Data and Process Modelling

8b.BPMN - Stylistic guidelines

Marco Montali

KRDB Research Centre for Knowledge and Data
Faculty of Computer Science
Free University of Bozen-Bolzano

A.Y. 2015/2016

Style

The BPMN specification supports modeler in the creation of syntactically correct BPMN diagrams.

The method supports modelers in the creation of structurally consistent BPMN diagrams.

What about clarity, readability and comprehensiveness? **Style** is needed.

- Best practices of good modeling.
- Formalized as style rules that can be validated against the model.

Main style principle at level 1

The process logic must be completely and unambiguously understood from the diagram alone.

Adherence to style rules should be considered as a strong requirement.

Rule 1: Labels

Rule 1

Use icons and labels to make the process logic clear from the printed diagram.

- All activities must be labeled, including subprocesses.
- If multiple end states are present, they must be labeled.
- Gates must be labeled.
- Pools and message flows must be labeled.
- Identify task types and event triggers with icons.
- If you still have the feeling that some clarifications are needed, add text annotations.

Rule 2: Process Hierarchy

Rule 2

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

- Top-level diagram should capture the end-to-end process and how it interacts with external entities.
- Child-level diagrams expand details.
- Child processes provide a mechanism for evolving the process without touching the top level.
- If there are diagrams that do not fit in one page, use carefully off-page connectors.

Rule 3: Black-Box Pools

Rule 3

Use a black-box pool to represent the Customer or other external requester or service provider.

- Showing the inner process of such external participants is an error.
- In fact, we do not now all the internal details of the external processes.
- We cannot expand a process only partly.
- We cannot attach a message flow to the boundary of a process pool (i.e., a white-box pool).

Rule 4: Customer-Facing Processes

Rule 4

Begin customer-facing processes with a message start event receiving a message flow from the customer pool.

- The message itself represents the case.
- Attaching the message to a task would just imply the possibility of an information exchange (weaker).

Rule 5: Process Orchestration

Rule 5

If you can, model internal organizational units as lanes within a single process pool, not as separate pools: they imply independent processes.

- The end-to-end process corresponds to a single process orchestration, which involves the different organizational units.

Exception

When there is no 1:1 alignment of the notion of case across organizational units.

Order Processing and Billing

Example

An order process starts when the sales department receives an order. The order is entered into the system, leading to two possible outcomes: failed or ok. In the first case, the process immediately ends sending a message to the customer so as to notify the failuer. In the second case, two activities are performed, the first one targeted to fulfilling the order, the second focused on the update of the customer account. The process then terminates notifying the customer that the order is complete.
Obviously, each order must be billed. Billing is done on a monthly basis, generating a unique statement for all the orders fulfilled during that month.

Rule 6: Pools Naming

Rule 6

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

- The specification encourages modelers to always name pools with the participant name.
- However, each participant must have only one corresponding process.
- This is enforced in the XML using an ID attribute beside the NAME of the pool. In fact, the same company can have multiple processes.
- However, the ID is not visible to the reader.
- Furthermore, the same company may have message flow interconnecting its different internal processes. Using process names for such pools makes message exchange easier to grasp.

Rule 7: End States

Rule 7

Indicate sucess and exception end states of a process or subprocess with separate end events, and label them to indicate the end state.

- This distinction must reflect the fact that a conceptual distinction is important.
- This is especially true if the parent process needs to distinguish behavior based on the reached end state inside a subprocess.
- Typical example: accept/reject.

Rule 8: Activity Labels

Rule 8

Label activities VERB-NOUN.

- Activities denot work or actions to be performed, not functions or states.

Example

- Credit OK vs Check credit vs Credit chek.
- Loan rejected vs Loan approval vs Approve loan.
- Report received vs Receive report.

Rule 8: Activity Labels

Rule 8

Label activities VERB-NOUN.

- Activities denot work or actions to be performed, not functions or states.

Example

- Credit OK vs Check credit vs Credit chek.
- Loan rejected vs Loan approval vs Approve loan.
- Report received vs Receive report.

Rule 9: Start Event

Rule 9

Use start event trigger in top-level process to indicate how the process starts.

- Message: external request.
  - Label: *Receive [message flow name]*.
- Timer: scheduled (recurring) process.
  - Label: shedule rule.
- None: manual start by the process performer.
  - Typically unlabeled.

Rule 10: End States across Levels

Rule 10

If a subprocess is followed by a gateway labeled as a question, the subprocess should have multiple end events, and one of them should match the gateway label.

- This is important to maintain semantic alignment across process levels.
- Typical example: approval/rejection.

Rule 11: Message Flow

Rule 10

Show message flow with all message events.

- Message flows are optional in BPMN.
- But their explicit modeling clarifies interaction modalities and the business context.

Rule 12: Matching of Message Flows

Rule 12

Match message flows in parent- and child-level diagrams.

- This is necessary, again, for traceability.
- Matching is in terms of number and labels.
- Top-down: all messages attached to a collapsed subprocess should be precisely "located" inside the child process.
- Bottom-up: all messages used inside a subprocess should be properly shown at the parent level.

Rule 13: Labeling of Message Flows

Rule 13

Label message flows directly with the name of the message.

- A message flow consists of a message (e.g., rejection notice).
- A message is not a state (e.g., rejected).
- A message is not the action of sending or receiving it (e.g., send rejection).
- Important: in BPMN it is not possible to attach a data object to a message flow.

Rule 14: End State Duplication

Rule 14

Two end events in a process level should not have the same name.

- Do they represent distinct end states? Then two different labels are needed.
- Are they actually the same end state? Combine them.

Example

A process starts by checking the account history. The outcome of this activity indicates whether the history is ok or not. If not, the process terminates due to bad credit. If so, then the credit score is checked, leading again to one between two outcomes. If the credit score is ok, the process terminates in a "credit ok" state. Otherwise, the process terminates due to bad credit.

Rule 15: Activity Repetition

Rule 15

Two activities in a process model should not have the same name.

- Are they the same activity? Then use a **call activity** referencing the corresponding global task/process.
- Are they different? Then differentiate labels.
- It is not uncommon to see the name of a subprocess repeated in one of its tasks. Cf. example in the previous slide.

Rule 16: Subprocess Start Event

Rule 16

A subprocess should have a single None start event.

- Multiple start events are allowed in the top-level process to distinguish different starting points/modalities.
- In a subprocess, multiple none start events create ambiguity.
  - Do they denote alternative starting points? Then model the decision explicitly.
  - Do they denote parallel branches? Then clarify this.

Example

The register applicant activity starts either by receiving the application or (if the application is already on file) by receiving the payment. The registration is then completed by filling in all the necessary details.

Rule 17: Subprocess Name

Rule 17

A process pool in child-level diagram (if drawn) should be labeled with name of the top-level process, not the name of the subprocess.

- E.g., the check credit subprocess of the order process should not have a pool called "check credit": either it has none, or it has a pool labeled "order process".
- Different pools would in fact implicitly imply different participants (and possibly even different, separate processes).
- Check credit is not independent from order process!

Rule 18: Process Scope

Rule 18

In a hierarchical process, a child-level diagram may not contain any top-level processes.

- A top-level diagram may disclose elements of more than one process (collaboration).
- Child processes should not do that. This would lead to a mixture of levels of abstractions within a level.
- Still, child processes may contain many black-box pools for external participants.

Rule 19: Merge Parsimony

Rule 19

Do not use an XOR gateway to merge alternative paths, unless into another gateway. Just connect the sequence flows directly.

- This makes the diagram lighter.

Rule 20: Synch Parsimony

Rule 20

Do not use an AND gateway to join parallel paths into a None end event.

- A join is in fact always implied at a None end event.
- Recall the semantics of process termination under normal circumstances.

Additional Official BPMN 2.0 Rules

Rule 21

A sequence flow may not cross a pool (process) boundary.

Rule 22

A sequence flow may not cross a subprocess boundary.

Rule 23

A message flow may not connect nodes in the same pool.

Rule 24

A sequence flow may only connect to an activity, gateway, or event, and both ends must be properly connected.

Rule 25

A message flow may only connect to an activity, message/multiple event, or black-box pool, and both ends must be properly connected.