MTAT.03.231
Business Process Management

Lecture 3 – Process Modeling I

Marlon Dumas

marlon.dumas @t ut . ee
Business Process Lifecycle

1. Process identification
2. Process architecture
3. Process discovery
   - Conformance and performance insights
4. Process monitoring and controlling
   - Executable process model
5. Process implementation
   - To-be process model
6. Process analysis
   - Insights on weaknesses and their impact
7. Process redesign
   - As-is process model
1. Introduction
2. Process Identification
3. **Essential Process Modeling**
4. Advanced Process Modeling
5. Process Discovery
6. Qualitative Process Analysis
7. Quantitative Process Analysis
8. Process Redesign
9. Process Automation
10. Process Intelligence
Purposes of process modeling

- Communication
- Documentation
- Analysis (e.g. simulation)

- Automation
- Testing
Business Process Model and Notation (BPMN)

• OMG standard (nowadays BPMN 2.0)
• Both for conceptual and executable models
• Supported by numerous tools: bpmn.org lists over 70 tools, incl.
  • Signavio (we’ll use the academic edition – academic.signavio.com)
  • Bizagi Process Modeler
  • Cameo Business Analyst
A BPMN process model is a graph consisting of four types of core elements:

- activity
- start
- end
- gateway
- sequence flow
Let’s start modeling

Order-to-cash

An order-to-cash process is triggered by the receipt of a purchase order from a customer. Upon receipt, the purchase order has to be checked against the stock to determine if the requested item(s) are available. Depending on stock availability the purchase order may be confirmed or rejected. If the purchase order is confirmed, an invoice is emitted and the goods requested are shipped. The process completes by archiving the order.
Let’s start modeling – break it down

Order-to-cash

• An order-to-cash process is triggered by the receipt of a purchase order from a customer.
• Upon receipt, the purchase order has to be checked against the stock to determine if the requested item(s) are available.
• Depending on stock availability the purchase order may be confirmed or rejected.
• If the purchase order is confirmed, an invoice is emitted and the goods requested are shipped. The process completes by archiving the order.
Let’s start modeling – break it down

Order-to-cash

• An order-to-cash process is triggered by the receipt of a purchase order from a customer.
• Upon receipt, the purchase order has to be checked against the stock to determine if the requested item(s) are available.
BPMN Model

Order-to-cash

Purchase order received → Check stock availability
Let’s start modeling – break it down

Order-to-cash

• An order-to-cash process is triggered by the receipt of a purchase order from a customer.
• Upon receipt, the purchase order has to be checked against the stock to determine if the requested item(s) are available.
• Depending on stock availability the purchase order may be confirmed or rejected.
• If the purchase order is confirmed, an invoice is emitted and the goods requested are shipped. The process completes by archiving the order.
BPMN Model

Order-to-cash

Naming conventions
- Event: noun + past-participle verb (e.g. insurance claim lodged)
- Activity: verb + noun (e.g. assess credit risk)
Execution of a process model
The “token game”

- Order #1
- Order #2
- Order #3

Purchase order received → Check stock availability

Items in stock → Confirm order → Emit invoice → Ship goods → Archive order → Order fulfilled

Items not in stock → Reject order → Order rejected
A little bit more on events...

A *start event* triggers a new process instance by generating a token that traverses the sequence flow (“tokens source”)

An *end event* signals that a process instance has completed with a given outcome by consuming a token (“tokens sink”)

![Start event](image1)

![End event](image2)
Order-to-cash example revisited...

[...] If the purchase order is confirmed, **an invoice is emitted and the goods requested are shipped (in any order)**. The process completes by archiving the order. [...]
First try

Order-to-cash
A little more on gateways: XOR Gateway

An XOR Gateway captures decision points (XOR-split) and points where alternative flows are merged (XOR-join).

XOR-split → takes one outgoing branch

XOR-join → proceeds when one incoming branch has completed
Example: XOR Gateway

Invoice checking process

- Invoice received:
  - Check invoice for mismatches:
    - Mismatches exist but can be corrected:
      - Re-send invoice to customer:
        - Mismatches exist but cannot be corrected:
          - Block invoice
          - No mismatches:
            - Post invoice
  - Mismatches exist but cannot be corrected:
    - Block invoice
  - No mismatches:
    - Post invoice

Invoice handled
A little more on gateways: AND Gateway

An AND Gateway provides a mechanism to create and synchronize “parallel” flows.

AND-split ➔ takes all outgoing branches

AND-join ➔ proceeds when all incoming branches have completed
Example: AND Gateway

Airport security check

- Boarding pass received
- Proceed to security check
- Pass security screening
- Proceed to departure level
- Pass luggage screening
- Departure level reached
Revised order-to-cash process model
Between XOR and AND

Order distribution process

A company has two warehouses that store different products: Amsterdam and Hamburg. When an order is received, it is distributed across these warehouses: if some of the relevant products are maintained in Amsterdam, a sub-order is sent there; likewise, if some relevant products are maintained in Hamburg, a sub-order is sent there. Afterwards, the order is registered and the process completes.
Solution 1

Order distribution process

- XOR-split
- XOR-join
- AND-split
- AND-join
Solution 2

Order distribution process
**OR Gateway**

An *OR Gateway* provides a mechanism to create and synchronize \( n \) out of \( m \) parallel flows.

*OR-split* ➔ takes one or more branches depending on conditions

*OR-join* ➔ proceeds when all **active** incoming branches have completed
Solution using OR Gateway

Order distribution process

- Order received
  - Check order line items
    - order contains Amsterdam products
      - Forward sub-order to Amsterdam warehouse
    - order contains Hamburg products
      - Forward sub-order to Hamburg warehouse
- Register order
  - Order completed
What join type do we need here?
Beware: Beginner’s Mistake…

- Check stock availability
- Available
- Not Available
  - PO rejected
  - Ship Products
    - Send Invoice
    - PO fulfilled
Guidelines: Naming Conventions

1. Give a name to every event and task
2. For tasks: verb followed by business object name and possibly complement
   • Issue Driver Licence, Renew Licence via Agency
3. For message events: object + past participle
   • Invoice received, Claim settled
4. Avoid generic verbs such as Handle, Record…
5. Label each XOR-split with a condition
   • Policy is invalid, Claim is inadmissible
Poll: Which model do you prefer?
One more guideline…

• Model in blocks
  • Pair up each AND-split with an AND-join and each XOR-split with a XOR-join, whenever possible
  • Exception: sometimes a XOR-split leads to two end events – different outcomes (cf. order management example)
Rework and repetition

Address ministerial correspondence

In the minister’s office, when a ministerial inquiry has been received, it is registered into the system. Then the inquiry is investigated so that a ministerial response can be prepared.

The finalization of a response includes the preparation of the response itself by the cabinet officer and the review of the response by the principal registrar. If the registrar does not approve the response, the latter needs to be prepared again by the cabinet officer for review. The process finishes only once the response has been approved.
Quick Note: Implicit vs. explicit gateways
How this process starts? How it ends?

- Collect mail
- Sort mail
- Register mail
- Check mail for compliance
- Compile document requisition
- Capture matter details
- Pay fee
- Capture party details
- Print physical file
- Document requisition compiled
- Document response prepared
- Physical file printed
- New mail arrived
- New email arrived
What’s wrong with this model? How to fix it?
Process Modelling Viewpoints

- **When?**
- **Flows**
- **Gateways**
- **Which?**
- **Organization**
- **Who?**
- **Lanes & Pools**
- **What?**
- **Tasks**
- **Events**
- **Data / Materials**
- **Data Objects, Stores**
- **Which?**
- **Lanes & Pools**
- **Organization**
- **Who?**
Organizational Elements in BPMN – Pools & Lanes

**Pool**
Captures a resource class. Generally used to model a business party (e.g. a whole company)

**Lane**
A *resource sub-class* within a pool. Generally used to model departments (e.g. shipping, finance), internal roles (e.g. Manager, Associate), software systems (e.g. ERP, CRM)
Order-to-cash process with lanes

- Order
  - to
  - cash
  - process
  - with
  - lanes
Message Flow

A *Message Flow* represents a flow of information between two process parties (Pools).

A Message Flow can connect:
- directly to the boundary of a Pool captures an *informative* message to/from that party.
- to a specific activity or event within that Pool captures a message that triggers a specific activity/event within that party.
Order-to-cash process with a black-box customer pool
Pools, Lanes and Flows: syntactic rules

1. A Sequence Flow **cannot** cross the boundaries of a Pool (message flows can)
2. Both Sequence Flow and Message Flow **can cross** the boundaries of Lanes
3. A Message Flow **cannot connect** two flow elements within the same pool
One more guideline…

• Start modeling with one single “white-box” pool
  • Initially, put the events and tasks in only one pool – the pool of the party who is running the process
  • Leave all other pools “black-boxed”
  • Once you have modeled this way, and once the process diagram inside the white-box pool is complete, you can model the details (events and tasks) in the other pools if that is useful.
  • In this course we will only model processes with one single white-box pool – all other pools are black-box
Process Modelling Viewpoints

Which?

Data / Materials

Data Objects, Stores
The purchase order document serves as an input to the stock availability check. Based on the outcome of this check, the status of the document is updated, either to “approved” or “rejected”. If the order is approved, an invoice and a shipment notice are produced.
Model with information artifacts
A Data Object captures an artifact required (input) or produced (output) by an activity.

- Can be physical or electronic

A Data Store is a place containing data objects that must be persisted beyond the duration of a process instance.

It is used by an activity to store (as output) or retrieve (as input) data objects.
Quick Note: BPMN Text Annotations

A Text Annotation is a mechanism to provide additional text information to the model reader

- **Doesn’t affect** the flow of tokens through the process

![Diagram showing the process with annotations](image.png)
BPMN Poster (link in “Readings” page)