Business Process Modeling

Version 3.0.0 – 7/10/2020

Licensing Note

BP Aspects

- Information
  - Conceptual modeling
    - UML Class diagrams
    - (Entity-Relationships)
- Process flow
  - Process modeling
    - UML Activity Diagrams
    - BPMN
- Interaction
  - Functional modeling
    - Use cases

Objectives

- Describe, as precisely as possible, a process (or workflow)
- Communicate, document, analyze, validate the workflow
- Implement (execute) it
  - Only formal notations allow this step

Issues

- Formal notations
  - Executable
  - But model can be very complex for high level of detail
- Semiformal
  - Not executable
  - But can be starting point for high level analysis

Notations

- Formal
  - UML Activity Diagrams
  - BPMN
  - BPEL
- Semi formal
  - IDEF0
  - Data Flow Diagrams

Process Modeling

BPMN

BPMN

- Business Process Modeling Notation
  - Business Process Diagram (BPD)
- Business Process Management Initiative
  - http://www.bpmi.org/
- Endorsed by major players

BPMI.org

- Business Process Modeling Language (BPML)
  - Meta-language for the modeling of business processes
  - provides an abstracted execution model for business processes based on FSM
- Business Process Modeling Notation (BPMN)
  - provides a graphical notation for expressing business processes
  - provides a binding between graphical elements and the constructs of BPML

BPMN

Business Process Model and Notation:

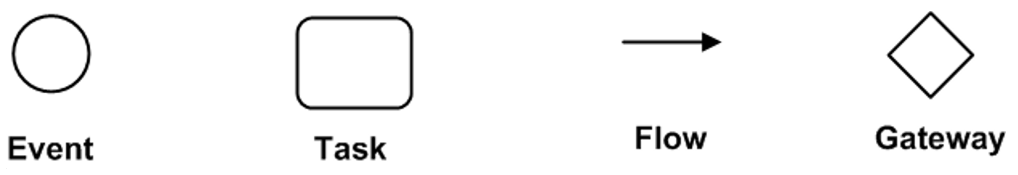
- a graphical representation for specifying business processes in a business process model

Goal

- Capture
  - Activities
  - Rules
  - Responsibilities

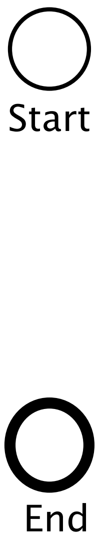
BPMN – Elements

- Four basic element categories
  - Events
  - Activities/Tasks
  - Connecting objects/Flow
  - Gateways



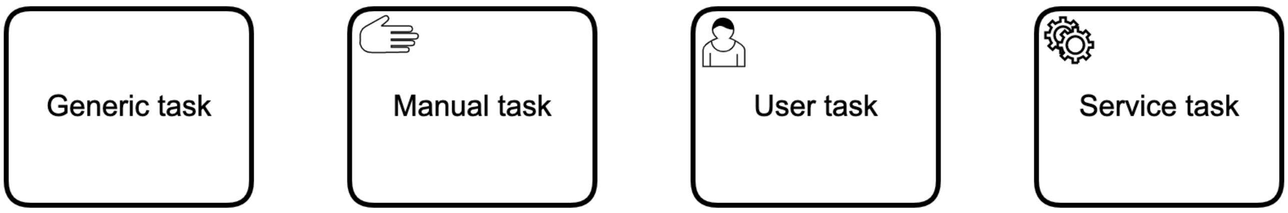
Terminal events

- Start event
  - Represents the starting point of the process execution
  - Creates a new token
- End event
  - Indicate that the processing has completed
  - Destroys all tokens



Activities

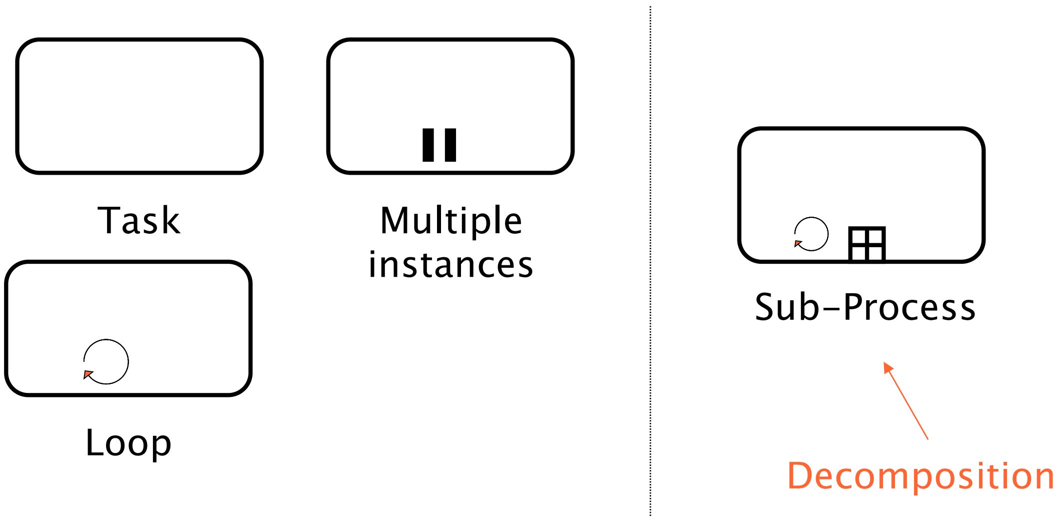
Task that are performed in the process by humans, by automation, or by subprocesses.



Activities

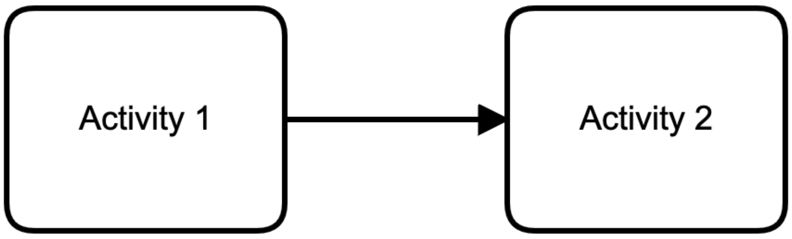
- Manual task
  - The user performs some activity outside the IS but eventually a track is recorded inside the IS
- User task
  - The user interacts with the system, by either getting or entering information
- Services task
  - The IS performs some operation without the support of the user

Activities



Sequence Flow

Indicates the order of execution of the activities

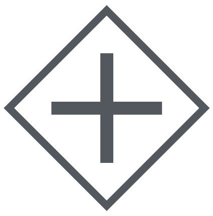


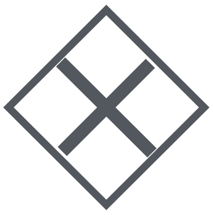
Connecting Objects

- Sequence Flow →
  - Shows the order that activities will be performed in a Process
- Message Flow ○┈┈┈▷
  - Shows the flow of messages between two separateProcess Participants (e.g. two Pools)
- Association ┈┈┈> ┈┈┈
  - Associates data, text, and other Artifacts with FlowObjects

Gateways

Route the flow of execution


Parallel (AND)


Exclusive (XOR)

Execution Semantics

A process instance can be created any time it is required

A token is created every time a process is activated

The token marks the current phase of the process instance

The token brings information specific for the corresponding instance of the process

Execution Semantics

- A token flows through the diagram
- The token is created on the start event
- The token complies with the process rules
- The token is eventually destroyed at end event

Execution Semantics

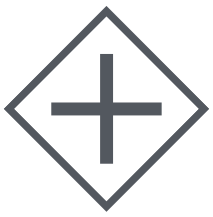
- When a token arrives at an action
  - The action is enabled: can be performed
    - The information systems informs the intended user she can start the action
  - No time is defined for starting the activity
    - It starts when the user wishes
  - No duration is defined for the activity
    - It takes as much time as the user needs
- The token can leave the action as soon as the activity is completed.

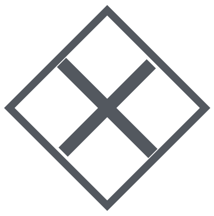
Execution semantics

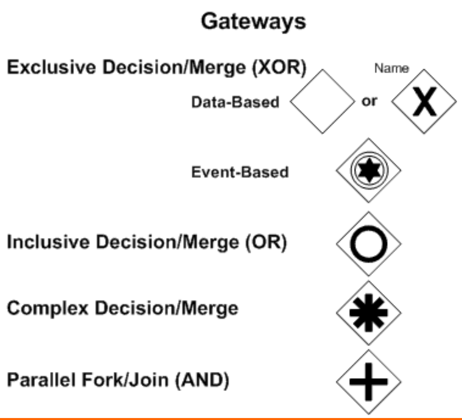
- Users taking part in the process execution have a worklist associated with them
- A new item is added to the worklist when a task assigned to the used is enabled
  - i.e. the token arrives in the activity
- The worklist of a user contains all the tasks the user is expected to perform
  - It includes tasks from all the projects she is involved in

Gateways

- Convegence/divergence point for the sequence flow


Parallel (AND)


Exclusive (XOR)



Basic patterns

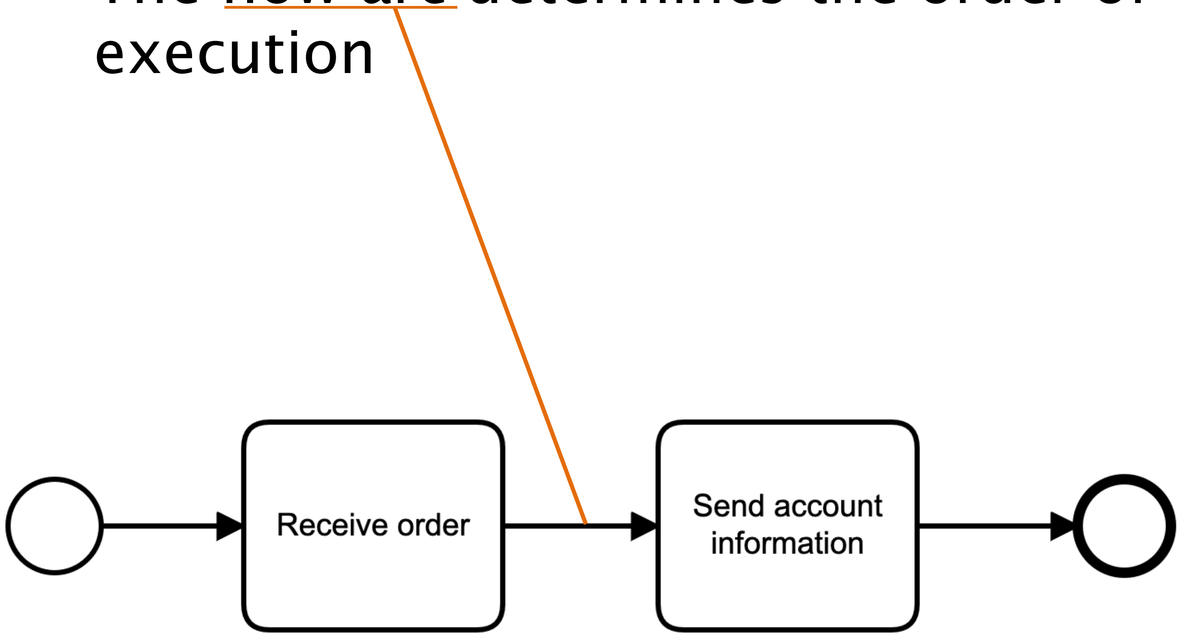
- Sequence
- Parallel split
- Synchronization
- Exclusive choice
- Merge
- Multiple choice

Sequence

- An activity is enabled after the completion of a preceding activity
  - A.k.a. serialization
  - It is the essential building block
  - Can be used to build a sequence of consecutive steps that take place, in turn, one after the other

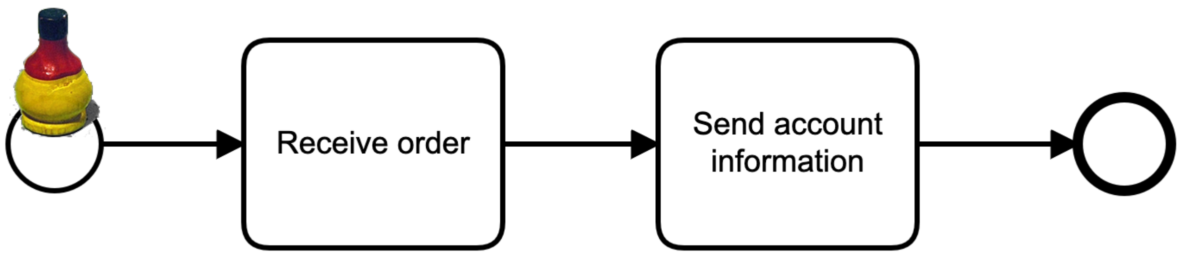
Sequence

- The flow arc determines the order of execution



Sequence – Semantics

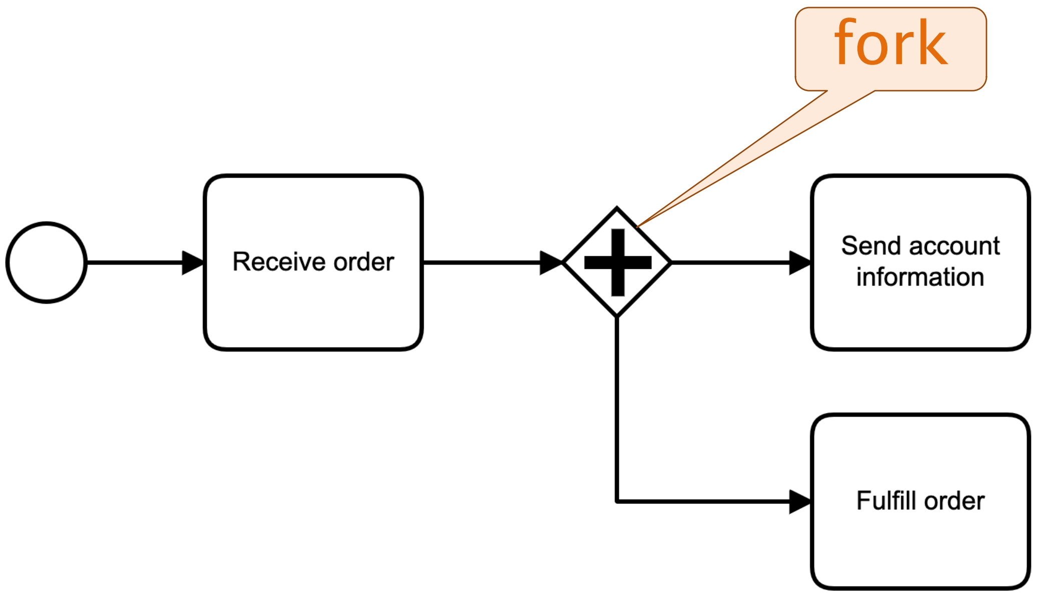
- The token flows through the diagram
- Following the arcs
- Stopping at actions
  - Performing actions



Parallel split

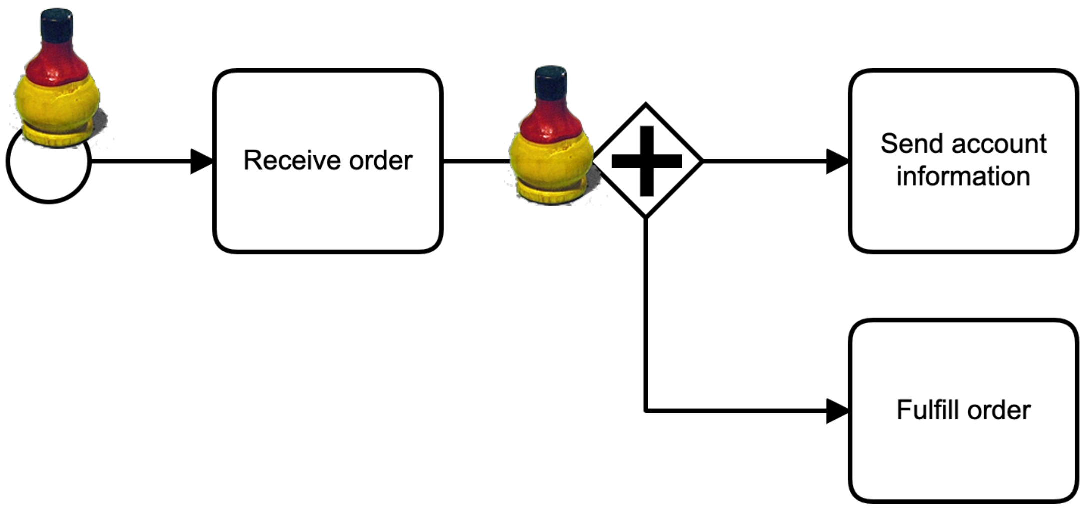
- From a certain point on a thread diverges into several parallel threads that can be executed concurrently
    - A.k.a. fork, AND-split
- Represents both
    - Actions taking place at the same time (concurrently)
    - Actions performed in no specific order
        - Possibly even serialized

Parallel split



Parallel split – Semantics

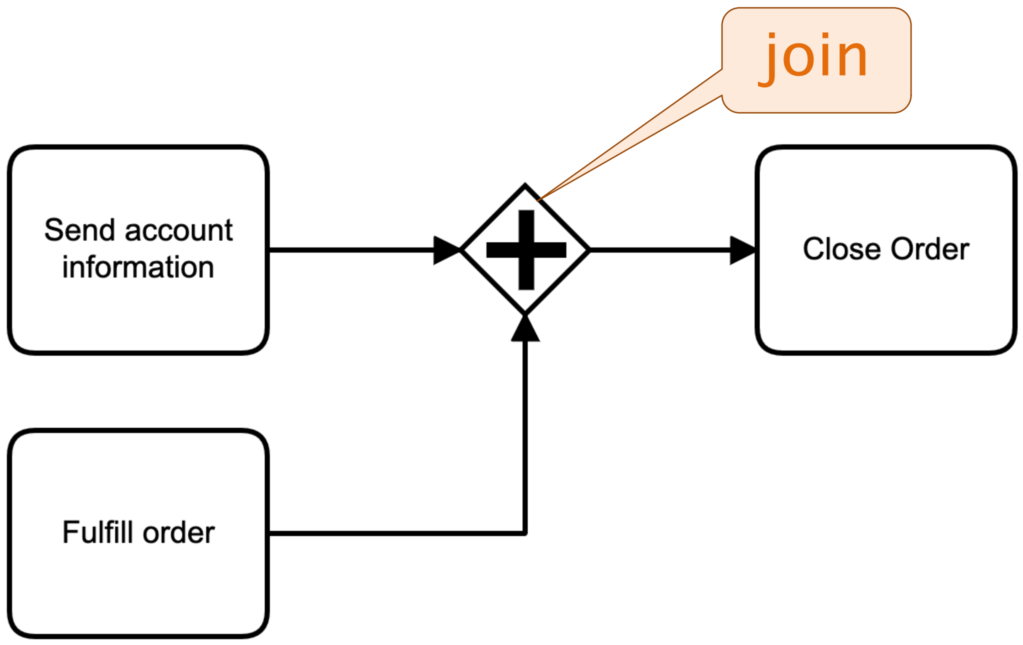
- When the token reaches the fork it is cloned as many times as the outgoing arcs



Synchronization

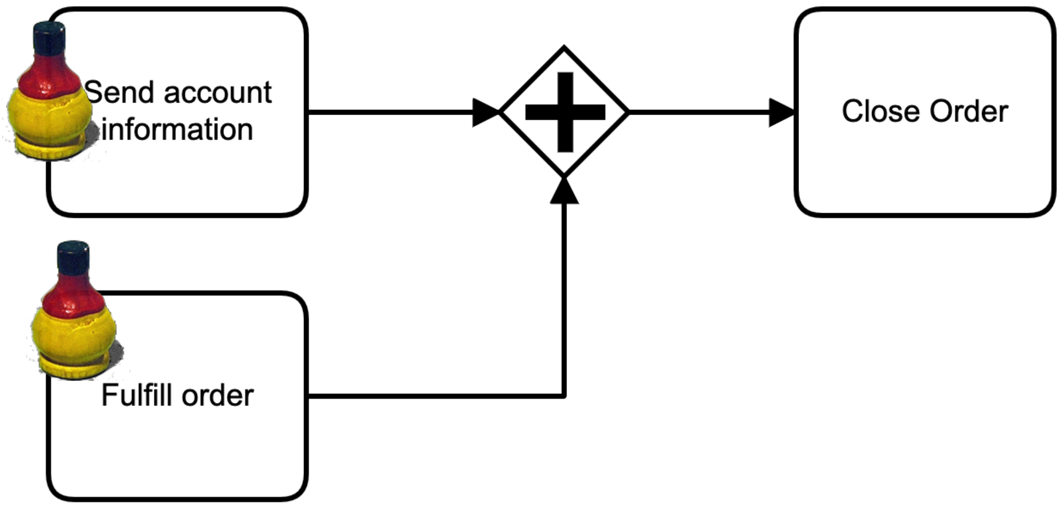
- Define a synchronization point or rendezvous
  - A.k.a. join
  - After a group of actions have been executed in parallel or independently
- Before proceeding with further activities all the previous ones must be completed

Synchronization



Synchronization– Semantics

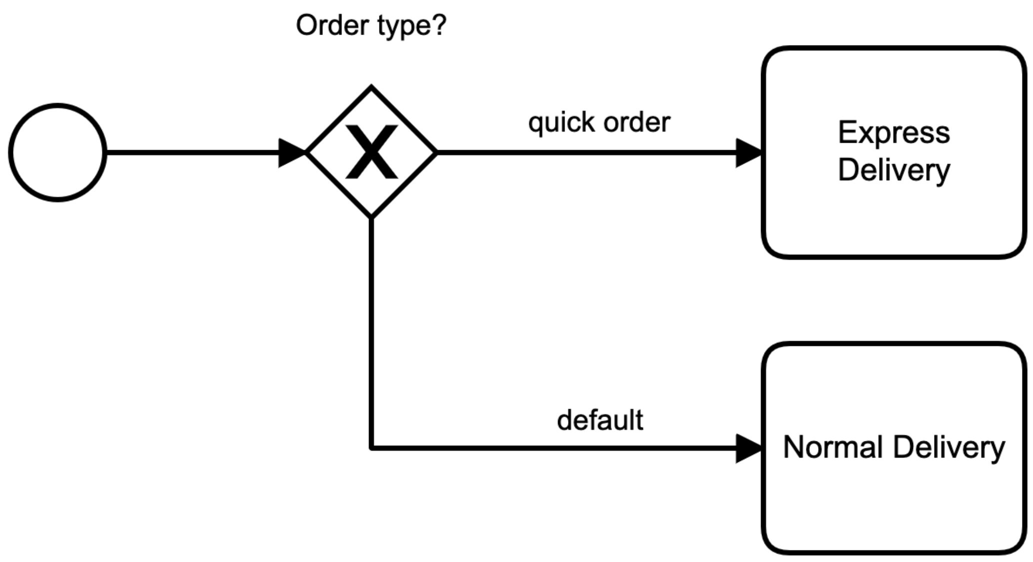
- When one token per incoming arc has reached the join, they are merged into a single token



Exclusive choice

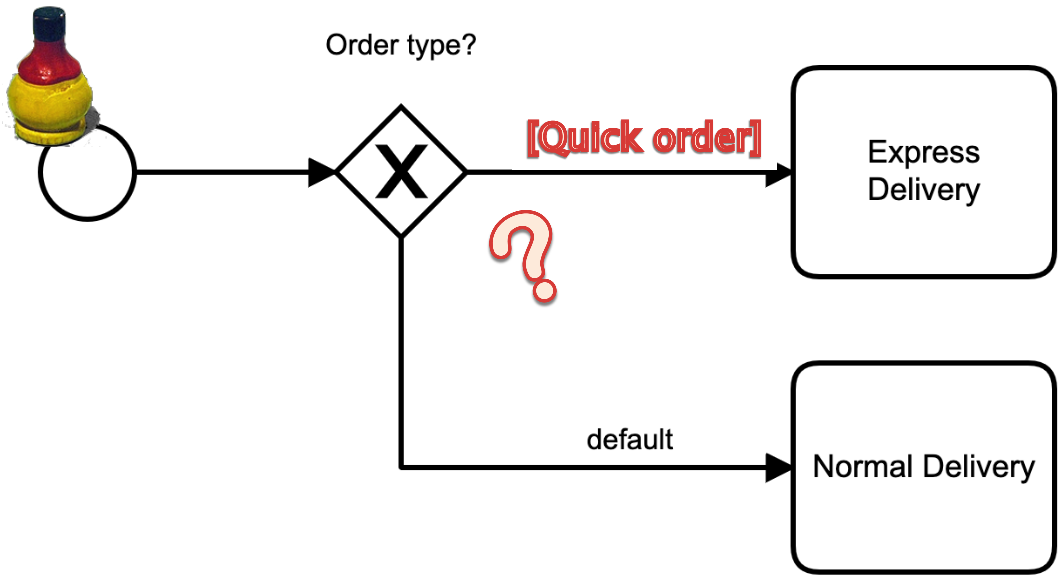
- A diversion of the thread into several alternative paths
  - Exactly one alternative is picked and followed during execution
  - A.k.a. conditional routing, decision
- Each path is characterized by a guard
  - Represents a condition that, when true, enable the execution of the corresponding path

Exclusive choice



Exclusive choice - Semantics

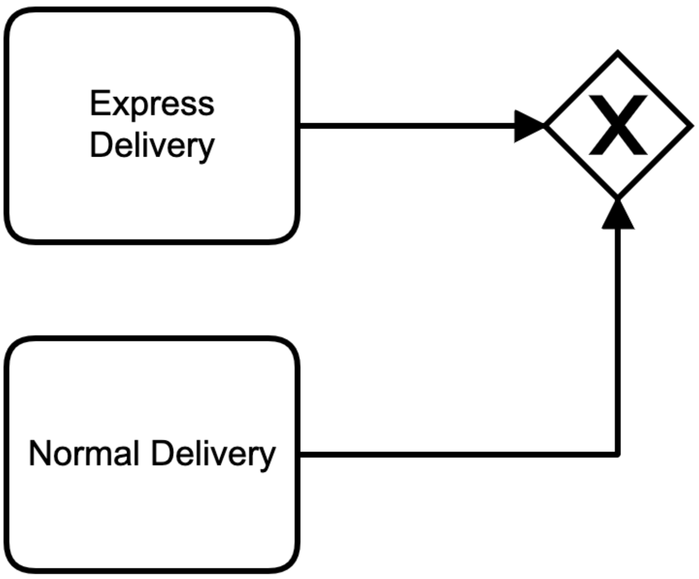
The token getting to the decision arc whose condition is evaluated as true



Merge

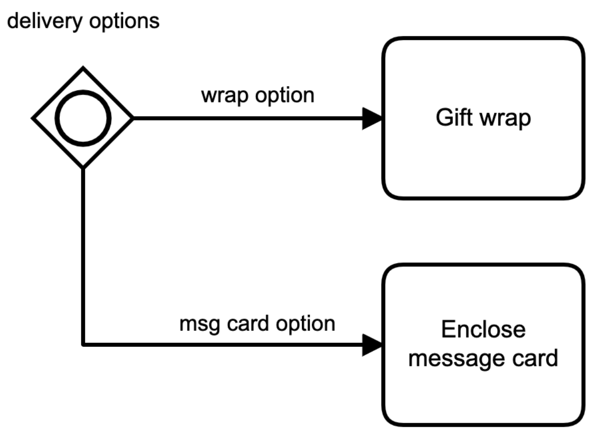
- The convergence of two or more threads into a single one
  - Any incoming thread activates the outgoing path
- Beware: no synchronization is performed

Merge

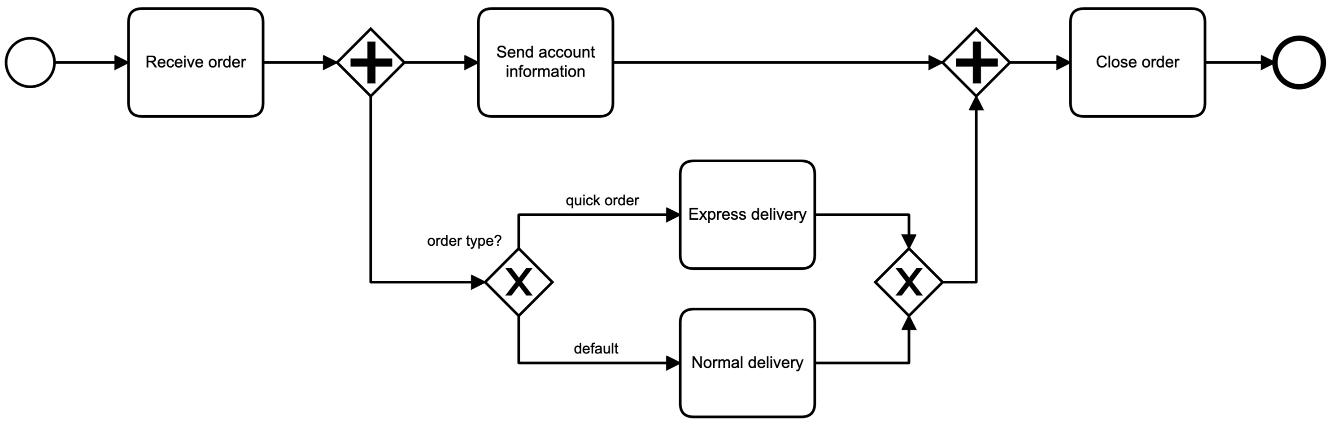


Multiple choice

- All paths with a true condition guard are followed
  - If no path is chosen, there is an abnormal stop to the flow



Example

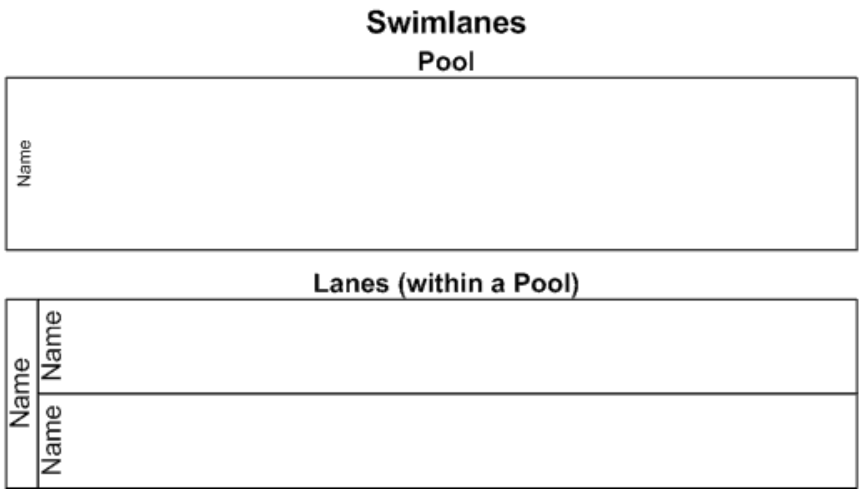


Pools and Lanes

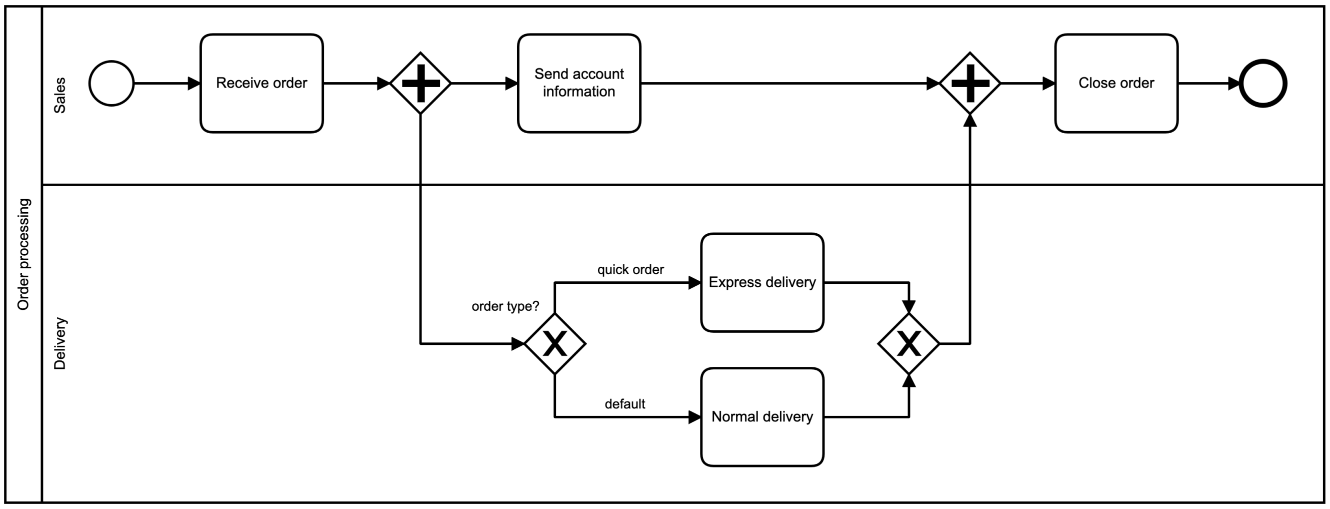
- A Pool represents a Participant (business entity) in a Process
- A Lane is sub-partition of a Pool
- Example
  - Customer
  - Enterprise
    - Manufacturing
    - Accounting
- Sequence Flow cannot cross the boundaries of a pool

Pools and Lanes

- A Pool represents a Participant (business entity) in a Process
- A Lane is sub-partition of a Pool



Pool and Lanes example

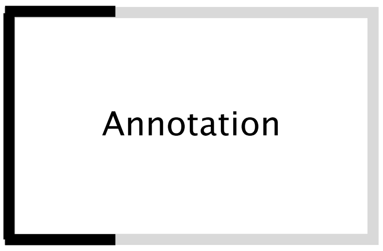


Readability Elements

Readability elements include:
- Text Annotations
- Links

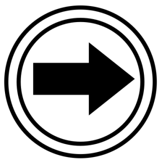
Text Annotation

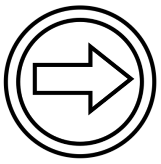
Allow the user to attach notes to a model with explanations for clarity.



Links

Allow the user to cut a process that has become too long to read easily, and simply continue the process on another line.


Throw Link


Catch Link

Structured processes

- Each action has exactly one input flow and one output flow
- Fork and Join must be coupled
- Decision and Merge must be coupled

Prescriptive vs. Descriptive

- Initial goal: understand the procedure currently in place
  - Descriptive
- Next goal: provide guidance for defining IS-supported procedures
  - Prescriptive
- Advice: avoid unnecessary constraints

BPMN

ADVANCED PATTERNS

Complex structures

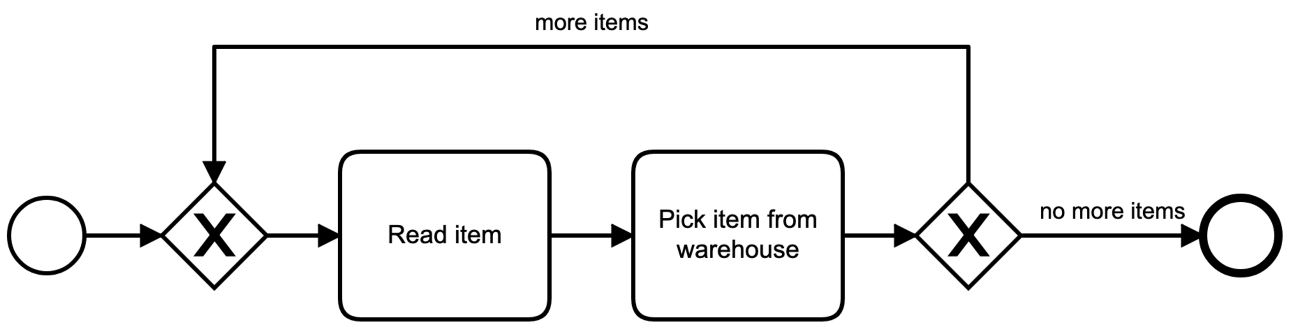
- Cycles / loops
- Arbitrary cycles
- Implicit termination
- Complex activities
- Multiple choice

Structured Loop

- One or more activities are repeated until a specific condition become true
- Realized by means of decision and merge nodes
  - First a merge node
  - Then a condition

Loop – Repeat

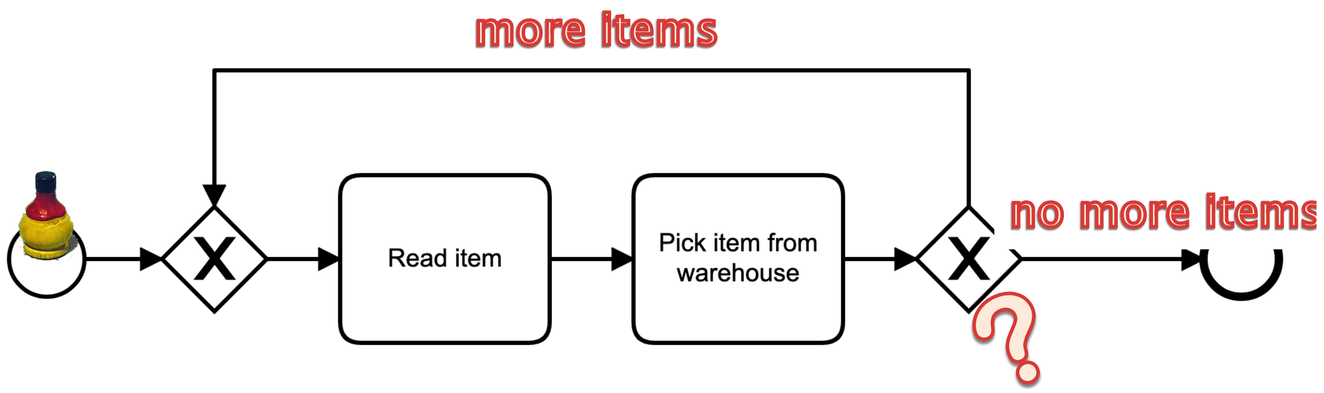
```
do {
   read_item();
   pick_item();
} while( more_items );
```



SOftEng http://softeng.polito.it

Loop – Semantics

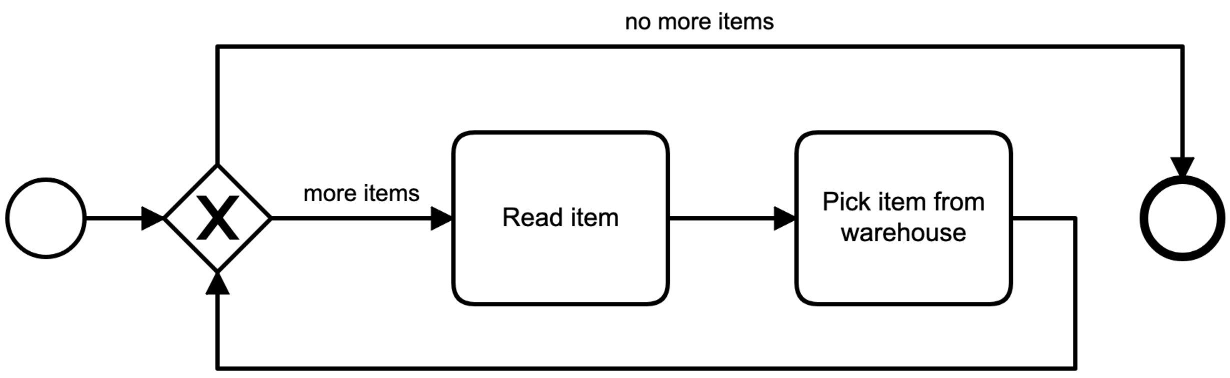
```
do {
   read_item();
   pick_item();
} while( more_items );
```



SOftEng http://softeng.polito.it

Loop – While

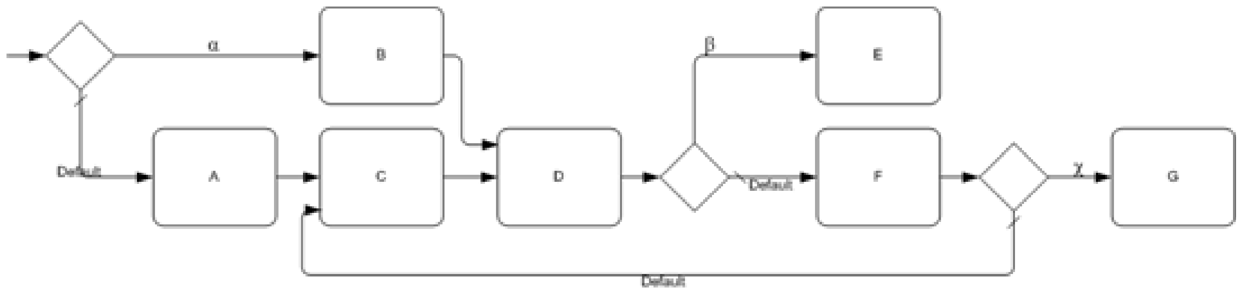
```
while( more_items ){
  read_item();
  pick_item();
}
```



Arbitrary cycles

- Loop that is unstructured or not block structured.
- That is, the looping segment of the process may allow more than one entry or exit point.
- Important for the visualization of valid, but complex, looping situations in a single diagram

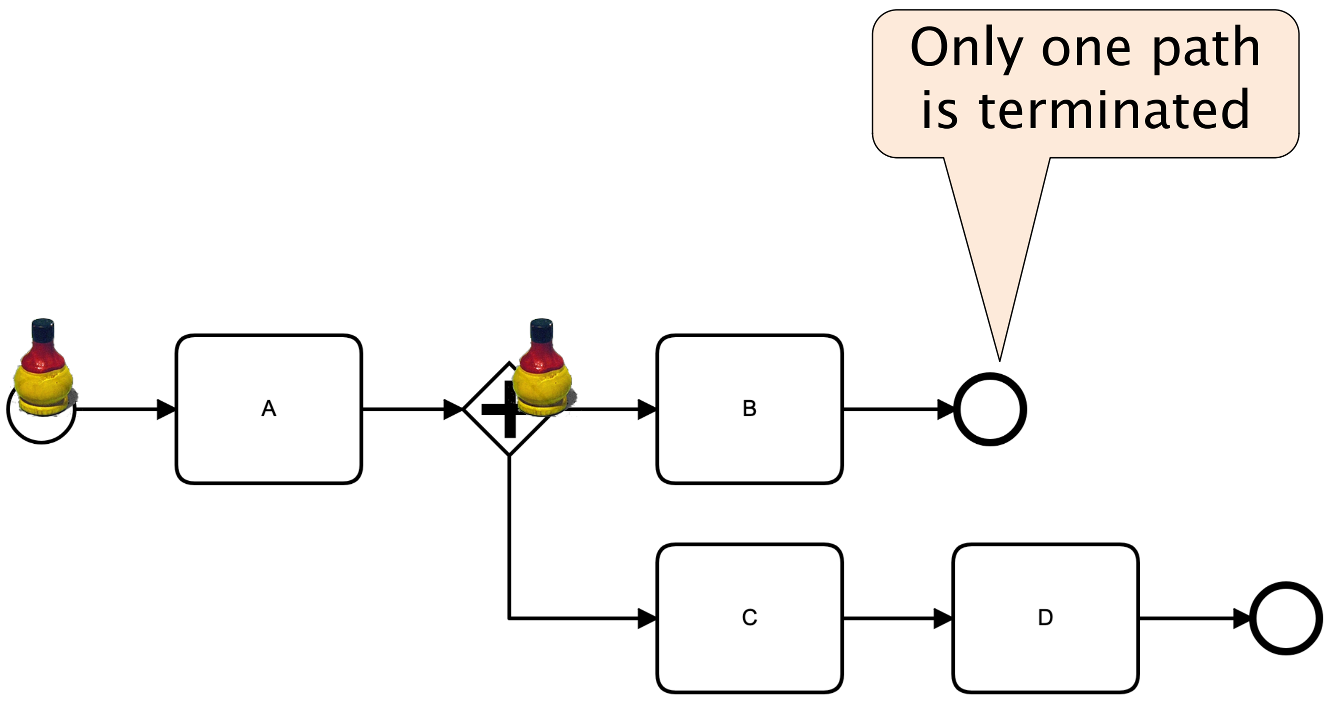
Arbitrary cycles



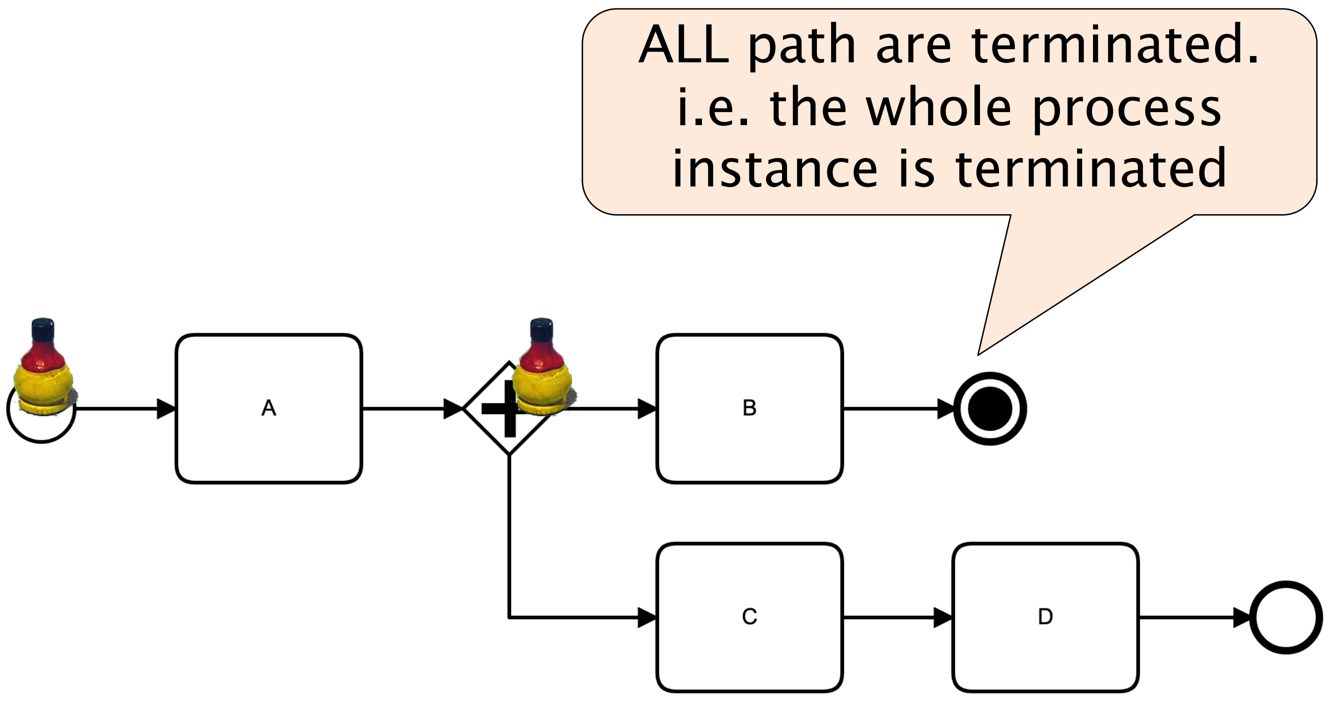
Implicit termination

- A specific path of a process can be concluded without other parallel paths be required to end as well.
- The normal case require the whole process to end when any end node is reached.

Implicit termination – semantics



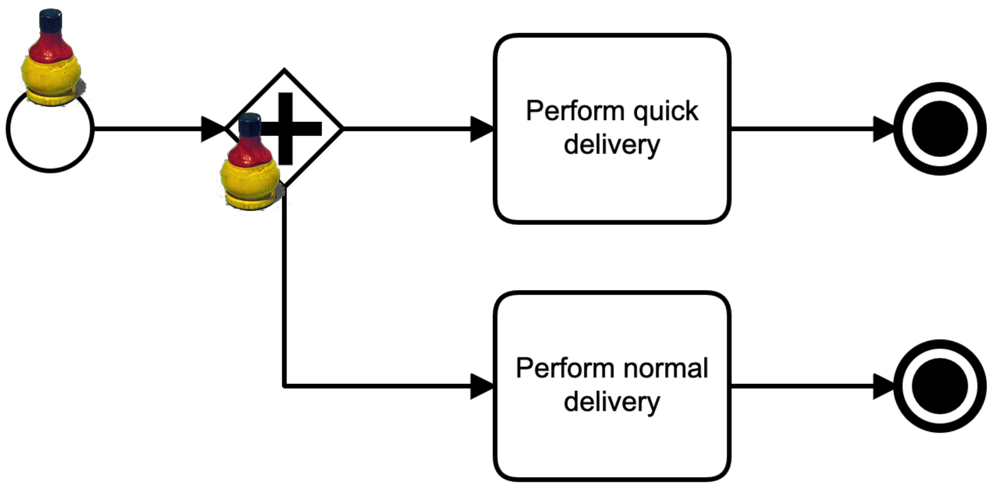
Explicit termination – semantics


ALL path are terminated. i.e. the whole process instance is terminated
A
B
C
D

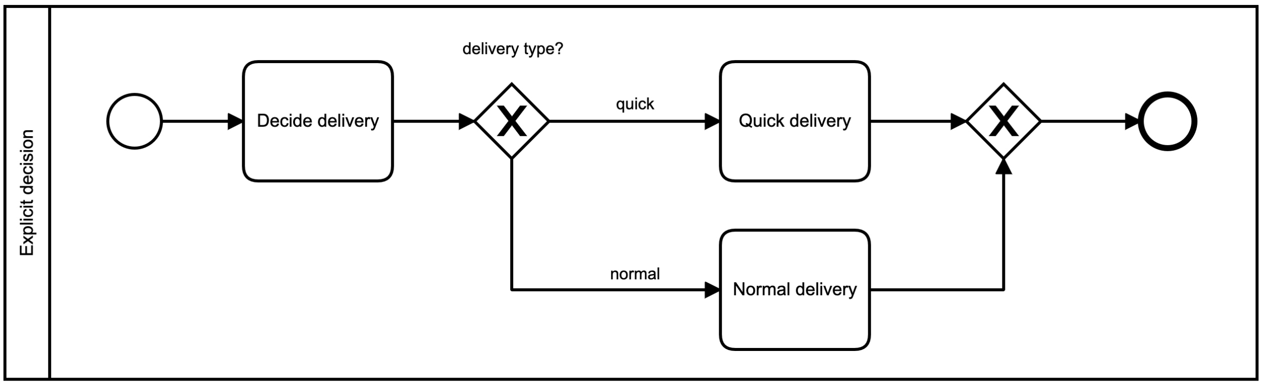
Implicit decision

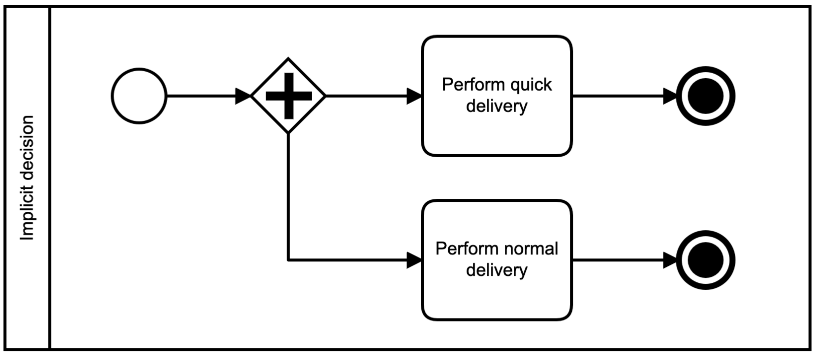
- Convergence of two or more branches such that the first activation of an incoming branch results in the subsequent activity being triggered while subsequent activations of remaining incoming branches are ignored.

Implicit decision semantics



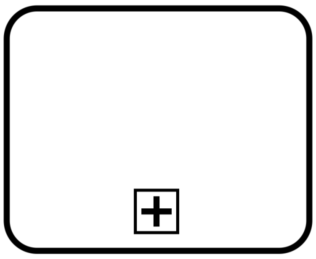
Implicit vs. explicit decision





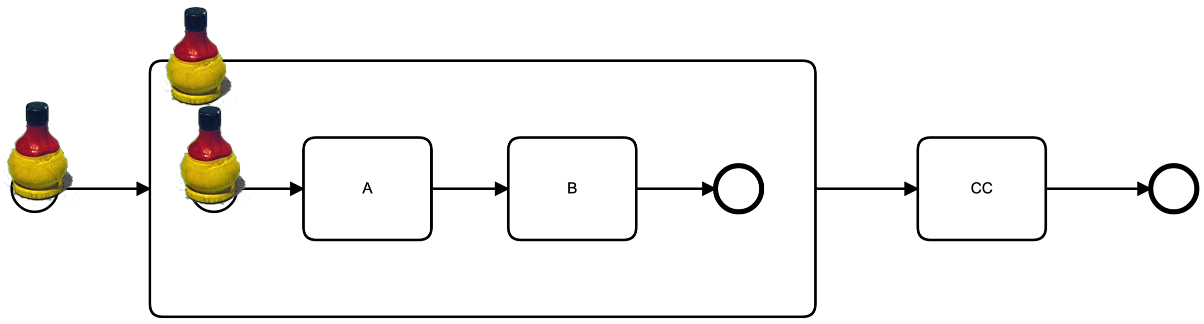
Complex activity / Subprocess

- Represent a complex (sub-)process in a single action
  - Call behavior
- The contents of the complex action can be represented in an additional diagram

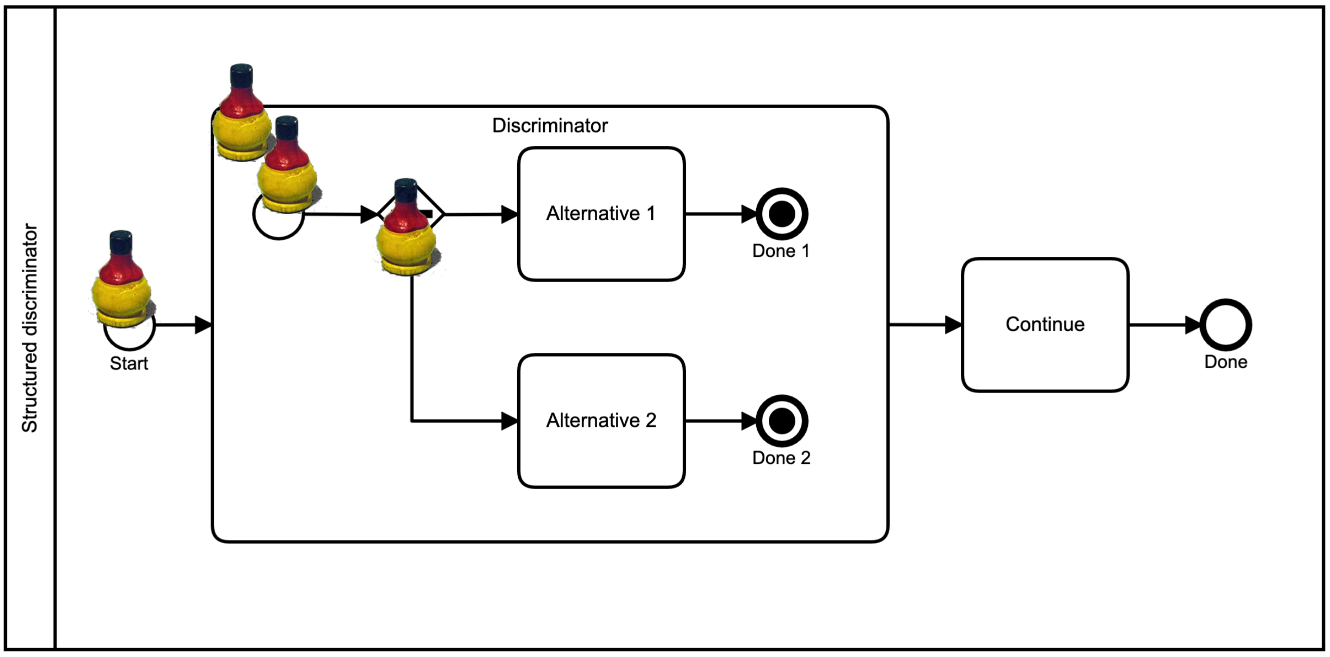


Expanded sub-process

- An expanded process can be embedded
  - Useful to provide a context where activties are executed



Implicit decision in subprocess



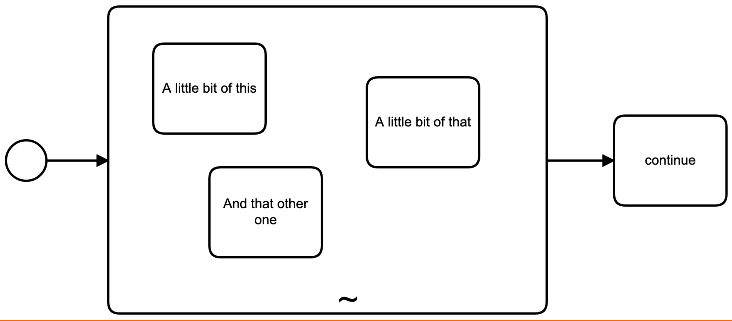
Ad-hoc

Marked with a ~

- Models of weakly structured processes

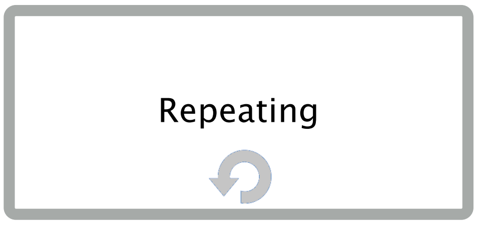
The enclosed activities can be:

- executed in any order
- executed several times
- possibly skipped



Repeating

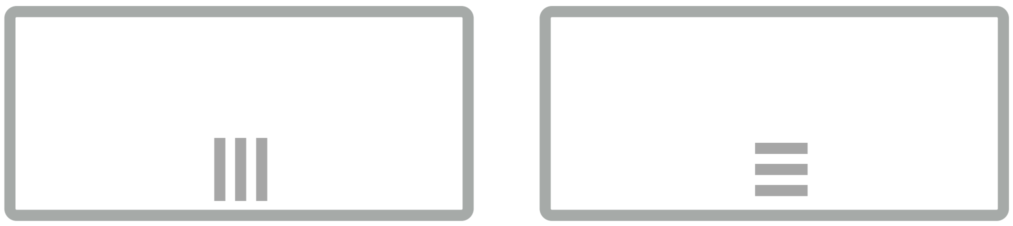
Used to repeat behaviour, such as multiple launches of the same task, or repeating the same task multiple times.



Multi instance activity

Activity to be performed many times with different data sets.

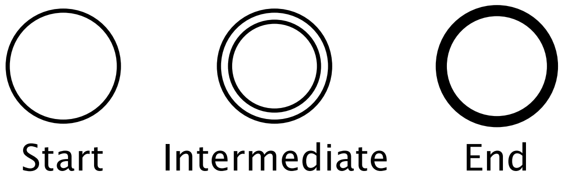
- The value of the loop condition attribute determines the number of times that the Activity is performed.
- The individual instances of a Multi-Instance Activity might occur in parallel or in sequence.

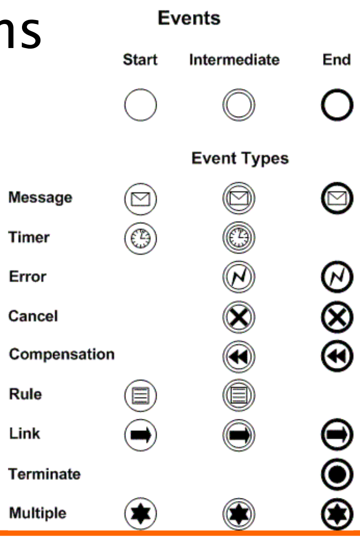


EVENTS

Events

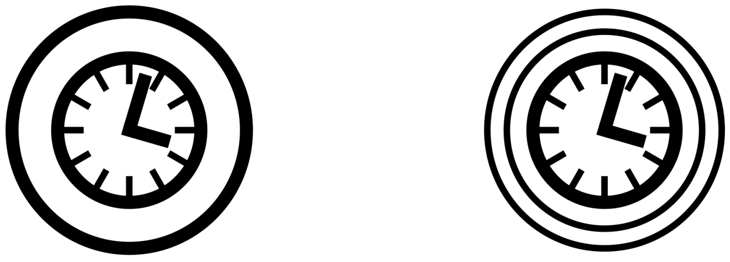
- Something that happens during a process
  - Used to start or end a process, and to manage specific actions during a workflow





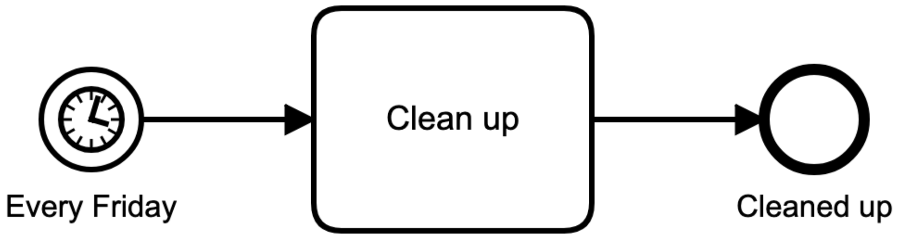
Timers

Used to launch periodic activities, or to ensure that an a happens after a specified deadline



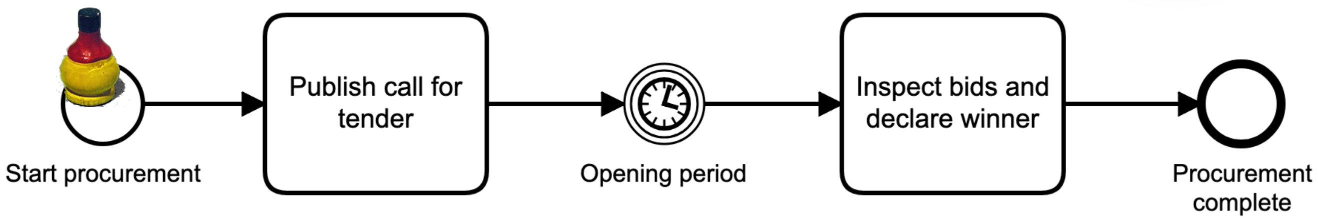
Start timer

- The timer can be specified as
  - A fixed time / date
  - An interval
  - A recurring interval



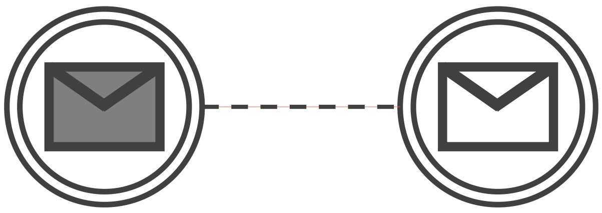
Intermediate timer semantics

- When the token arrives at the intermediate timer event a timer is started
- When it expires the token is released



Messages and Message Flow

Used to transfer actions or data from one pool or process to another and to correlate related processes.



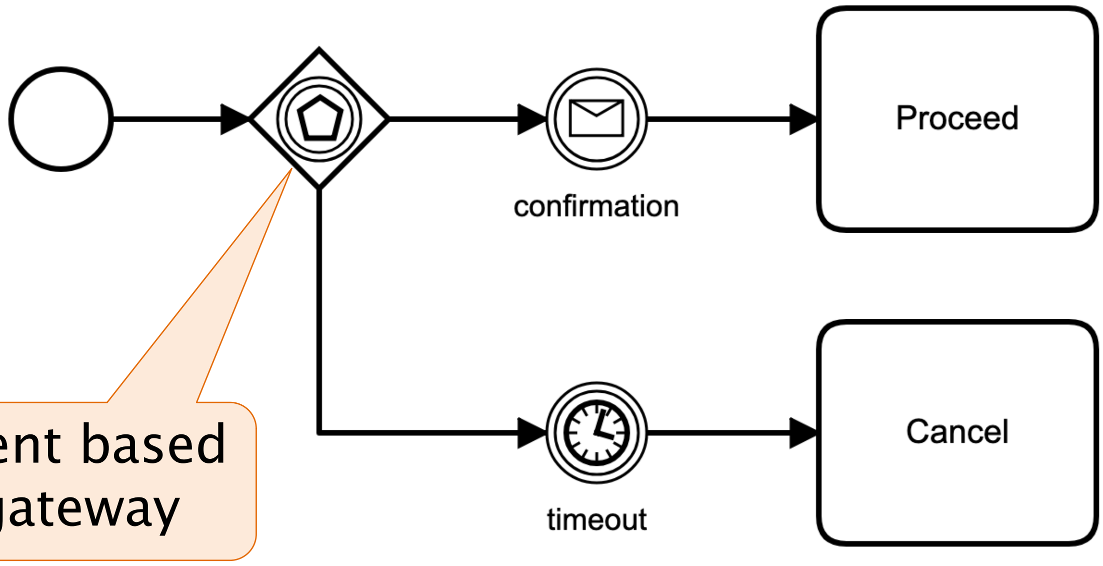
Throw Message   Catch Message

A message is a direct communication between two business participants. These participants must be in separate Pools

Deferred choice

- A divergence point in a process where one of several possible branches should be activated.
- The actual decision on which branch is activated is made by the environment and is deferred to the latest possible moment.
- Uses the event-based gateway

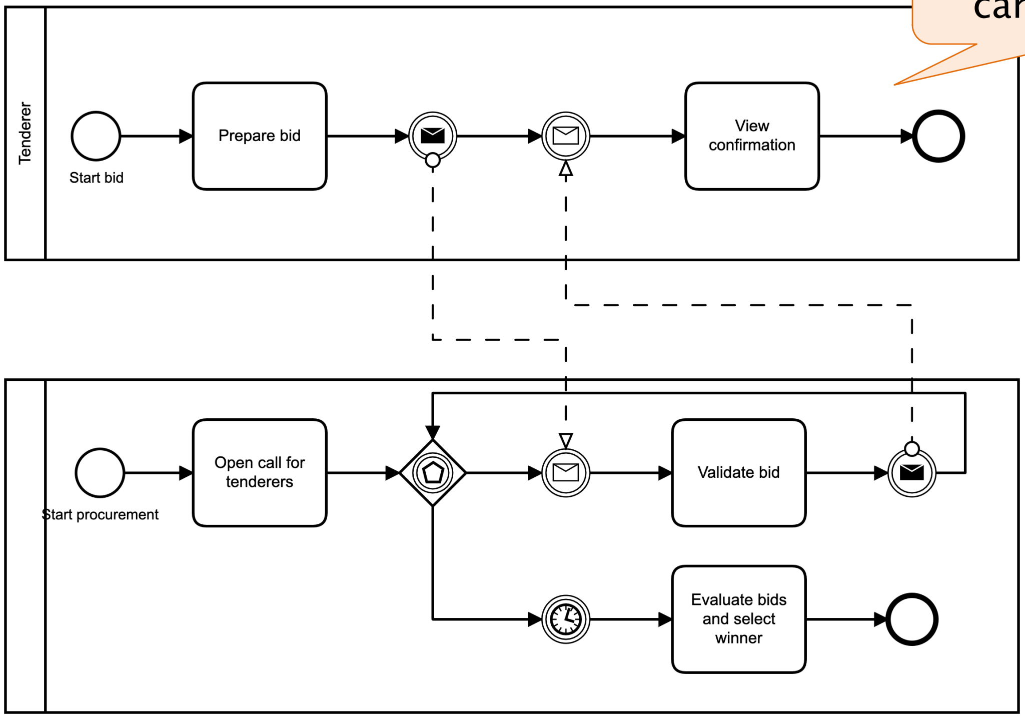
Deferred choice



Event based gateway

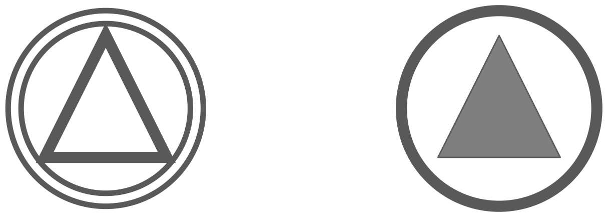
Message coordination

Multiple tenderers can participate



Signals

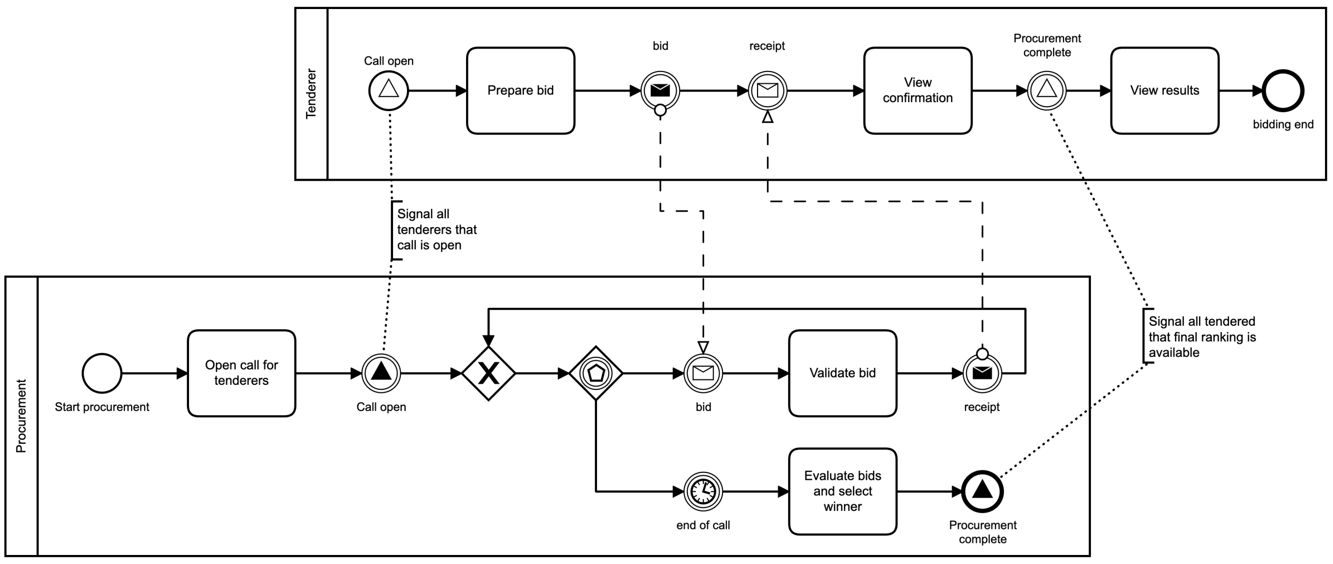
Used to send data to multiple activities simultaneously.



Throw Signal   Catch Signal

Signals are broadcast communications from a business participant or another Process. Signals have no specific target or recipient - i.e. all Processes and participants can see the signal and it is up to each of them to decide whether or not to react.

Signals



Synchronous vs. Asynchronous

Messages and signals shown above are received when a process decides to wait for them

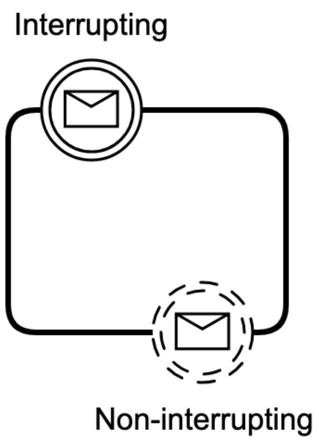
- Sender and receiver "at the same time" (synchronously) are ready for the signal or message exchange

In some cases events occur at unpredictable or unexpected times

- They must be handled asynchronously

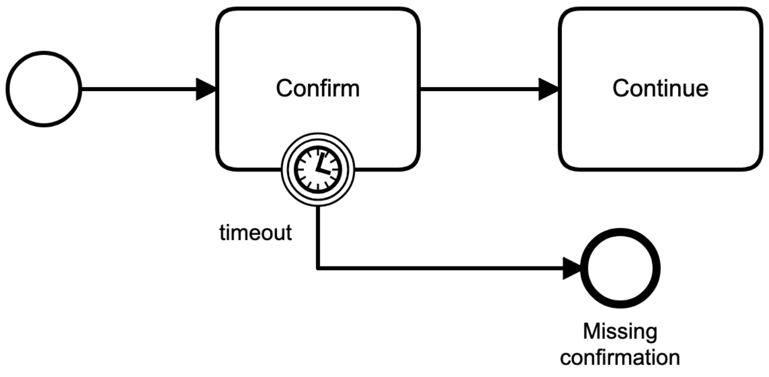
Boundary events

- Events on the boundary of an activity catch asynchronous events occurring during the execution of the activity
  - Interrupting
  - Non-interrupting



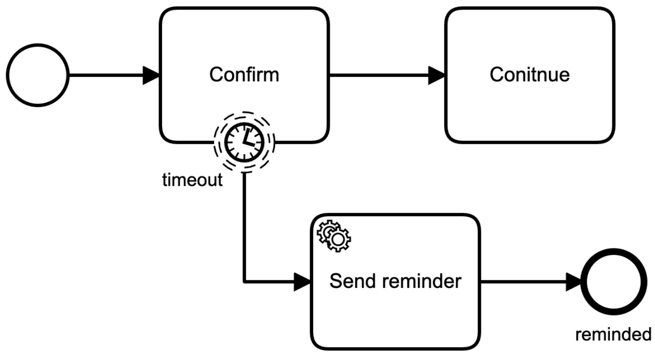
Boundary timeout

- The token triggers a timer
- Upon expiration the token is sent out via the boundary outgoing flow
  - That interrupts the activity



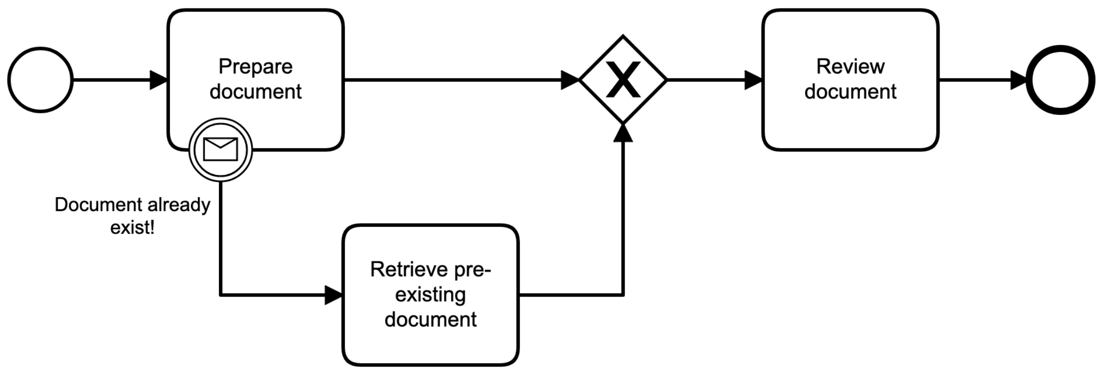
Non-interrupting timeout

- The token triggers a timer
- Upon expiration the token is cloned and sent out via the boundary outgoing flow in a parallel thread
  - Activity completion sets the timer off



Boundary message

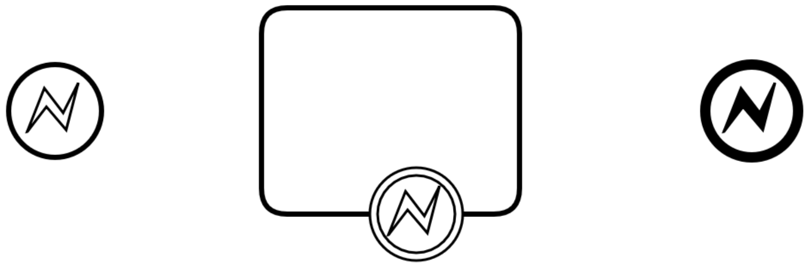
- The activity takes place as usual
- If and when a message is received the activity is stopped and the alternative path is followed



Error Events

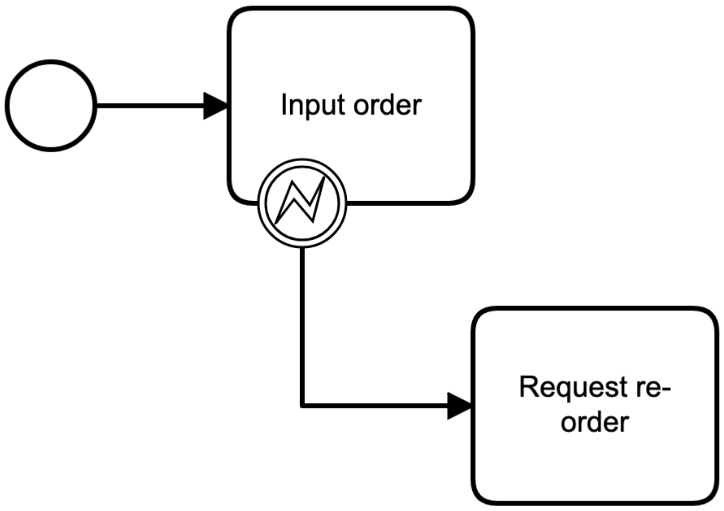
When an error occurs the task stops

- The Error Intermediate Event can only be used as boundary



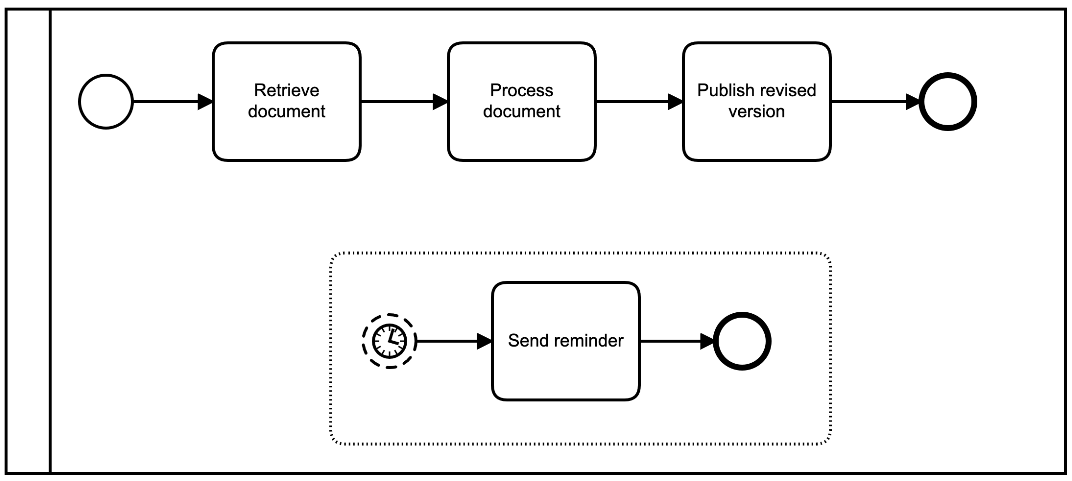
Exception management

- Used to define behaviour when the system encounters a technical error.



Event sub-process

Handle asynchronous events that occur during any activity in the enclosing context



Tools

Online

SIGNAVIO

- https://academic.signavio.com

BPMN.io

- https://bpmn.io

Applications

- Camunda Modeler
  - https://camunda.com/download/modeler/

References

- Camunda, BPMN Modeling Reference
  - https://camunda.com/bpmn/reference/
- N. Russell et al., Workflow Control-Flow Patterns - A Revised View
  - http://www.workflowpatterns.com/documentation/documents/BPM-06-22.pdf
- Workflow Patterns web site
  - http://www.workflowpatterns.com