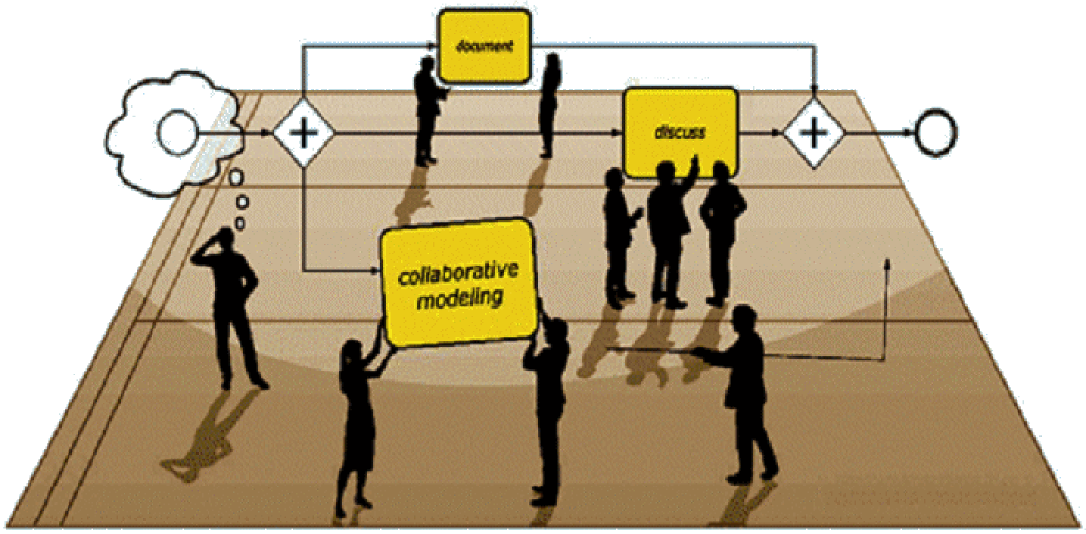
University of Pisa

MSc in Computer Engineering

Business Processes Management

"Large and complex organizations are a tangible manifestation of advanced technology, more than machinery itself." (J.K. Galbraith)

Modeling and Simulation of Business Processes using BPMN 2.0



Lectures

Mario G. Cimino, Department of Information Engineering, Center for Logistics Systems

Pisa, March-May 2013, Monday 15.30-18.30, Room: ADInform1

Course Schedule and Materials

✓The course will take about 30 hours in total:

2/3 of the time (March-April) for learning notation and tools (in parallel)
1/3 of the time (May) for working on the group project

✓Materials:

http://www.iet.unipi.it/m.cimino/bpm
user: *business*
password: *pr0cess* (note: *0* is a zero)

✓References:

OMG BPMN 2.0 Specification:
http://www.omg.org/spec/BPMN/2.0/PDF/
http://www.omg.org/spec/BPMN/2.0/examples/PDF

T. Allweyer, Introduction to the Standard for Business Process Modeling 2009, http://bpmn-introduction.com

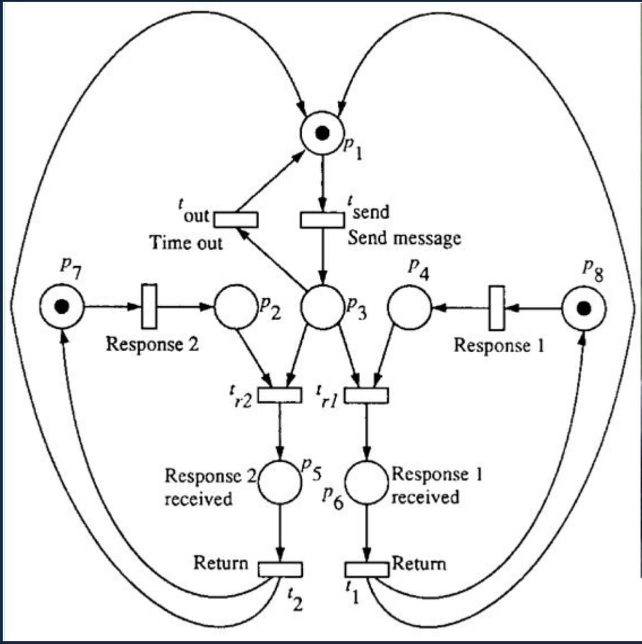
Logizian 10.x Simulacian, a business workflow design and simulation tool, http://www.visual-paradigm.com/product/lz/

Business Process Model and Notation (BPMN 2.0)

- **BPMN is…**
  - ✓ *human-readable*: a standard visual notation for modeling business processes;
  - ✓ *accessible*: easy to understand for various roles: who analyzes and defines processes, who leads the technological implementation, who is responsible for management and control;
  - ✓ *machine-readable*: a notation serializable to XML for process execution (e.g. WS-BPEL 2, SOA environments).

- **BPMN is not…**
  - ✓ a language for representing data flows and object flows, although this can be done at a certain abstraction level;
  - ✓ a notation to represent structures, functional decompositions, data models, organization strategies, business rules.

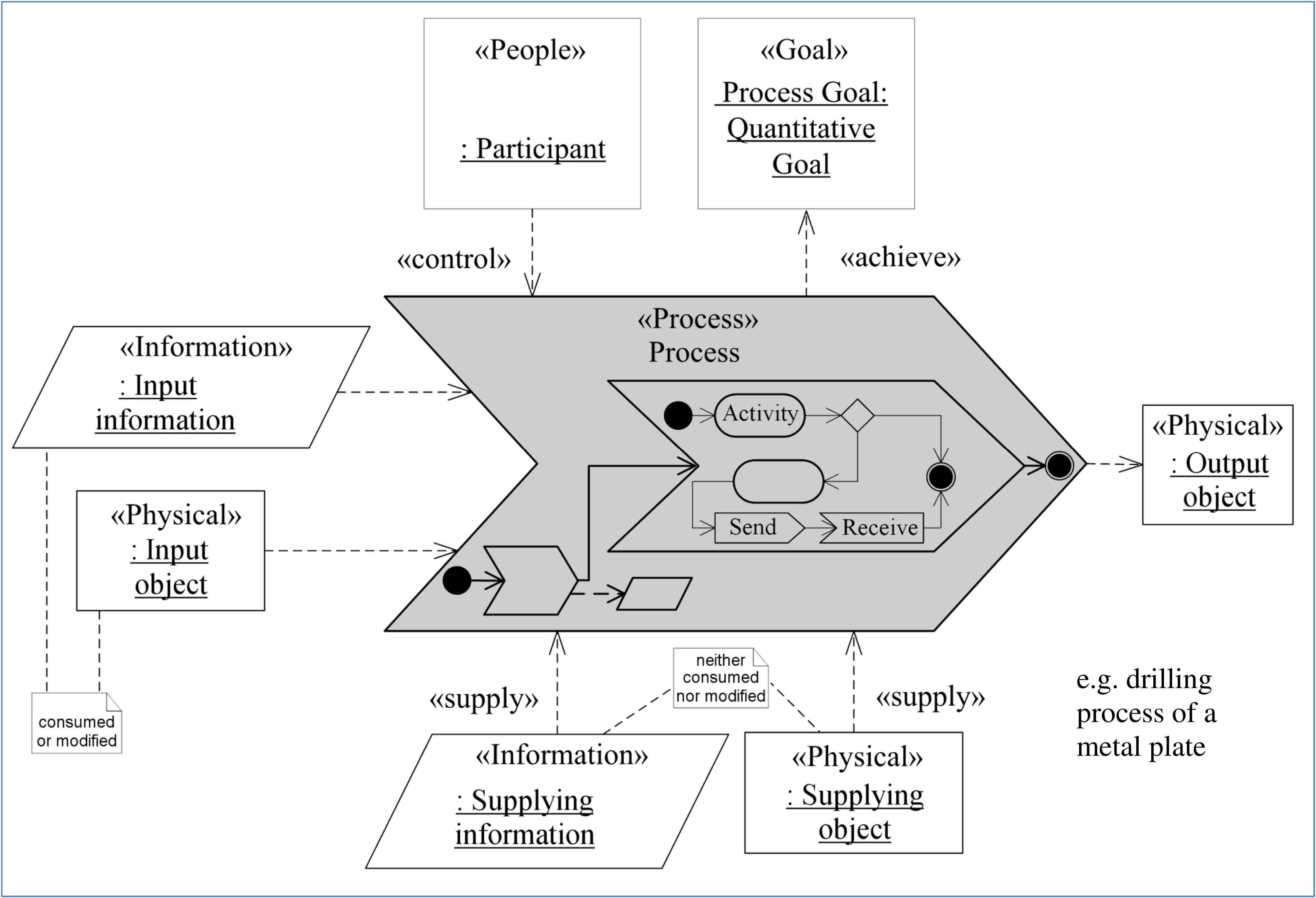
- **“Relatives” of BPMN** (languages for worlfkow-based analysis)
  - ✓ *Petri Nets* (1962): formal language to model distributed systems, usable by computer scientist and designers of specialized software. It consists of a visual representation and a corresponding mathematical notation (graphs), which allow advanced analyses such as validation, verification (e.g.. *soundness* to identify deadlock, livelock, ...)



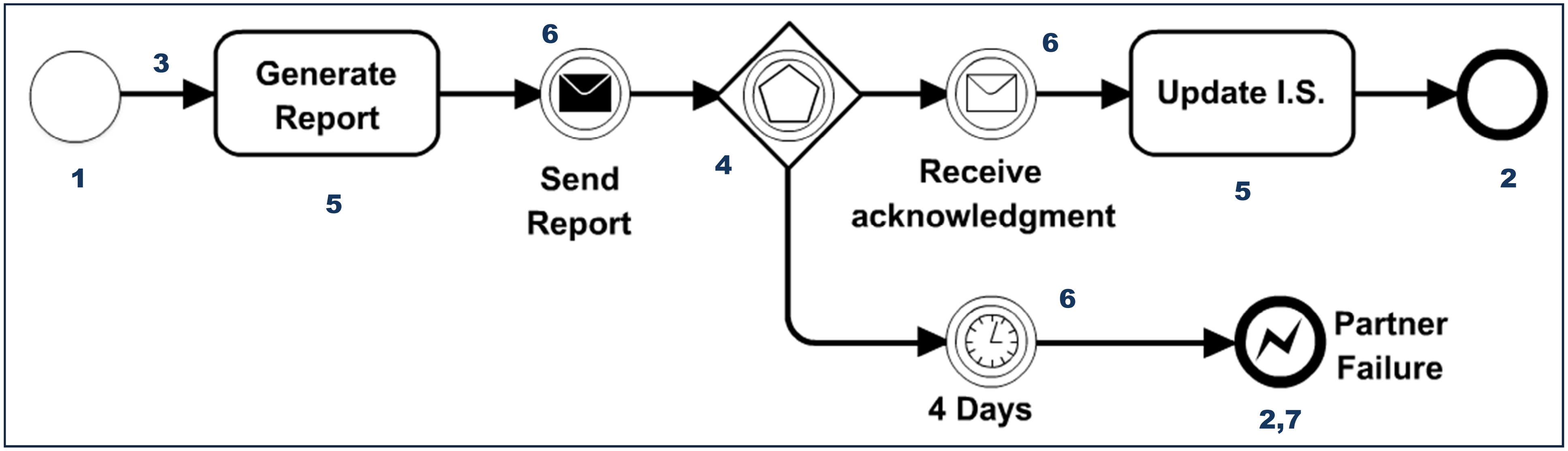
Carl Adam Petri (1926-2010)

  - ✓ *UML Activity Diagram* (OMG, 1997): language for visual modeling for the object-oriented paradigm, usable by software engineers. The extended UML of Eriksson e Penker (2000) is suitable for business process modeling, and usable also by business level (non-technical) roles.

Generic example of a process diagram, with **UML extension of** Eriksson-Penker

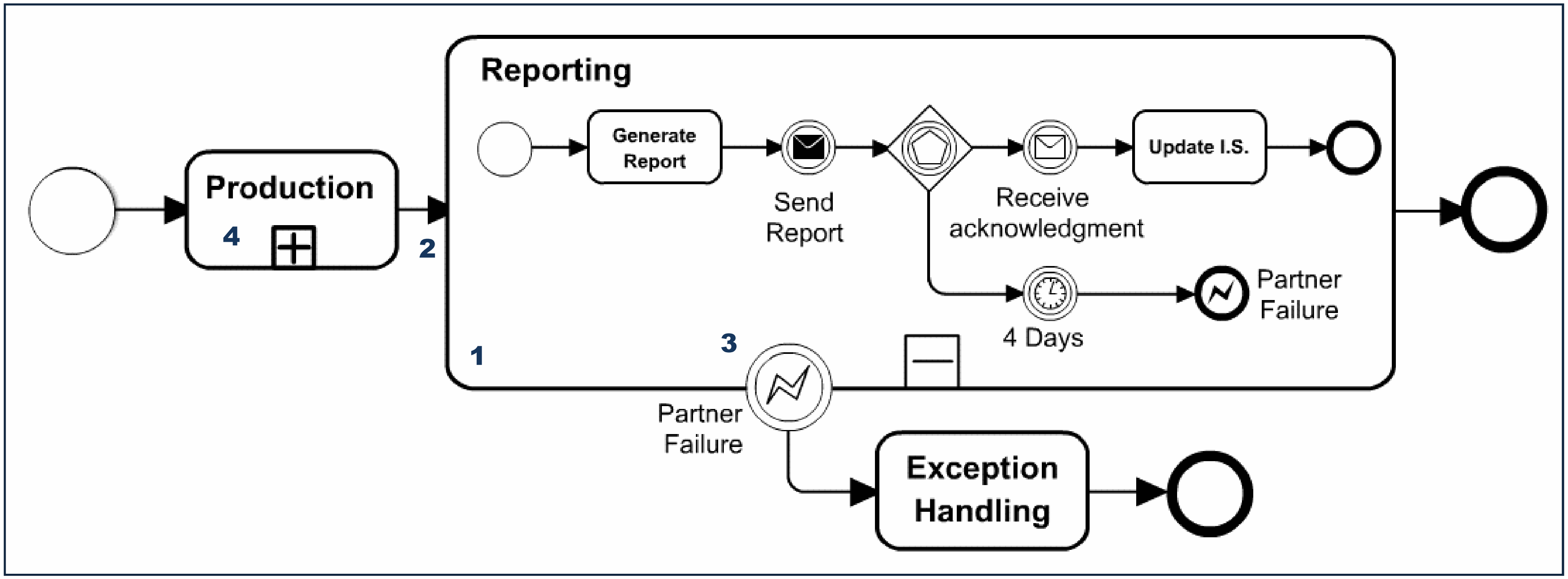


- *BPMN* (OMG, 2005) is specialized in representing the behavior of processes concerning the **control flow**, with the concept of a *token* traversing the process structure.
- A *Start Event*[1] generate a token that will be consumed by an *End Event*[2]. The path of tokens is managed by a network of *Sequence Flow*[3], *Gateway*[4], *Activity*[5] and *Intermediate Event*[6], within the process.
- *Race pattern*: there is a race between two intermediate events[6] after the event-based gateway[4], i.e., "receive acknowledgment" and "four days elapsed". When the message is not received before the 4 days cutoff date, the execution is diverted from normal execution flow in order to raise a 'Partner Failure' error (throw semantic)[7].



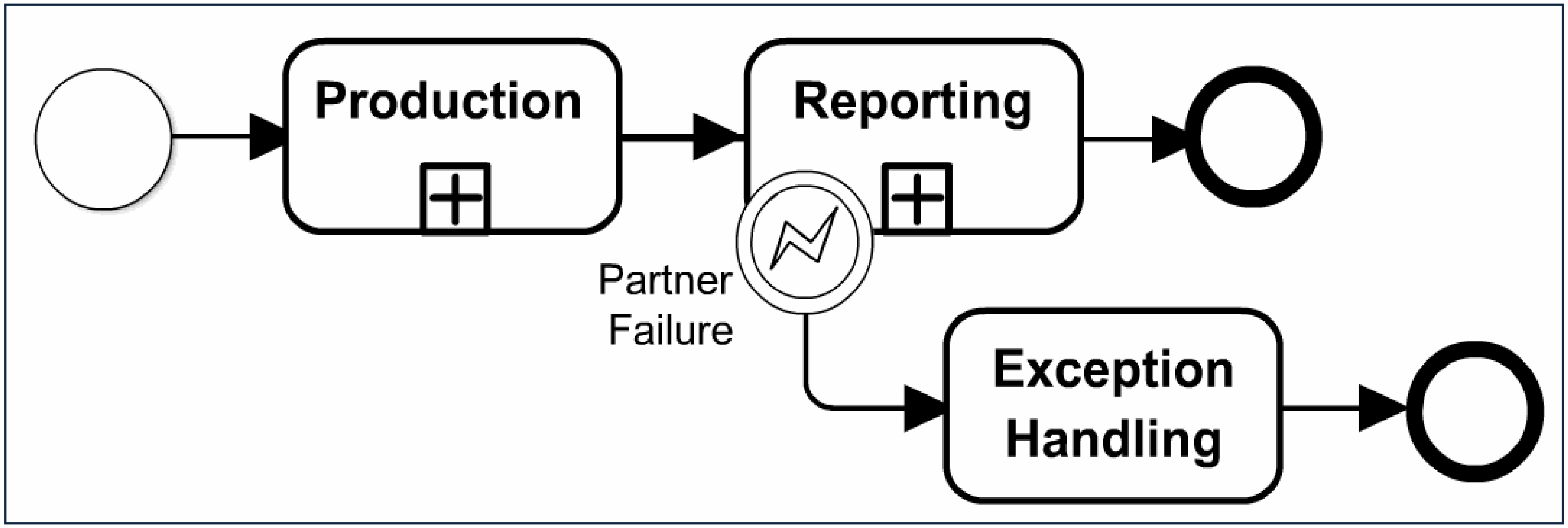
http://www.iet.unipi.it/m.cimino/bpm/res/mov01.swf

- *Interruption pattern*: the previous model is reused and embedded as the 'Reporting' sub-process (expanded notation)^1. An intermediate event ('catch' semantic) has been added to the boundary of that sub-process. The model contains also a collapsed sub-process (details are not visible), denoted by a "plus" sign^4.
- The 'Partner failure' event ('catch' semantic) gets activated when the execution points reaches the 'Reporting' sub-process^2, and gets un-activated when the 'Reporting' sub-process completes successfully (i.e., an end event is reached).



http://www.iet.unipi.it/m.cimino/bpm/res/mov02.swf

- Note how the partner failure event ('send' semantic) raised by the reporting sub-process is caught by the 'Partner failure' event[3] ('receive' semantic) and how the execution flow get diverted from that point.
- Note that if a sub-process of a diagram is expanded in the diagram, the elements inside the sub-process cannot be connected to elements outside the sub-process.
- BPMN allows the *structured modeling* of processes, i.e., views at different levels of abstraction: from the level "0" (the least amount of detail) the processes are decomposed into sub-processes, up to activities (that are atomic, the most amount of detail). For instance, in the IMB methodology the analysis stops at the level "3".

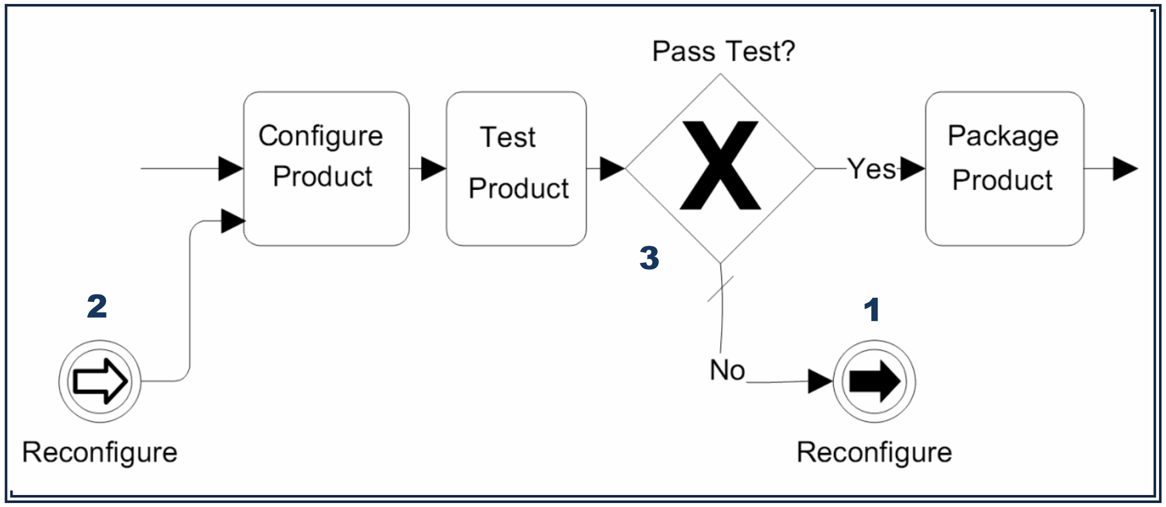


http://www.iet.unipi.it/m.cimino/bpm/res/mov03.swf

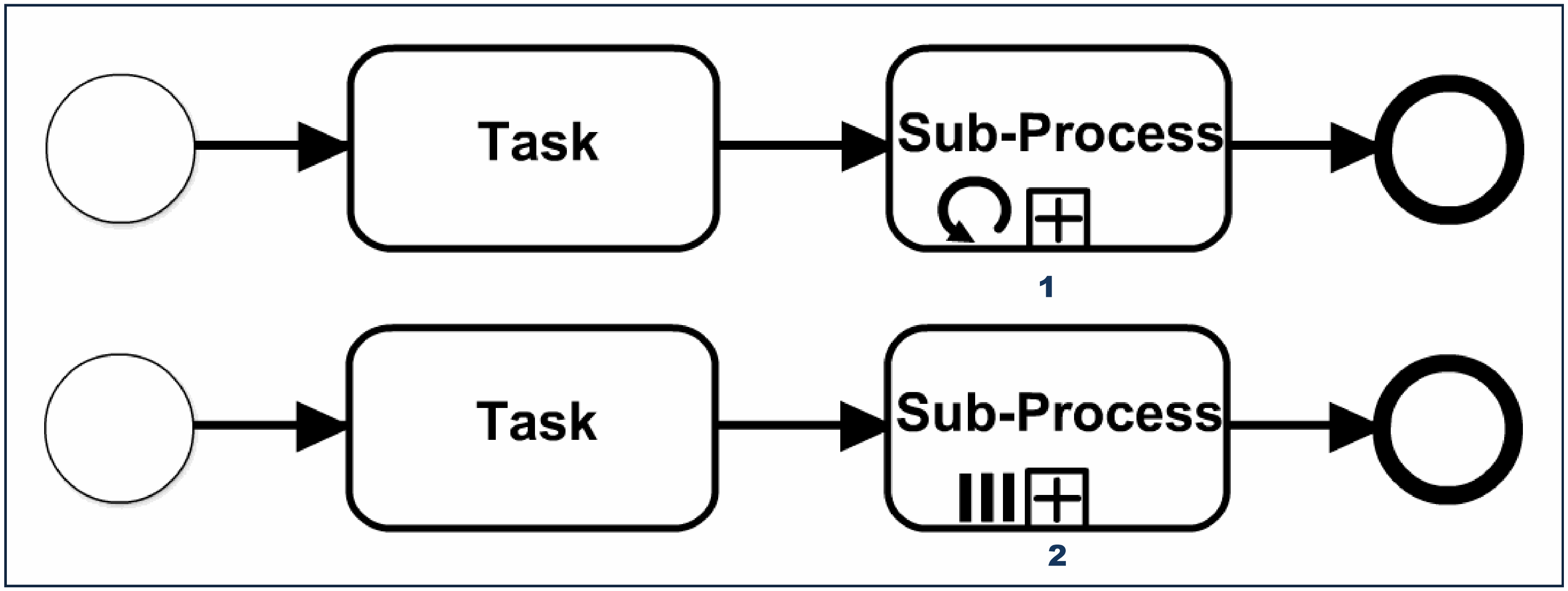
- At the level 0 process is a high-level *not executable* process, i.e., which has been modeled for the purpose of documenting process behavior at a modeler-defined level of detail. Thus, information needed for execution, such as formal condition expressions are typically not included. In contrast, an *executable* process is modeled for the purpose of being executed (e.g. a WS-BPEL process).

- BPMN allows process *segmentation*, at a given level, to create different modular segments. For instance, in the IBM methodology it is recommended that the maximum number of process activities per page should be *six*.

- The *off-page connector*, generally used for printing, is an object showing where a Sequence Flow leaves one page and then restarts on the next page. A *Link Intermediate Event* with throw[1] and catch[2] semantic can be used as an Off-Page Connector. In figure, the flow of the exclusive gateway[3] labeled with "No" leads to come back, and then to a cycle.

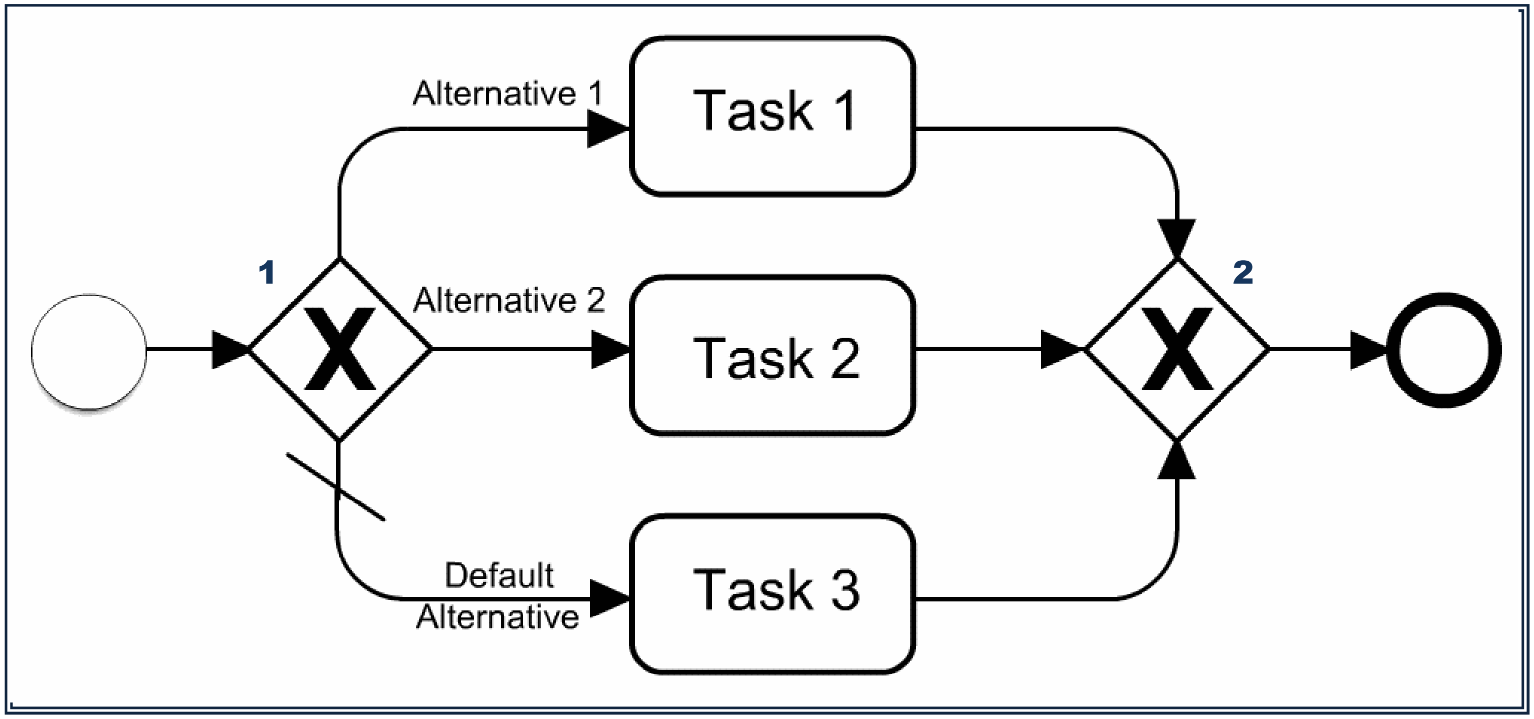


- As a result of a ‘Sequential Loop’ marker[1] on the Sub-Process, the sub-process will be instantiated several times sequentially. As a result of a Parallel Loop’ marker[2] on the Sub-Process, the sub-process will be instantiated several times in parallel.
- The number of loops to execute might be: (i) defined at design time, (ii) affected at runtime from some process data, (iii) computed at runtime.



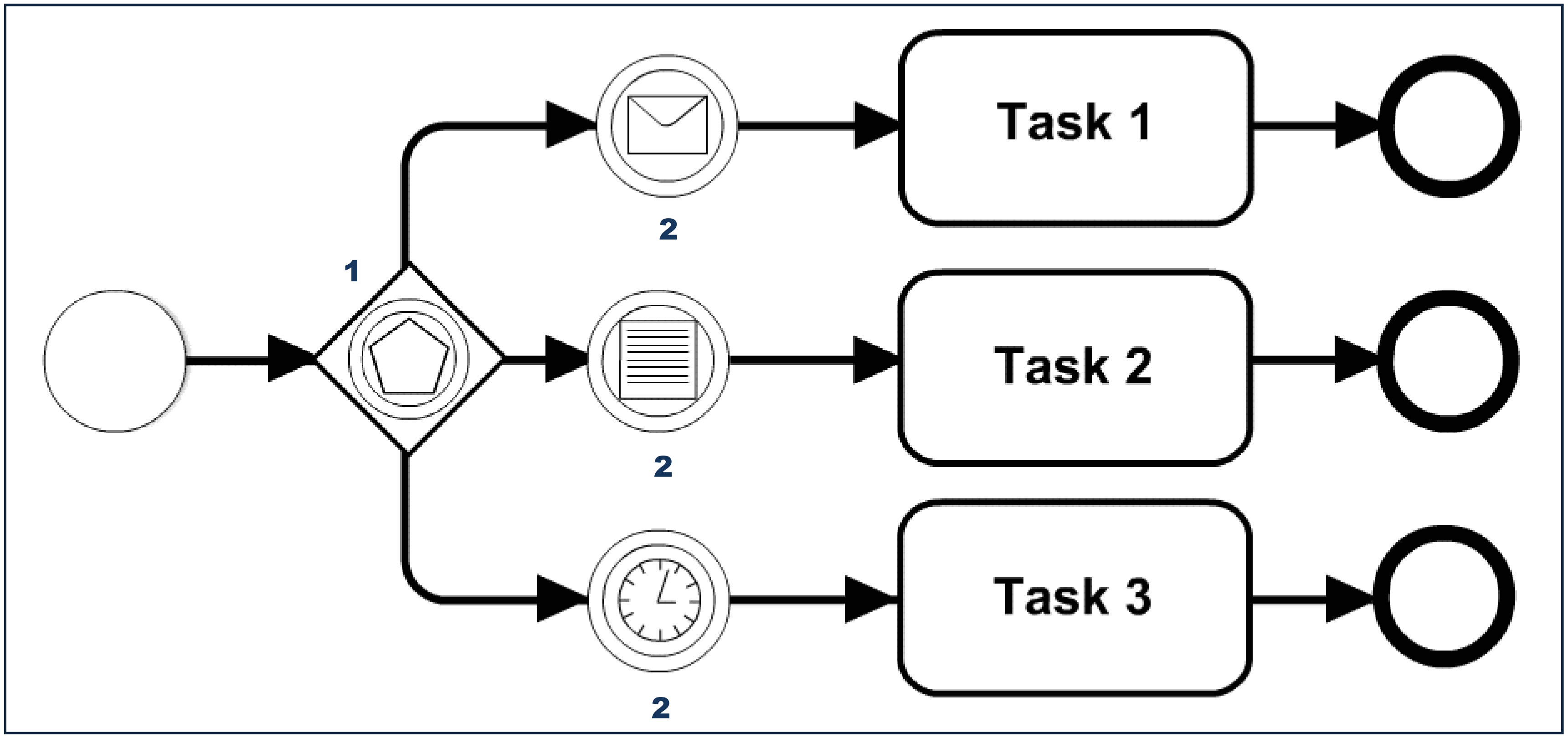
```
http://www.iet.unipi.it/m.cimino/bpm/res/mov04.swf
http://www.iet.unipi.it/m.cimino/bpm/res/mov05.swf
```

- Gateways are decisions points used to constrain the execution flow, fork an execution point into several or merge several into one. A gateway is represented by a diamond and the kind of a gateway is specified by a marker.
- An *exclusive gateway* can be used as a decision point[1] where several outgoing sequence flows are possible. Such flows are all constrained by a condition allowing only one of them to be used by a token. Such a condition will be evaluated based on the process data. The gateway can be also used as a way to merge[2] several sequence flows into one. The incoming token moves straight through the gateway and goes on.



http://www.iet.unipi.it/m.cimino/bpm/res/mov06.swf

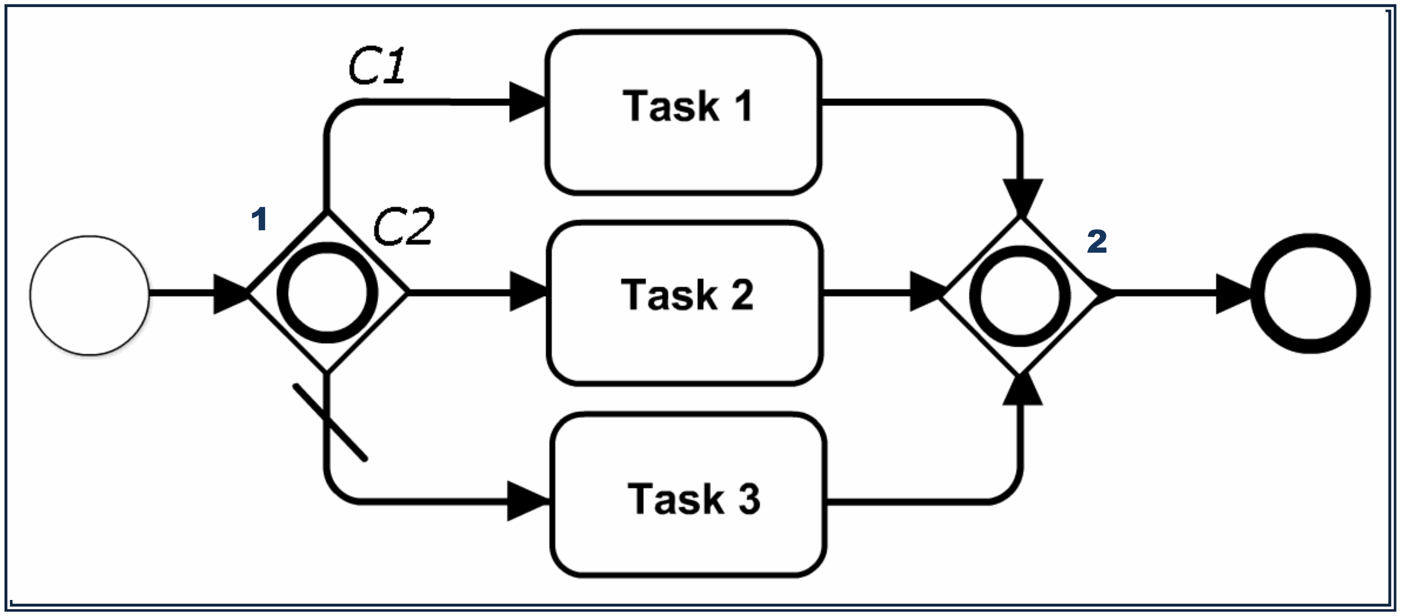
- The *Exclusive Event Based gateway*[1] is similar to the Exclusive Data Based gateway. The only difference is that, instead of evaluating a set of alternatives to determine only one outgoing flow, the event based gateway will start a race between the different events[2] the process might receive, the first one to be received wins the race and that determines which outgoing sequence flow should be used.



http://www.iet.unipi.it/m.cimino/bpm/res/mov07.swf

- Exercise: identify the event before task 2 via the BPMN poster and the specification.

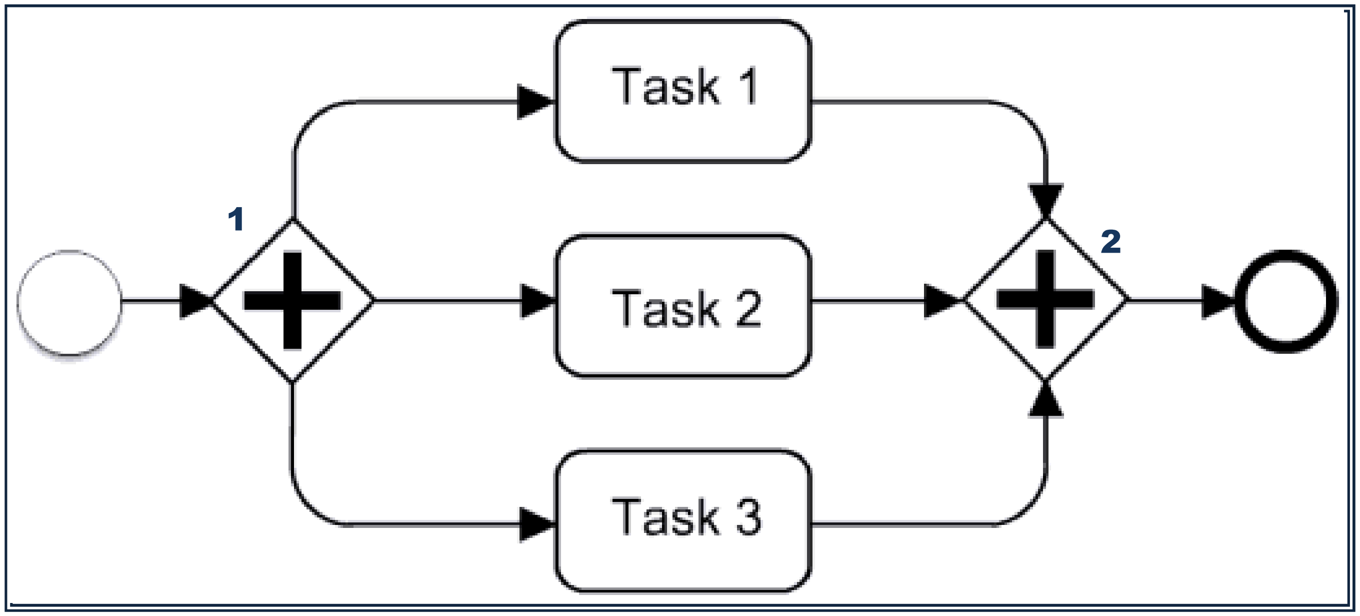
- An *Inclusive Gateway*[1] can be used as a decision point where several outgoing sequence flows are possible, they are all constrained by conditions, each outgoing sequence flow with a condition evaluated as being true will be followed. Effectively it might spawn several execution points.
- Used as a merge[2] the Inclusive Gateway will synchronize all the execution points produced upstream but at most one for each incoming Sequence Flow



http://www.iet.unipi.it/m.cimino/bpm/res/mov08.swf

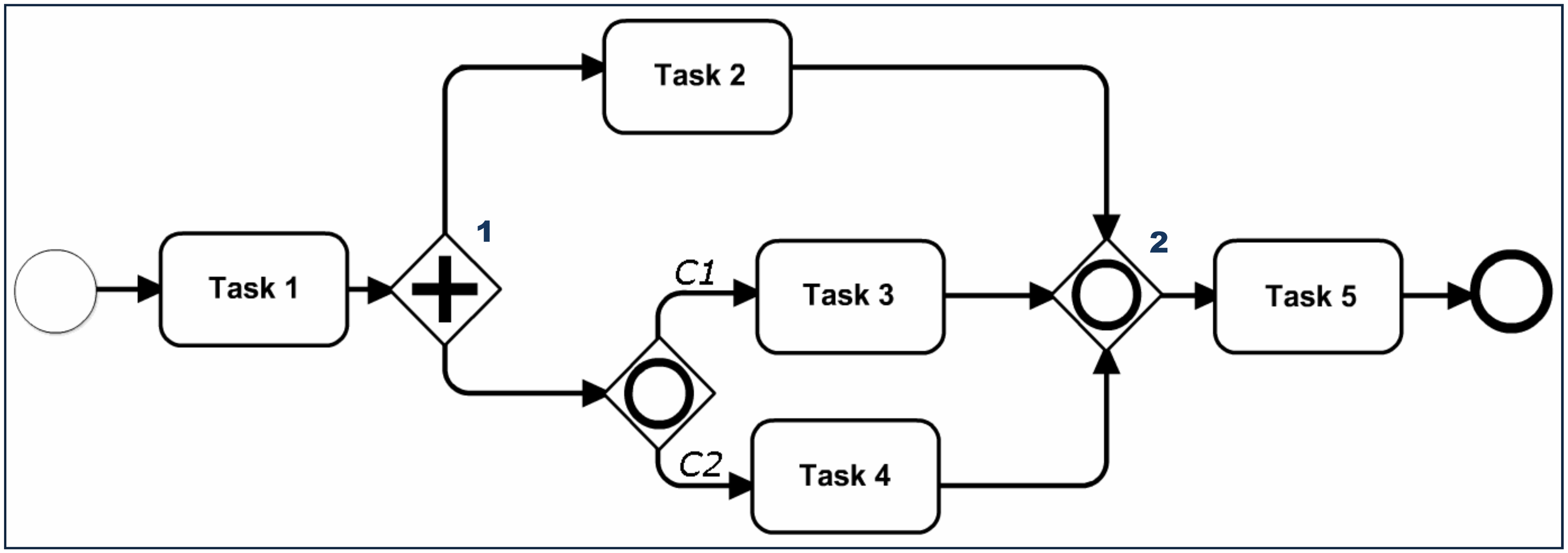
- A gateway's exit can also be marked with a small diagonal slash as *default* sequence flow. It will be selected automatically, (only) if no condition of the other sequence flows is true. This ensures the actual selection of at least one sequence flow.

- A *Parallel Gateway*[1] provides a mechanism to fork and synchronize flows. There are no conditions associated to this gateway.



http://www.iet.unipi.it/m.cimino/bpm/res/mov09.swf

- In the following model, notice how the first inclusive gateway produces two tokens because both conditions 'C1' and 'C2' are evaluated as being true. The second inclusive gateway[2] will not only synchronize the token produced by the upstream inclusive gateway, but also the one coming from the upstream parallel gateway
- Exercise: what happens if the second inclusive gateway is replaced by a parallel gateway? Consider a scenario in which *C1* (or *C2*) is false.



http://www.iet.unipi.it/m.cimino/bpm/res/mov10.swf