BPMN Meets DMN:
Business Process and Decision Modeling

Week 2:
Basic Business Process Modeling

An openHPI Course

Prof. Dr. Mathias Weske

Hasso Plattner Institute at the University of Potsdam
Business Process Technology

Week 2
Basic Business Process Modeling

- 2.1 Fundamentals of Modeling Languages
- 2.2 Process Activities
- 2.3 Exclusive and Parallel Gateways
- 2.4 Inclusive Gateways and Loops
- 2.5 Start Events and End Events
- 2.6 Intermediate Events

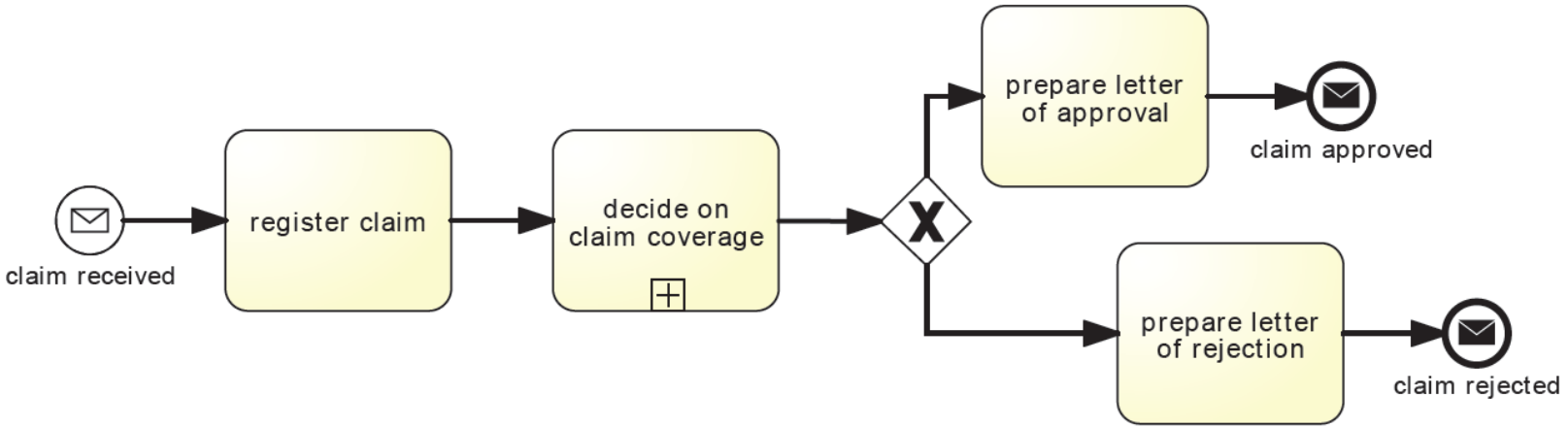
Video Clip 2.1

Fundamentals of Modeling Languages

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Basic Business Process Modeling

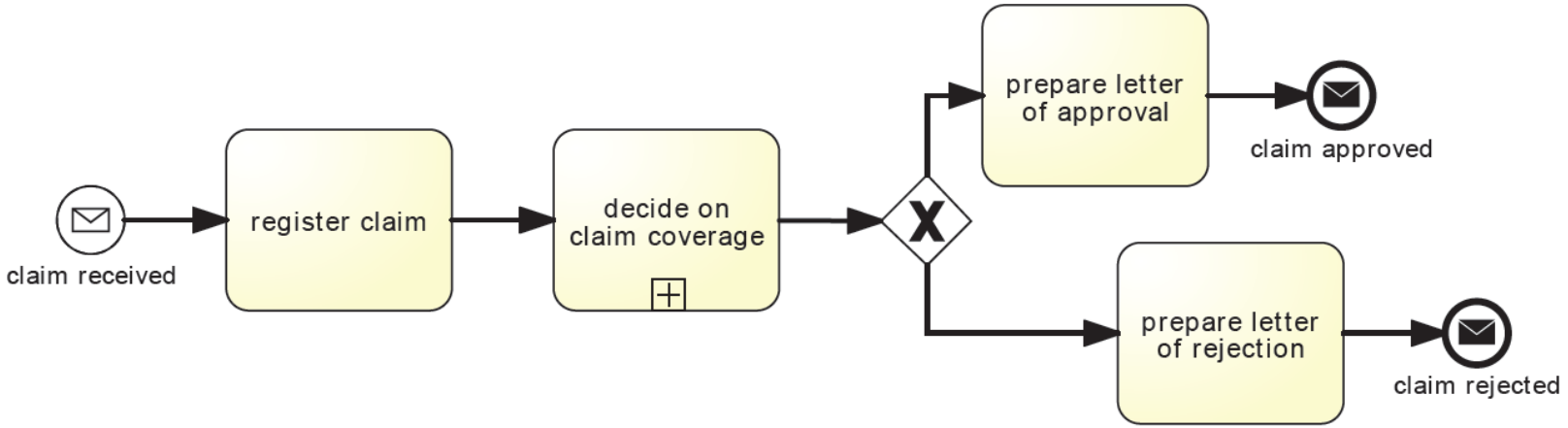
- Business process models are abstract representations of originals, with the features
  - mapping, abstraction, and pragmatics



- Business process modeling languages express business process models
  - Need to provide language constructs to represent the important aspects of business processes

Basic Business Process Modeling

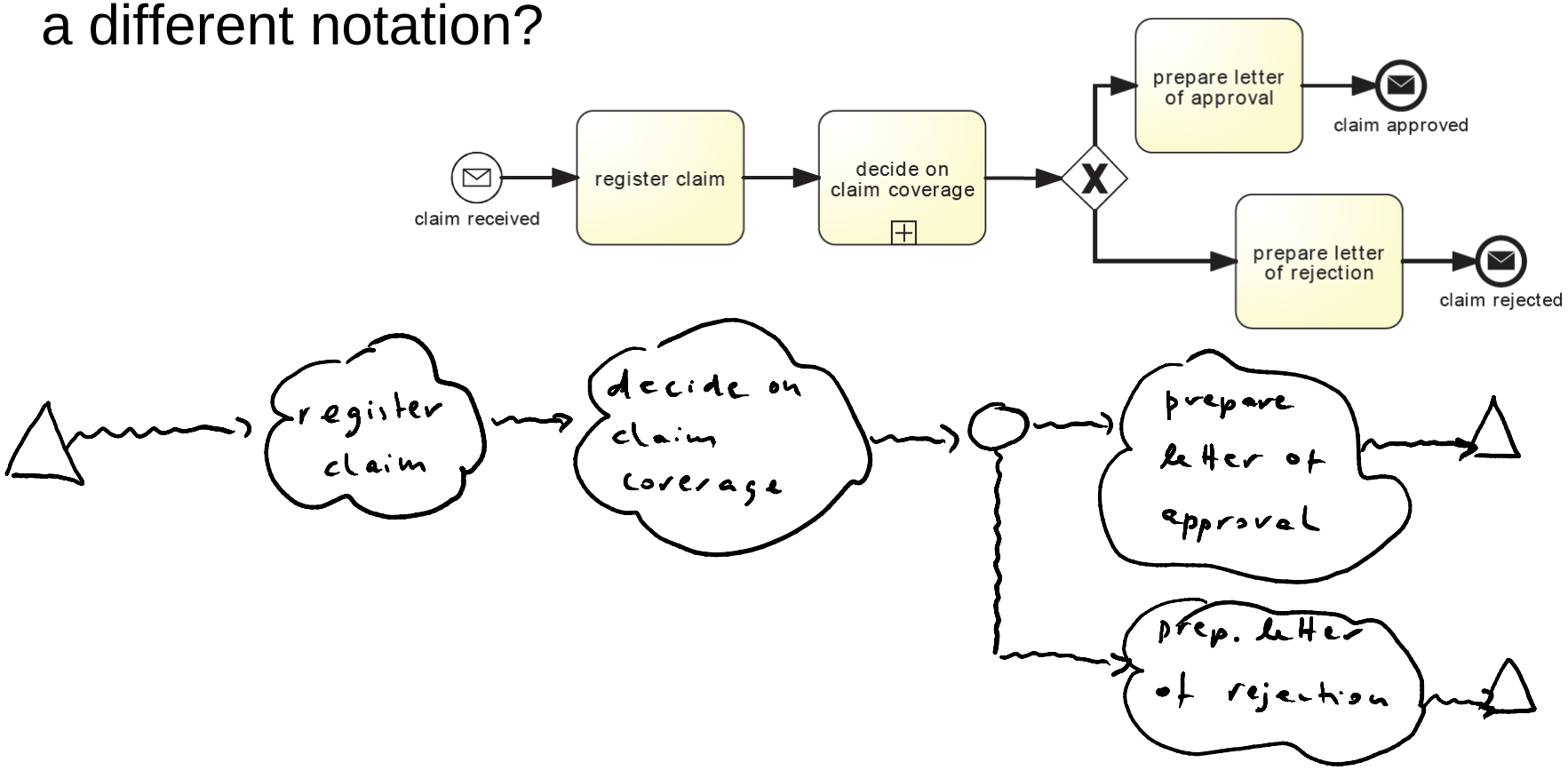
- Languages consist of
  - Concepts and their relationships (abstract syntax)
  - Means to express the concepts (concrete syntax / notation)
  - Meaning of the concepts (semantics)



- Concepts introduced
  - Activity, sequence flow, gateway, and event

Basic Business Process Modeling

- Could we create a language with the same concepts and a different notation?

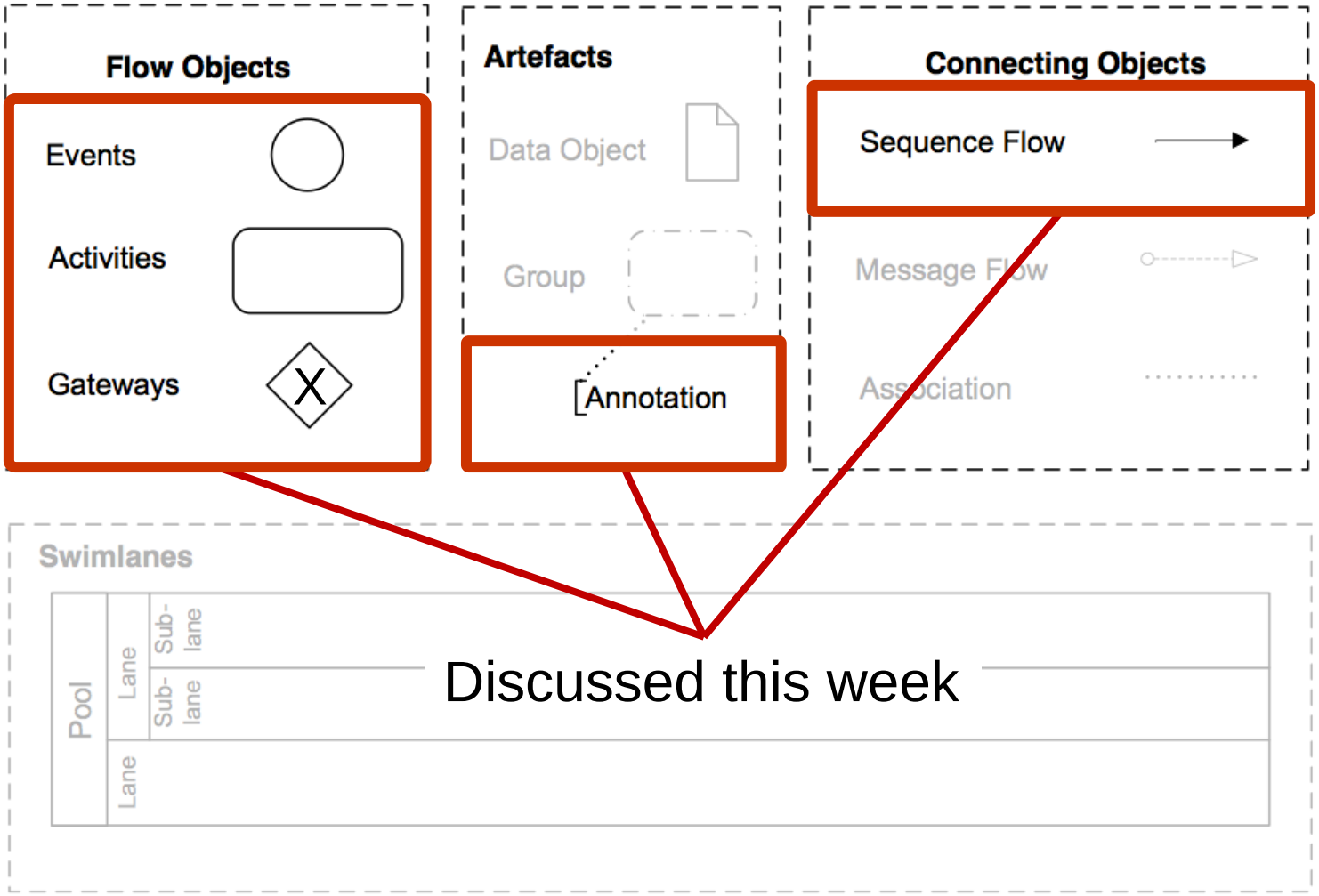


- Let‘s not do that, because a standard for concepts and notation eases communication

Business Process Model and Notation

- BPMN, Version 2
  - Industry standard by the Object Management Group
  - 1.x Versions: “Business Process Modeling Notation”
- Expressive process modeling language
  - Expressive, but also „flexible“
  - Allows to use different sets of modeling concepts
    - Many processes can be modeled with a minimal set of concepts
- BPMN is not suited to model
  - Process landscapes, organizational structures
  - Data structures, IT infrastructures
- In most projects, other modeling languages will complement BPMN

Notation Categories and their Elements

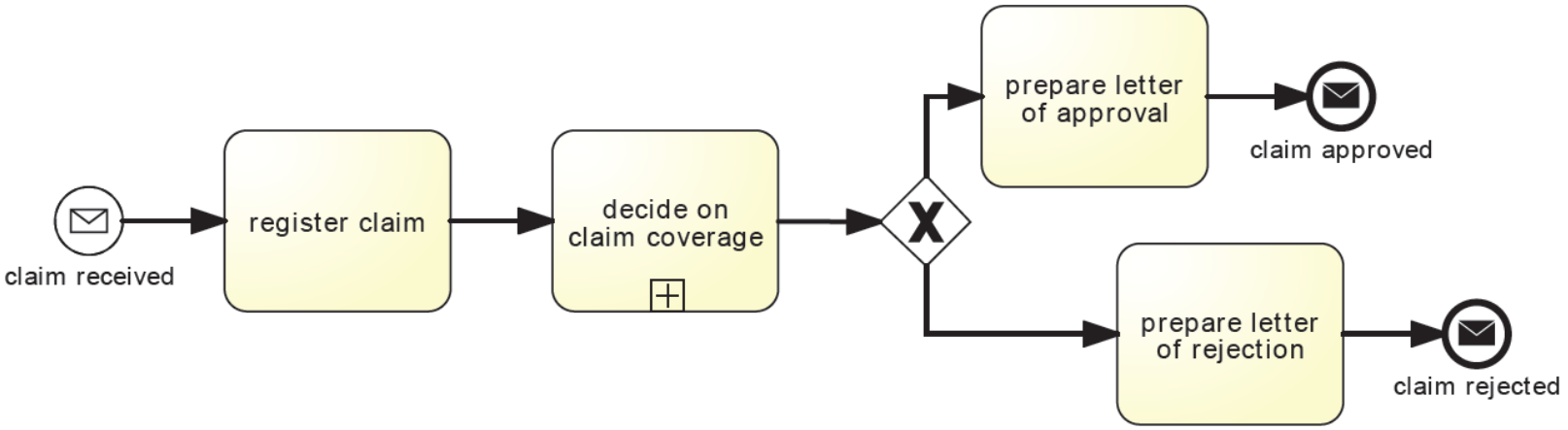


M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

Fig. 4.76. BPMN: categories of elements

Video Clip 2.1
Fundamentals of Modeling Languages

- Abstract syntax
- Concrete syntax
- Semantics
- BPMN



Video Clip 2.2

Process Activities

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Process Activities

- "A business process consists of a set of activities …"
  - Activities are units of work that require time to be performed
  - In this week we only cover tasks, i.e., atomic activities that are not broken down into smaller units of work
- Sample scenario: order fulfillment

The process starts by entering the details of an order in an IT system.
The ordered product is obtained from the warehouse, packaged and shipped to the customer.
An invoice is sent to the customer. Upon receiving the payment, the order is archived.

Activities and Sequence Flow

- "… that are performed in coordination"
  - The activities are not executed independently from each other, there are execution constraints
- Execution order of activities defined by *sequence flow*
  - Notation: directed arcs between activities
  - A → B means that B can only be started, once A has been terminated
  - After A has been terminated, the sequence flow arc is signaled, telling B that it is ready for execution
- Now we can design a sequential process model

Activity Instances

- Process instances contain activity instances

- Each activity instance traverses a series of states
  - At any point in time, an activity instance is in exactly one state
- In the example
  - At the beginning, only *enter order details* is ready for execution
  - After this activity instance has terminated, *obtain product from warehouse* is ready for execution, etc.

Activity Instances

- States of an activity instance
  - Init: it is initialized, but not ready for execution
  - Ready: it is ready for execution, it can be started
  - Running: it is currently being executed
  - Terminated: it has terminated
  - Skipped: it is no longer required

Activity Instances

- The states of an activity instance can be represented by a state transition diagram
  - The states and their transitions are an essential part of the semantics of a process model

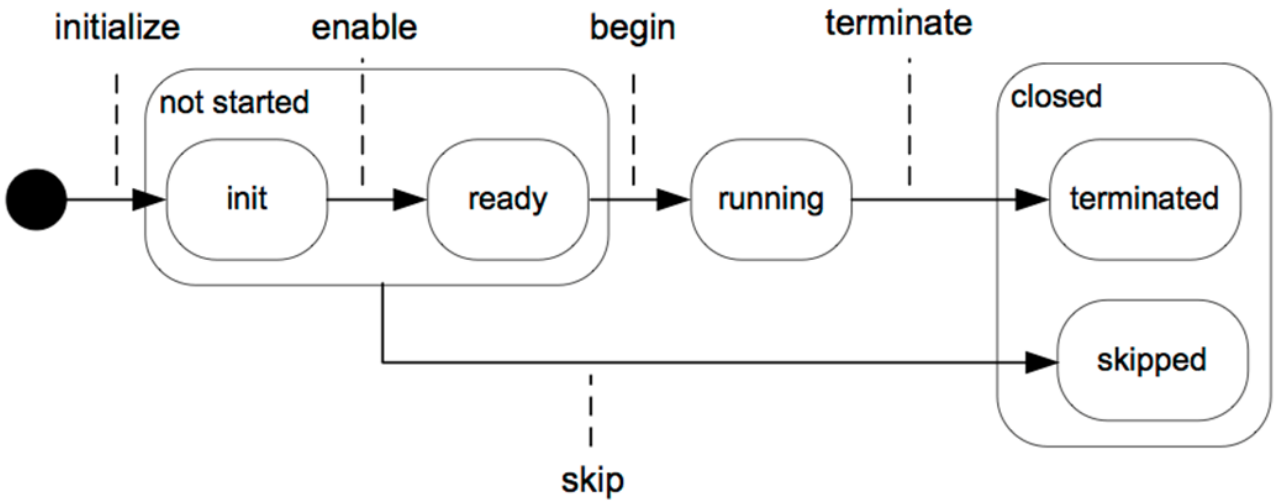


Fig. 3.9. State transition diagram for activity instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

- Notice
  - This is a simplified variant of the BPMN activity state transition diagram, which is sufficient for our course

Example

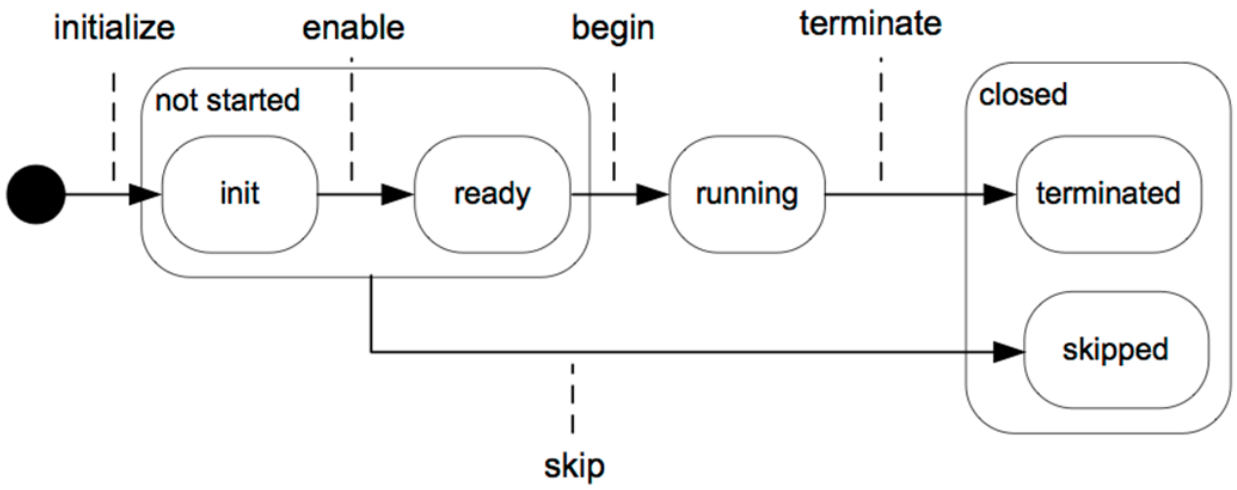


Fig. 3.9. State transition diagram for activity instances

M. Weske: Business Process Management,
© Springer-Verlag Berlin Heidelberg 2012, 2007

Modeling Guideline

Label activities in the "verb noun" style.
Use a consistent vocabulary.

- Motivation
  - The "verb noun" style, e.g., *package product*, indicates an activity
  - The same noun should be used throughout a model, if it refers to the same domain concept
  - Consistent vocabulary should also be ensured across models, e.g., with the help of a glossary of terms of an organization

Activities and Task Types

- A task can be of a certain type, which characterizes the task further
  - Task types are represented by markers in the upper left corner

- Task Types (excerpt)
  - Message task: Sending / Receiving a message to / from a partner
  - User task: Task is conducted manually
  - Service task: Task is carried out automatically, e.g., by software

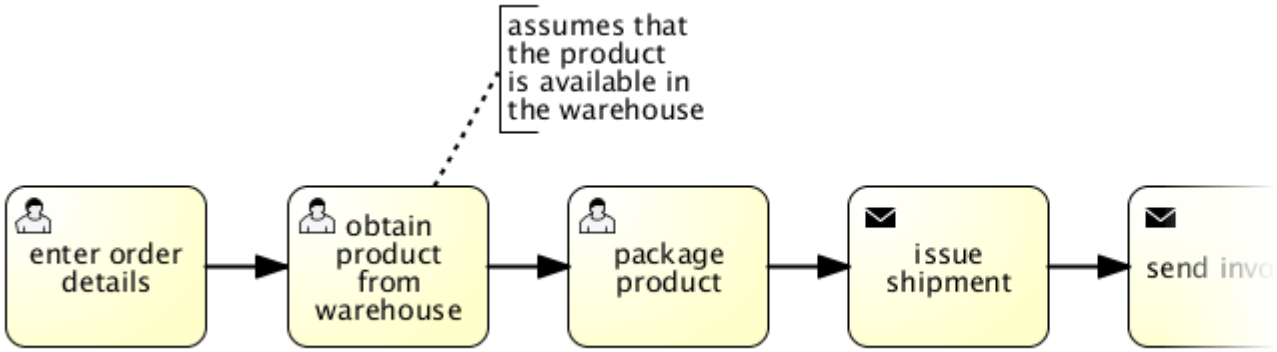
BPMN Attributes

- BPMN is more than a visual diagram representation
- Attributes hold additional information of BPMN elements
  - Not all attributes have a visual representation
  - In general, the diagram does not show the complete information of the process model
- Task types
  - Stored as attributes, which do have a visual representation

- Notice
  - We will discuss attributes w/o visual representation later in this course

Annotations

- To improve the understandability of business process models, they can be enriched with annotations
  - Annotations can be attached to any model element to describe it in more detail
- Annotations do not have any execution semantics
  - But they can relate useful information to specific model elements



Video Clip 2.2
Process Activities

- Activities are units of work
- Order fulfillment scenario
- Sequence flow
- Activity instances, and their state transitions
- Modeling guideline
- Task types, attributes, and annotations

Video Clip 2.3

Exclusive and Parallel Gateways

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Application Scenario (cont.)

The process starts by entering the details of an order for a product into an IT system.

This is followed by checking whether the ordered product is available in the inventory.

If the product is available, it will be obtained from the warehouse.

Otherwise, the materials for the product need to be obtained and the product be manufactured.

Subsequently, the ordered product is packaged and shipped to the customer.

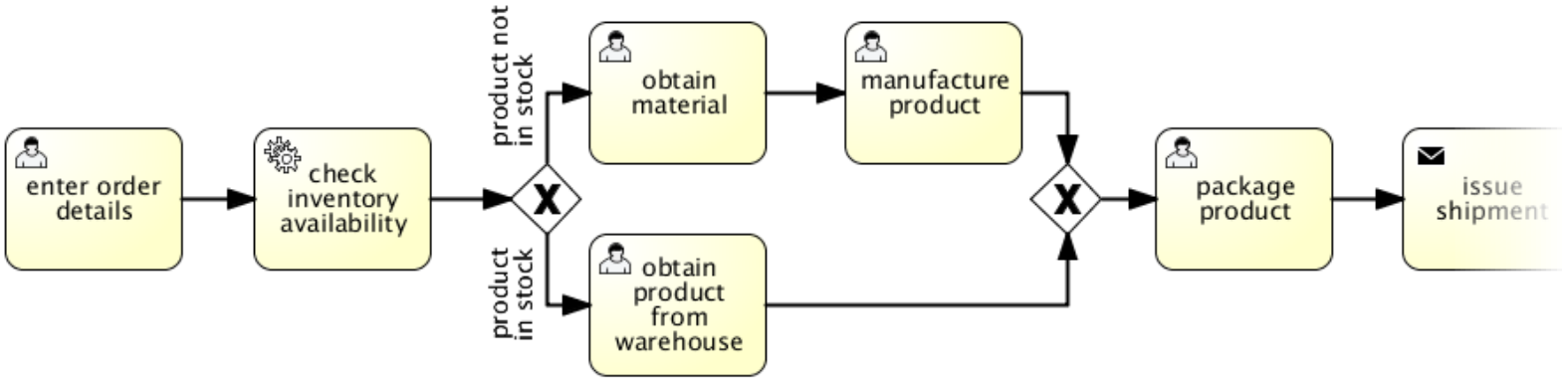
An invoice is sent to the customer. Upon receiving the payment the order is archived.

Basic Gateways

- Sequence flow can only express ordering of activities in terms of sequences, which is often not sufficient
  - Richer execution conditions can be expressed by gateways
- Gateways have the generic shape of a diamond and are supplemented with a symbol that specifies their behavior
  - Gateways can be used for splitting and joining sequence flow
- In this section, we introduce
  - Data based exclusive gateway (exclusive gateway, XOR gateway)
  - Parallel Gateway (AND Gateway)

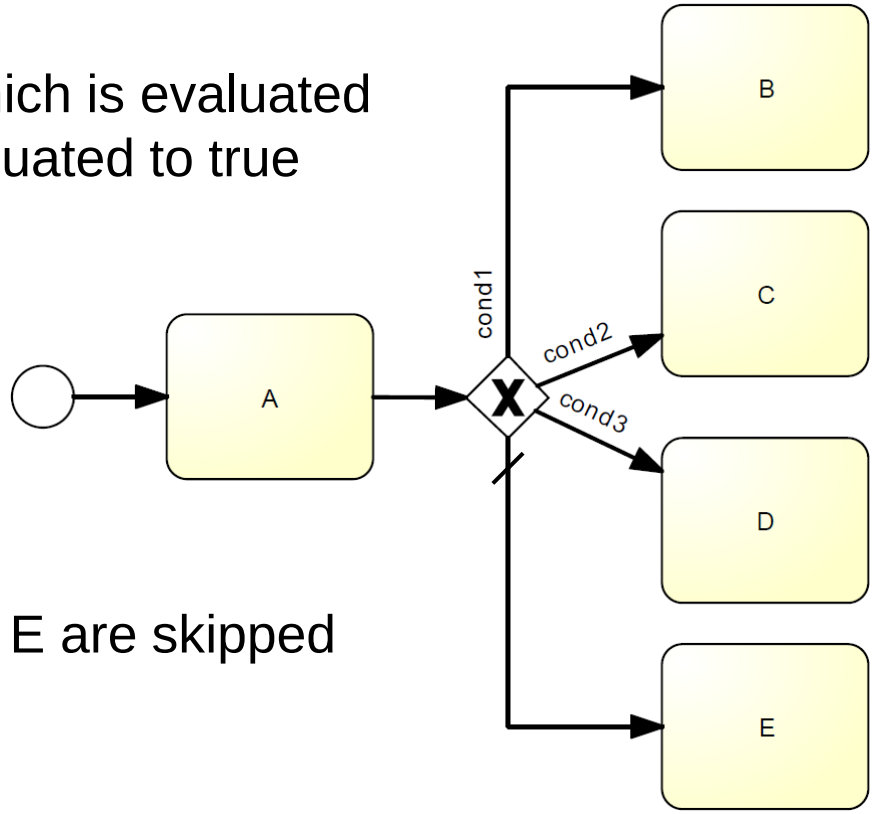
Exclusive Gateway

- The exclusive gateway can be used as split and join node
- XOR split (exclusive or split)
  - From a set of alternatives, exactly one is chosen
  - Conditions attached to outgoing sequence flow arcs are evaluated
- XOR join (exclusive or join)
  - Merges alternative paths
  - For each incoming sequence flow that is signaled, an outgoing sequence flow is signaled



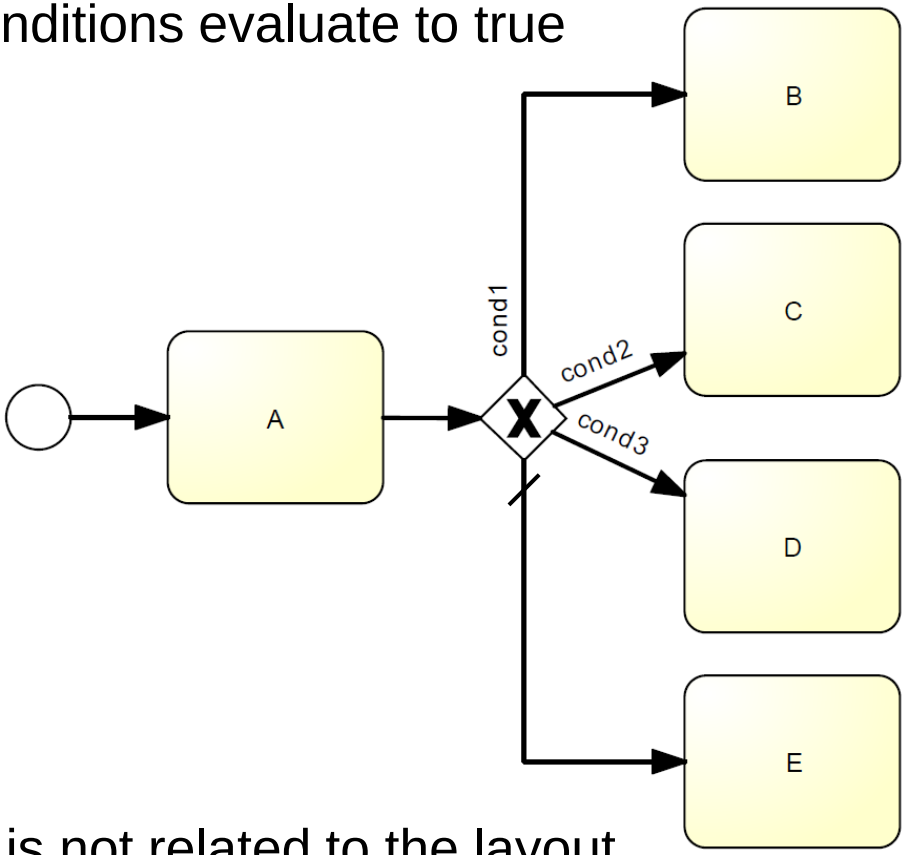
Exclusive Gateway Semantics

- Semantics of the XOR split, in more detail
  - Outgoing edges of the split have conditions attached
  - Conditions are evaluated in “specific order”
  - The arc of the first condition evaluated to true is signaled
  - All other arcs are discarded
  - There can be a default flow, which is evaluated last; its condition is always evaluated to true
- Example
  - Let the evaluation ordering be
    - cond1, cond3, cond2, default
    - Let cond1 evaluate to false
    - Let cond3 evaluate to true
  - Activity D is enabled, B, C, and E are skipped



Exclusive Gateway Semantics

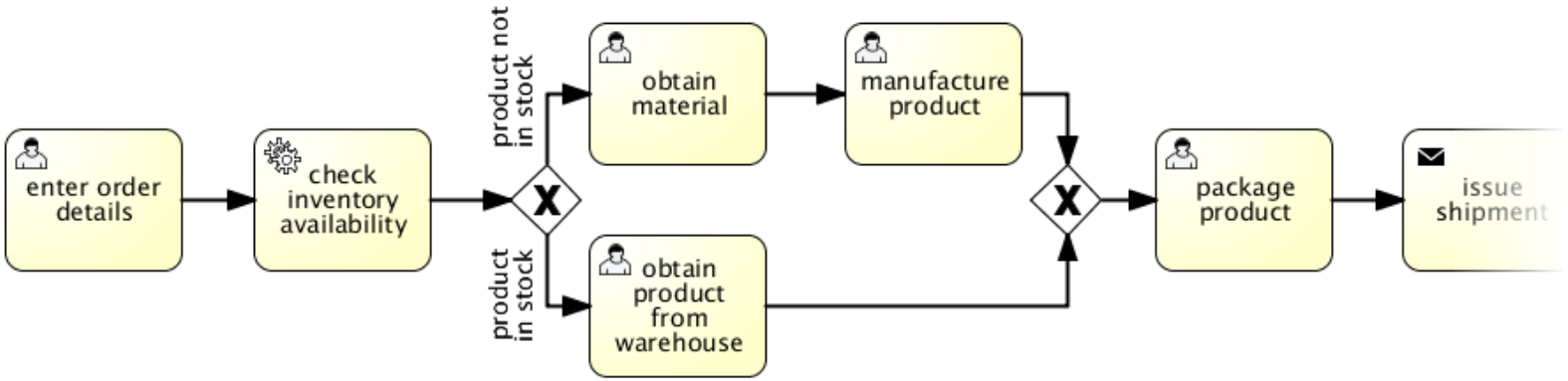
- XOR semantics (there is exactly one arc selected) is preserved, even if the conditions are „overlapping“
  - In a given situation, several conditions evaluate to true
- Example
  - cond1: Cost < 200
  - cond2: Cost < 1000
  - cond3: Cost < 10000
  - Evaluation ordering
    - cond1, cond3, cond2, default
  - For Cost=500, two conditions evaluate to true, but D starts
- Notice
  - The ordering of the evaluation is not related to the layout



Modeling Guideline

Always use default flow with an exclusive gateway.

- Motivation
  - If neither condition evaluates to true and there is no default flow, the process cannot continue
  - If you want to be sure which path is taken in a certain situation, make sure conditions of an exclusive gateway are non-overlapping



Application Scenario (cont.)

… the materials for the product need to be obtained and the product be manufactured.

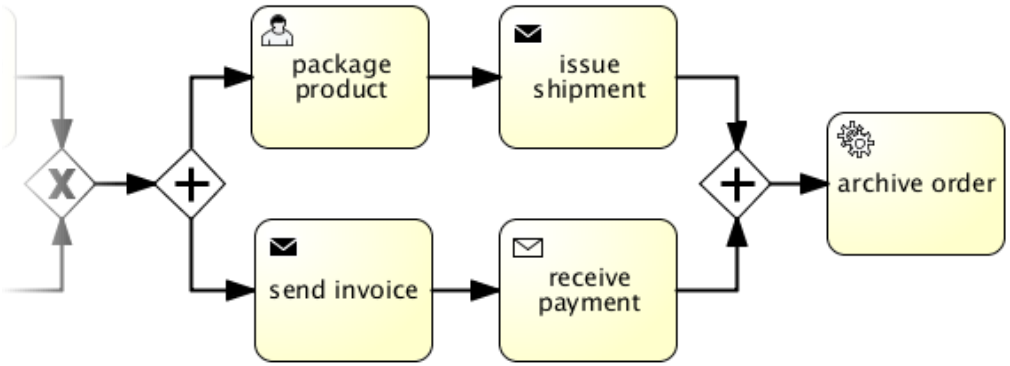
Subsequently, the ordered product is packaged and shipped to the customer.

At the same time, an invoice is sent to the customer, from whom a payment is received later.

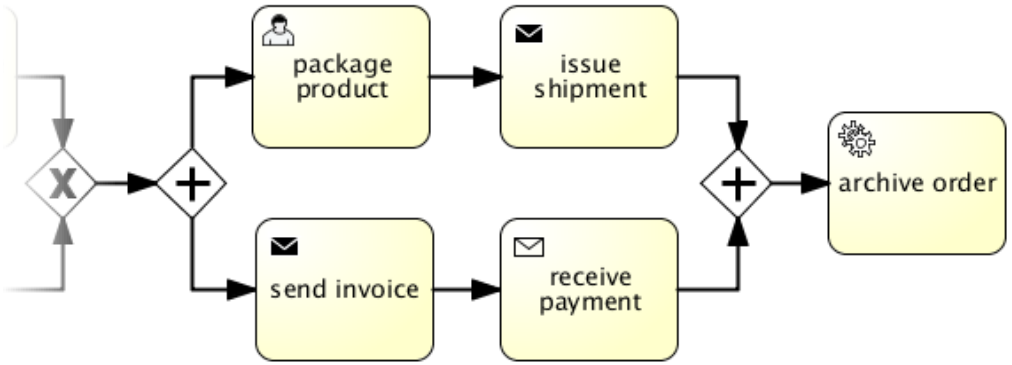
Once the product has been shipped successfully and the payment has been received, the order can be archived.

Parallel Gateway

- AND split (parallel split)
  - All outgoing sequence flow arcs are signaled
  - Allows modeling independent, concurrent execution paths
  - AND splits do not require parallel execution, but allow for arbitrary interleaving and execution in any order



Parallel Gateway



- AND join (synchronizing join)
  - Synchronizes all incoming sequence flow arcs
  - Only if all incoming sequence flow arcs have been signaled, the outgoing arc is signaled

Video Clip 2.3
Exclusive and Parallel Gateways

- Exclusive Gateway
- Parallel Gateway
- Syntax and Semantics
- Modeling Guideline

Video Clip 2.4

Inclusive Gateways and Loops

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Application Scenario (cont.)

… if the product is available, it will be obtained from the warehouse.

Otherwise, the materials for the product need to be obtained and the product be manufactured.

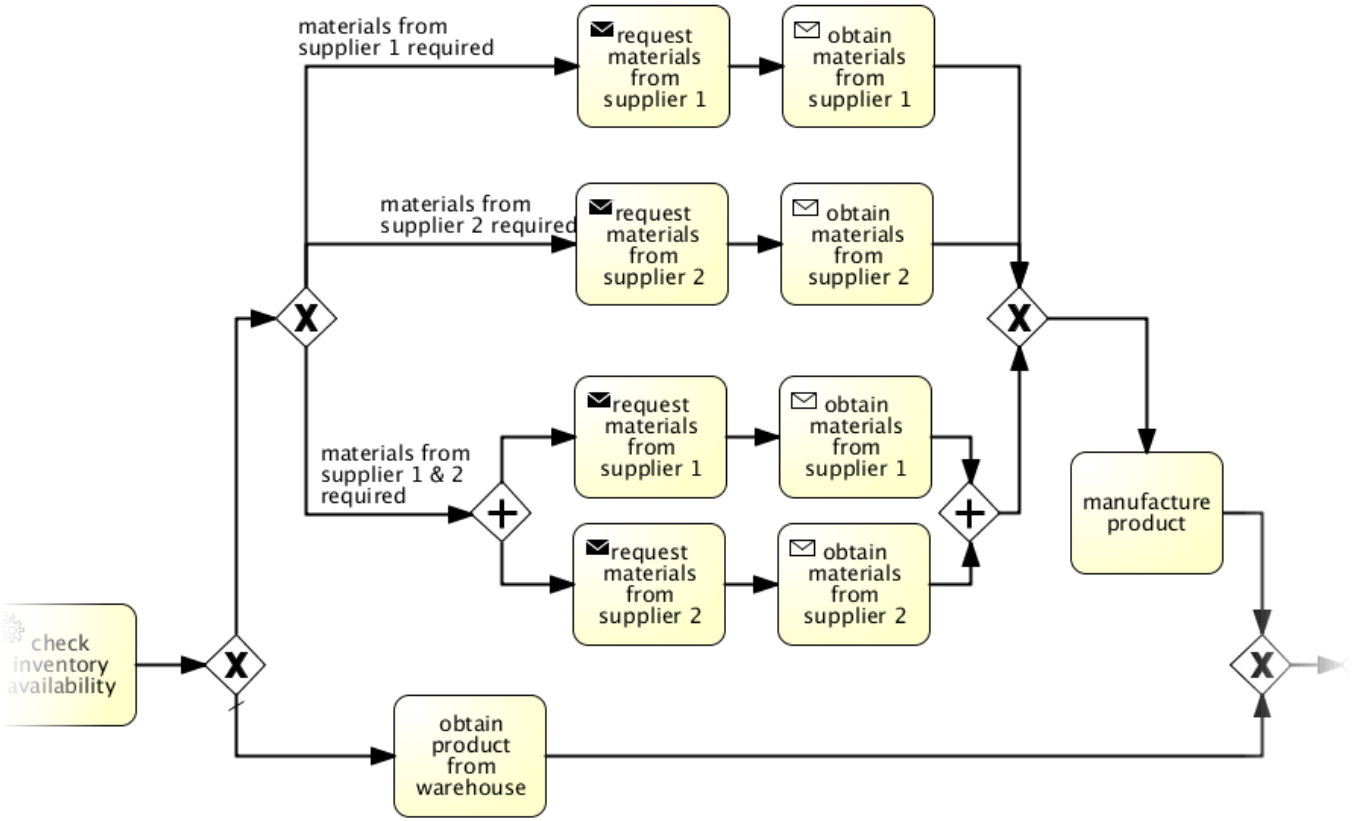
Depending on the ordered product, materials from supplier 1 or supplier 2 may be sufficient to manufacture the product.

In some cases, however, materials from both suppliers are required for manufacturing the product.

Subsequently, the ordered product is packaged and shipped to the customer.

Complex Conditional Execution

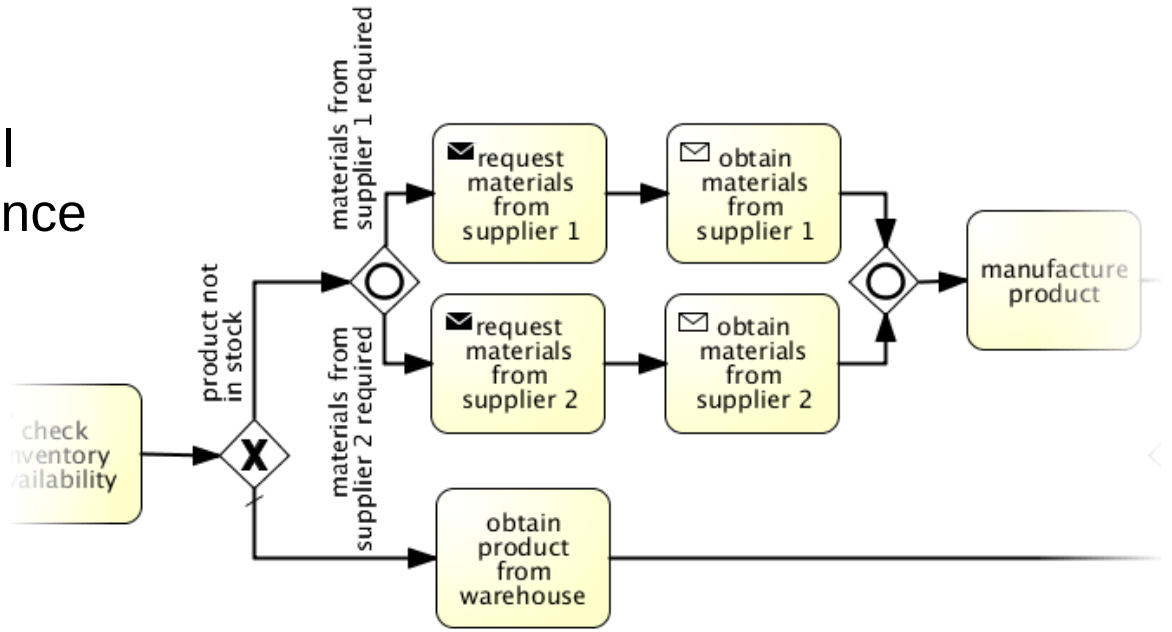
- Can be modeled by combining XOR and AND gateways



- Notice
  - The duplication of tasks increases complexity of the model

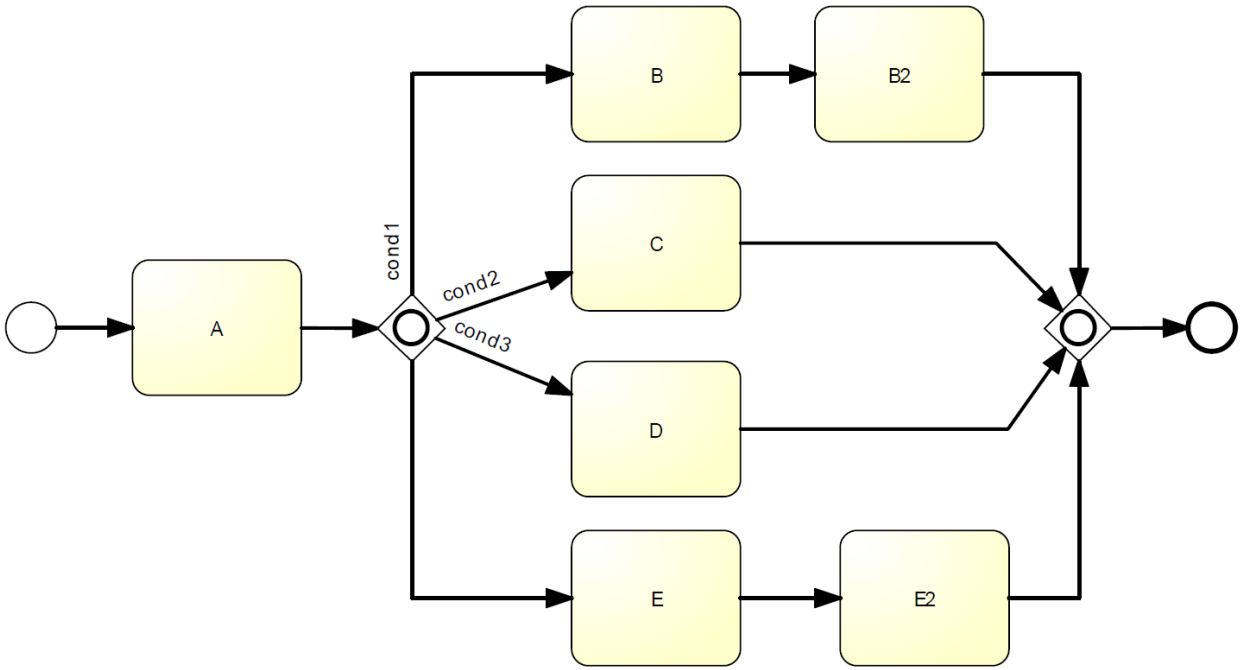
Inclusive Gateway

- OR split (inclusive or split)
  - Any nonempty subset of outgoing sequence flow arcs is signaled, depending on conditions
  - Each condition evaluated to true signals an outgoing arc
  - Default flow is used only if all other conditions are false
- OR join
  - Synchronizes all incoming sequence flow arcs that can be signaled



Inclusive Gateway

- Remarks on the OR join
  - This modeling construct provides a high degree of flexibility, but it also has a non-local execution semantics
    - The join cannot decide locally whether to signal its outgoing edge
  - Block structured OR joins are quite easy to handle, using *false token passing*



  - Arbitrarily structured OR joins are really complex [Kindler2004]

Application Scenario (cont.)

The process starts by entering the details of an order for a product into an IT system.

Then the order is checked for correctness and completeness.

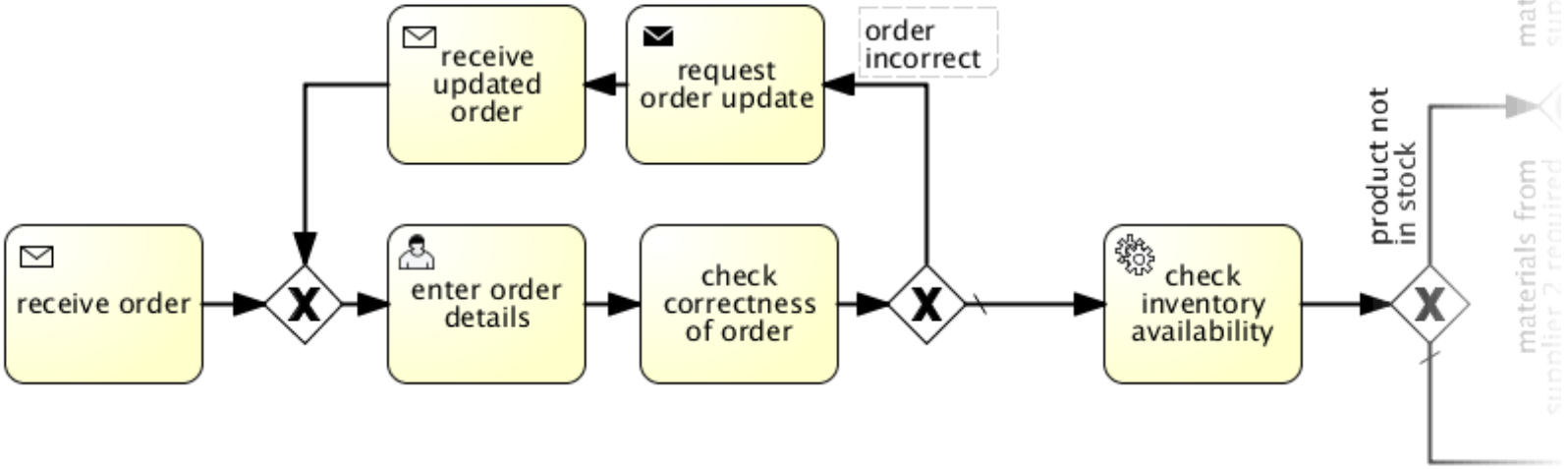
If the order details are not valid, an order update is requested, which needs to be registered in the IT system and checked again.

This is repeated until the order is valid.

This is followed by checking whether the ordered product is available in the inventory.

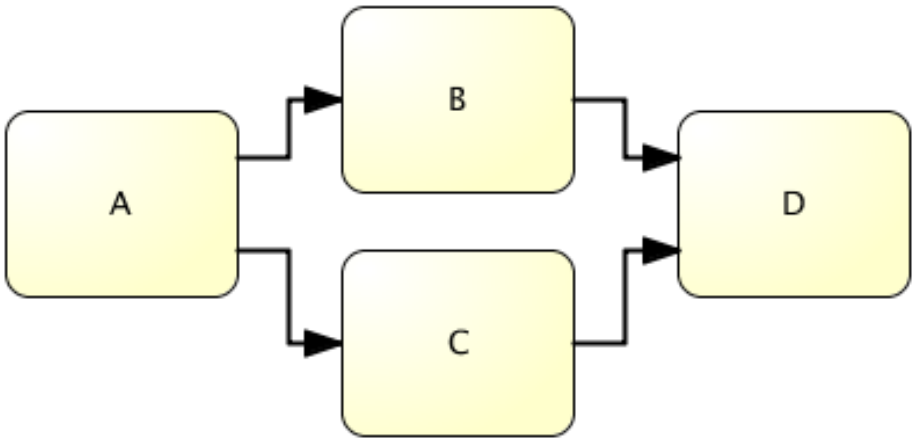
Loops

- Iterations can be modeled with the exclusive gateway
  - The branching condition of the XOR split acts as the loop condition
  - Loop iterations require the creation of new activity instances
  - There can be many activity instances for a given process activity in one process instance



Uncontrolled Flow

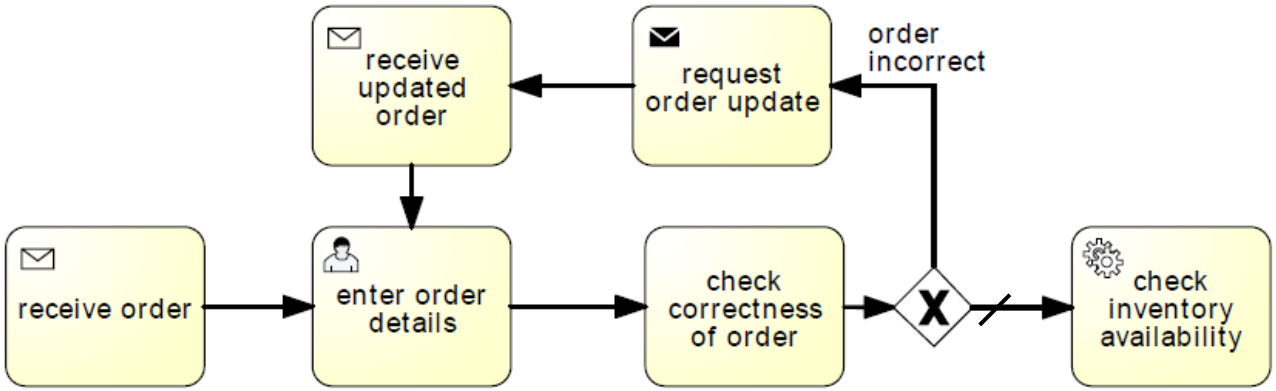
- Activities may have several incoming and outgoing sequence flow arcs
- Signaling these arcs is determined by the “uncontrolled” flow semantics
  - Each outgoing edge is signaled
  - Each incoming edge that is signaled executes the task
- Uncontrolled flow might lead to confusion, because
  - Split has AND semantics
  - Join has XOR semantics



Modeling Guideline

Use gateways for splits and joins

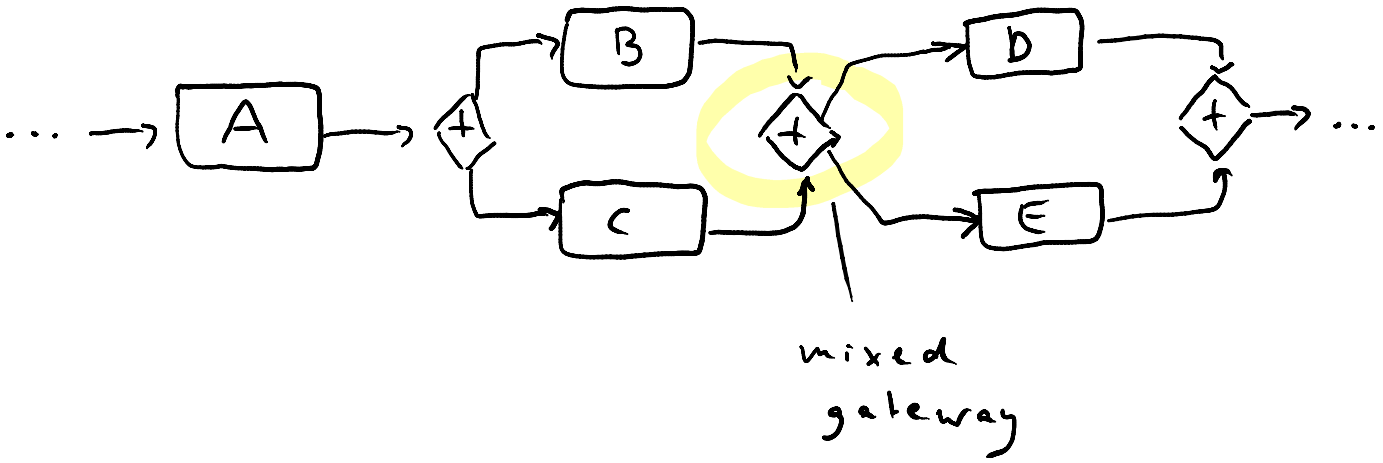
- Motivation
  - Uncontrolled flow semantics are often difficult to comprehend
  - Each activity should have one incoming and one outgoing sequence flow edge
- Exception: Loops



Modeling Guideline

Avoid mixed gateways

- Motivation
  - Split and join are different semantics and shall be treated separately for the sake of clarity
  - By changing the split behavior of a mixed gateway, we change the join behavior, which might lead to undesired results



Video Clip 2.4
Inclusive Gateways and Loops

- Inclusive Gateway
- Split and Join
- Loops
- Uncontrolled Flow
- Modeling Guidelines

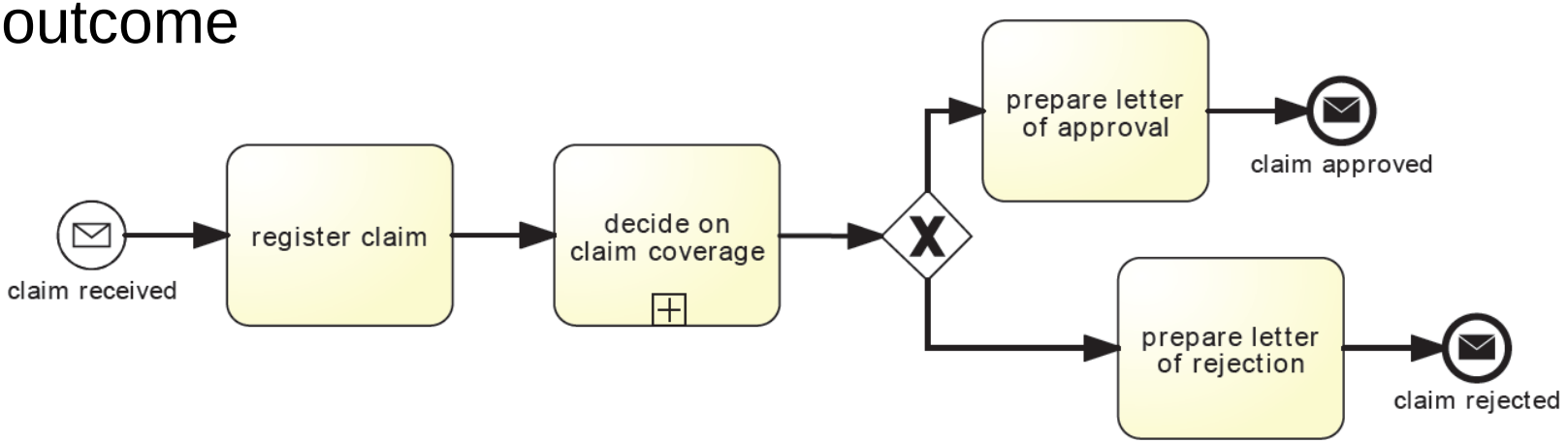
Video Clip 2.5

Start Events and End Events

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Start Events and End Events

- The interaction of a business process with its environment is represented by events
- Events are happenings of the real world that
  - have a business meaning
  - do not take time
- A start event models the start of a business process
- An end event models process completion, with an outcome



Application Scenario

The process starts when receiving an order.

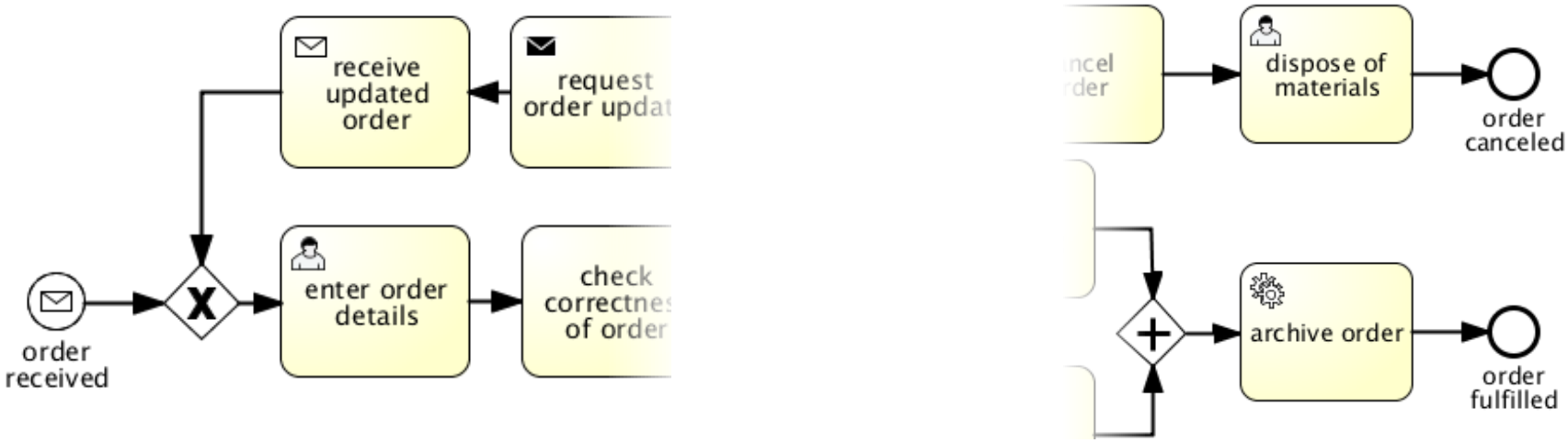
...

After archiving the order, the order is fulfilled, and the process terminates successfully.

If the order cannot be fulfilled, the process terminates unsuccessfully, the order is cancelled

Events in the Scenario

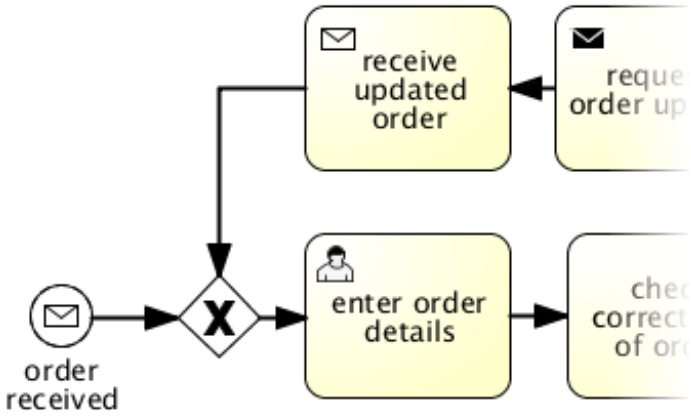
- Remark
  - Successful and unsuccessful termination is separated by event labels

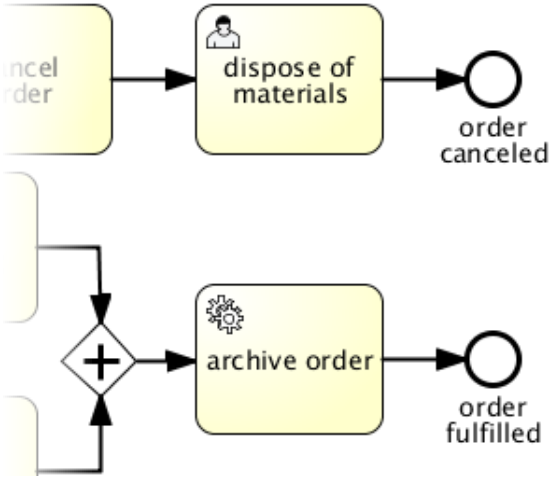


- Notice
  - Start event and end events have a different character, regarding event creation

Catching and Throwing Events

- An event can be *caught*
  - The event happens in the environment and its occurrence has an effect on a process instance
- An event can be *thrown*
  - The event is created by a process instance





BPMN Event Types

- Start events (Catching)
  - Cause the start of a new process instance
- End events (Throwing)
  - Signal the termination of a process instance
- Intermediate events (Catching or Throwing)
  - Can be caught/thrown during a process instance
  - Events do not take time, but a process might have to wait for an intermediate catching event to occur

BPMN Event Kinds

- BPMN defines event kinds that characterize the nature of the event, e.g.,
  - Timer has expired, error has occurred, message was received
  - These event kinds can be combined with event types, but not all combinations are allowed
- Each event kind has a symbol; e.g.,

 Timer intermediate event

 Error end event

 Message start event

 „None” start event (or blank start event)

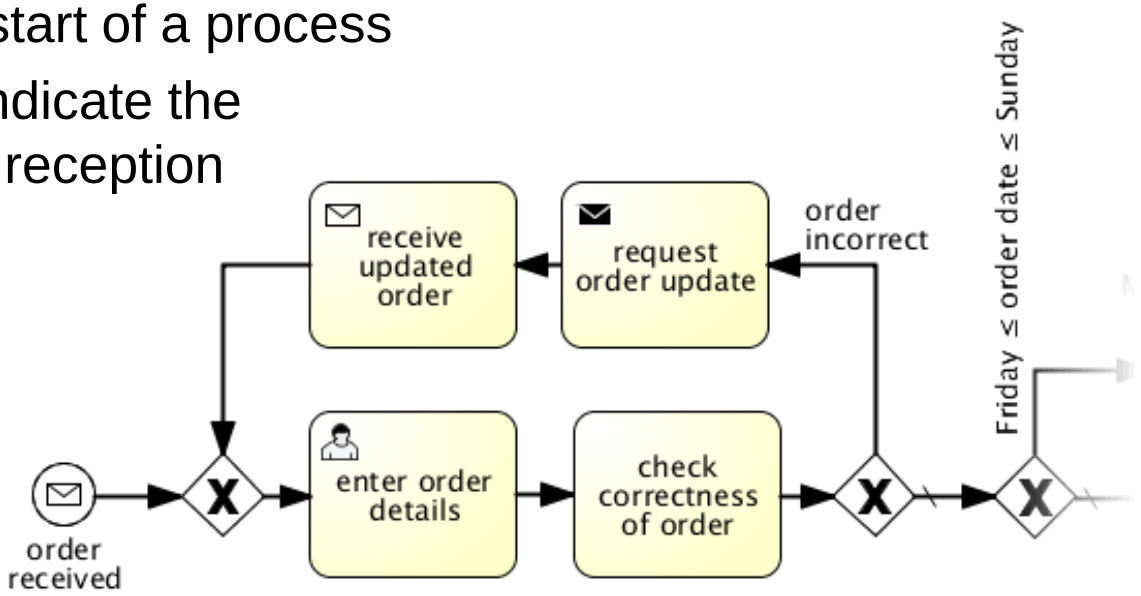
Modeling Guideline

Events do not consume time.
Label events in the “noun verb (participle)” style.

- Motivation
  - Events are observed changes of the state of a system, the process environment, or the process itself – they must not be used to model activities
  - To emphasize the occurrence of an event, label events using the “noun verb (participle)“ style, e.g., “order received“
    - Exception: timer events (“every Monday at 10 a.m.”)
  - Use a consistent vocabulary

Start Events and Event Kinds

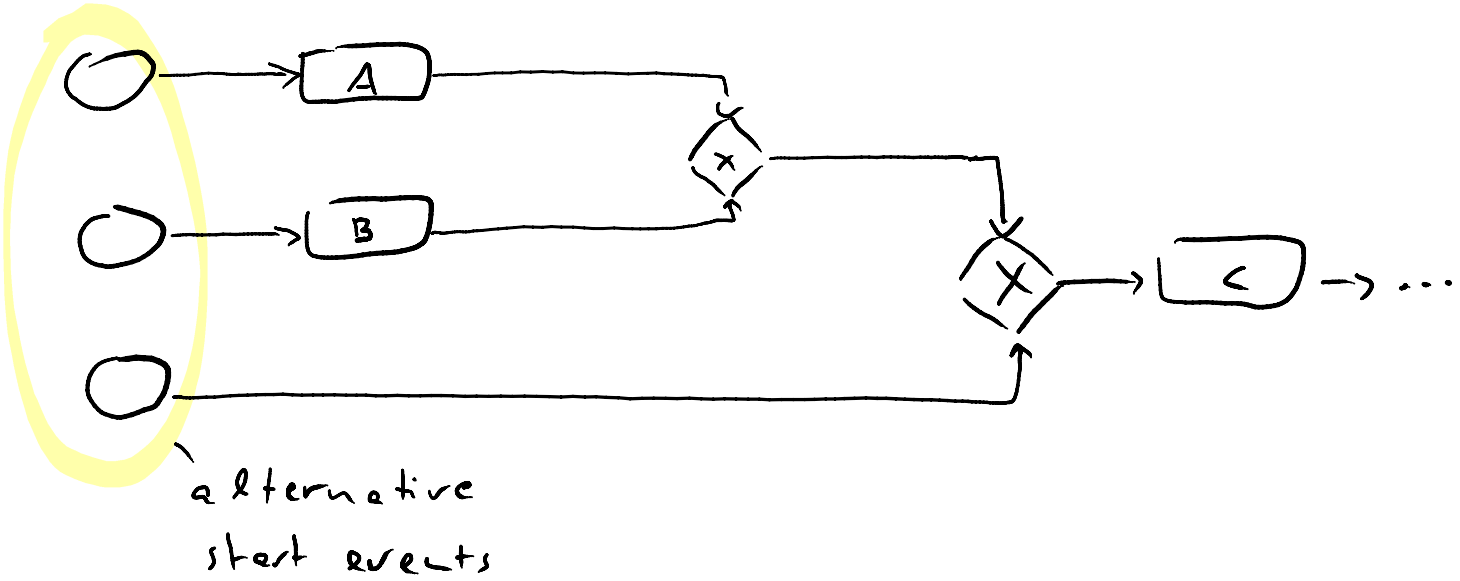
- A start events triggers the creation of a new process instance and starts its execution
- Kinds of start events
  - Blank start events are used, if a process is started manually or if the kind of start event is not relevant
  - Timer start events allow the definition of a particular date/time, or recurring dates for the start of a process
  - Message start events indicate the start of a process upon reception of a message



Modeling Guideline

Start each business process model with exactly one start event.

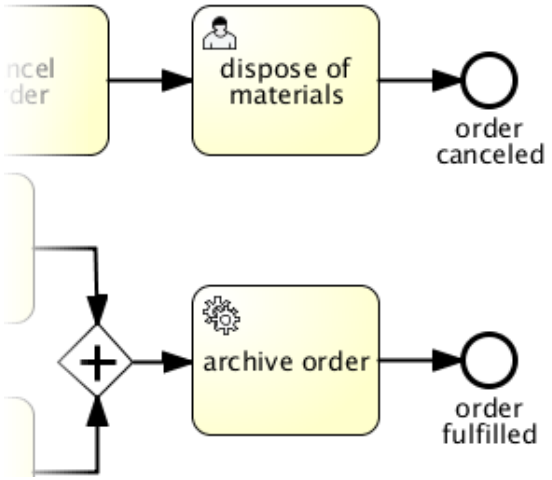
- Motivation
  - Process instantiation is easy to understand, if there is exactly one start event
  - If more than one start event is required, these shall be alternative



Modeling Guideline

Use a distinct end event for each possible outcome of the process.

- Motivation
  - Different results of a process often have different business meanings and shall be distinguished therefore by distinct end events
  - End events shall be exclusive
  - In this case it is clear when a process terminates



Video Clip 2.5
Start Events and End Events

- Process interacts with its environments by events
- Events do not take time
- Start and end events
- Throwing and catching events
- Event kinds: message, timer, etc.
- Modeling guidelines

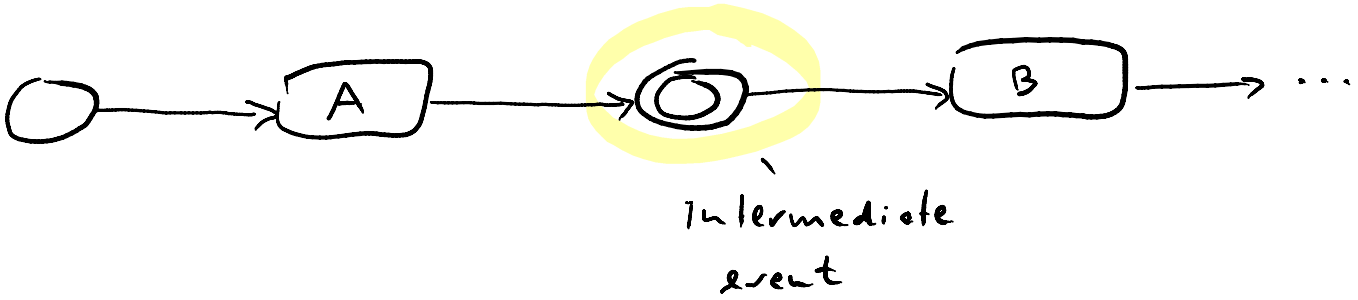
Video Clip 2.6

Intermediate Events

BPMN Meets DMN: Business Process and Decision Modeling:
An openHPI Course by Mathias Weske

Intermediate Events

- Intermediate events happen after the start of a business process and before its termination
- Intermediate events can be catching or throwing
  - Catching: the process waits for the occurrence of the event
  - Throwing: the process emits the event and continues with the outgoing sequence flow
- An intermediate event can be connected to other nodes of a process model by incoming and outgoing sequence flow arcs



Application Scenario (cont.)

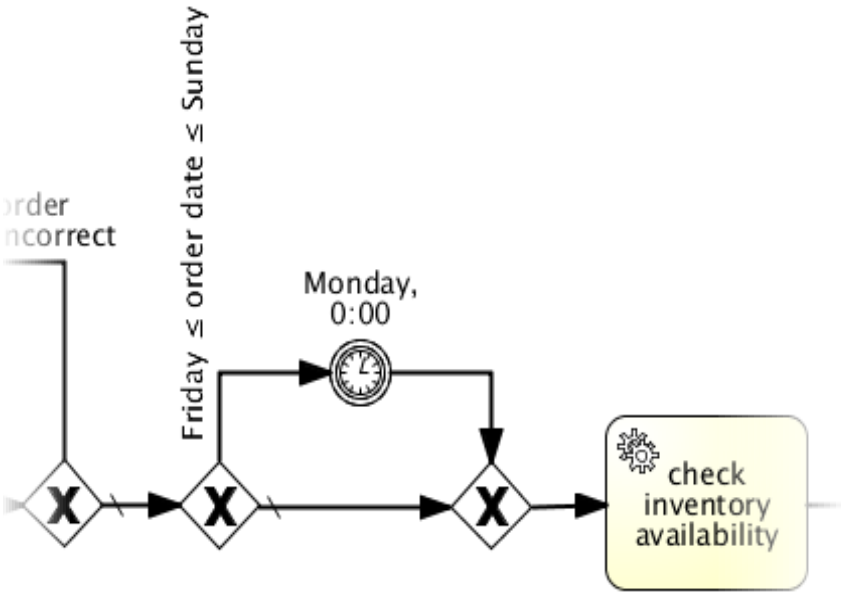
...

If the order is placed on a Friday or on a weekend, the inventory check shall be delayed until Monday, as the inventory is updated every Sunday evening.

...

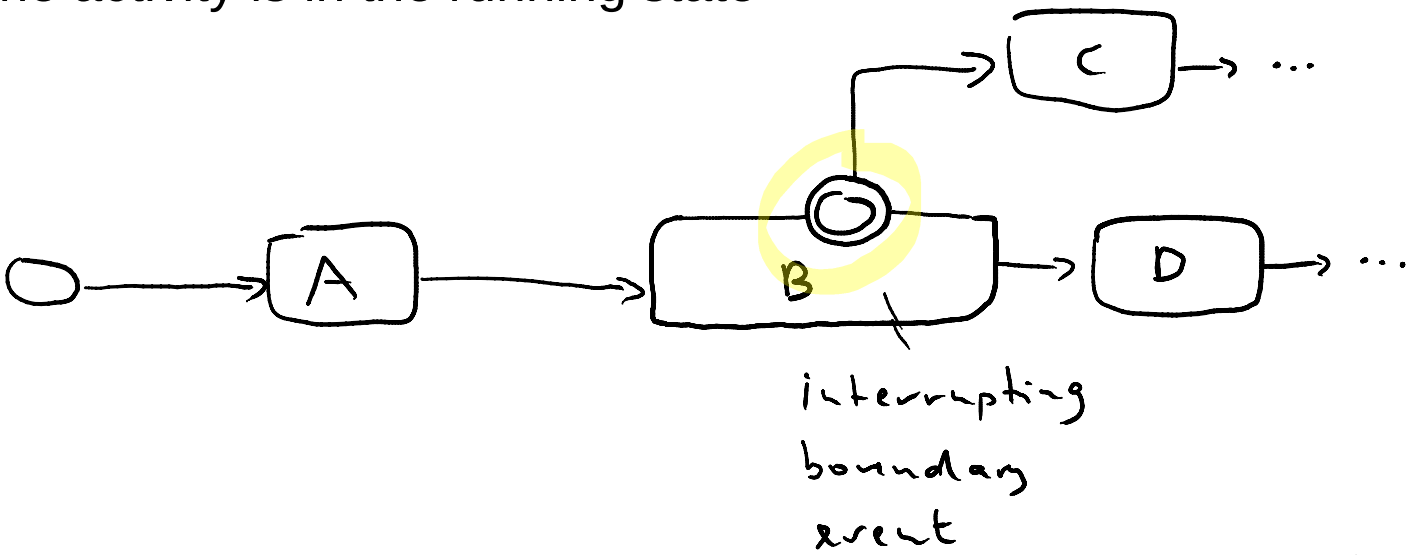
Intermediate Timer Event

- Intermediate timer events can only be catching
- They act as a delay mechanism
  - By a specific date / time, e.g., “December 16, 2013”
  - By a recurring date / time, e.g., “every Saturday at 10:00 am”
  - By a duration, e.g., “one week”



Boundary Event

- A boundary event is on the boundary of an activity
  - Boundary events are always catching events
  - The occurrence of a boundary event is only relevant, if it occurs while the activity is in the running state



- Interrupting boundary events: Interrupt the activity

- Non-interrupting boundary events: Do not interrupt it

Application Scenario (cont.)

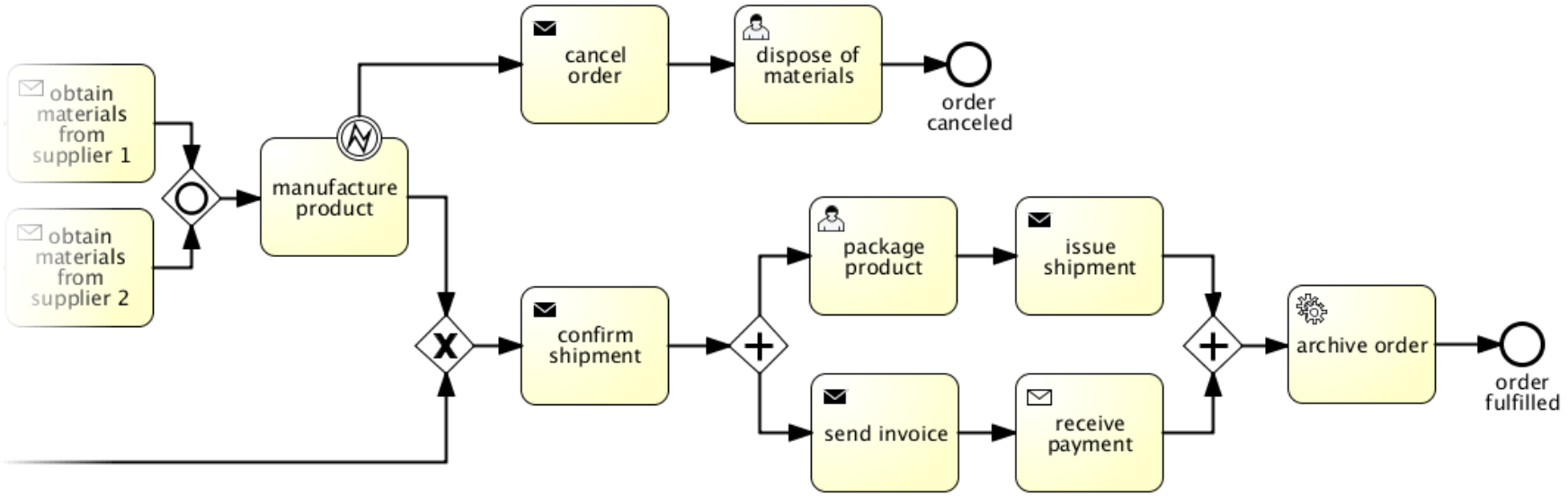
...

If the product is not in stock and an error occurred during manufacturing the product, the order shall be canceled and the obtained materials be disposed of. In that case, the process terminates unsuccessfully.

...

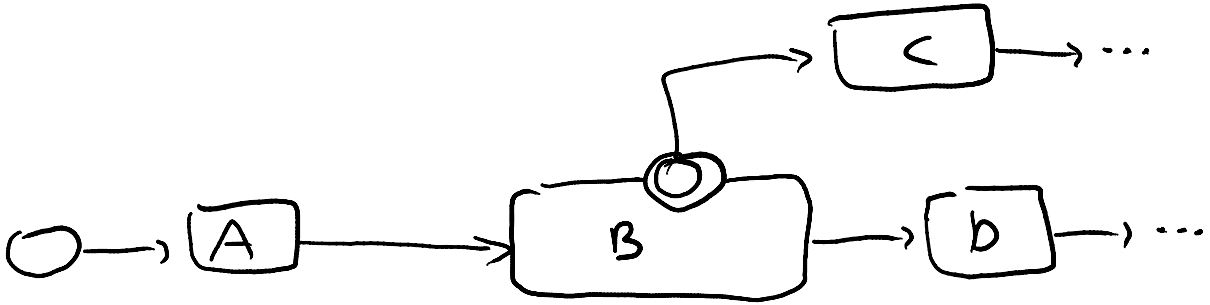
Interrupting Boundary Error Event

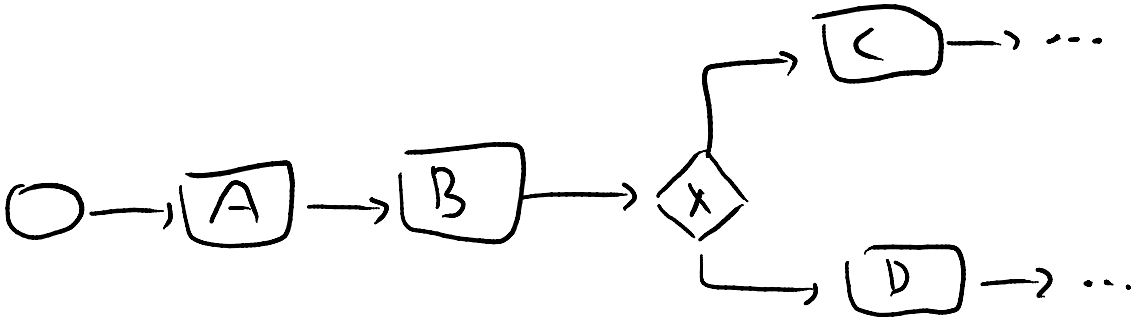
- When the boundary event occurs, *manufacture product* is interrupted and the flow continues with cancelling the order
  - Typically, activities to handle the error situation are started after a boundary error event occurred



Interrupting Boundary Event and XOR Split

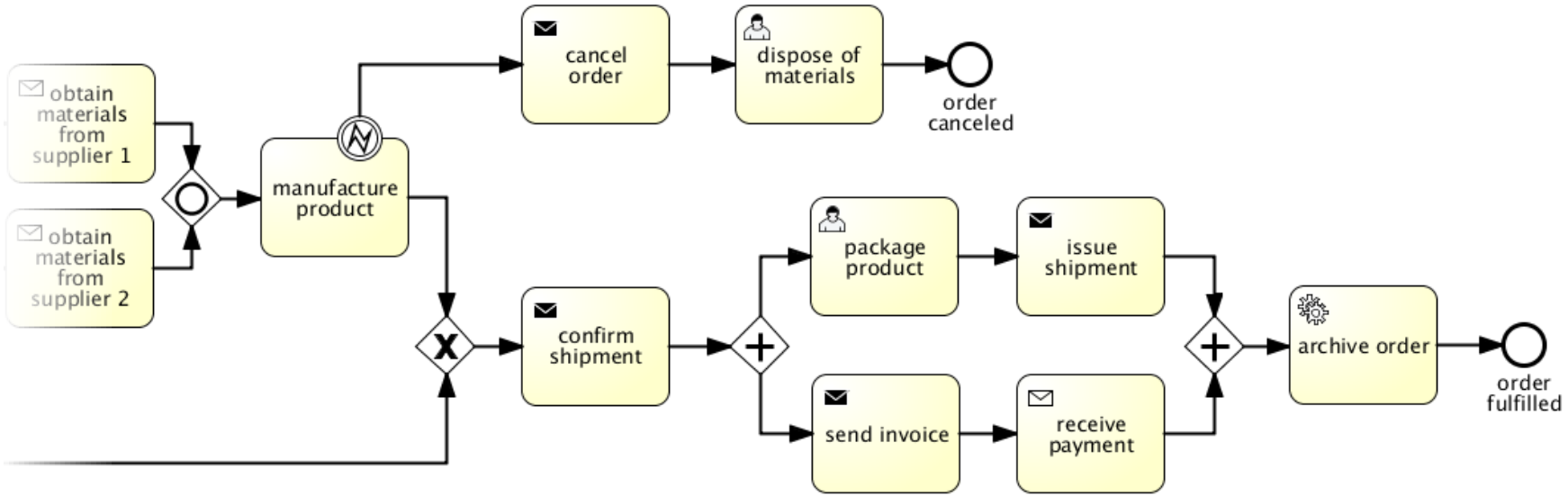
- What is the difference to an XOR split decision?





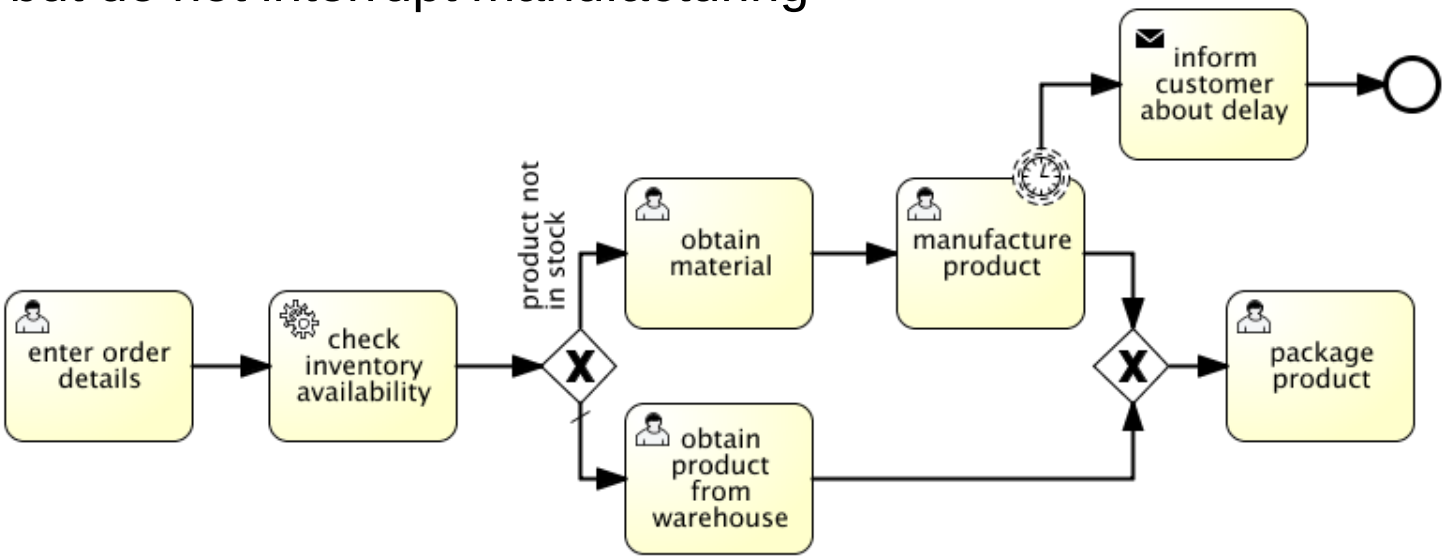
Interrupting Boundary Event and XOR Split

- There are subtle differences
  - Decision by event occurrence / by conditions
  - There is a difference in the execution semantics, since in the XOR case, the activity completes, while it does not in the case of the boundary event
  - From the modeling perspective, it is clearer that an error has occurred rather than a normal decision during a process



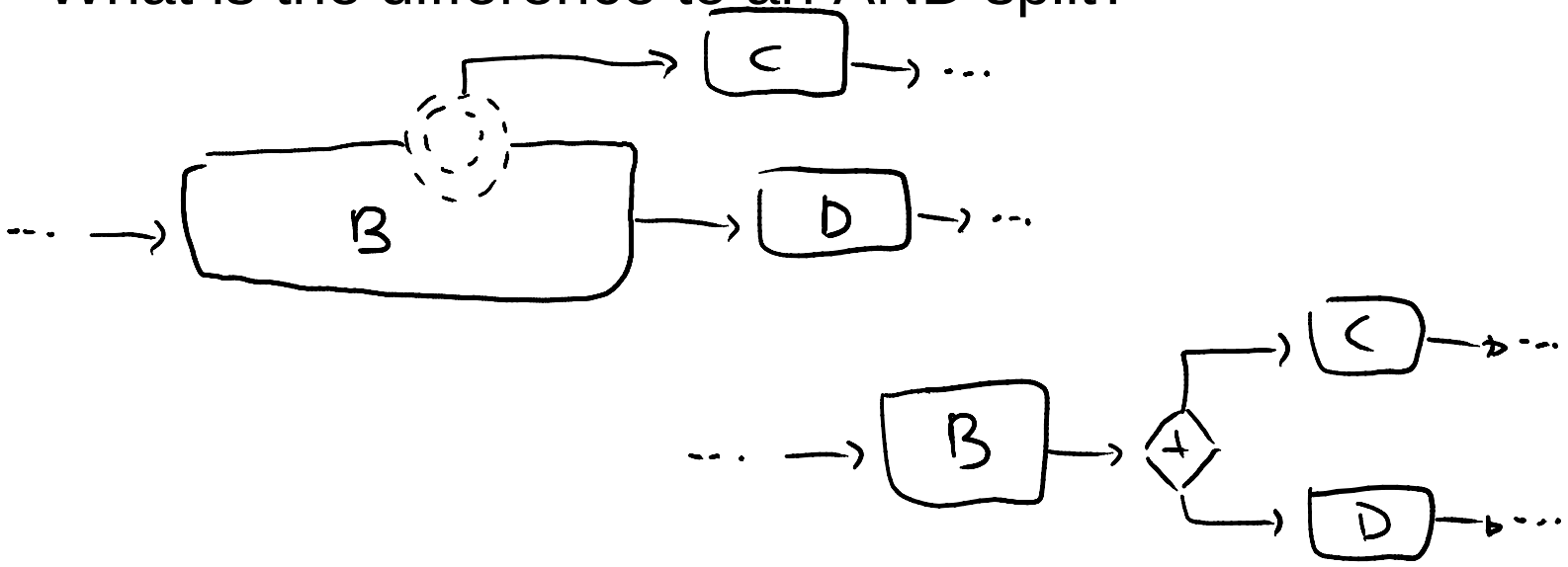
Non-Interrupting Boundary Event

- Boundary events might also be non-interrupting
  - In this case, the activity continues normally, while spawning a reaction to the event
  - Notation: circle with a dashed double border
- Example
  - Inform the customer if manufacturing takes longer than expected, but do not interrupt manufacturing



Non-Interrupting Event and AND Split

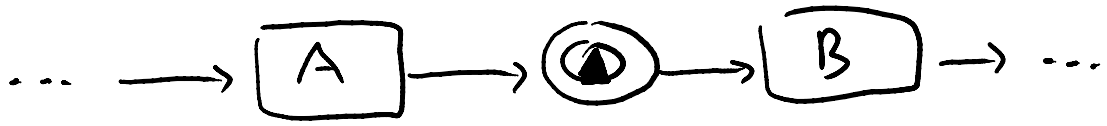
- What is the difference to an AND split?



- There are subtle differences as well
  - There is a difference in the execution semantics, since in the AND case, activities are sequential, while this is not the case with the boundary event
  - We need to be careful with joining the flows in an AND gateway, since the event can occur several times

Intermediate Signal Events

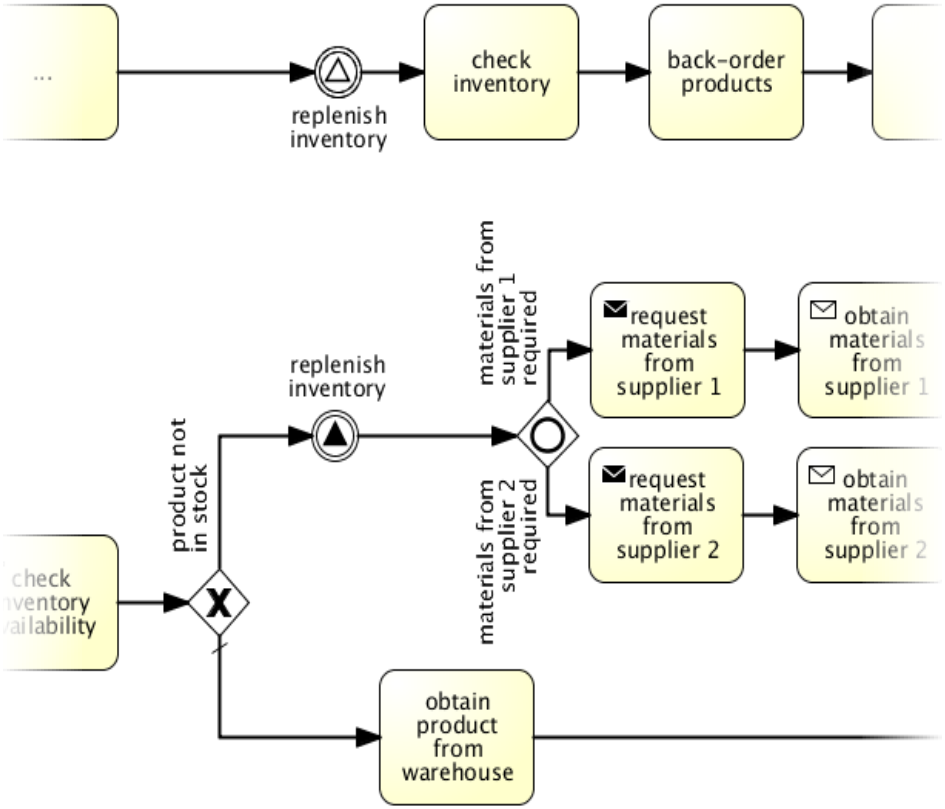
- Scope of events
  - The scope of throwing and catching events is a single process instance and its environment (timer, incoming messages, etc.)
- Signal events lift this assumption
  - Signal events can be caught by any other business process



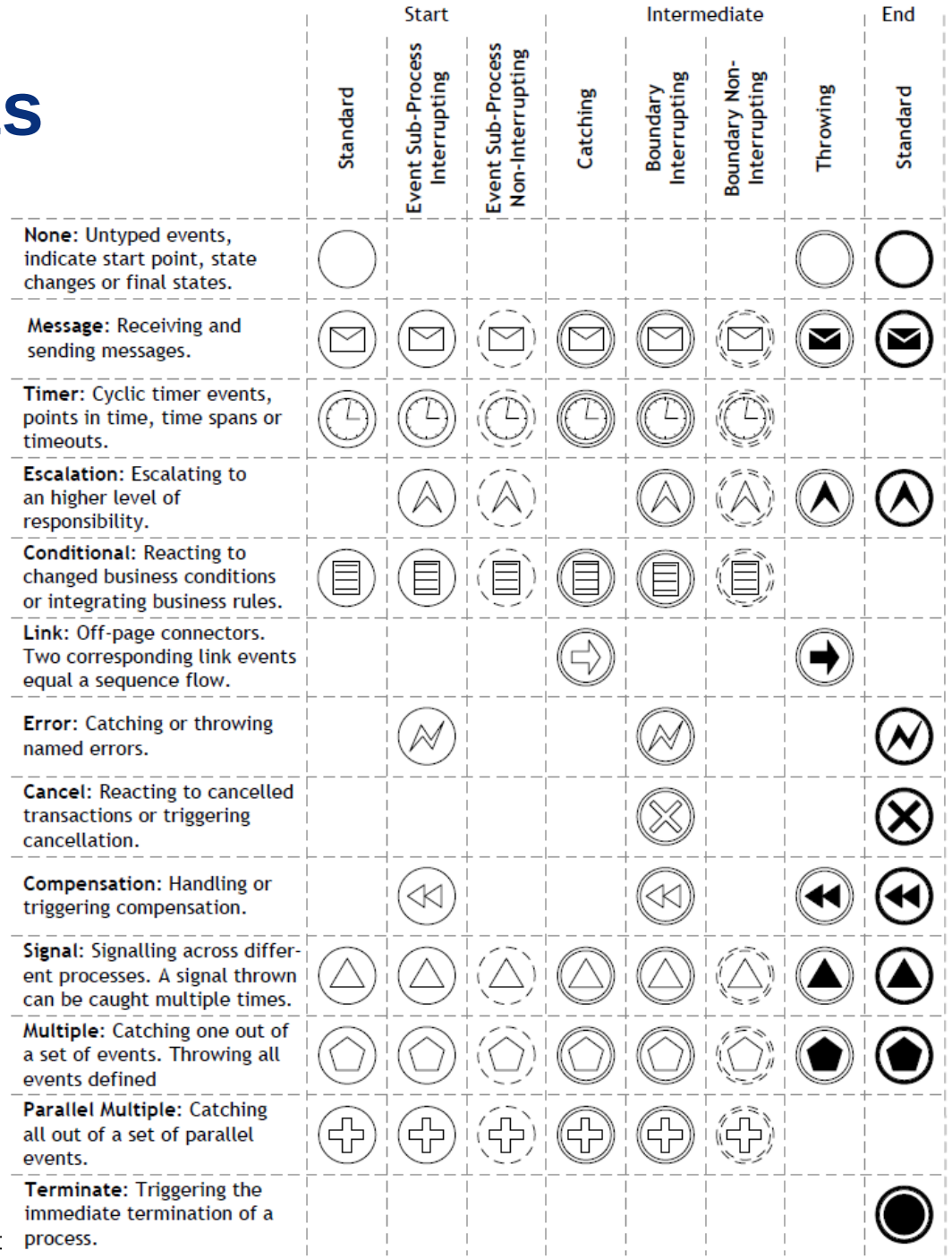
- A BPMN Signal is similar to a signal flare that shot into the sky is visible to everyone who might be interested (from the BPMN 2.0 Specification)

Intermediate Signal Events

- Example
  - A signal event may be used to inform another process to replenish the inventory, if a product is not available



Overview of Events



Source: BPMN Poster
http://www.bpmb.de/index.php/BPMNPoster

Video Clip 6
Intermediate Events

- If catching, can act as delay mechanism
- Boundary events restrict the scope to an activity
- Can be interrupting or non-interrupting
- Signal events

Summary of Week 2

- Activities
  - Represent units of work, take time
  - Activity instances follow a state transition diagram
  - Task types determine the nature of a task
- Gateways
  - XOR – for exclusive choices (1/m) and loops
  - AND – for interleaved ordering or concurrency (m/m)
  - OR – for inclusive choices (n/m)
  - Be careful with uncontrolled flow
- Events
  - Start events, end events
  - Intermediate events might be throwing, catching
  - Boundary events might be interrupting, non-interrupting