Business Process Patterns

Business Process Patterns are ***general solutions for typical situations*** occurring in the process of modeling the business processes.

Business Process Patterns ***complete the methodology*** with the further information about ***how to create models*** which are ***fully consistent*** with its principles and rules.

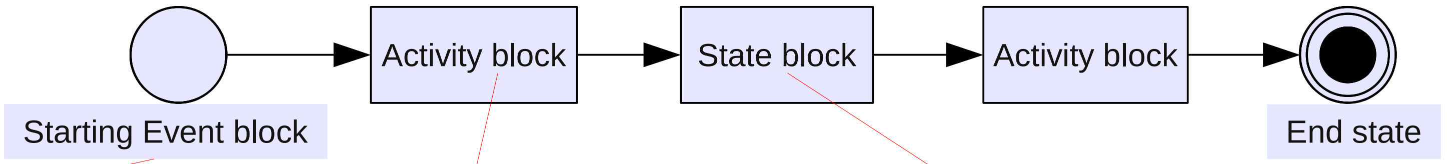
Business Process Patterns can be also used as the general examples of typical segments which the business process should consists of. These segments can be instantiated ***for the particular situations*** and used as ***basic building blocks of the business process model***.

- Basic Business Process Flow Pattern – defines the basic procedure and decision points of model creation
- BP patterns for particular situations:
  - complementary events
  - repeating state
  - <<inclusive OR>>

Basic Business Process Flow Pattern

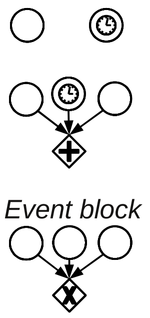
This pattern expresses the ***basic structure of the process model*** respecting the essential MMABP methodology rules.

Business process is described as a sequence of <u>Activity blocks</u> interrupted by <u>State blocks</u> starting with one <u>Starting Event block</u> and resulting in one or more <u>End states</u>.



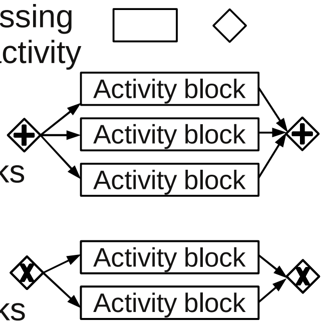
Event block can be:

- Single event
- Synchronization of parallel Event blocks
- Set of exclusive Event blocks



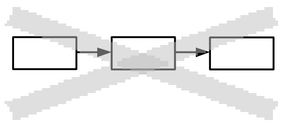
Activity block can be:

- Single processing or decision activity
- Structure of parallel Activity blocks
- Structure of exclusive Activity blocks



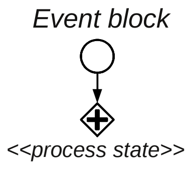
Activity block must not be:

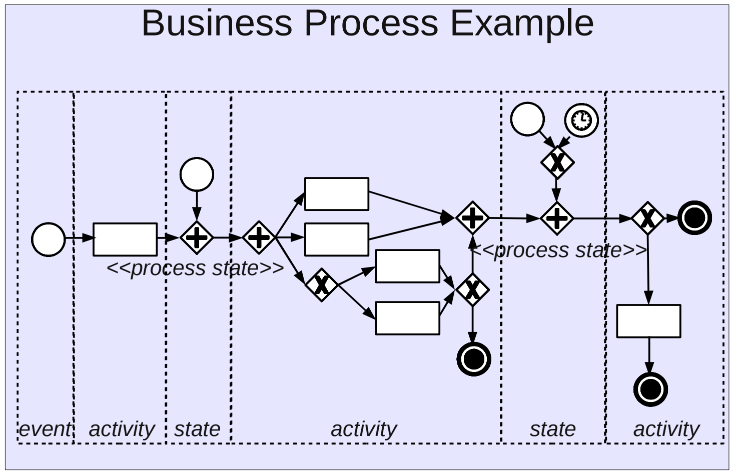
- Sequence of processing activities nor structures



State block is a:

- Synchronization of the process run with external event block





Business Processes Models

Model of the process run (once again)

Describes dynamics of the process – basic logic of the process:

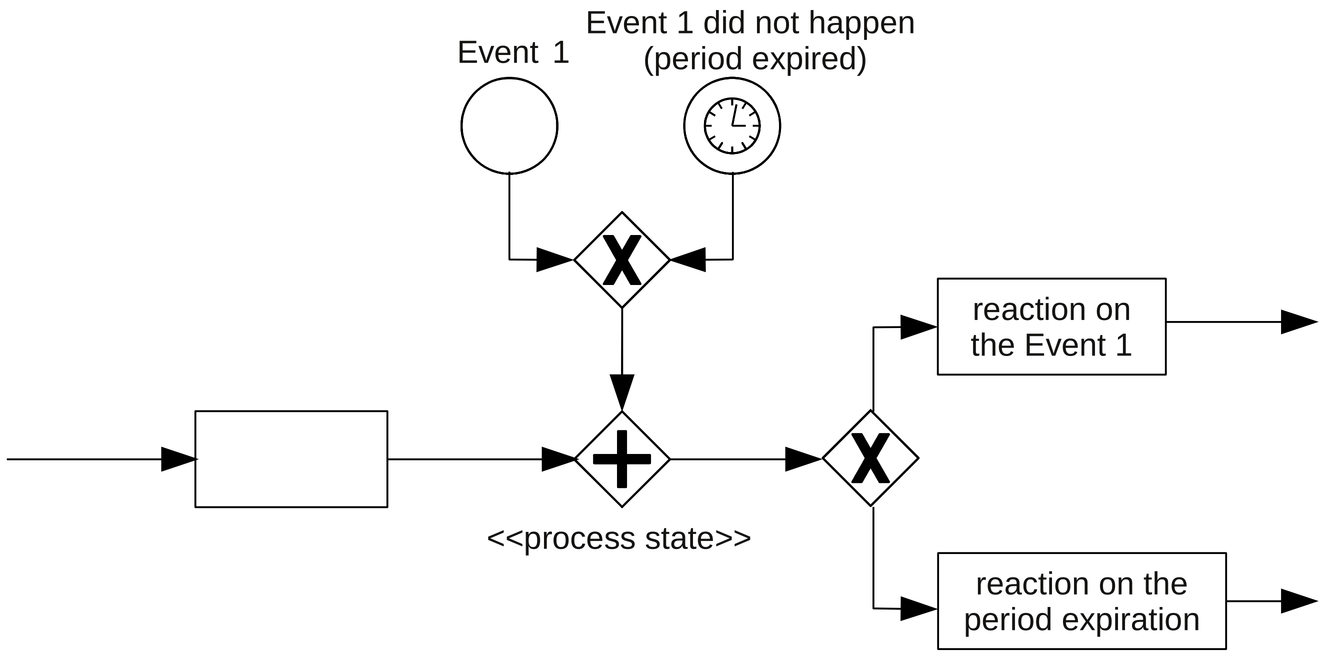
- **Every key process** should be described (and some of other - important - processes, if necessary);
- **Level of granularity** is given by external factors influencing the process - **events**);
  - One activity between two states (i.e. events, consequently),
  - Reason for more detailed division of an activity is no more objective. It is always subjective – relative to some other factor (technology, qualification, organization, etc.).
  - Identification of the border of possible (i.e. a sense making) optimization of a process.
- **process state** represents waiting for the event (only one / one of alternative ones / synchronization of several ones);
- **Actors, organization units**, and other important **external aspects** are mapped to the process activities;

BP patterns for particular situations

- complementary events pattern
- repeating state pattern
- <<inclusive OR>> pattern

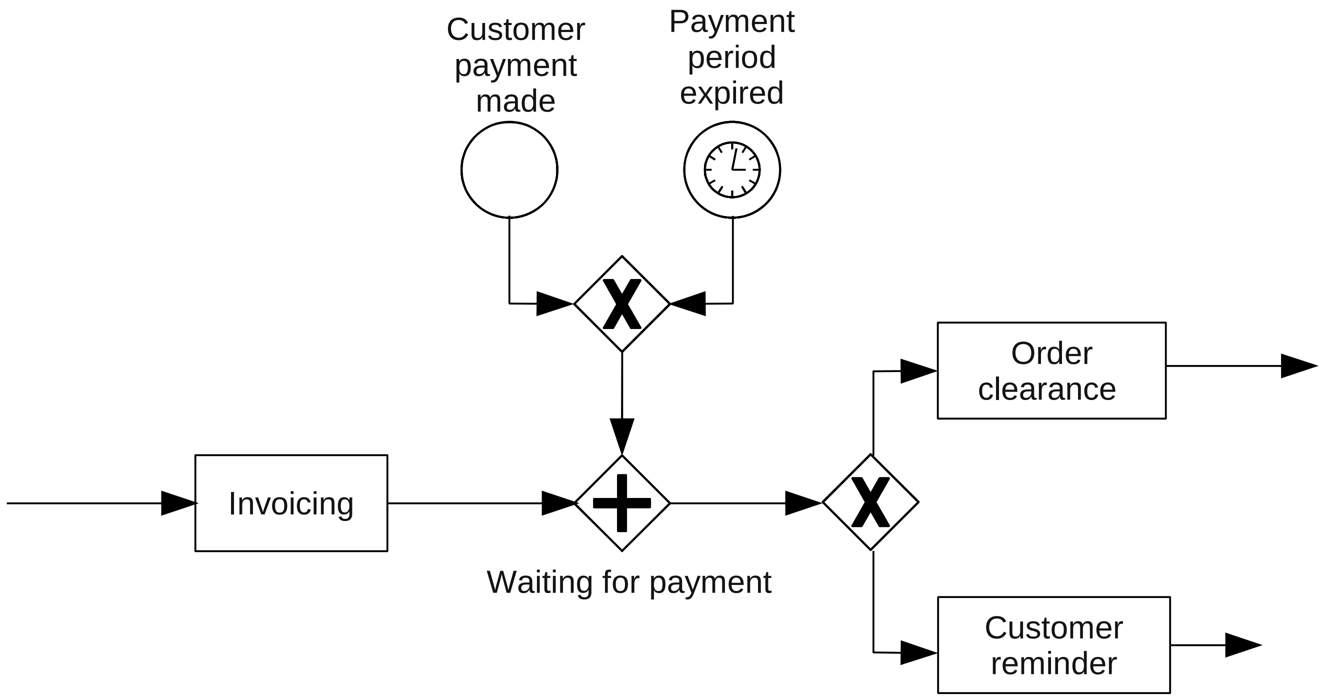
Complementary Events BP Pattern

This pattern should be used for expressing that some ***event can either happen or not***.
To find out that something did not happen it is necessary to determine the ***period***.
The fact that the event ***did not happen*** is expressed with a ***timer event*** expressing that the ***period has expired***.


Event 1
Event 1 did not happen (period expired)
<<process state>>
reaction on the Event 1
reaction on the period expiration

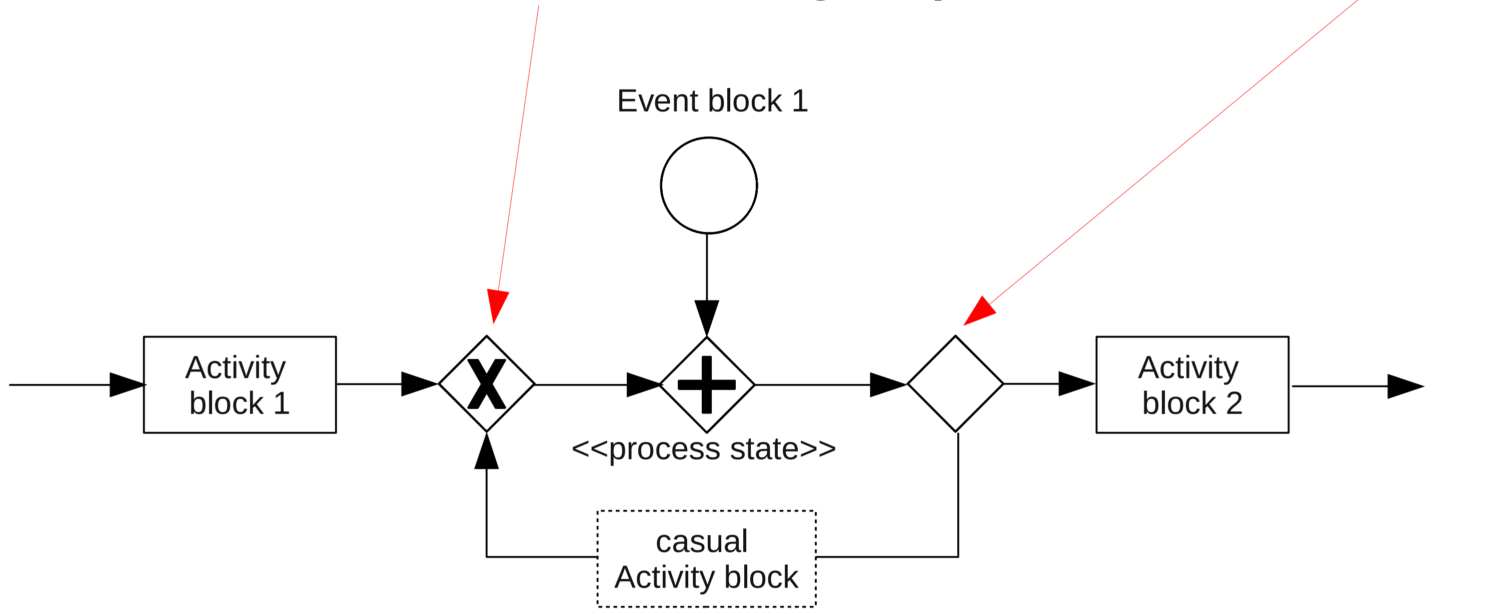
Complementary Events BP Pattern - example -

The fact that the event ***did not happen*** is expressed with a ***timer event*** expressing that the ***period has expired***.



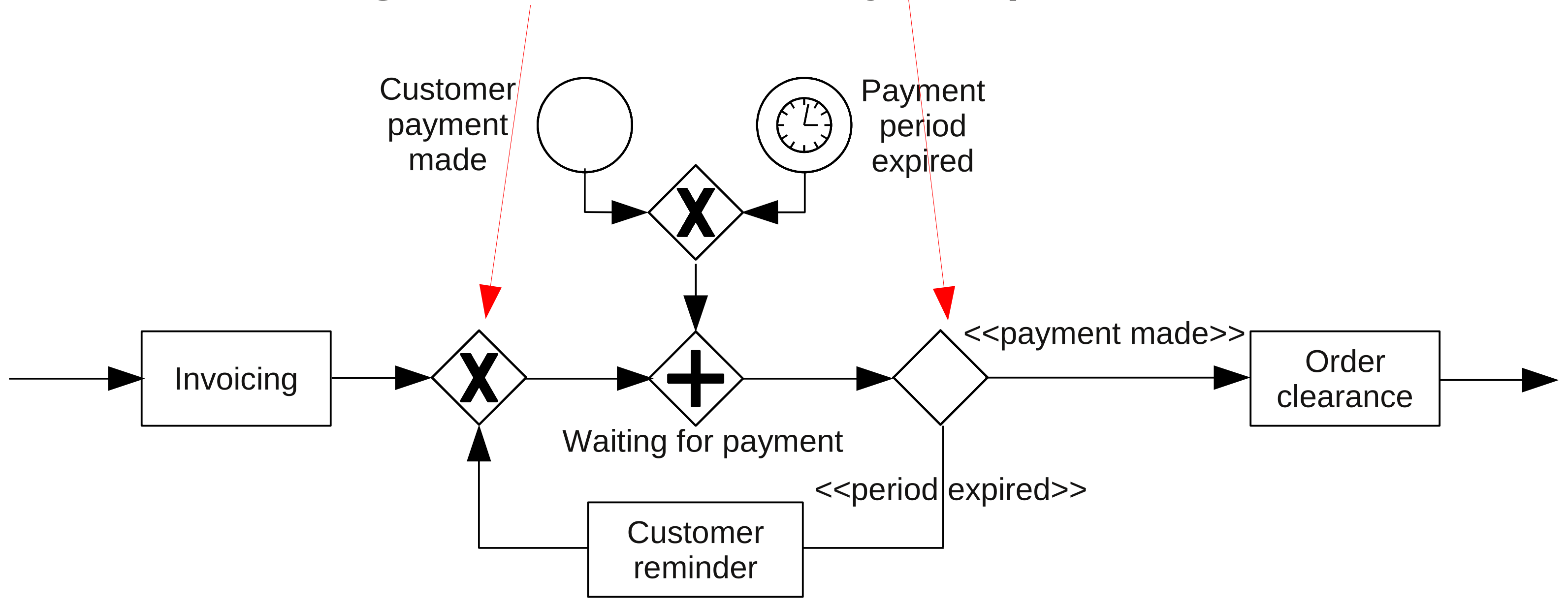
Repeating State BP Pattern

This pattern should be used for expressing that some ***state can occur repeatedly***.
As the cycle has to be finite the state must be ***immediately followed by the decision*** about its end.
Return to the state is an ***alternative to the original path*** to it.



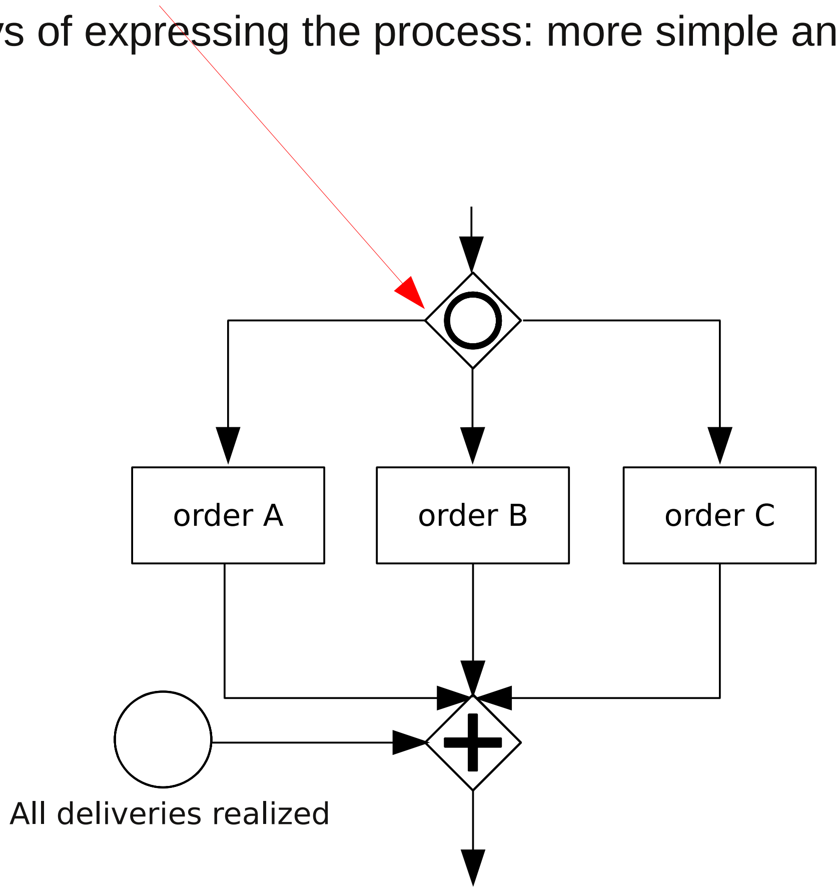
Repeating State BP Pattern - example -

When no payment is made within the given period the customer is reminded, new period is set, and the process is waiting for the payment again.
Once the payment is made the ***reminding cycle ends***.
The process has to ***distinguish between the very first period and the other ones***.



<<inclusive OR>> BP Pattern

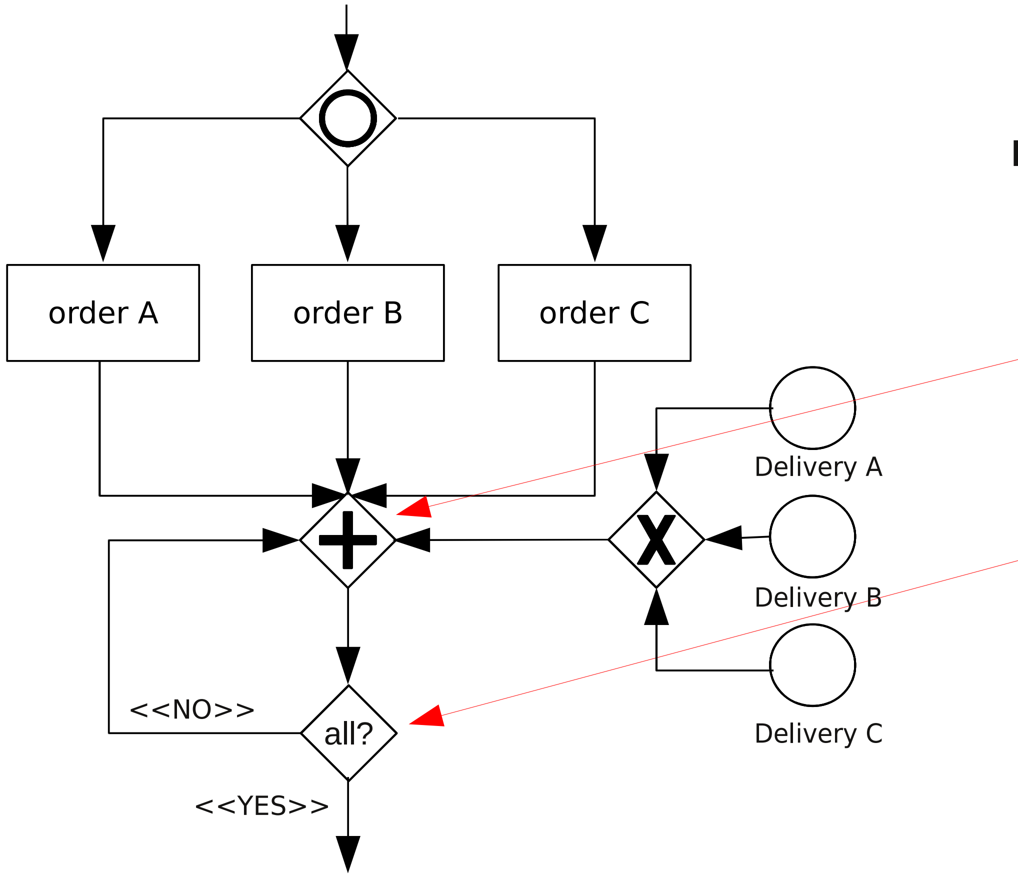
This pattern should be used to express that ***any combination of any number of described activities may occur***.
As the ***<<inclusive OR>>*** is a ***non exact construction*** on principle one has to choose from two basic ways of expressing the process: more simple and less exact or more exact and more complicated one.



Simple but less exact solution

- The event “All deliveries realized” is an abstract event which is not a real event (not existing in the Real World)
    - in fact, it should be a product of the process activities which are abstracted (not modeled) in this model.
- Consequently, such model cannot be used for the purpose which needs the exact definition of all possible elementary variants (like programming the process workflow for instance).
- The model is simple and expresses the essence of the problem although not absolutely precisely.

<<inclusive OR>> BP Pattern

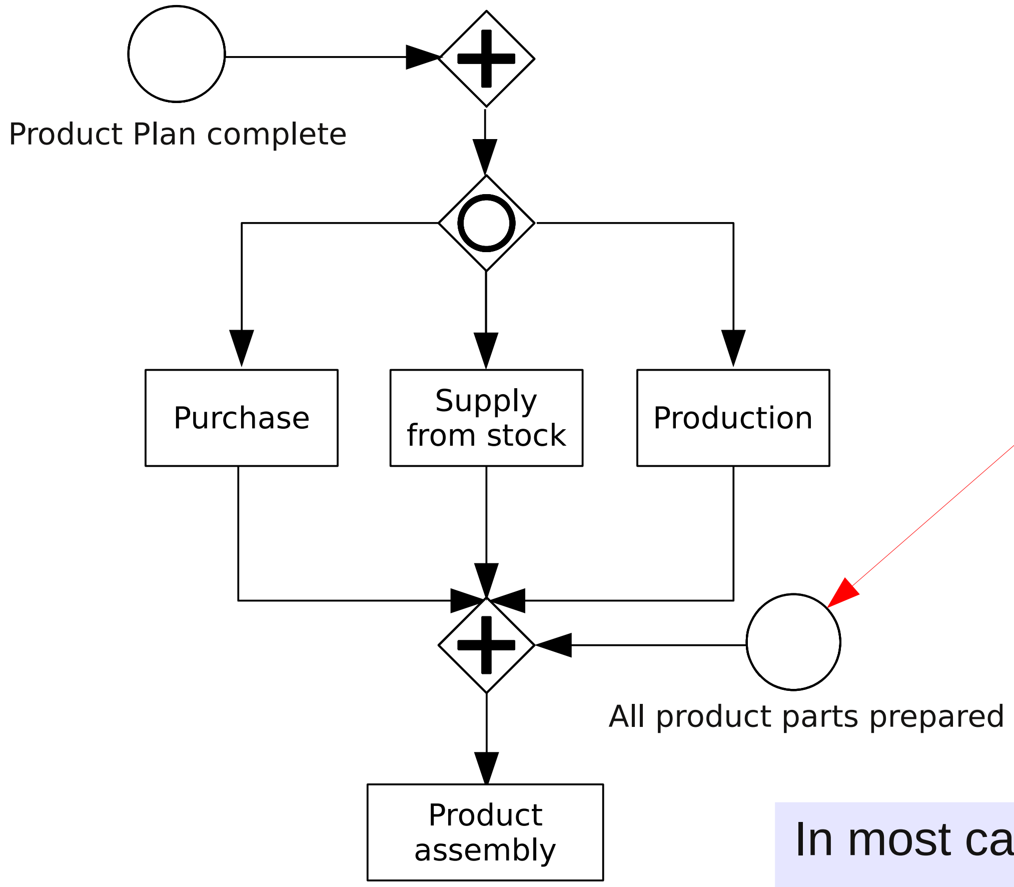


More exact but complicated solution

- The process state represents the waiting for one delivery (the result of the one of above expressed activities). The following decision has to be made for every occurring event
- Such model can be used for the purpose which needs the exact definition of all possible elementary variants.
- Nevertheless, the description has to be made on the level of particular time points although the needed message is just “all started actions were finished”.
- The model expresses the exact “technical” way of making quite a simple decision which can unnecessarily obscure relatively simple essence of the problem.

<<inclusive OR>> BP Pattern - example -

This pattern should be used to express that ***any combination of any number of described activities may occur***.



Compound product realization

Every part of the *Product* defined in the *Product Plan* can be either purchased or supplied from the stock or produced. Particular form of the part delivery depends on the situation (whether it is in stock, can be purchased or must be produced). Therefore, in general ***any part of the product can be delivered any of possible ways***.

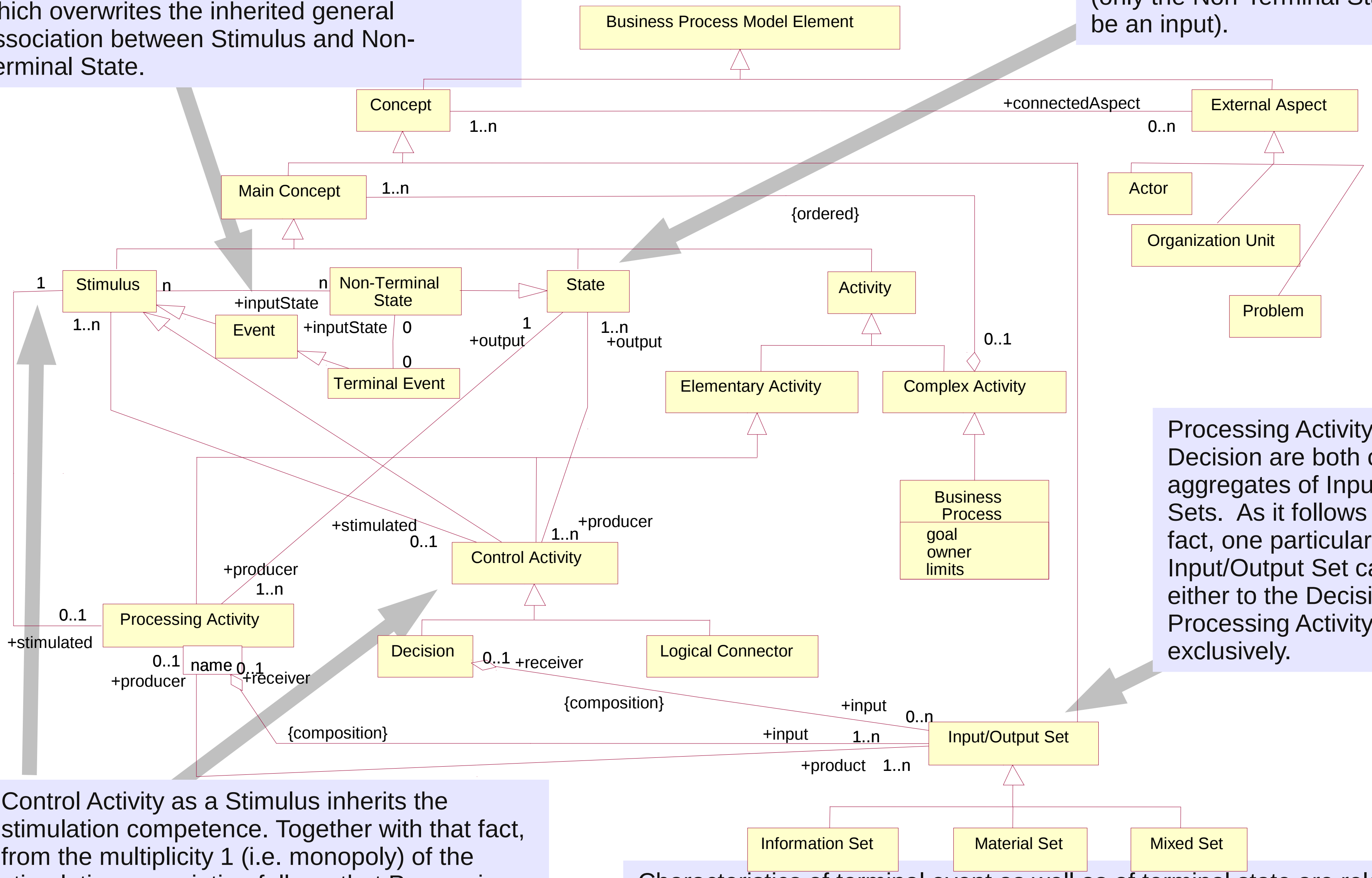
The event *All product parts prepared* is an ***abstract event*** which is in the reality represented by the fact that the last planned part of the product has been delivered.
This fact should be recognized by the process instance itself (by monitoring the deliveries, remembering the state of deliveries, and comparing it with the *Product Plan*).

In most cases in the reality it is not necessary to use more exact description.

Process meta-model

Each Stimulus has to have at least one input state except the first one (Terminal Event). Terminal Event has no input state. This exception is expressed by the specific zero multiplicity association with Terminal Event which overwrites the inherited general association between Stimulus and Non-Terminal State.

Each State has to be an input for at least one Stimulus except the last one (Terminal State), which has no succeeding activity (only the Non-Terminal State can be an input).



Processing Activity and Decision are both composite aggregates of Input/Output Sets. As it follows from that fact, one particular Input/Output Set can input either to the Decision or Processing Activity exclusively.

Control Activity as a Stimulus inherits the stimulation competence. Together with that fact, from the multiplicity 1 (i.e. monopoly) of the stimulation association follows that Processing Activity can be stimulated either by Event or by Control Activity exclusively.

Characteristics of terminal event as well as of terminal state are relative to the specified model. Usage of the model as a part (sub-process) of another model will change all terminal events of sub-process to regular ones and all terminal states to internal ones from the super-process point of view.