildung 41 Conversation Nodes an einem größeren Beispiel
Kapitel 2
BPMN

2.4 Weitere Beispiele