Business Process Modelling

Antonio Brogi

Department of Computer Science
University of Pisa

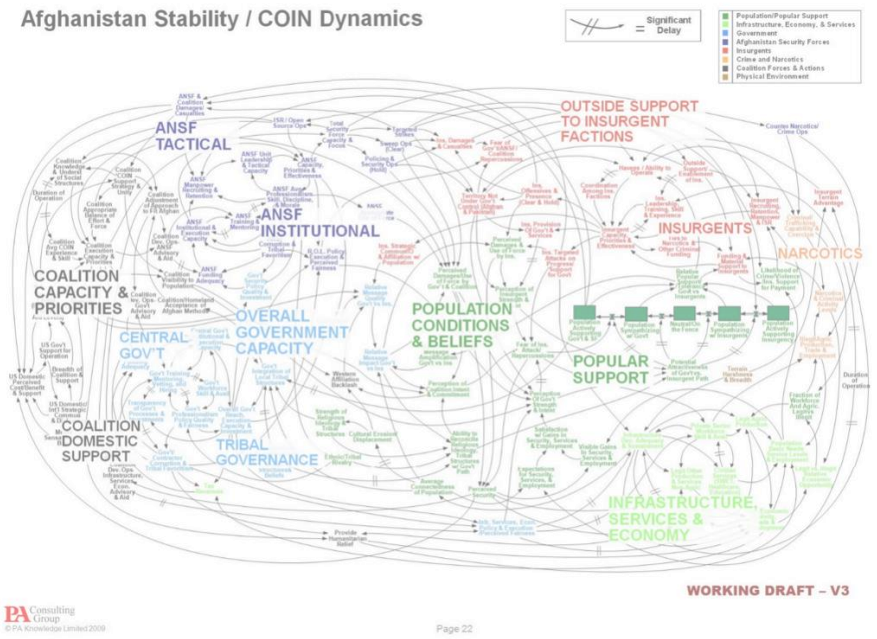
Déjà vu

Web Services

(1) Web sites vs. Web services

(2) Evolution of software development

(3) Business process management view



Business process models

A **business process** consists of a set of **activities** that are performed in coordination in an organizational and technical environment.
These activities jointly realize a business goal.
Each business process is enacted by a single organization, but it may interact with business processes performed by other organizations.

A **business process model** consists of a set of activity models and execution constraints among them.
A **business process instance** represents a concrete case in the operational business of a company, consisting of activity instances.

Each business process model acts as blueprint for a set of business model instances, and each activity model acts as a blueprint for a set of activity instances.

[*M. Weske. Business Process Management. Springer, 2007.*]

Example of business process model

"When we receive a new order, an invoice should be sent to the customer.
The order should be archived only after receiving the payment.
The requested products must be shipped to the customers."

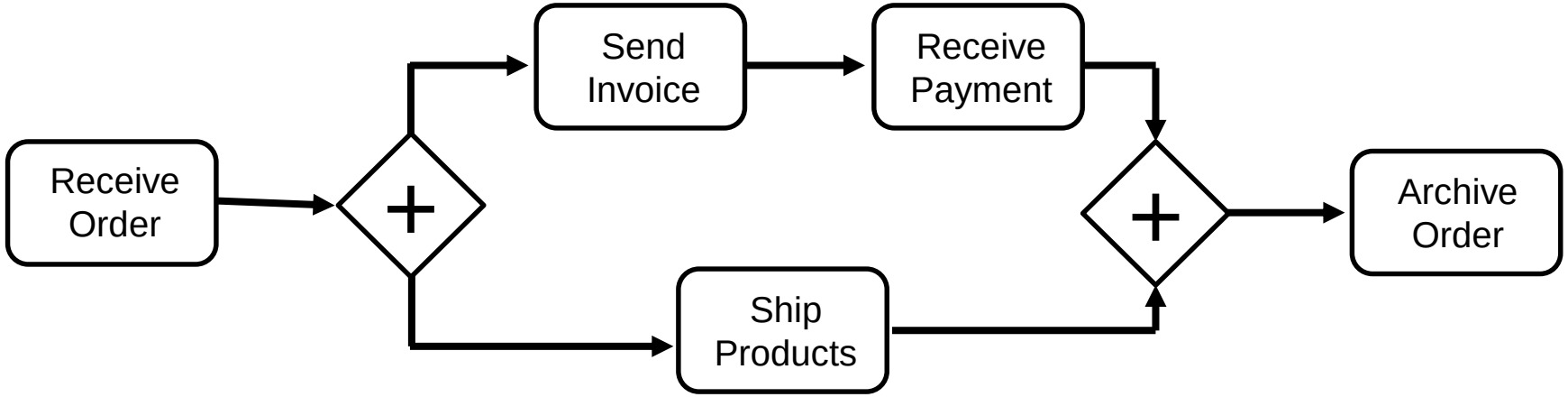
Activities:

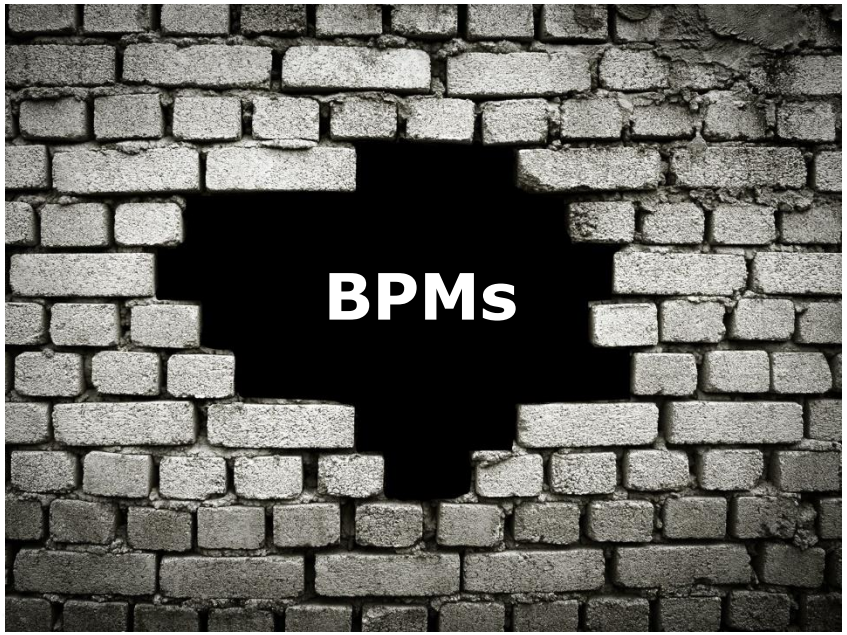
- Receive Order
- Send Invoice
- Archive Order
- Receive Payment
- Ship Products

Example of business process model

(e-commerce)

(traditional)




BPMs

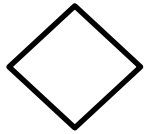
Business process models
BPMN

BPMN - Business Process Model and Notation

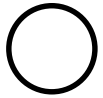
Graphical notation for business process modeling

activity

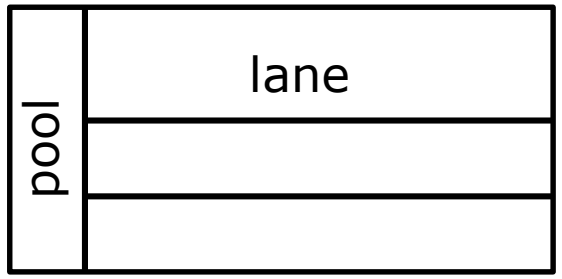
execution ordering



gateway



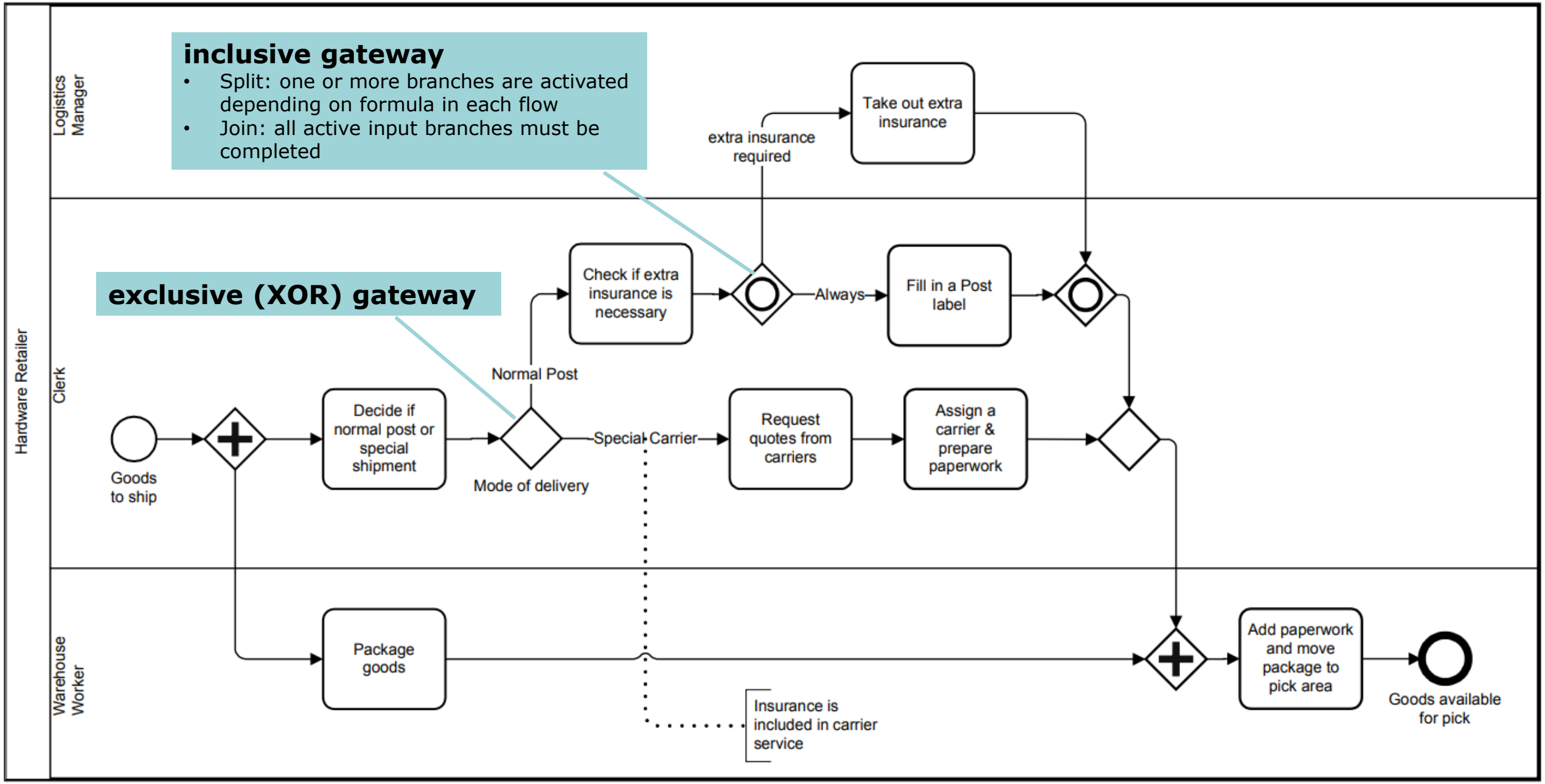
event



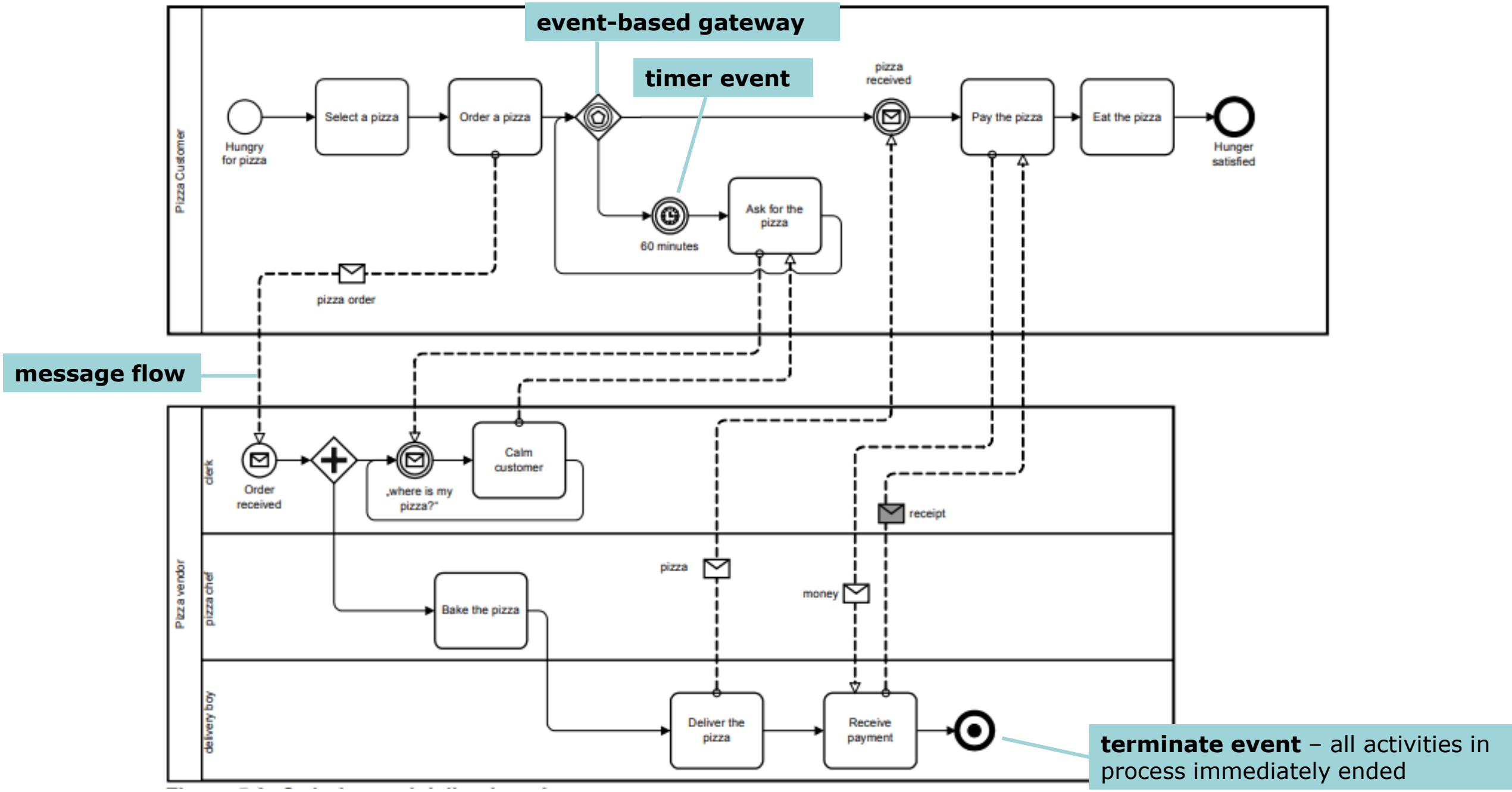
pool defines group of participants or external entity

lane defines participant role within process

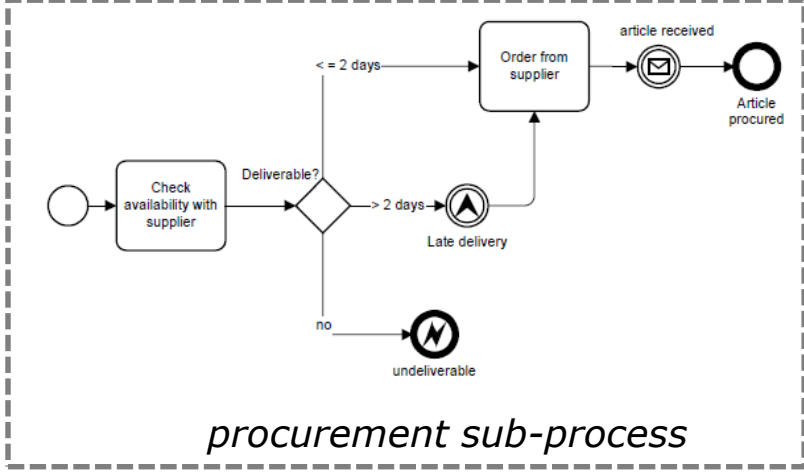
Example: Shipment process of hardware retailer

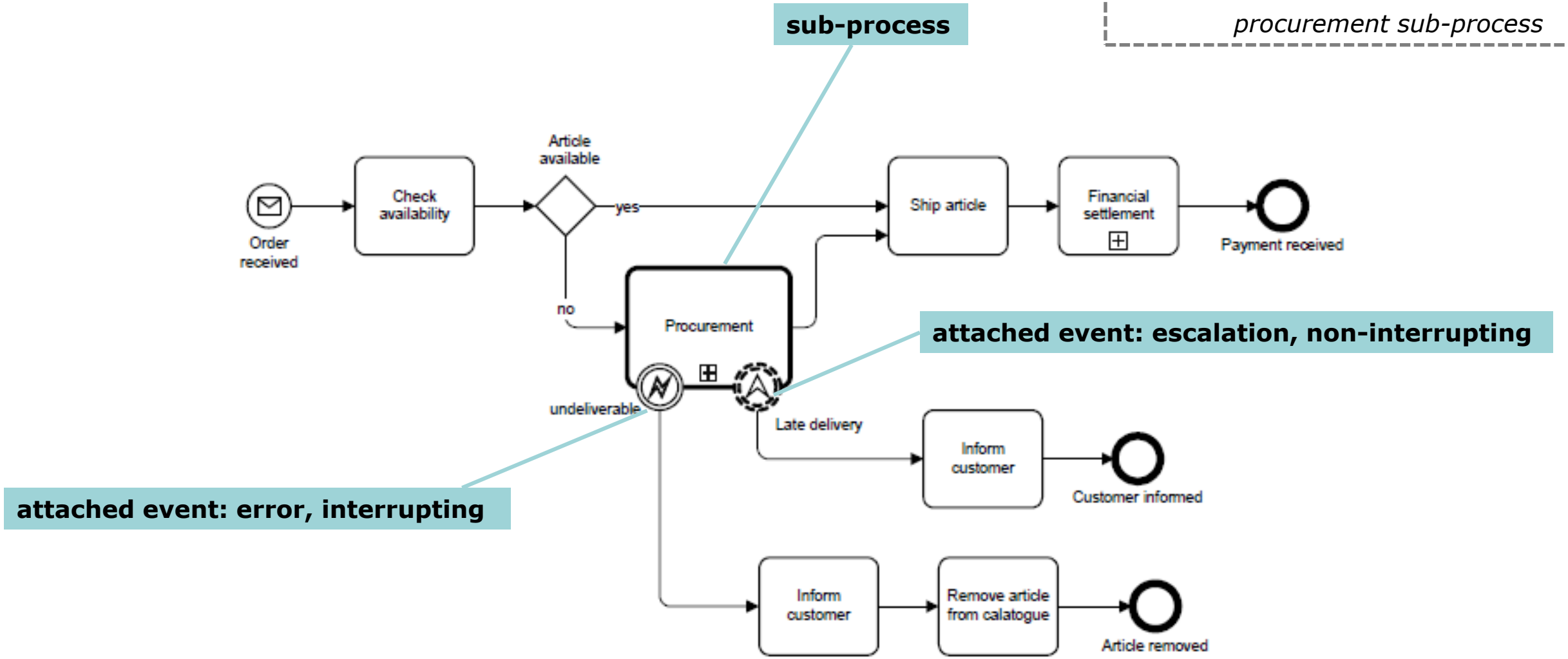


Example: B2B (pizza) collaboration

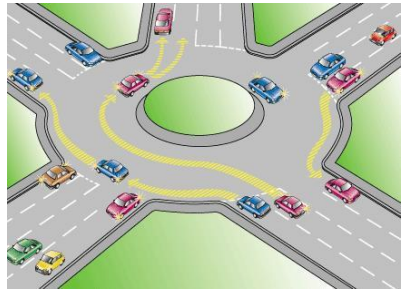


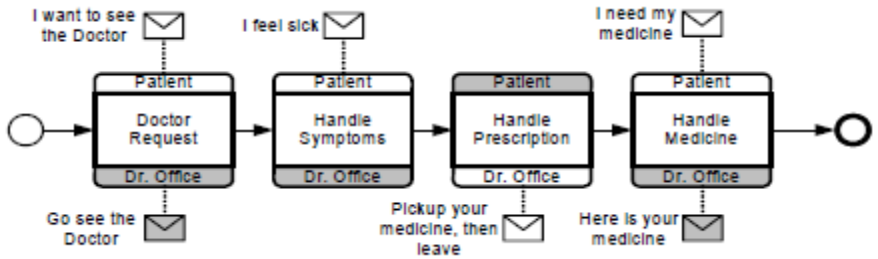
Example: Order fulfillment and procurement





Orchestration vs. Choreography





Many tools available for BPMN

camunda Products Solutions Services Customers Learn Contact Download DE

Workflow and Decision Automation Platform

Open source platform for workflow and decision automation that brings business users and software developers together.

Learn more Download

Camunda BPM 7.10 coming up!

Camunda BPM 7.10 will be released on November 30 - join us for a live webinar on Wednesday, December 5. We'll walk through key features in the release, and attendees will have the chance to ask our very own Daniel Meyer (VP of Engineering) questions and to share feedback.

Register Now

Model

Create BPMN workflow diagrams and DMN decision tables in an editor that both business users and developers love to use.

Execute

Execute your workflows and decisions in powerful engines that are paired with essential applications for process automation projects.

Enjoy

Never fear Business Process Management again as you will love Camunda. If you find that hard to believe, you should just give it a try.

Allianz DB ING DiBa T zalando

Get Product Updates. Email Address Subscribe

By submitting this form, you agree to Camunda's Privacy Policy.

Products	Services	Resources	Community	About	Languages
BPMN Engine	Enterprise Platform	Download Community	Community	Contact	English
DMN Engine	Consulting	Download Enterprise	Forum	Leadership	Deutsch
Modeler	Training	Download Modeler	Events	Locations	
Cockpit		Documentation		Careers	
Tasklist		BPMN Tutorial		Press	
Admin		BPMN Tool		Partners	
Optimize		DMN Tutorial		References	
				Brand	

Privacy Statement Terms Imprint Brand Camunda Services GmbH © 2018

Business Process Model and Notation (BPMN)

Version 2.0

Table of Contents

The **Inclusive Gateway** is activated if

- At least one incoming **Sequence Flow** has at least one *token* and
- For every directed path formed by sequence flow that
  - starts with a **Sequence Flow** ***f*** of the diagram that has a *token*,
  - ends with an *incoming* **Sequence Flow** of the inclusive gateway that has no *token*, and
  - does not visit the **Inclusive Gateway**.
- There is also a directed path formed by **Sequence Flow** that
  - starts with ***f***,
  - ends with an *incoming* **Sequence Flow** of the inclusive gateway that has a *token*, and
  - does not visit the **Inclusive Gateway**.

Upon execution, a *token* is consumed from each incoming **Sequence Flow** that has a *token*. A *token* will be produced on some of the outgoing **Sequence Flows**.

In order to determine the outgoing **Sequence Flows** that receive a *token*, all conditions on the outgoing **Sequence Flows** are evaluated. The evaluation does not have to respect a certain order.

For every condition which evaluates to *true*, a *token* MUST be passed on the respective **Sequence Flow**.

If and only if none of the conditions evaluates to *true*, the *token* is passed on the default **Sequence Flow**.

In case all conditions evaluate to *false* and a default flow has not been specified, the **Inclusive Gateway** throws an exception.

The **Complex Gateway** is in one of the two states: *waiting for start* or *waiting for reset*, initially it is in *waiting for start*. If it is *waiting for start*, then it waits for the `activationExpression` to become *true*. The `activationExpression` is not evaluated before there is at least one *token* on some *incoming* **Sequence Flow**. When it becomes *true*, a *token* is consumed from each *incoming* **Sequence Flow** that has a *token*. To determine which *outgoing* **Sequence Flow** receive a *token*, all conditions on the *outgoing* **Sequence Flows** are evaluated (in any order). Those and only those that evaluate to *true* receive a *token*. If no condition evaluates to *true*, and only then, the *default* **Sequence Flow** receives a *token*. If no default flow is specified an exception is thrown. The **Gateway** changes its state to *waiting for reset*.The **Gateway** remembers from which of the *incoming* **Sequence Flows** it consumed *tokens* in the first phase.

When *waiting for reset*, the **Gateway** waits for a *token* on each of those *incoming* **Sequence Flows** from which it has not yet received a *token* in the first phase unless such a *token* is not expected according to the join behavior of an **inclusive Gateway**.

More precisely, the **Gateway** being *waiting for reset*, *resets* when for every directed path formed by sequence flow that

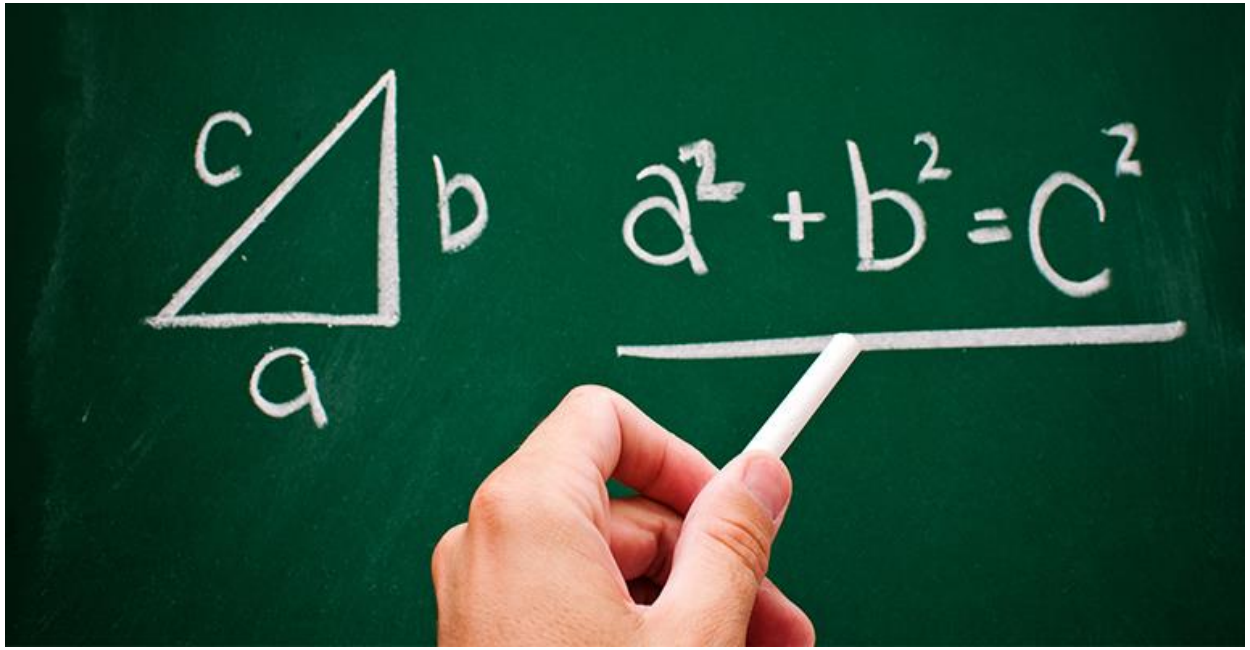
- starts with a **Sequence Flow** ***f*** of the diagram that has a *token*,
- ends with an *incoming* **Sequence Flow** of the **Complex Gateway** that has no *token* and has not consumed a *token* in the first phase, and that
- does not visit the **Complex Gateway**.

- There is also a directed path formed by **Sequence Flow** that
  - starts with ***f***,
  - ends with an *incoming* **Sequence Flow** of the **Complex Gateway** that has a *token* or from which a *token* was consumed in the first phase, and that,
  - does not visit the **Complex Gateway**.

If the **Complex Gateway** is contained in a **Sub-Process**, then no paths are considered that cross the boundary of that **Sub-Process**.

When the **Gateway** resets, it consumes a *token* from each *incoming* **Sequence Flow** that has a *token* and from which it had not yet consumed a *token* in the first phase. It then evaluates all conditions on the *outgoing* **Sequence Flows** (in any order) to determine which **Sequence Flows** receives a *token*. Those and only those that evaluate to *true* receive a *token*. If no condition evaluates to *true*, and only then, the *default* **Sequence Flow** receives a *token*. The **Gateway** changes its state back to the state *waiting for start*. Note that the **Gateway** might not produce any *tokens* in this phase and no exception is thrown. Note that the conditions on the *outgoing* **Sequence Flows** MAY evaluate differently in the two phases, e.g., by referring to the state of the **Gateway** (runtime attribute `waitingForStart`).

Note that if the `activationCondition` never becomes *true* in the first phase, *tokens* are blocked indefinitely at the **Complex Gateway**, which MAY cause a deadlock of the entire **Process**.



The possibility of **proving properties of business process models is a crucial** aspect of business process management

Business process models
BPMN
Workflow nets

Workflow nets

Extension of **Petri nets** (see next)

One of the best known tecnhiques for specifying business processes in a *formal* and *abstract* way

+ Graphical representation eases communications between different stakeholders
+ Process properties can be formally analysed
+ Various supporting tools are available

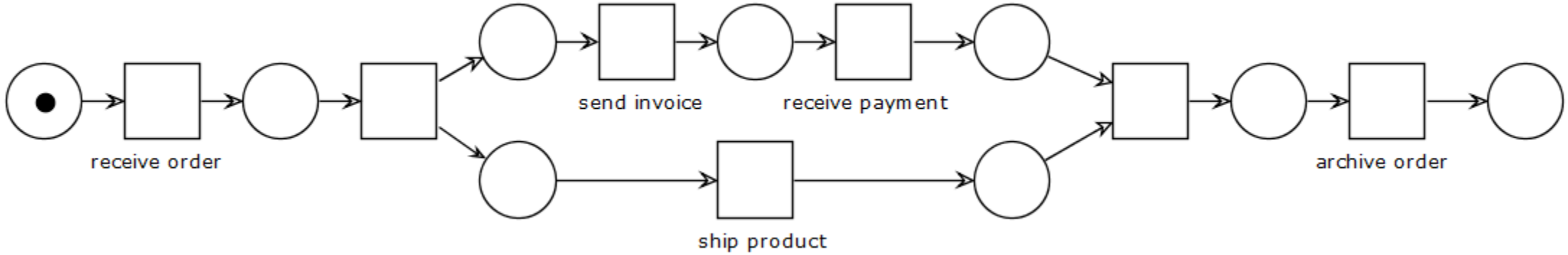
[*W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. The Journal of Circuits, Systems and Computers, 8(1):21-66, 1998.*]

Business process models
BPMN
Workflow nets
definition

Petri nets in 3 slides

Petri nets consist of **places**, **transitions** and direct **arcs** connecting places and transitions.

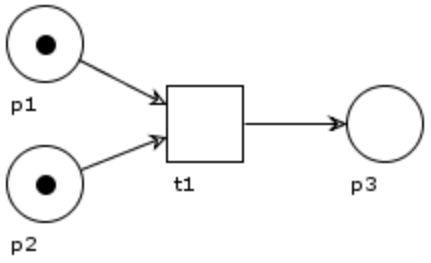
Transitions model activities, places and arcs model execution constraints.



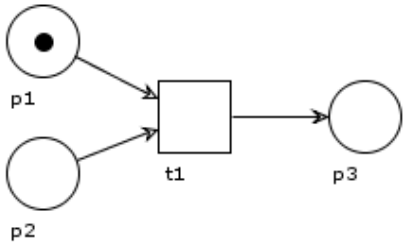
System dynamics represented by **tokens**, whose distribution over the places determines the state of modelled system.

Petri nets in 3 slides

A transition *can fire* if there is a token in each of its input places



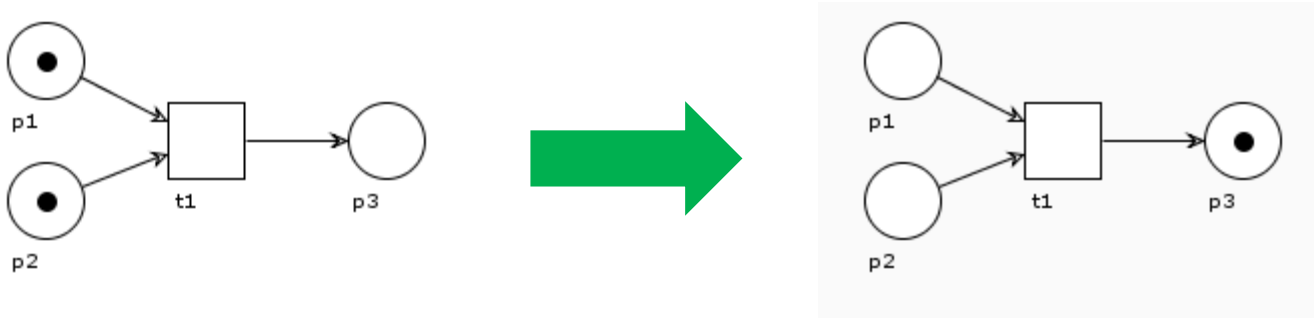
t1 can fire

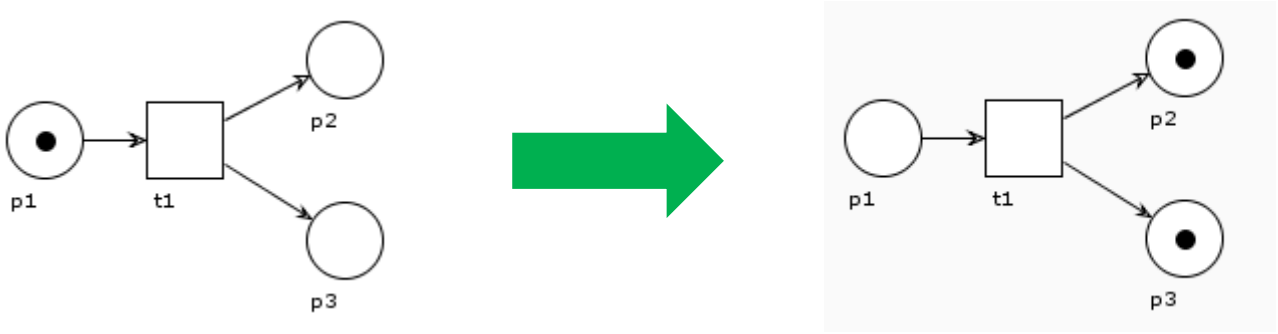


t1 cannot fire

Petri nets in 3 slides

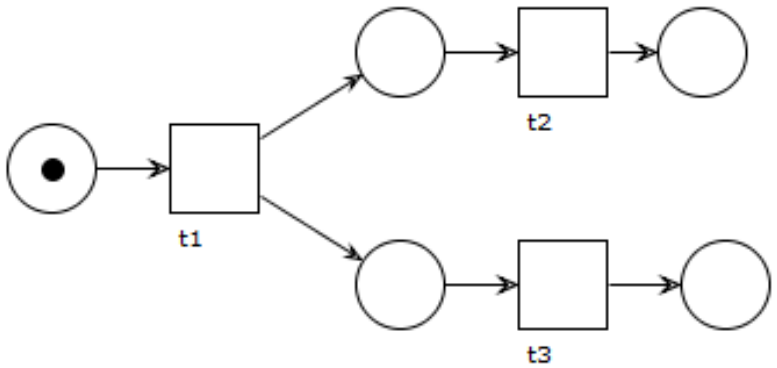
If a transition *fires*, one token is removed from each input place and one token is added to each output place



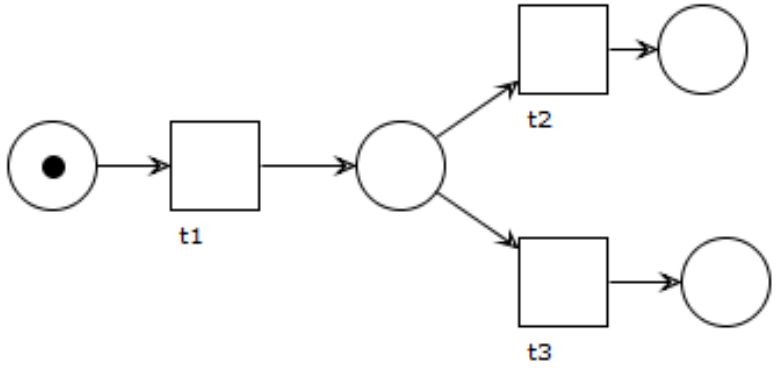


Review question

What is the difference between



and



?

Workflow nets

Idea: Enhance Petri nets with concepts and notations that ease the representation of business processes

Like Petri nets, workflow nets focus on the *control flow* behaviour of a process:

- transitions represent activities
- places represent conditions
- tokens represent process instances

Workflow nets

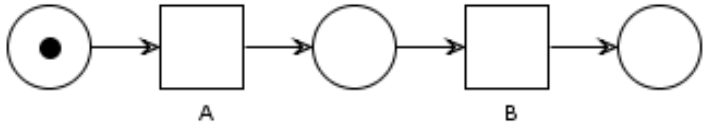
A Petri net is a ***workflow net*** iff

(1)There is an initial place with no incoming edge, and

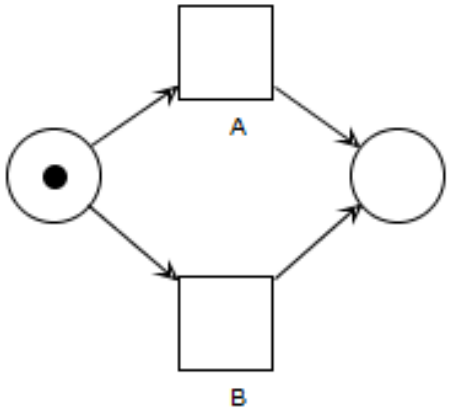
(2)There is a final place with no outging edge, and

(3)All places and transitions are located on some path from the initial place to the final place

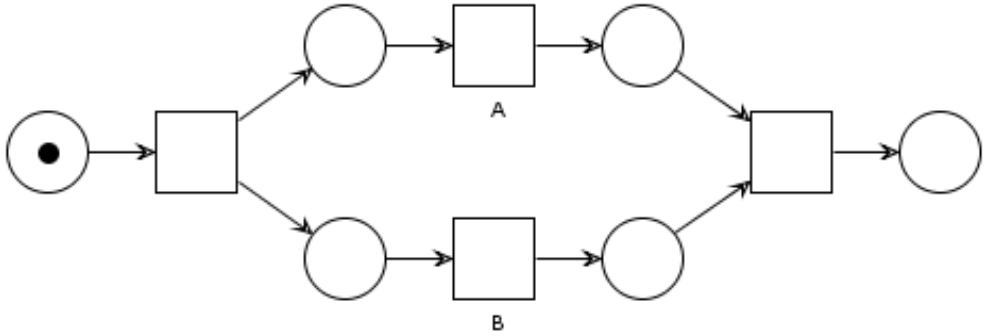
Business process models
BPMN
Workflow nets
- definition
- composition patterns



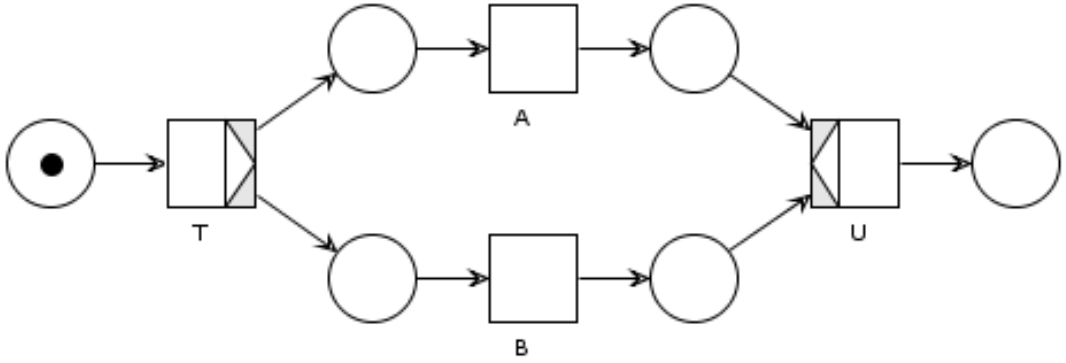
sequential



implicit OR *("A+B")*

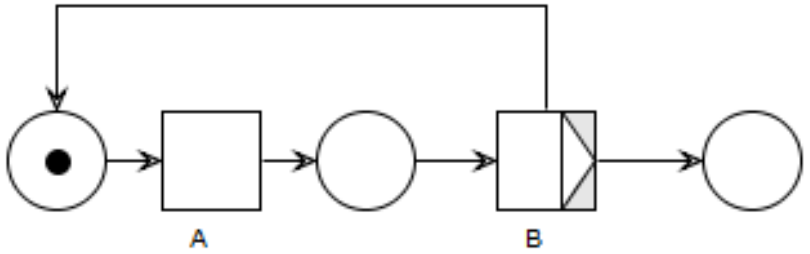


AND



explicit OR *("τ.A+τ.B")*

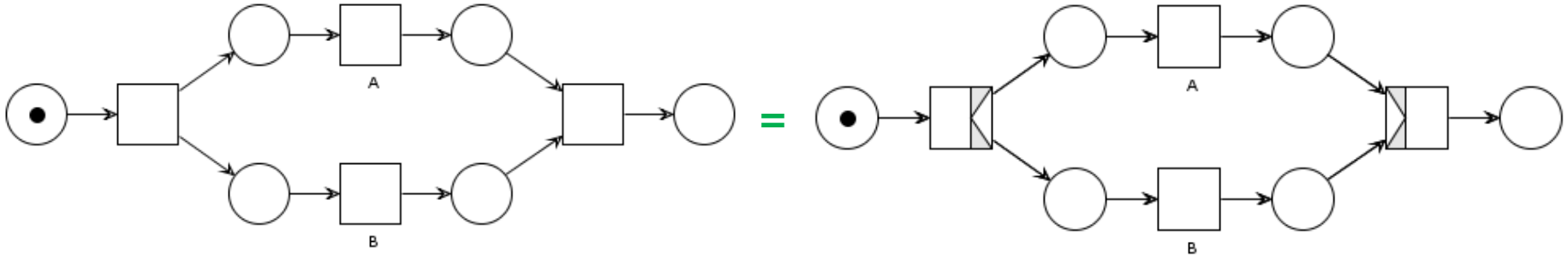
T will pace **only one** token in one of its output places.
U can fire if (at least) **one** of its input places contains a token.



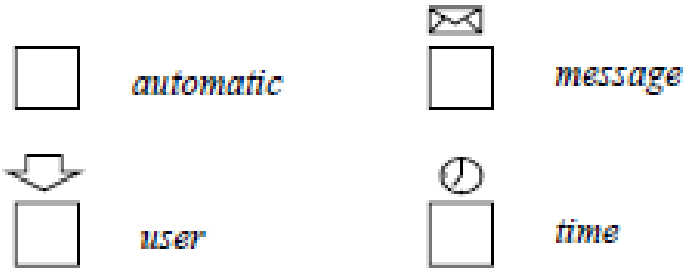
iteration

Workflow nets

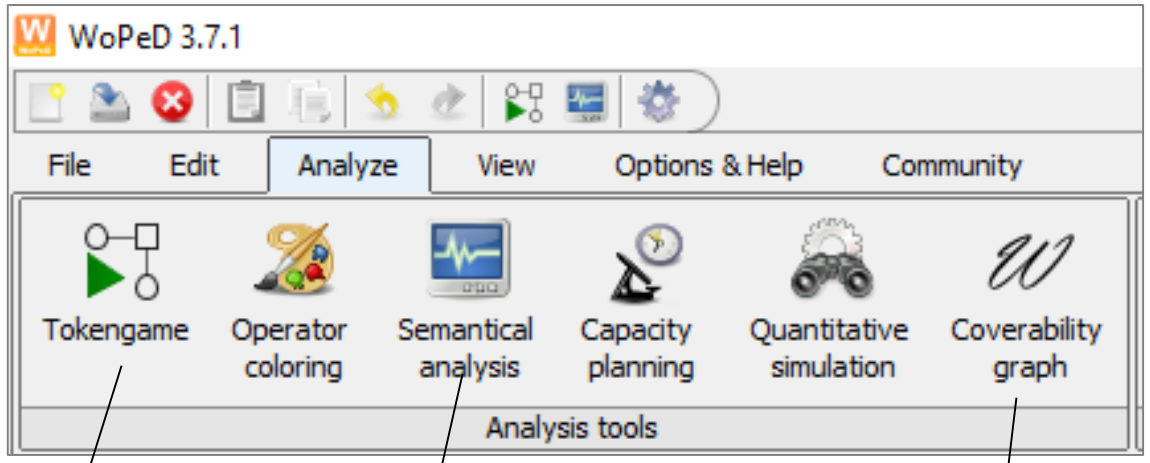
Equivalent “sugared” representation of AND-split and AND-join transitions



Transitions can be annotated with *triggers*, to denote who/what is responsible for an enabled transition to fire

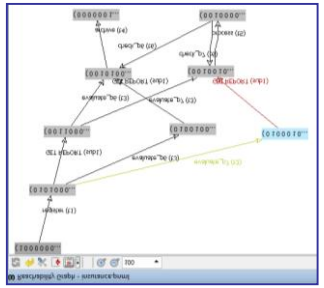


Business process models
BPMN
Workflow nets
- definition
- composition patterns
- WoPeD



simulation

formal verification
(soundness)



Business process models
BPMN
Workflow nets
- definition
- composition patterns
- WoPeD
- soundness

Soundness

A workflow net is ***sound*** iff

(1) every net execution starting from the *initial marking* (one token in the initial place, no tokens elsewhere) eventually leads to the *final marking* (one token in the final place, no tokens elsewhere), and

(2) every transition occurs in at least one net execution

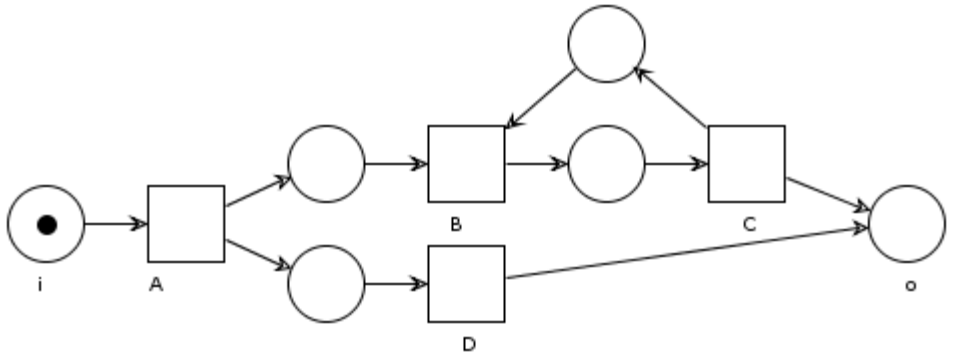
Examples

A workflow net is ***sound*** iff

(1) every net execution starting from the *initial marking* (one token in the initial place, no tokens elsewhere) eventually leads to the *final marking* (one token in the final place, no tokens elsewhere), and

(2) every transition occurs in at least one net execution

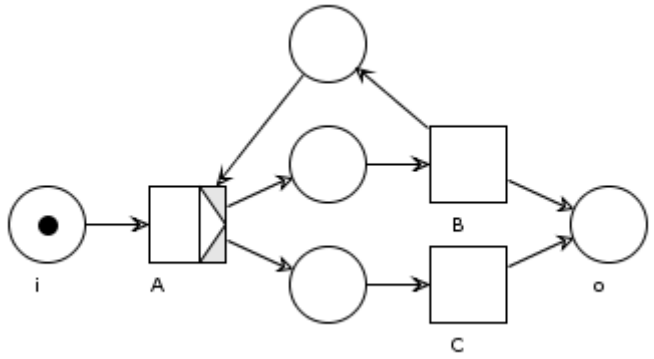
Is



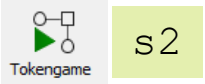
sound?

s1

Is

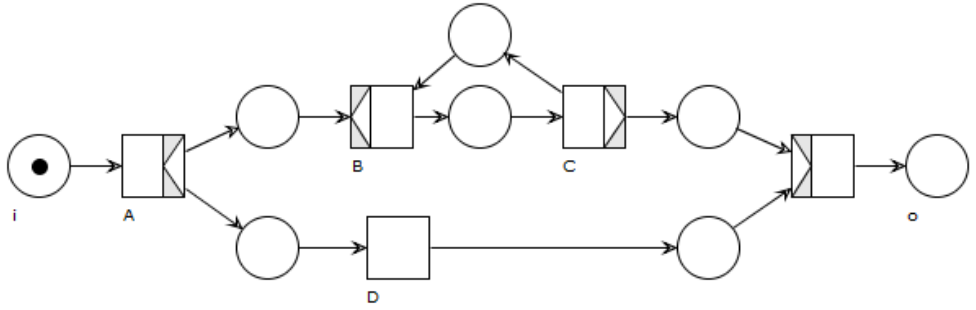


sound?



s2

Is



sound?

s3

Examples (cont.)

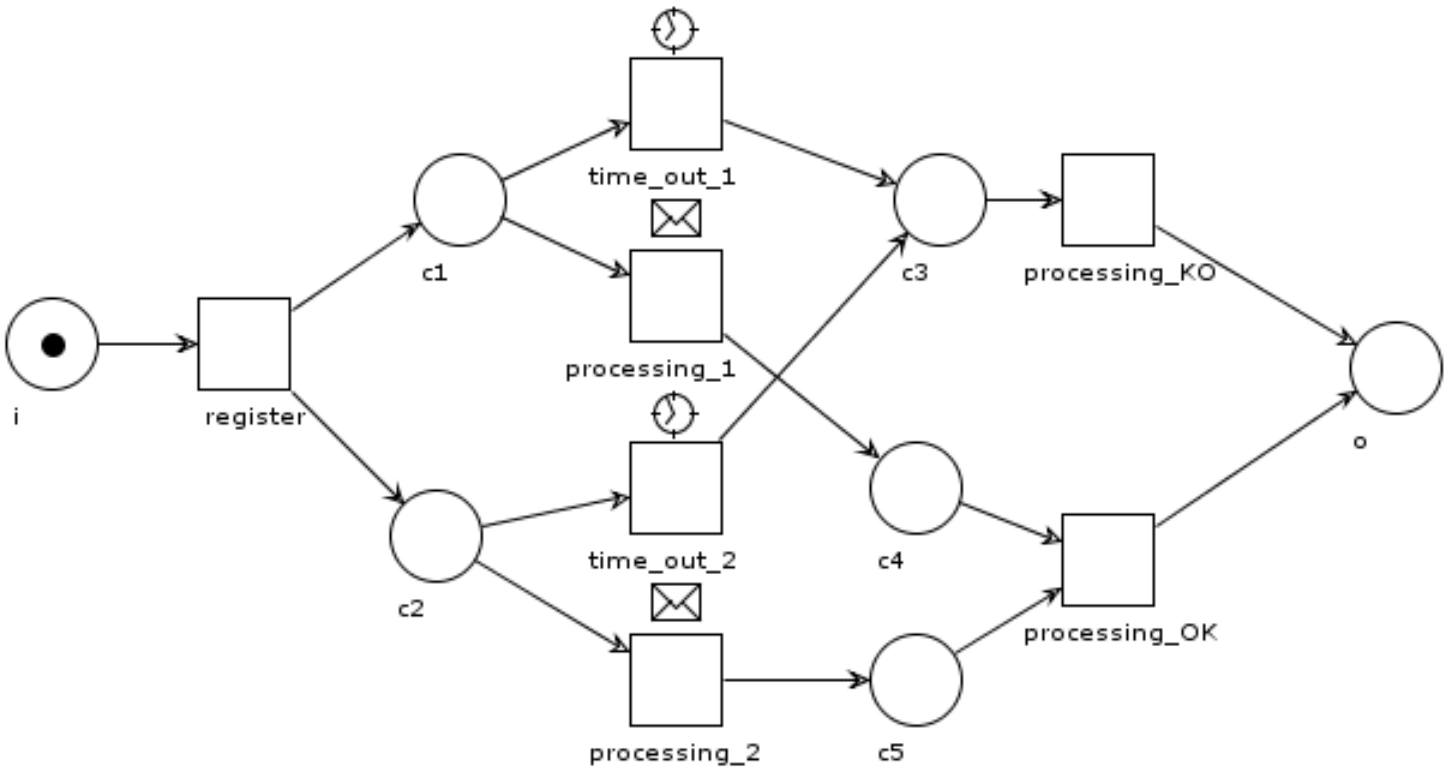
A workflow net is ***sound*** iff

(1) every net execution starting from the *initial marking* (one token in the initial place, no tokens elsewhere) eventually leads to the *final marking* (one token in the final place, no tokens elsewhere), and

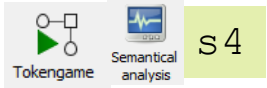
(2) every transition occurs in at least one net execution

Consider the following net specifying a business process to process of complaints

Idea: processing succeeds if both *processing_1* and *processing_2* activities are performed



Is the above net sound?



s4

Soundness (cont.)

How to establish whether a net is sound?

Definition. A Petri net is **live** iff for every reachable state M' and every transition t, there is a state M" reachable from M' which enables t.

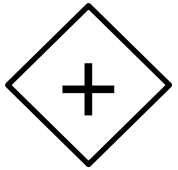
Definition. A Petri net is **bounded** iff for each place p there is a natural number n such that for every reachable state the number of tokens in p is less than n.

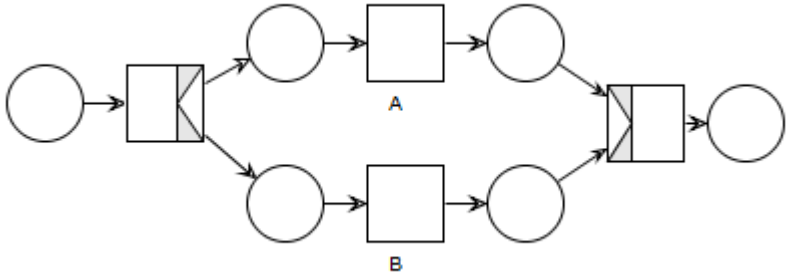
Theorem. A workflow net N is *sound iff* (Ň,i) is *live* and *bounded*.

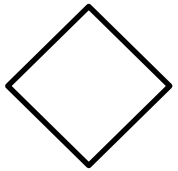
where Ň is N extended with a transition from the final place o to the initial place i

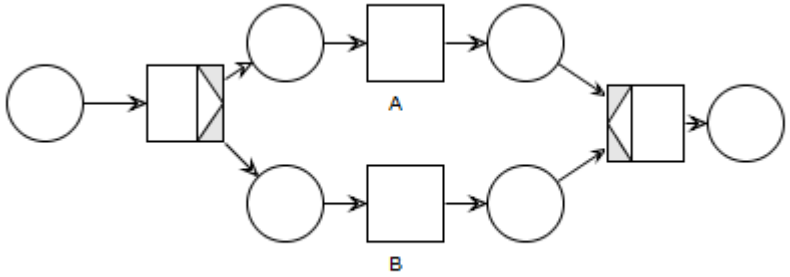
Business process models
BPMN
Workflow nets
- definition
- composition patterns
- WoPeD
- soundness

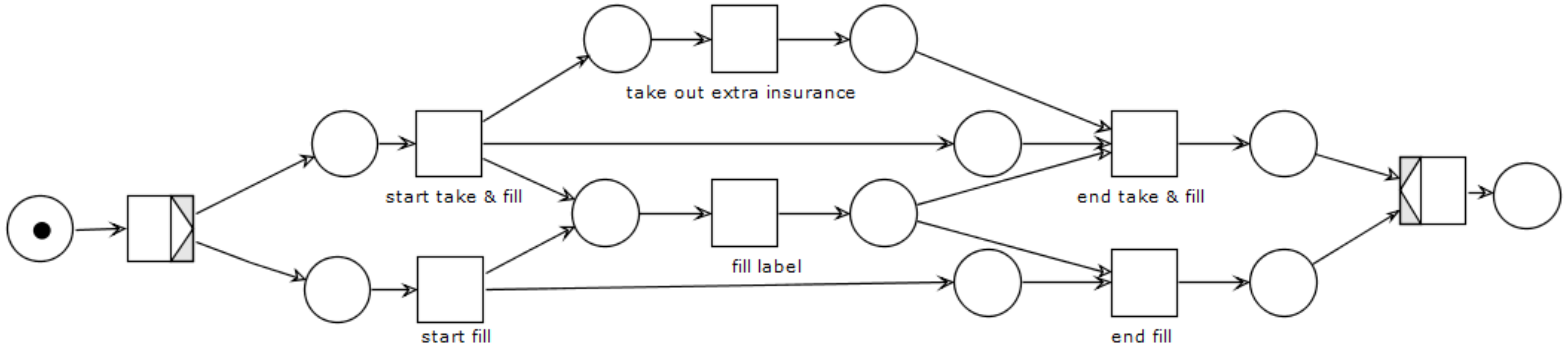
From BPMN to workflow nets




A
B

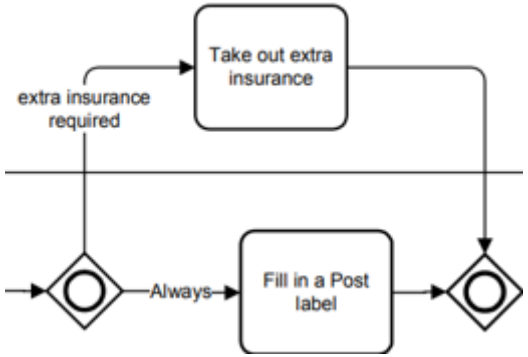


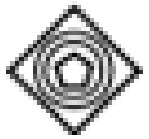

A
B



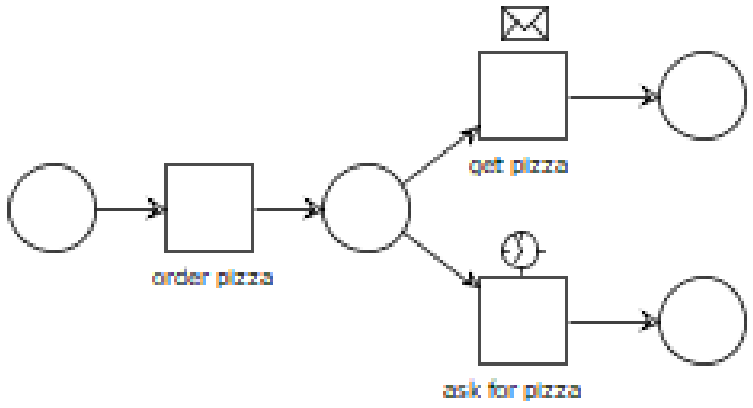
inclusive gateway

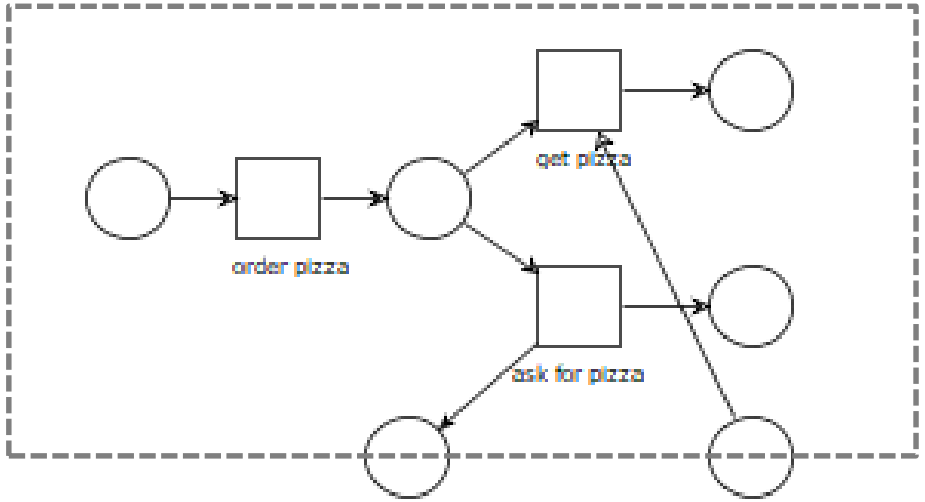
- Split: one or more branches are activated depending on formula in each flow
- Join: all active input branches must be completed





event-based gateway





terminate event – all activities in process immediately ended