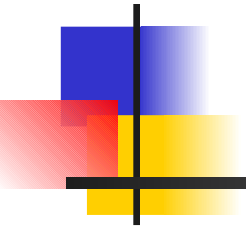
Workflow Concepts and Techniques



Hala Skaf-Molli

LORIA- INRIA LORRAINE

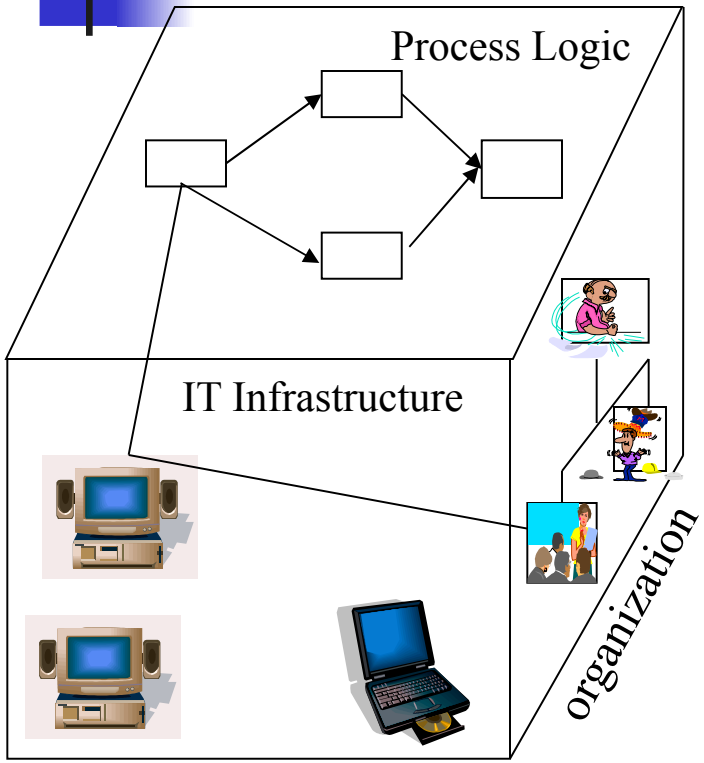
B032

University Henri Poincaré, Nancy1

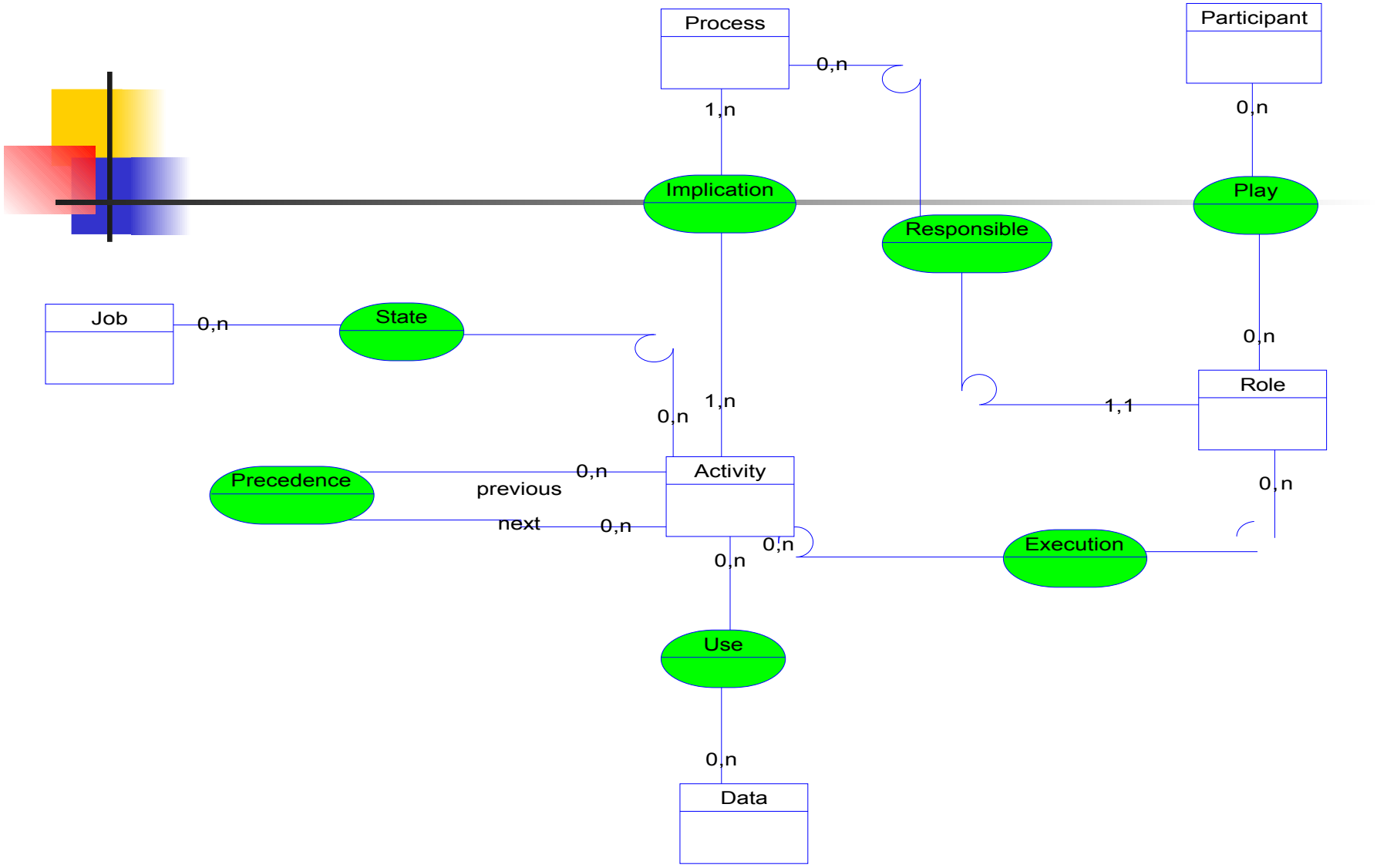
Skaf@loria.fr

http://www.loria.fr/~skaf

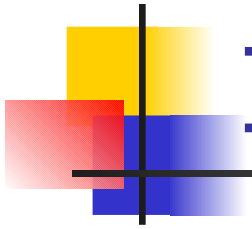
Dimensions of Workflow



- Three dimensions:
  - Process Logic
    - Which Activities ?
    - In which Order ?
  - Organization dimension
    - Who ?
  - IT infrastructure ..
    - What ?



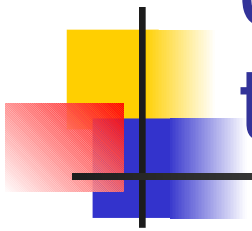
Workflow Conceptual Architecture



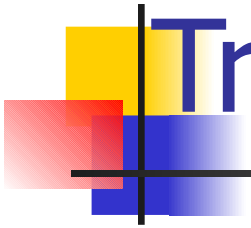
In practice

- Step 1:
  - Process Modeling
- Step 2:
  - Process Execution

There is a subtle but important distinction between the following terms:



- Task
  A logical step which may be executed for many cases.
- Work item = task + case
  A logical step which may be executed for a specific case.
- Activity = task + case + (resource) + (trigger)
  The actual execution of a task for a specific case.
- Work items and activities are task instances.

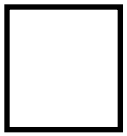


Triggers

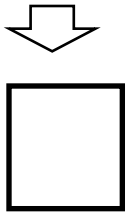
- The workflow system cannot force things to happen in reality:
  - The arrival of an electronic message (EDI) which is needed to execute a task.
  - A resource which starts to work on a case.
  - The arrival of a paper document.
  - A phone call to confirm a purchase order.
- A workflow system is a reactive system, i.e. it is triggered by the environment.
- Some tasks require a trigger.

We identify four kinds of tasks:

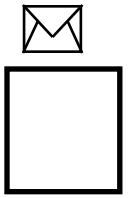
- **Automatic**
  No trigger is required.



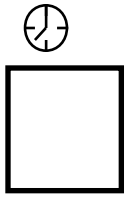
- **User**
  A resource takes the initiative.



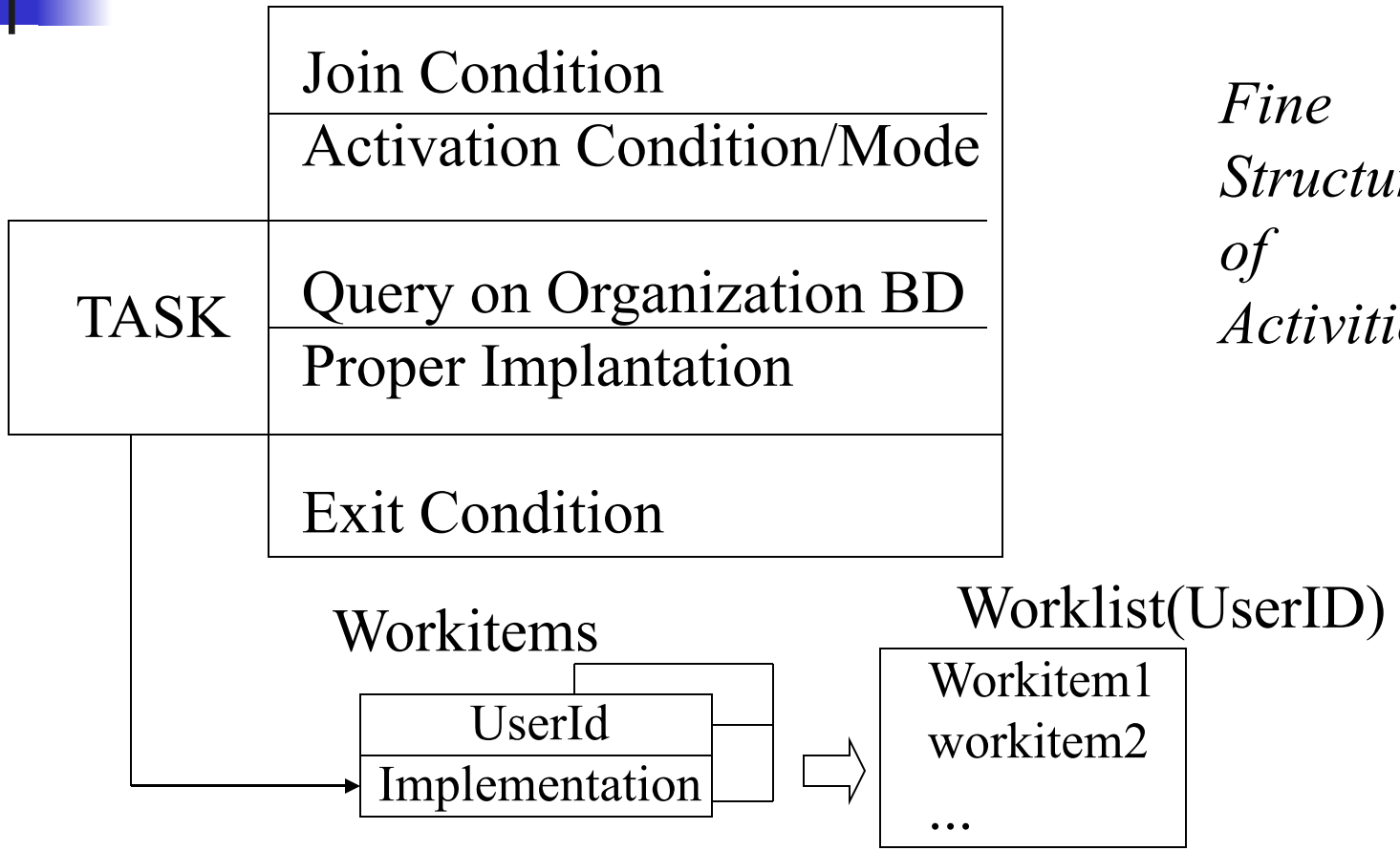
- **External**
  A external event (message, phone call) is required.



- **Time**
  The task requires a time trigger.



More About Activities



Fine Structure of Activities

Running Activities

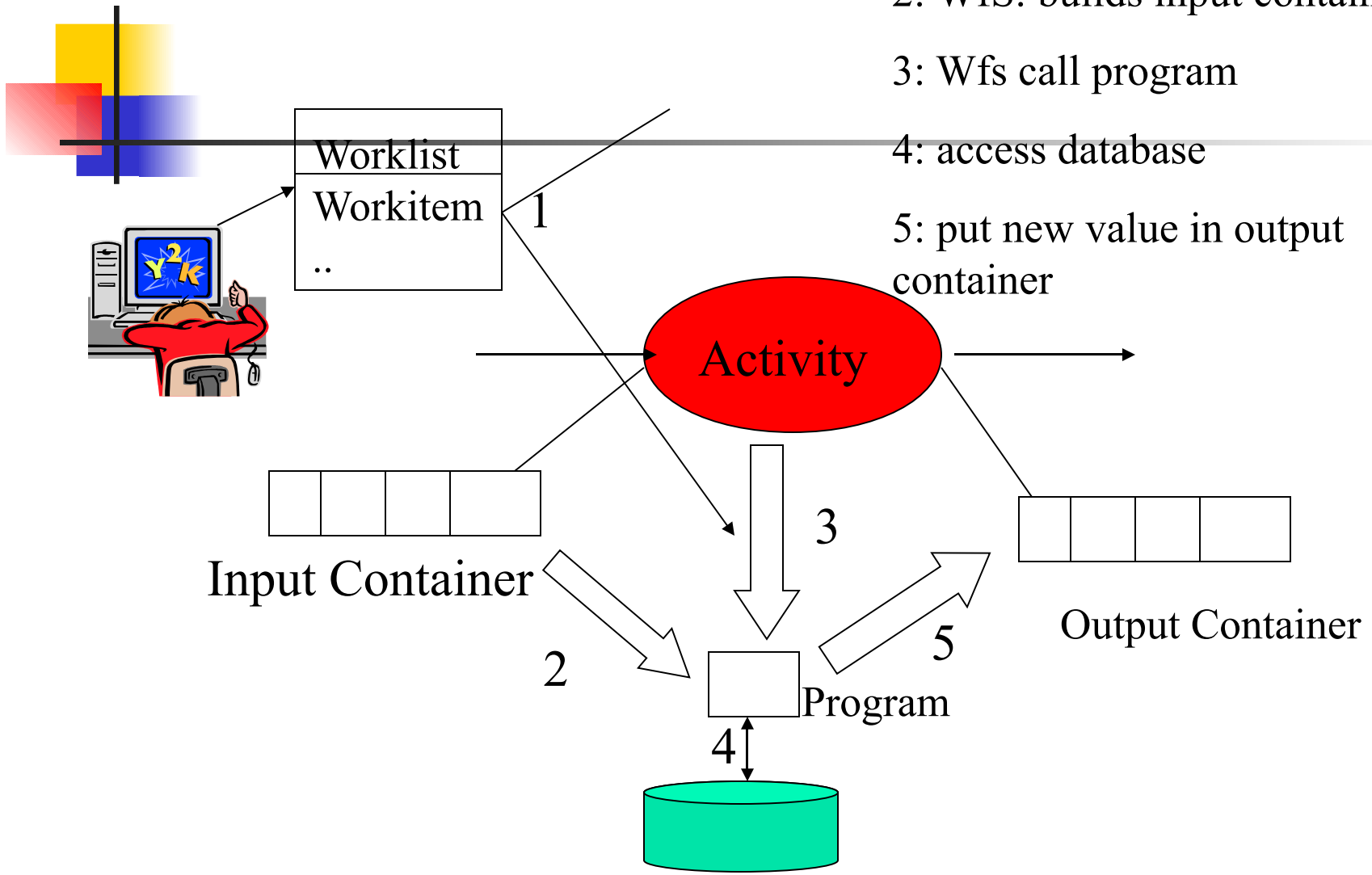
1: user click (manual activity)

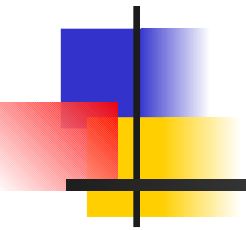
2: WfS: builds input container

3: Wfs call program

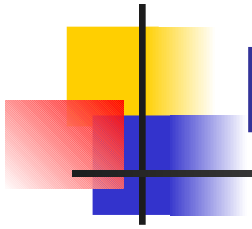
4: access database

5: put new value in output container





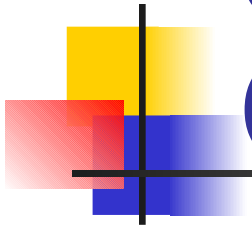
Process Modeling



Petri nets as a basis

- There are many modeling techniques and tools
  - BPEL, BPMN, DFD, ISAC, SADT, PN, HLPN, PA, FC, UML, ...
  - Simulation tools, design tools, CASE tools, WFMS, Focus on the essential concepts rather than (system-)specific languages.
- Workflow modeling in terms of workflow nets (a subset of Petri nets).

Classical Petri net (www.workflowcourse.com)



- Simple process model
  - Just three elements: **places**, **transitions** and **arcs**.
  - Graphical and mathematical description.
  - Formal semantics and allows for analysis.
- History:
  - Carl Adam Petri (1962, PhD thesis)
  - In sixties and seventies focus mainly on theory.
  - "Hidden" in many diagramming techniques and systems.

Elements

(name)

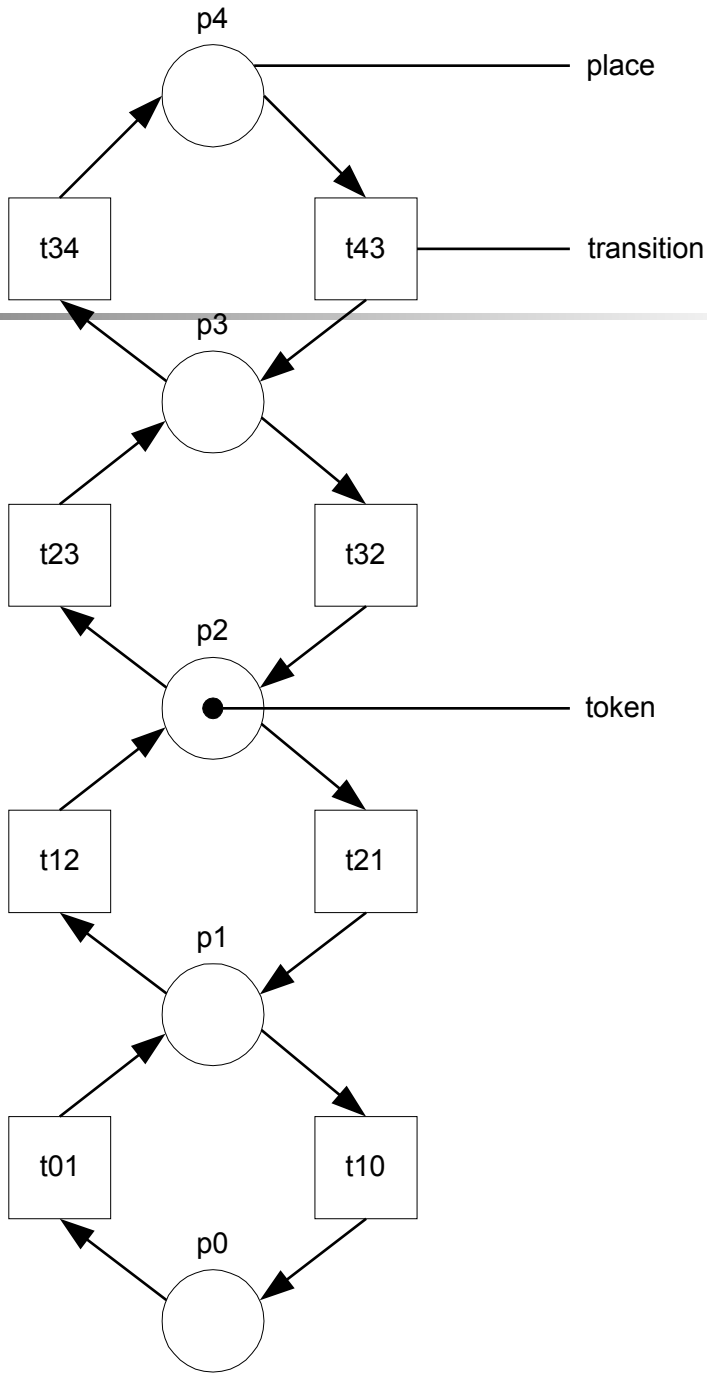
place

(name)

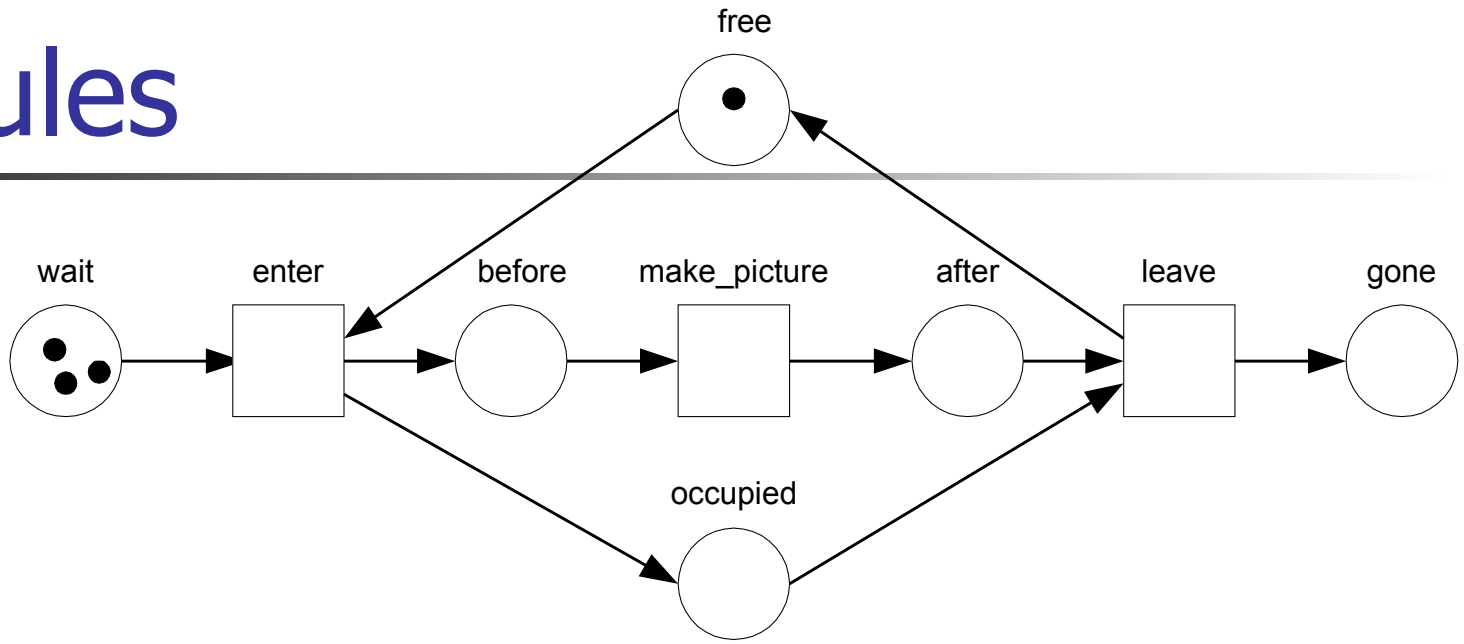
transition

arc (directed connection)

token



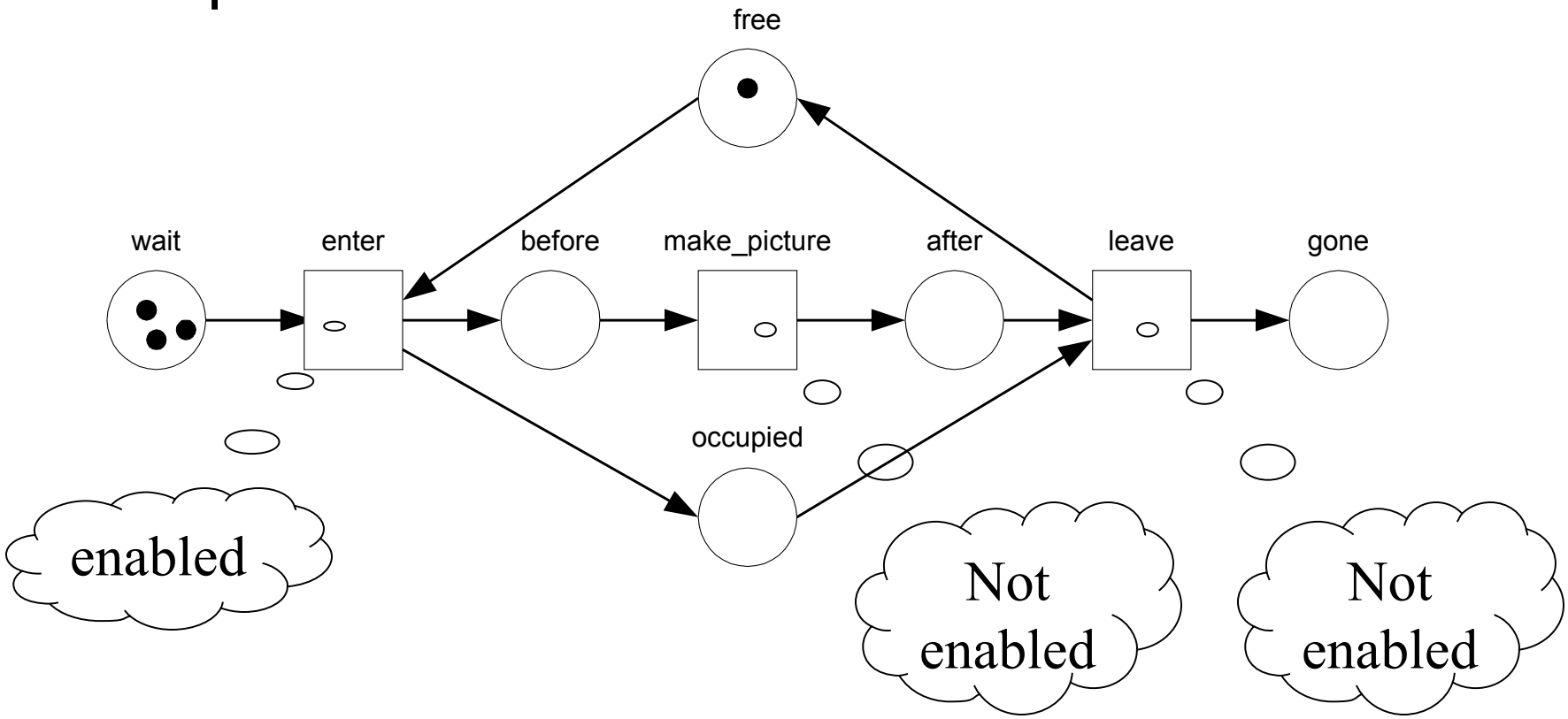
Rules



- Connections are directed.
- No connections between two places or two transitions.
- Places may hold zero or more tokens.

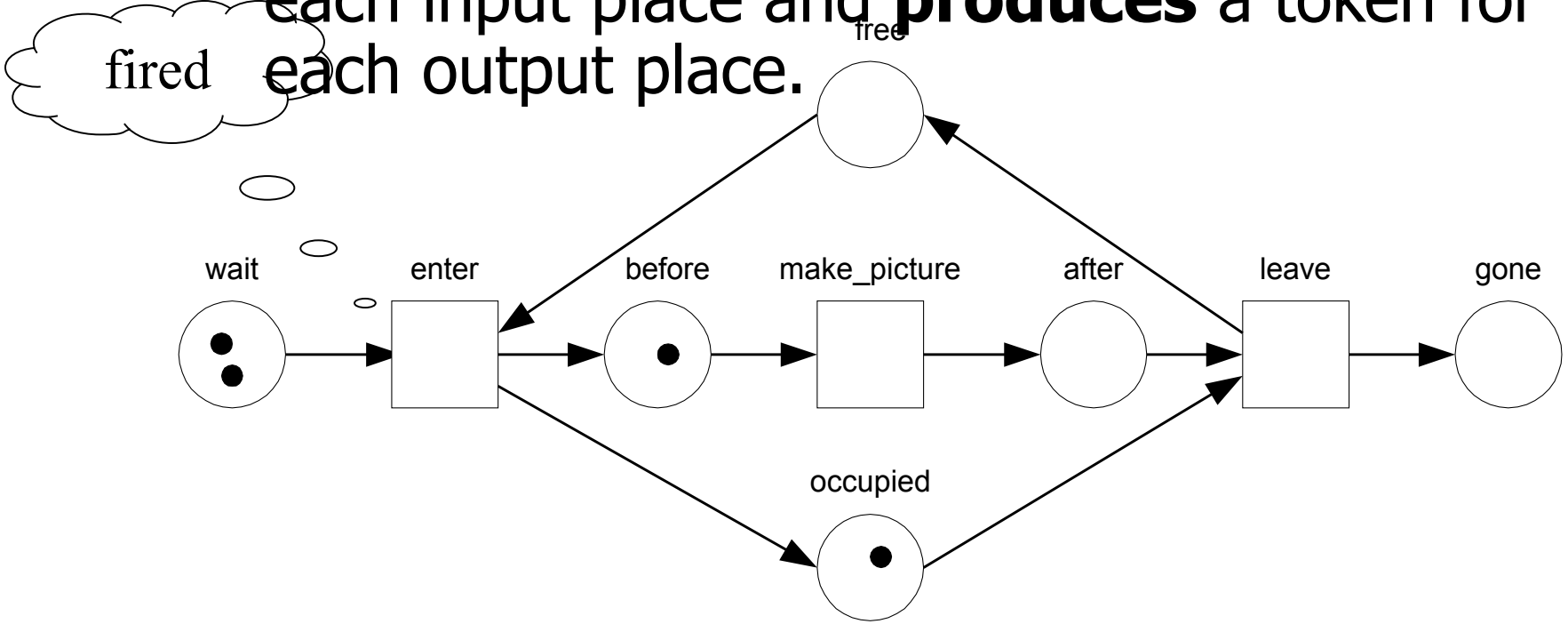
Enabled

- A transition is **enabled** if each of its input places contains at least one token.



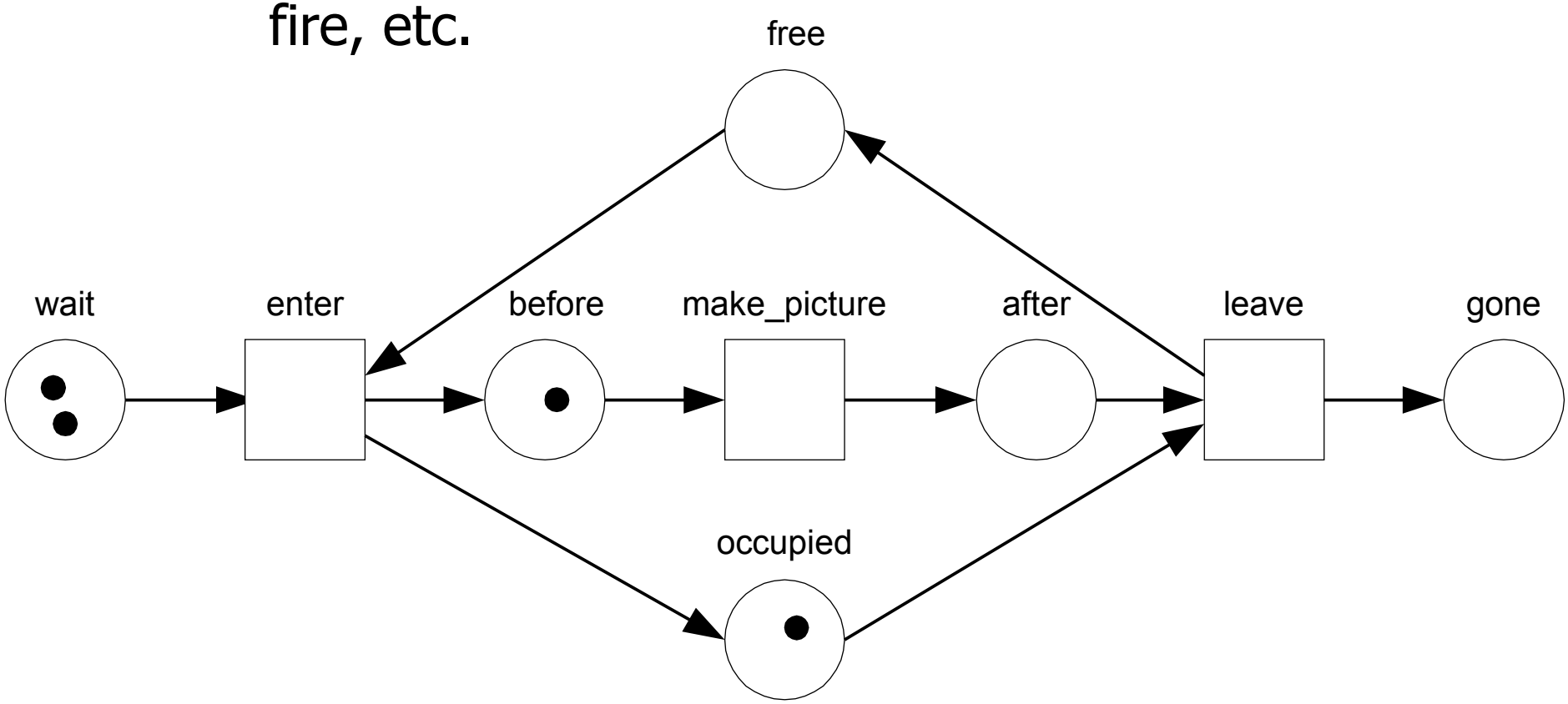
Firing

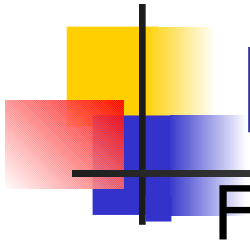
- An **enabled** transition can **fire** (i.e., it occurs).
- When it **fires** it **consumes** a token from each input place and **produces** a token for each output place.



Play "Token Game"

- In the new state, *make_picture* is enabled. It will fire, etc.

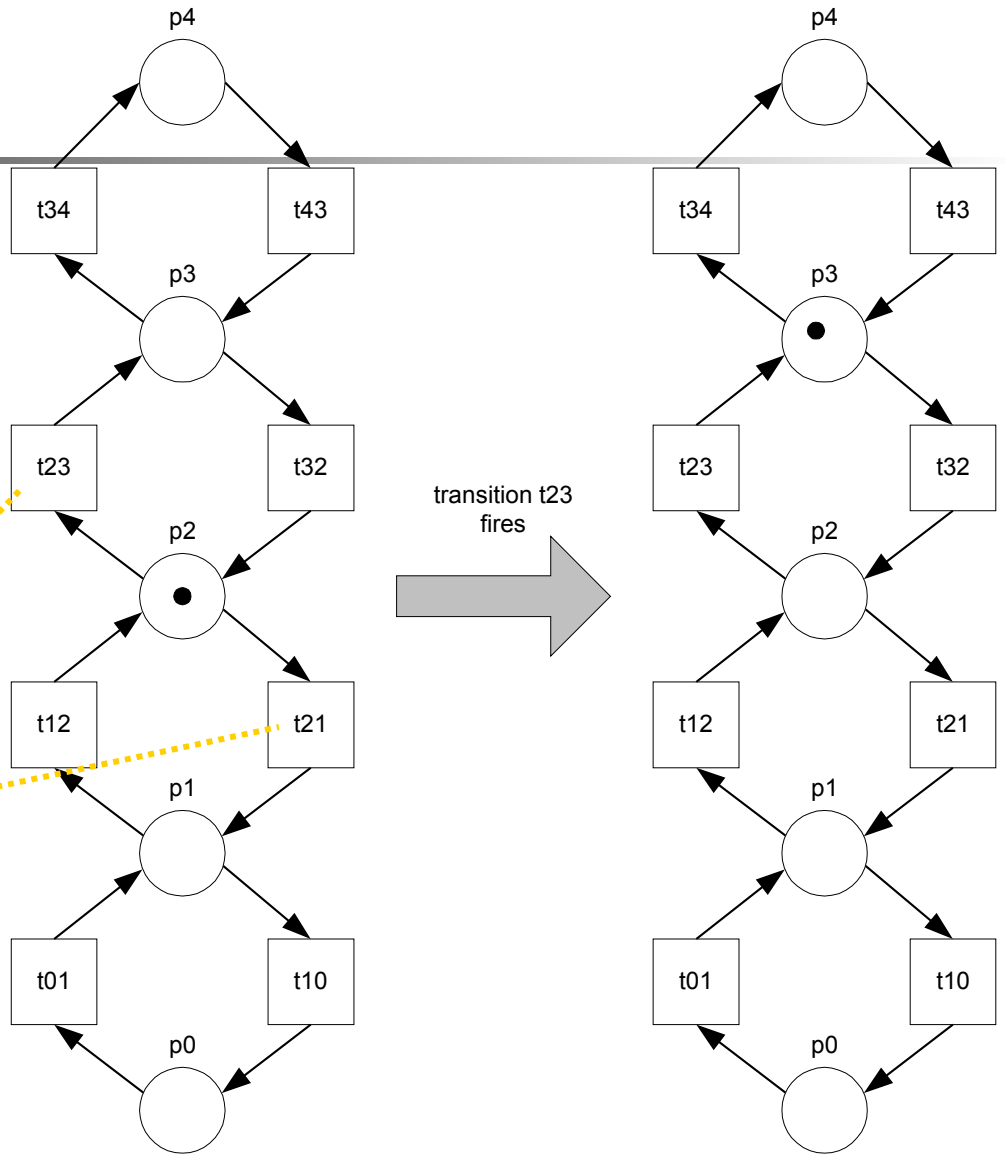




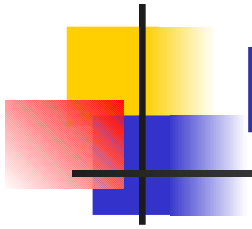
Remarks

- Firing is **atomic**.
- The number of tokens may vary if there are transitions for which the number of input places is not equal to the number of output places.
- The network is static.
- The **state** is represented by the distribution of tokens over places (also referred to as **marking**).
- Multiple transitions may be enabled, but only one fires at a time

Non-determinism



Two transitions are enabled but only one can fire

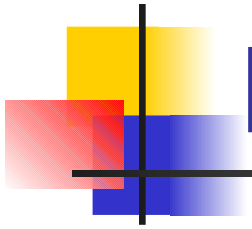


Formal definition

A classical Petri net is a four-tuple (P,T,I,O) where:

- P is a finite set of places,
- T is a finite set of transitions,
- I : P x T -> **N** is the input function, and
- O : T x P -> **N** is the output function.

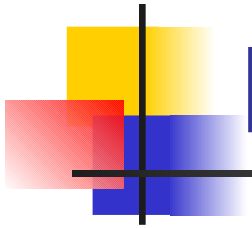
Any diagram can be mapped onto such a four tuple and vice versa.



Formal definition (2)

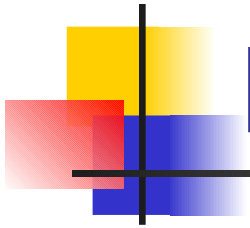
The state (marking) of a Petri net (P,T,I,O) is defined as follows:

- s: P-> **N**, i.e., a function mapping the set of places onto {0,1,2, … }.



Modeling

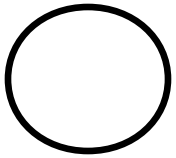
- Place: passive element
- Transition: active element
- Arc: causal relation
- Token: elements subject to change



Role of a token

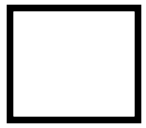
- Tokens can play the following roles:
  - a physical object, for example a product, a part, a drug, a person;
  - an information object, for example a message, a signal, a report;
  - an indicator of a condition: the presence of a token indicates whether a certain condition is fulfilled.

Role of a place



- a type of **communication medium**, like a telephone line, a middleman, or a communication network;
- a **buffer**: for example, a depot, a queue or a post bin;
- a **geographical location**, like a place in office or hospital;
- a possible **state or state condition**: for example, the floor where an elevator is, or the condition that a specialist is available.

Role of a transition

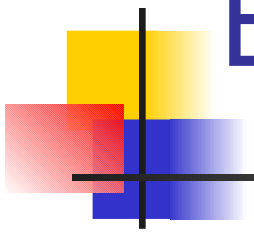


- an **event**: for example, starting an operation, the death of a patient, a change seasons or the switching of a traffic light from red to green;
- a **transformation of an object**, like adapting a product, updating a database, or updating a document;
- a **transport of an object**: for example, transporting goods, or sending a file.

High level Petri Net

- Limitation of classical Petri net:
  - Complexe
- Three extenions:
  - Color
  - Time
  - Hierarchy

Extenion with color

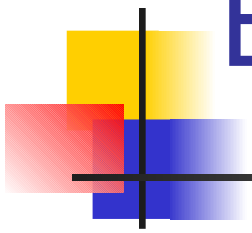


- Each token has a value (color) with refers to specific features of the object modeled by the token

Extenion with time

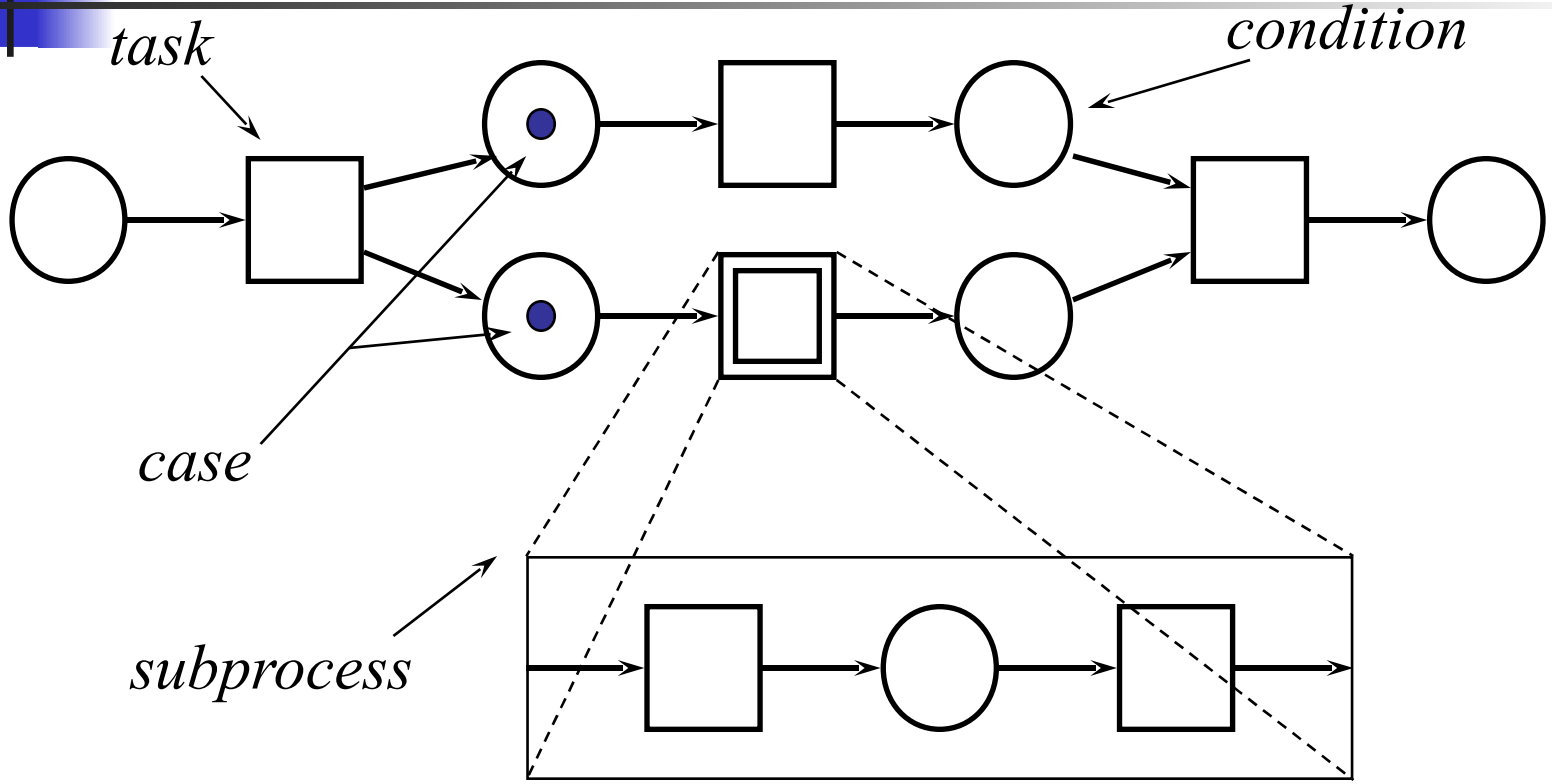
- For performance analysis : durations, delays, etc
- Each token has a **timpestamp** and transitions determine the **delay** of a produced token

Extension with hierarchy

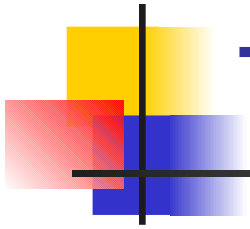


- To structure complex Petri nets

Mapping a process definition into Petri nets



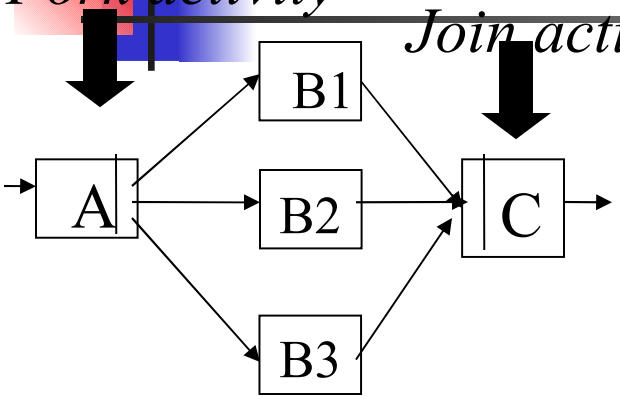
case variables



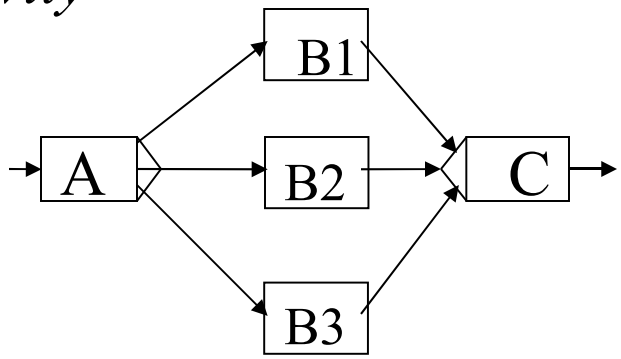
Typical network structures

- Causality (sequential routing)
- Parallelism (AND-split - AND-join)
- Choice (XOR-split – XOR-join)
- Iteration (XOR-join - XOR-split)
- Capacity constraints
  - Feedback loop
  - Mutual exclusion
  - Alternating

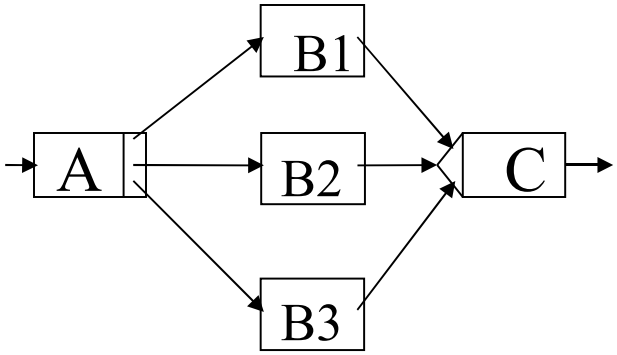
Activity and Flow Control



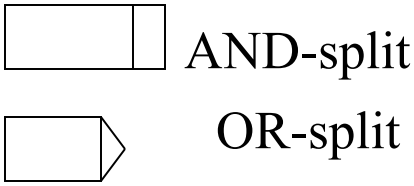
Parallel Processing



Conditional Branching



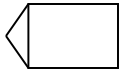
Parallel Branching with final selection



AND-split

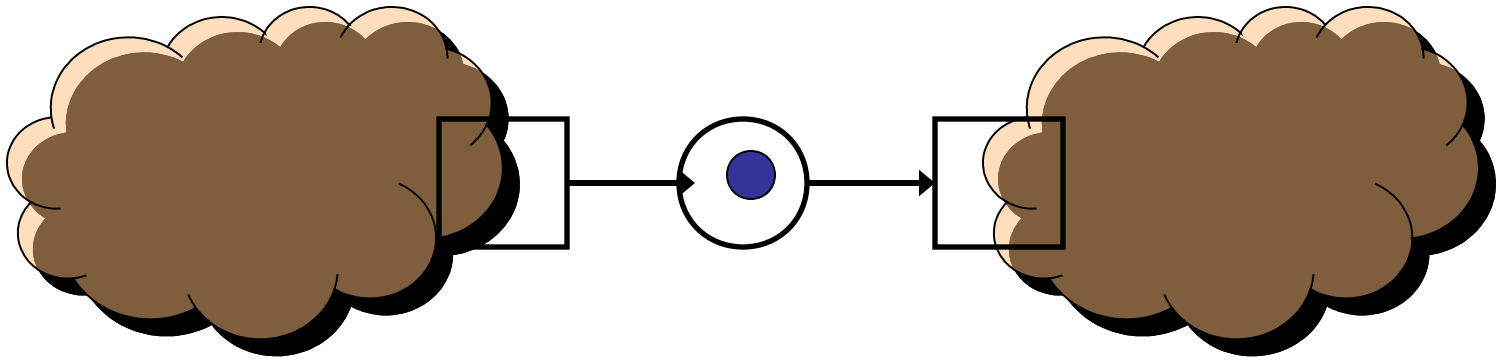
OR-split

AND-join

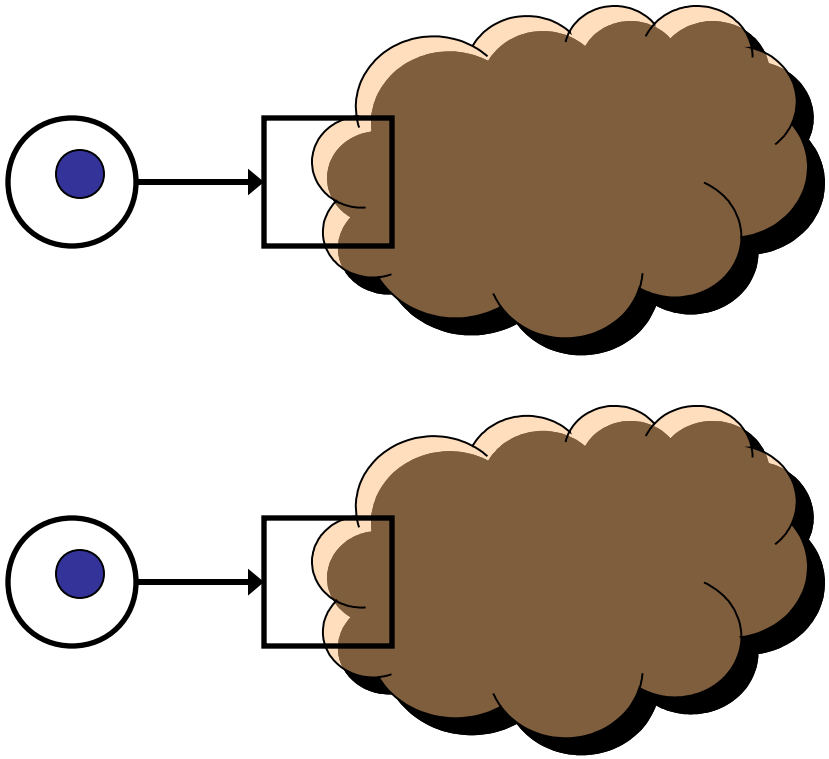


OR-join

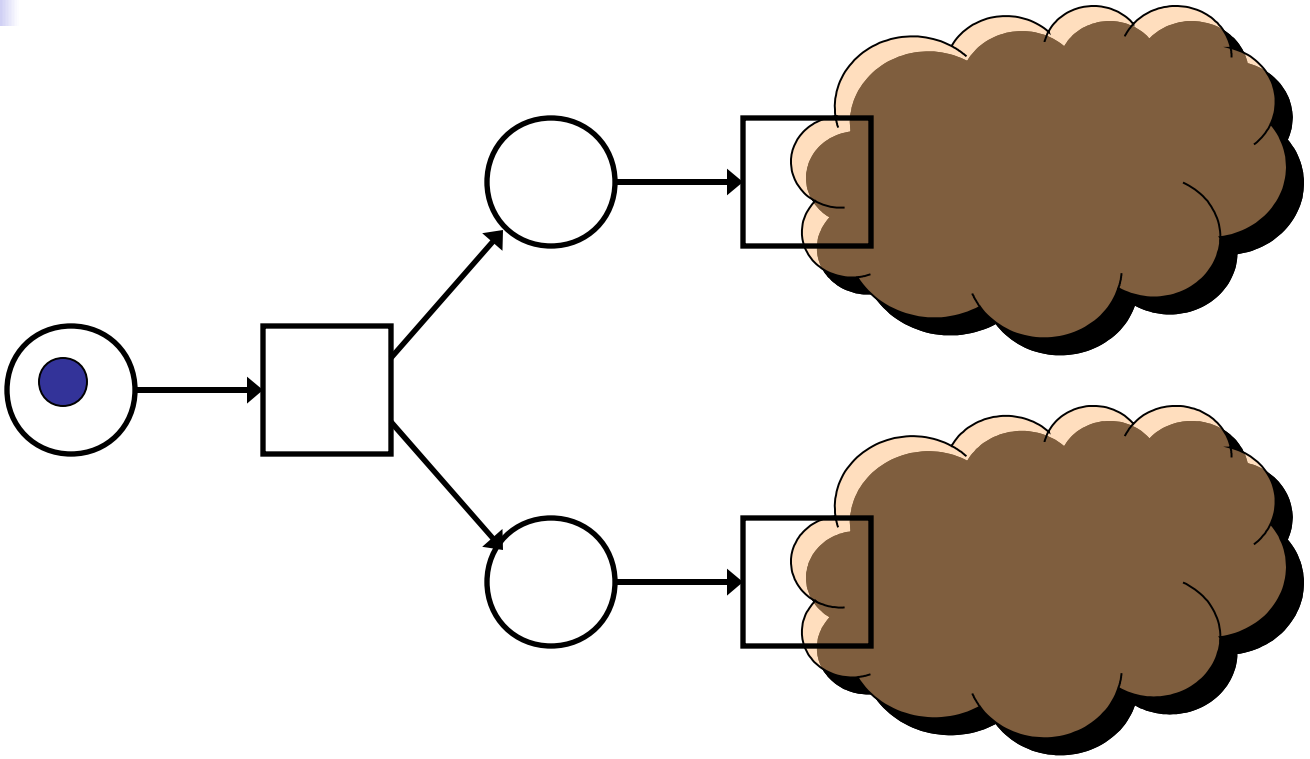
Causality



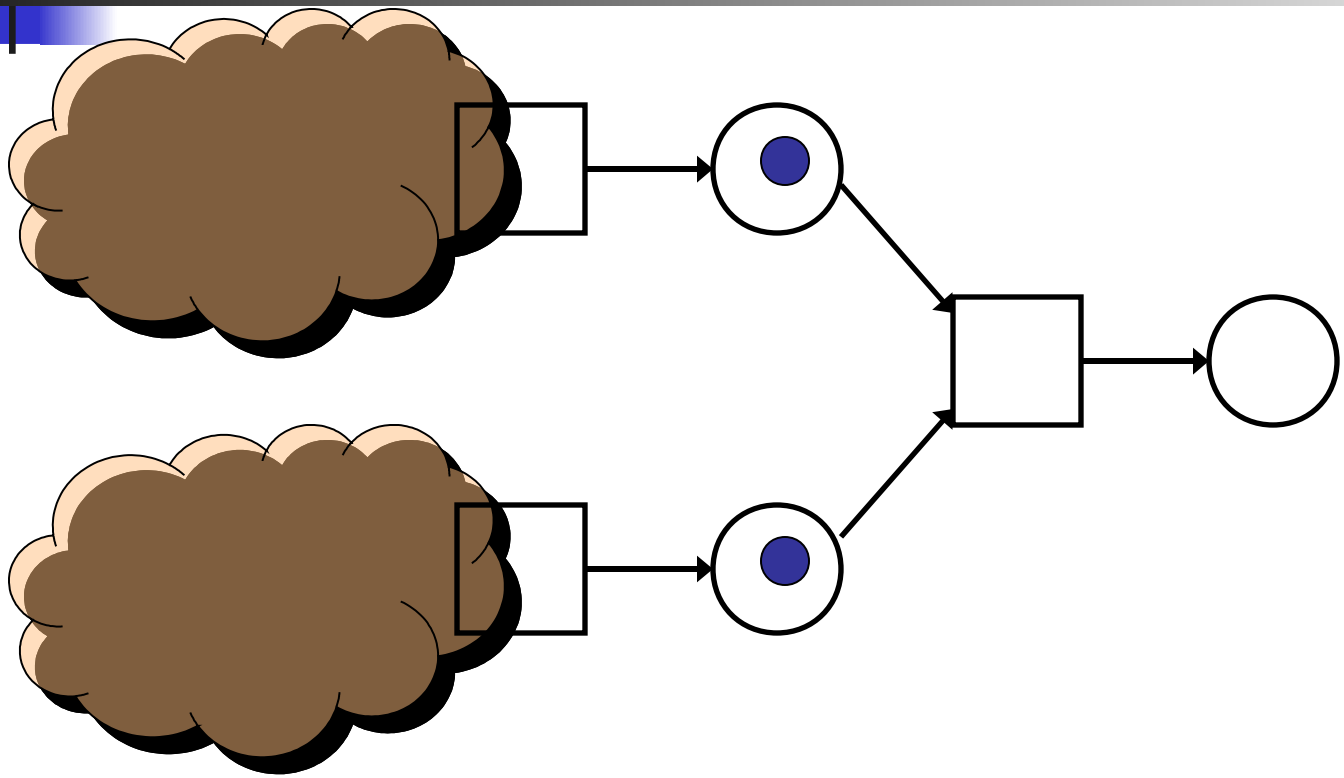
Parallelism



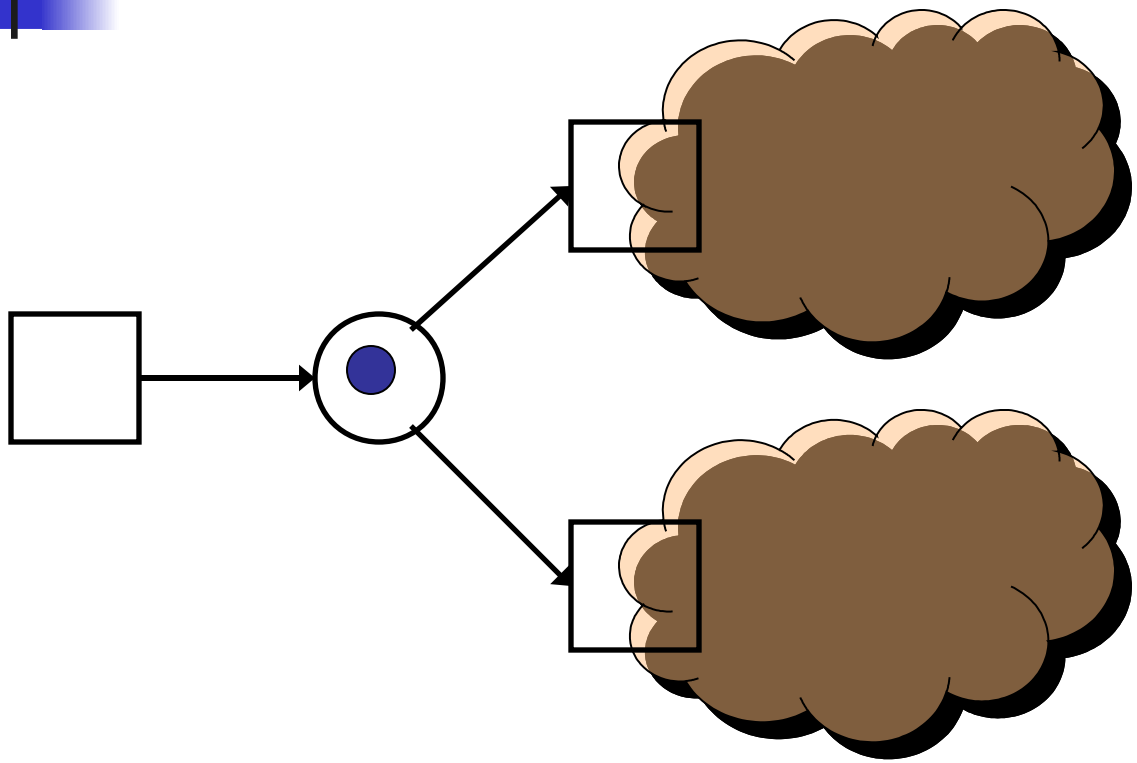
Parallelism: AND-split



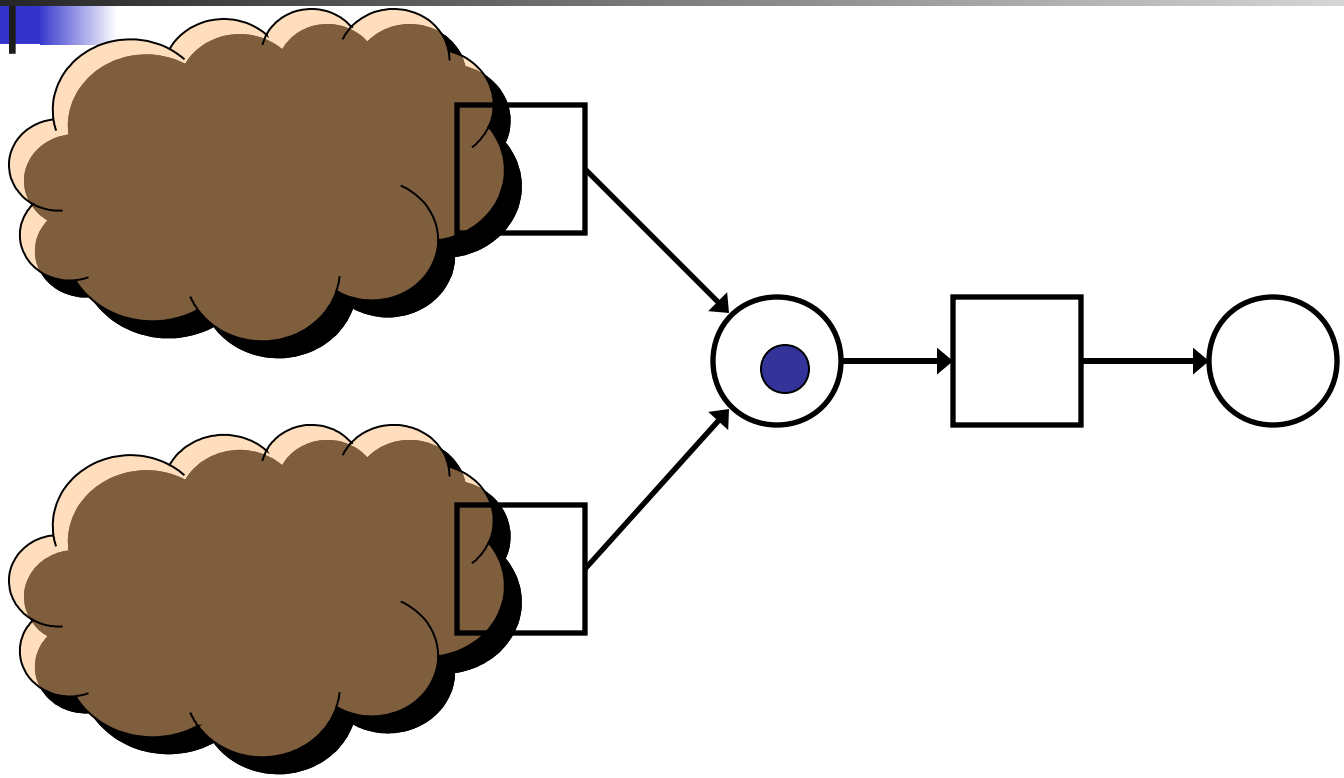
Parallelism: AND-join



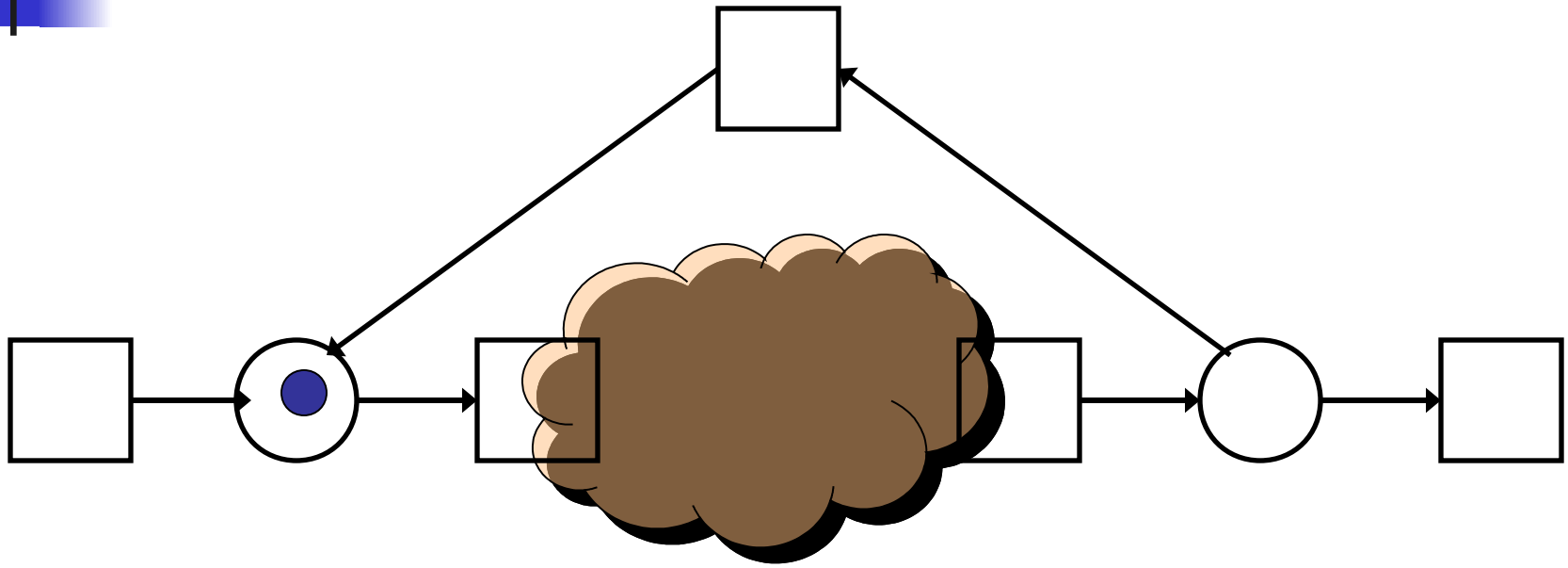
Choice: XOR-split



Choice: XOR-join

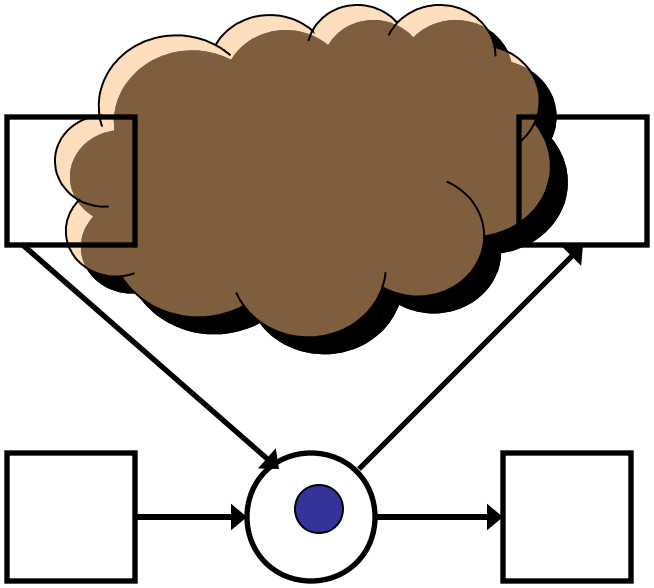


Iteration: 1 or more times



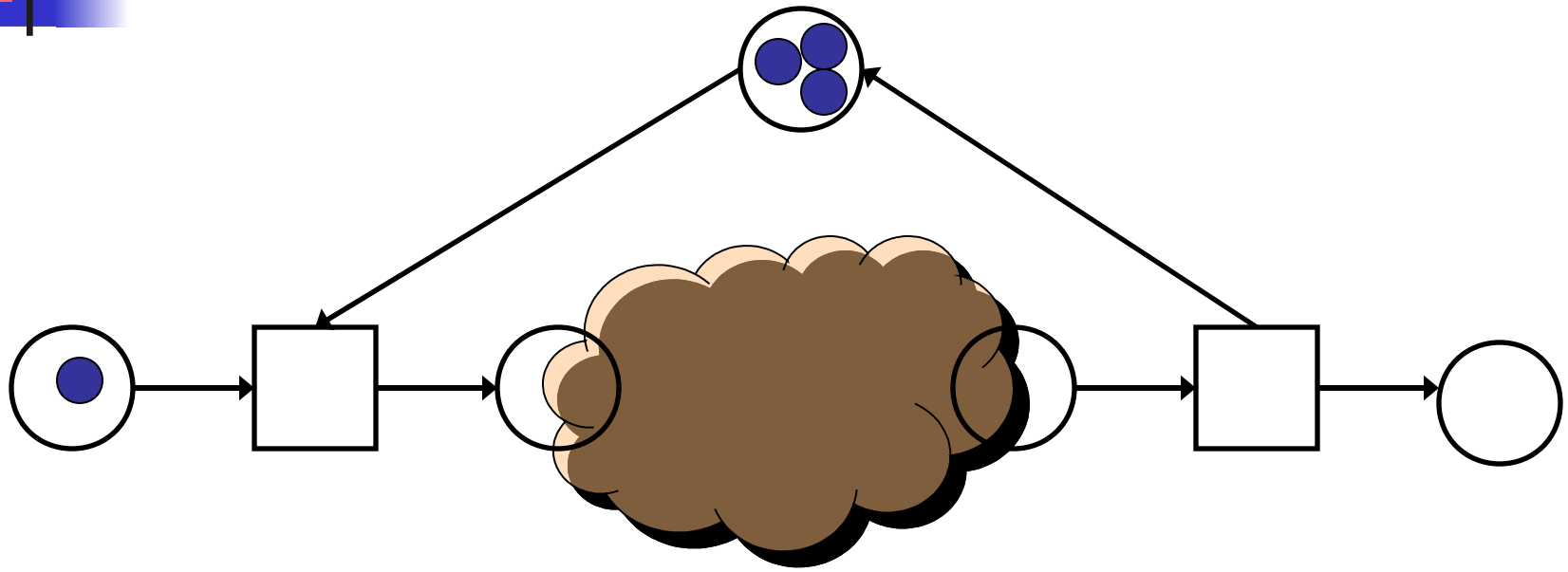
XOR-join before XOR-split

Iteration: 0 or more times



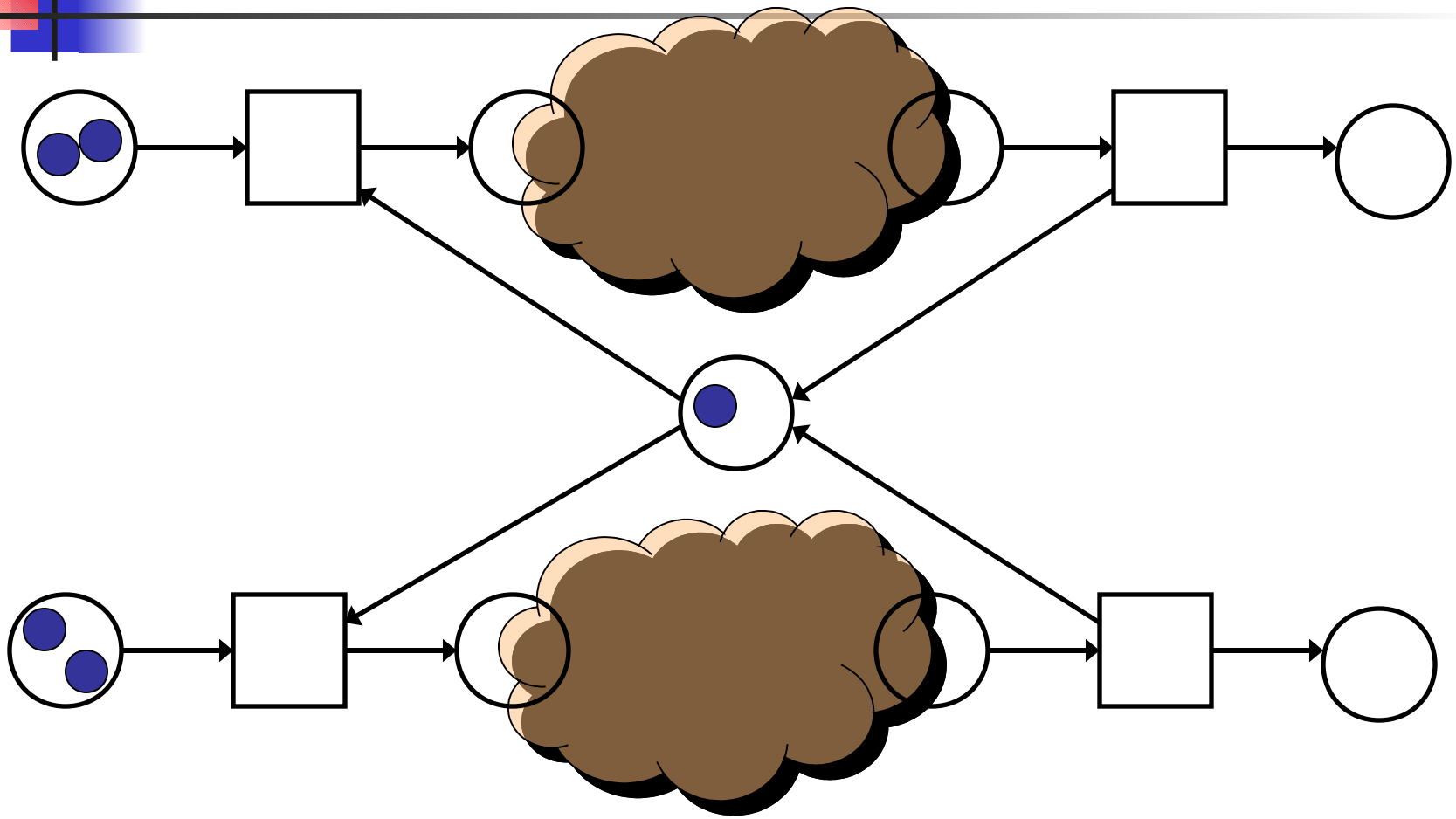
XOR-join before XOR-split

Capacity constraints: feedback loop



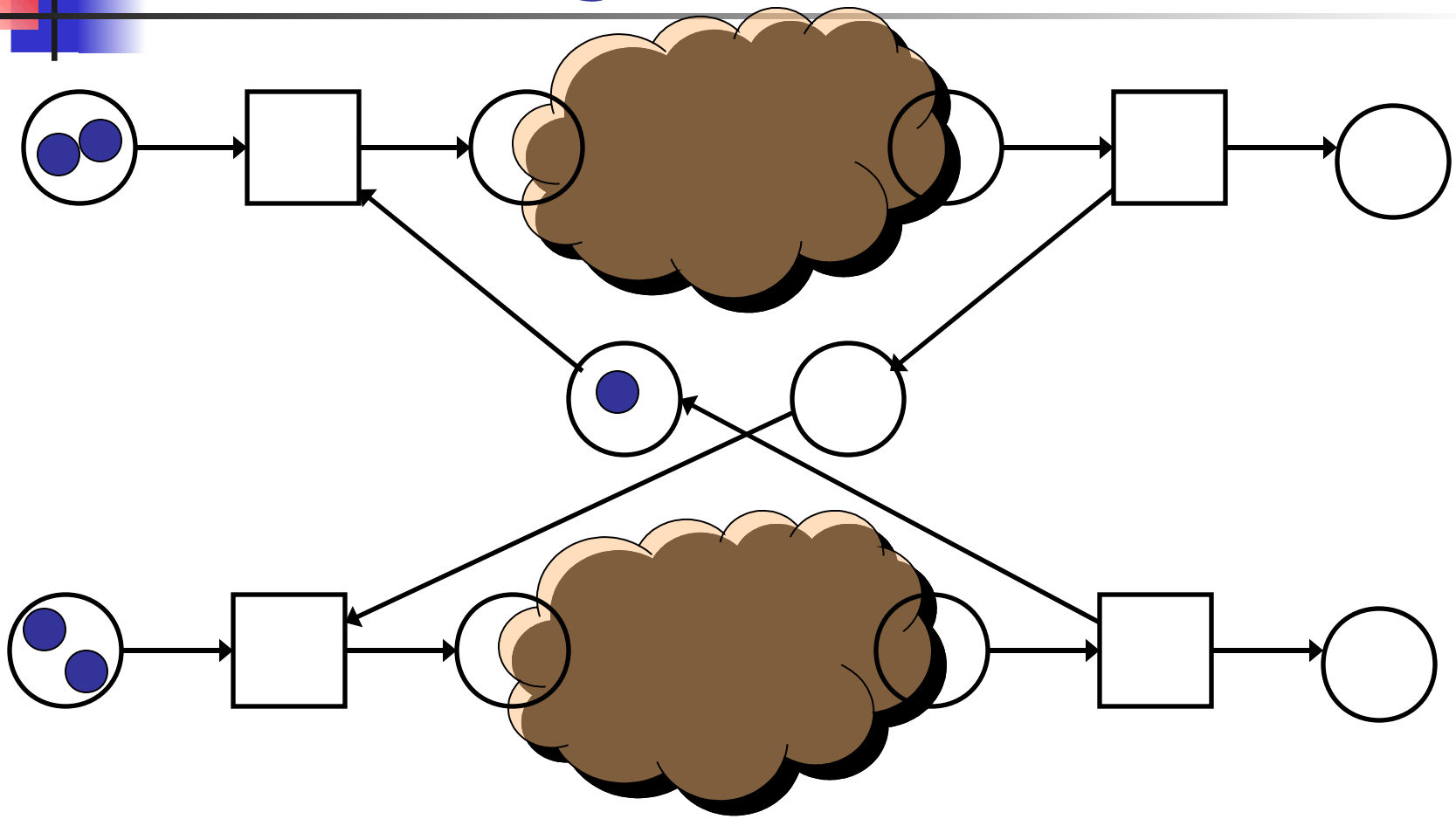
AND-join before AND-split

Capacity constraints: mutual exclusion



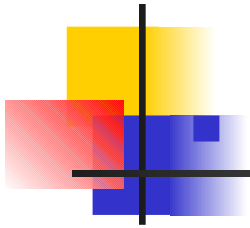
AND-join before AND-split

Capacity constraints: alternating



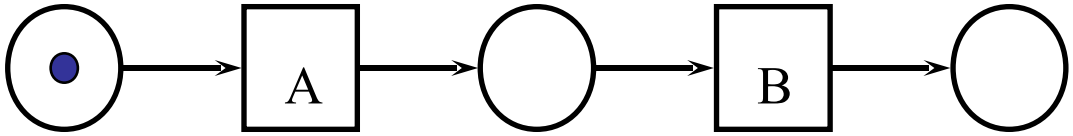
AND-join before AND-split

Activities



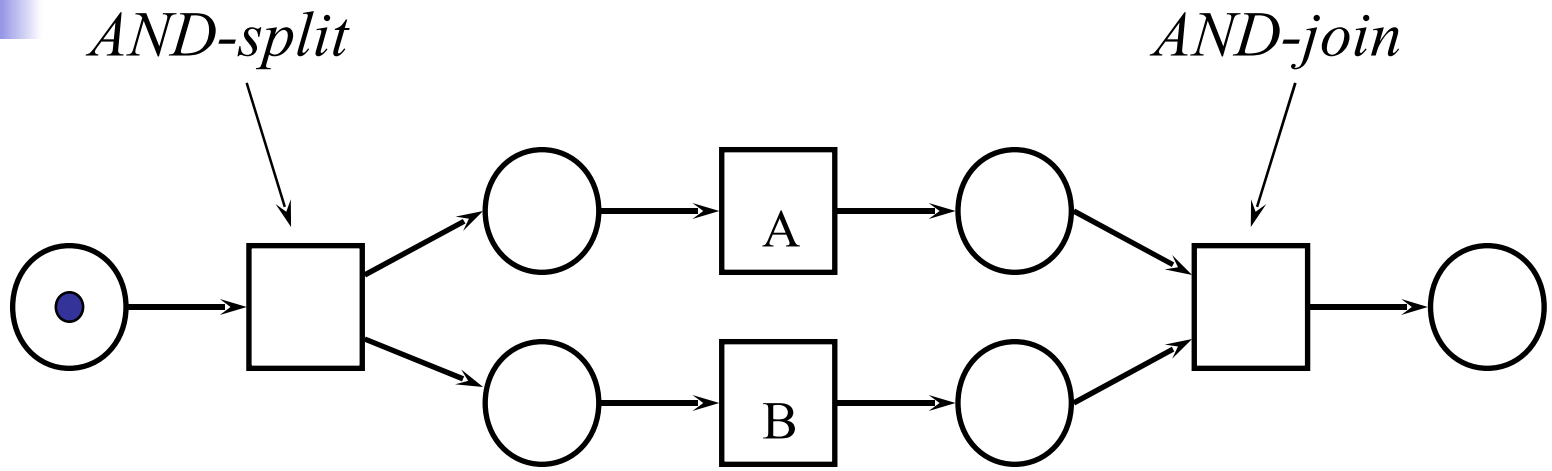
- Sequential
  "first A then B"
- Parallel
  "A and B at the same time or in any order"
  - AND-split
  - AND-join
- Choice
  "A or B"
  - OR-split
  - OR-join
- Iteration
  "multiple A's"

Sequential routing



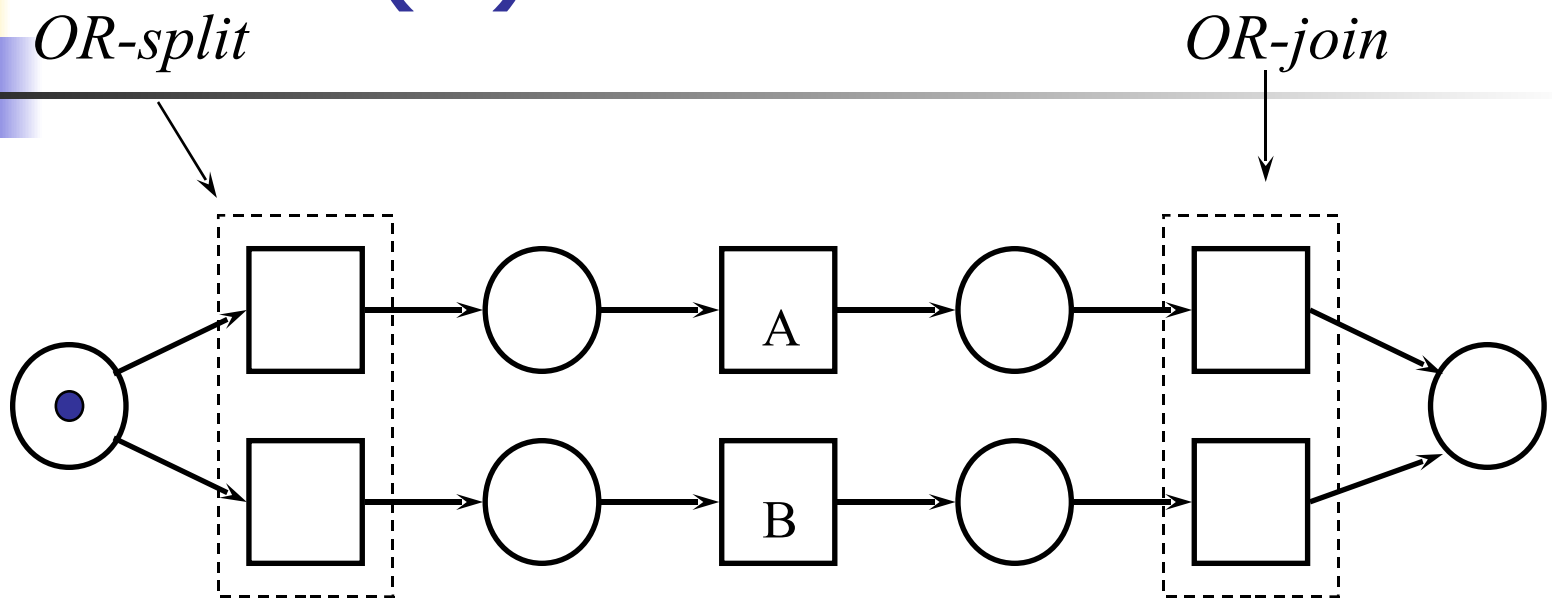
"First A then B"

Parallel routing



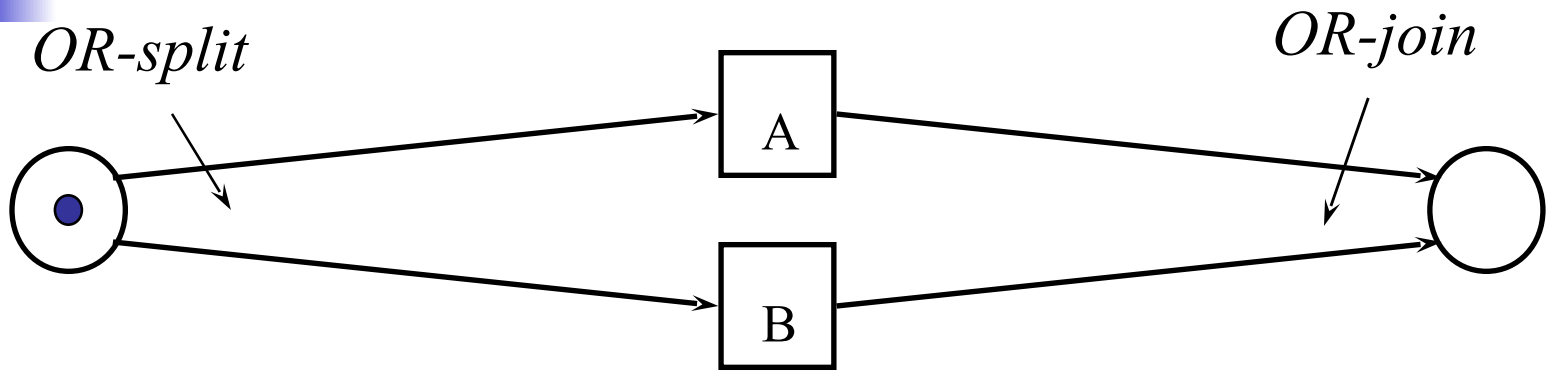
"A and B at the same time or in any order"

Choice (1)



"A or B"

Choice (2)

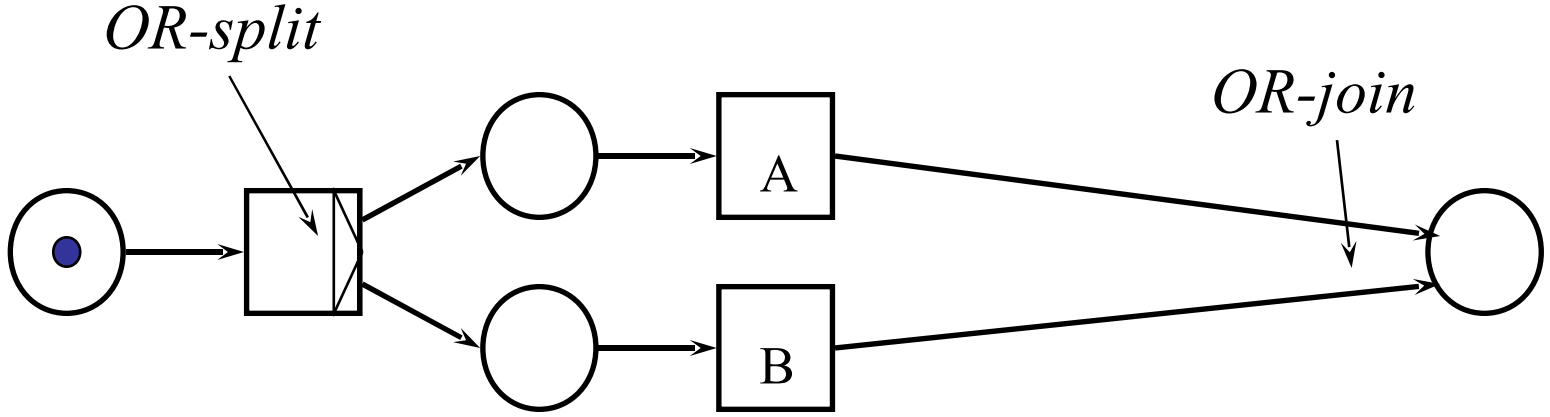


Implicit choice: it depends on the "eagerness" of A and B!

Choice (3)

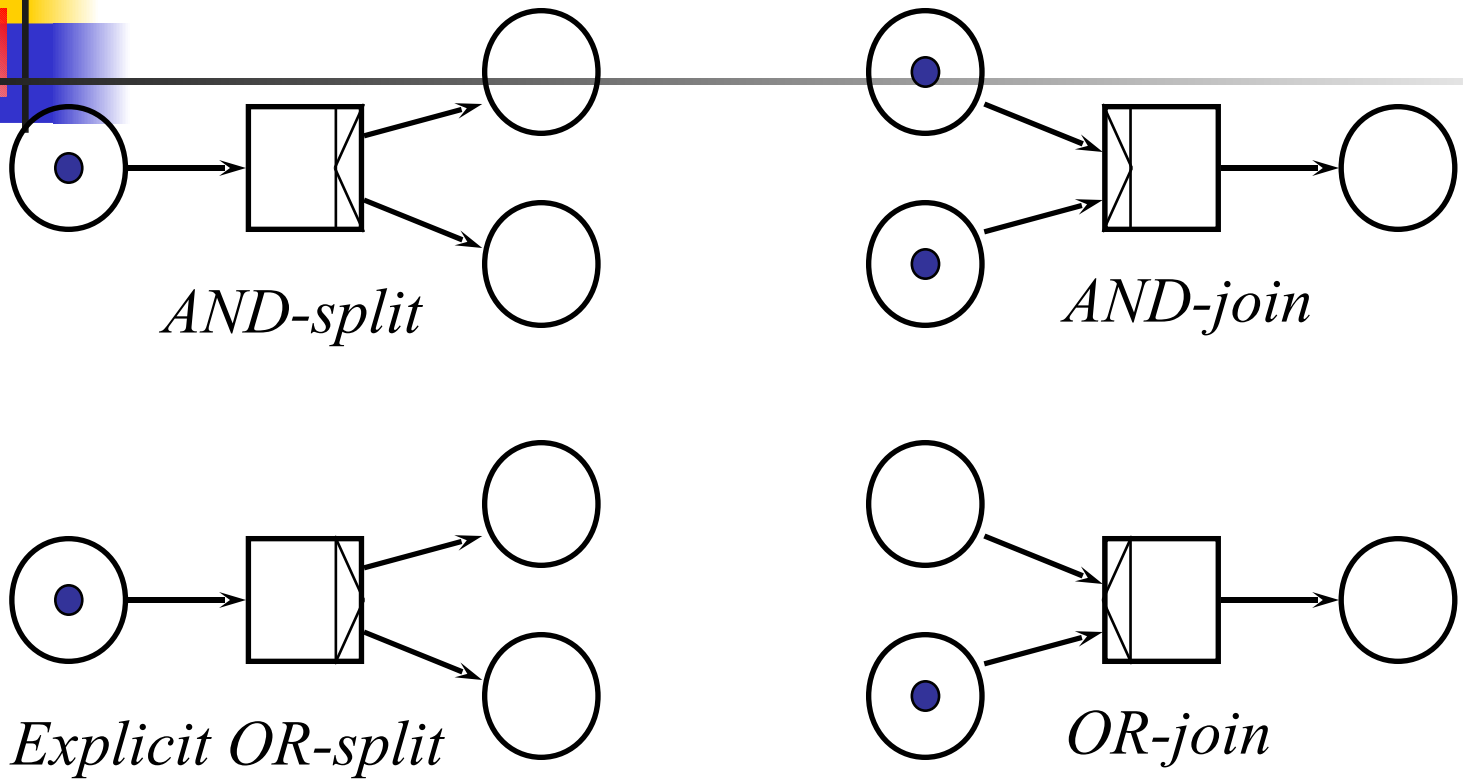
We use high-level Petri nets:

- tokens have values: case variables
- transitions determine the number of tokens produced: explicit OR-split

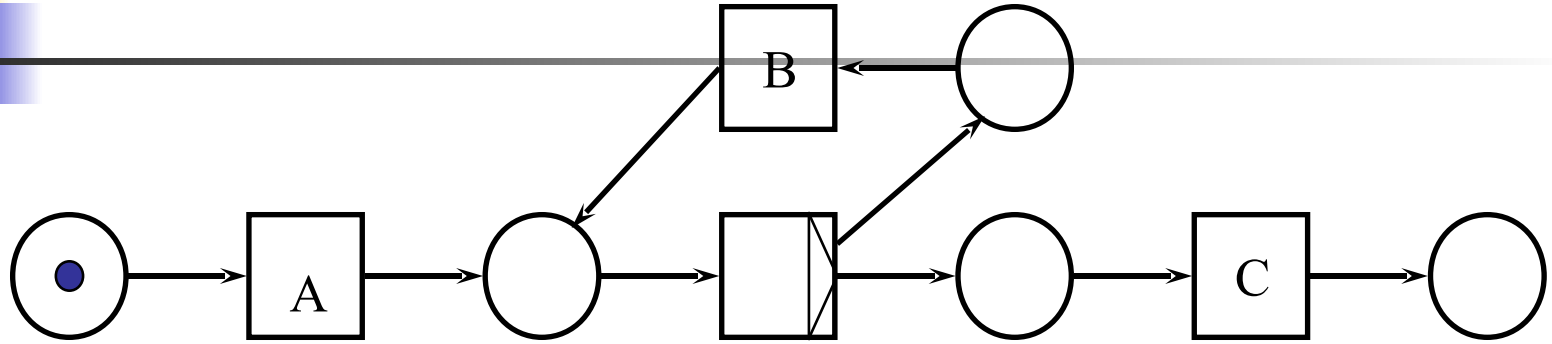


Choice is explicit and may be based on logistic attributes!

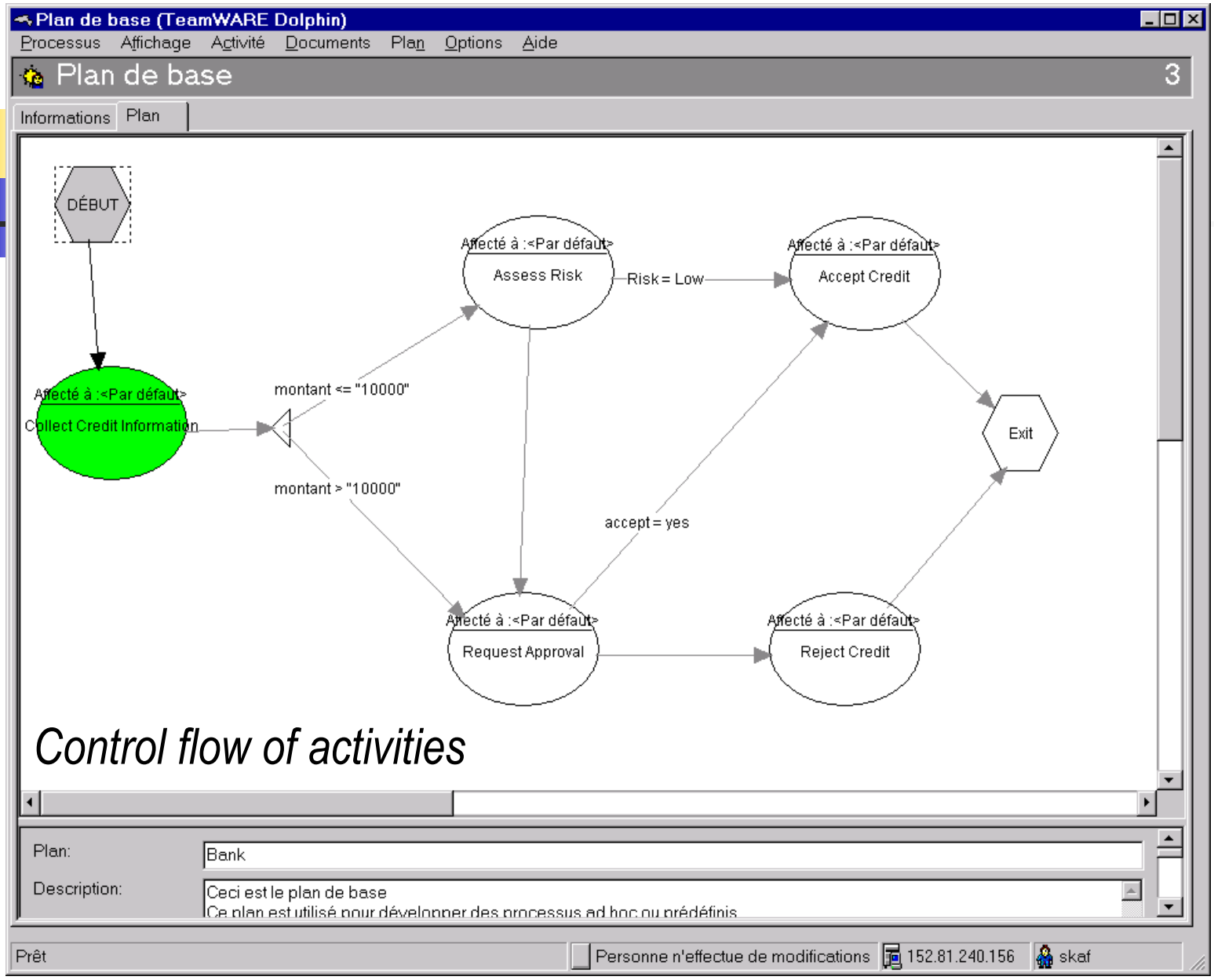
Syntactic sugaring

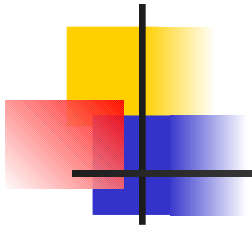


Iteration



B may be executed several times.

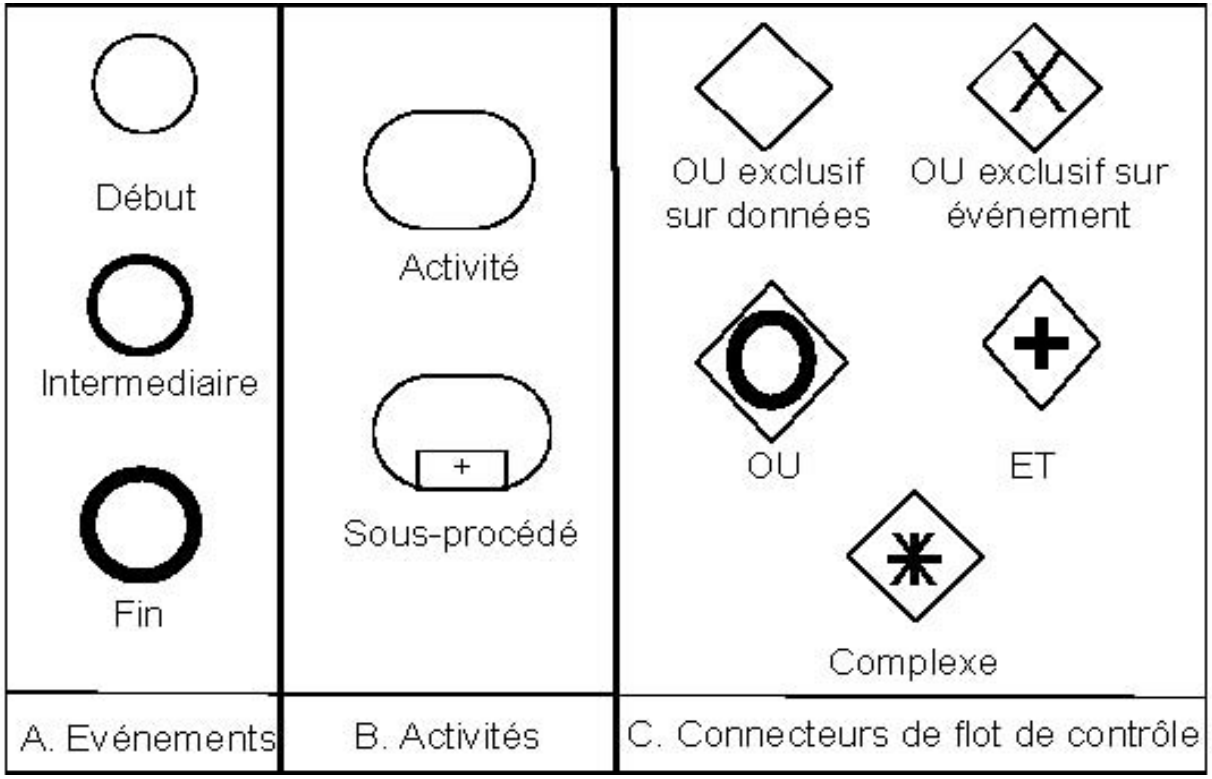

Plan de base (TeamWARE Dolphin)
Processus Affichage Activité Documents Plan Options Aide
Plan de base 3
Informations Plan
DÉBUT
Affecté à :<Par défaut>
Collect Credit Information
montant <= "10000"
montant > "10000"
Affecté à :<Par défaut>
Assess Risk
Risk = Low
Affecté à :<Par défaut>
Accept Credit
Exit
accept = yes
Affecté à :<Par défaut>
Request Approval
Affecté à :<Par défaut>
Reject Credit
Control flow of activities
Plan: Bank
Description: Ceci est le plan de base
Ce plan est utilisé pour développer des processus ad hoc ou prédéfinis
Prêt
Personne n'effectue de modifications
152.81.240.156
skaf



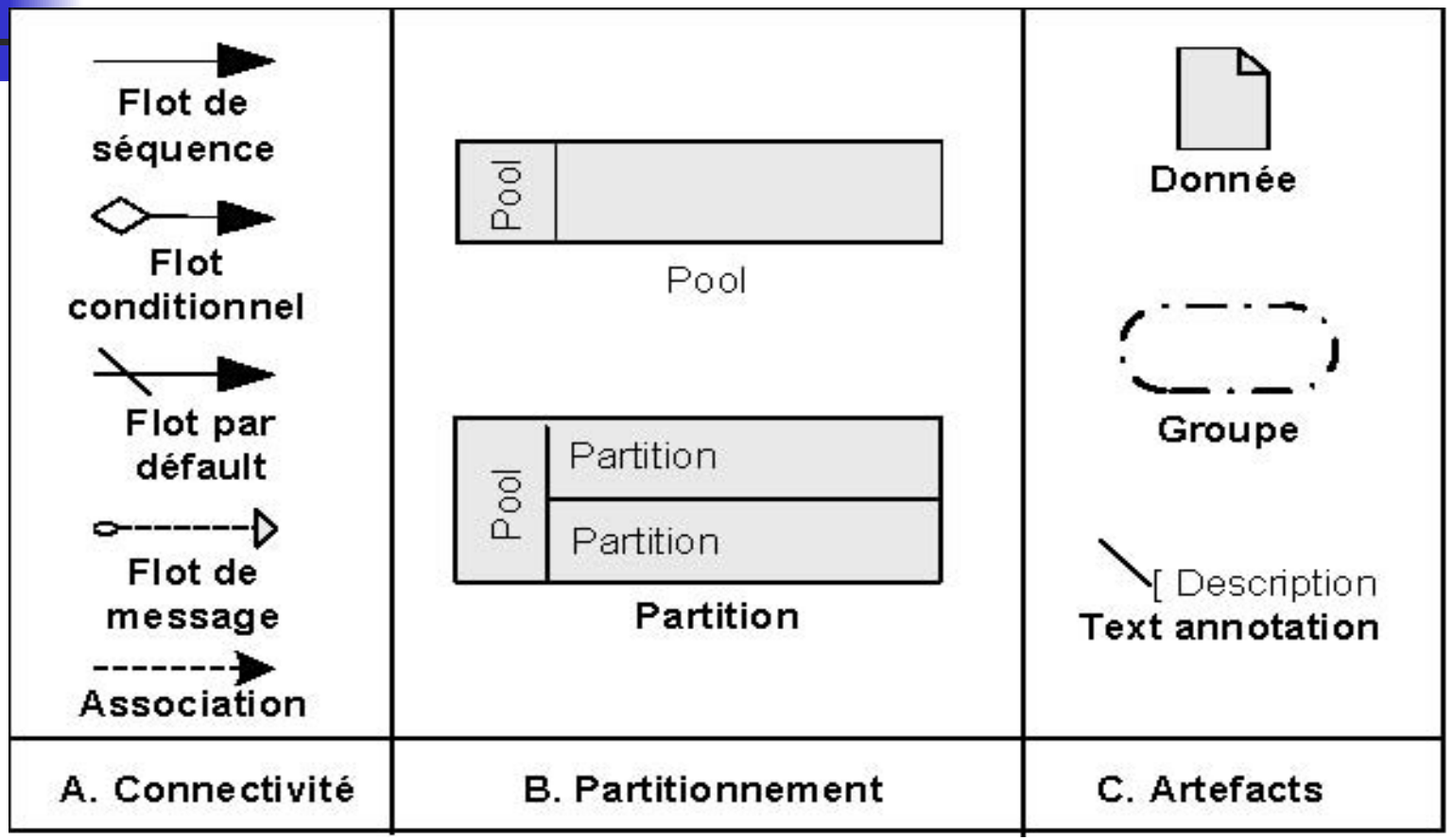
Notations

BPMN

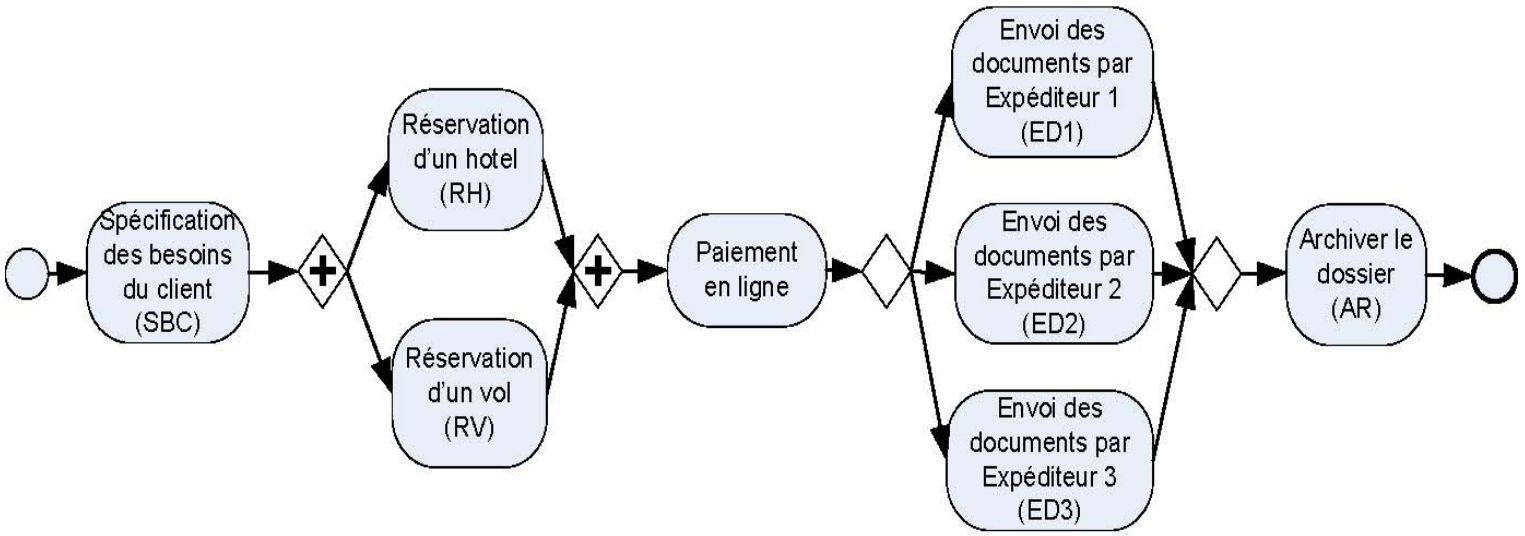
Symboles BPMN (1)



Symboles BPM (2)



Travel process



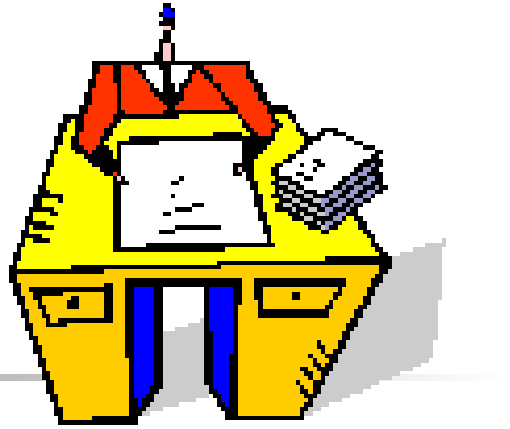
Insurance company

- Insurance company X processes claims which result from traffic accidents with cars where customers of X are involved in. Therefore, it uses the following procedure for the processing of the insurance claims.
- Every claim, reported by a customer, is registered by an employee of department CD (CD = Car Damages). After the registration of the claim, the insurance claim is classified by a claim handler of rank A or B within CD. There are two categories: simple and complex claims.
- For simple claims two tasks need to be executed: check insurance and phone garage. These tasks are independent of each other.

Insurance company (2)

- The complex claims require three tasks to be executed: check insurance, check damage history and phone garage. These tasks need to be executed sequentially in the order specified.
- Both for the simple and complex claims, the tasks are done by employees of department CD. After executing the two respectively three tasks a decision is made. This decision is made by a claim handler of rank A and has two possible outcomes: OK (positive) or NOK (negative).
- If the decision is positive, then insurance company X will pay. An employee of the finance department handles the payment. In any event, the insurance company sends a letter to the customer who sent the claim. An employee of the department CD writes this letter.

Complaints handling



- Each year travel agency Y has to process a lot of complaints (about 10.000). There is a special department for the processing of complaints (department C). There is also an internal department called logistics (department L) which takes care of the registration of incoming complaints and the archiving of processed complaints. The following procedure is used to handle these complaints.

Complaints handling (2)

- An employee of department L first registers every incoming complaint. After registration a form is sent to the customer with questions about the nature of the complaint. This is done by an employee of department C. There are two possibilities: the customer returns the form within two weeks or not. If the form is returned, it is processed automatically resulting in a report which can be used for the actual processing of the complaint. If the form is not returned on time, a time-out occurs resulting in an empty report. Note that this does not necessarily mean that the complaint is discarded. After registration, i.e., in parallel with the form handling, the preparation for the actual processing is started.

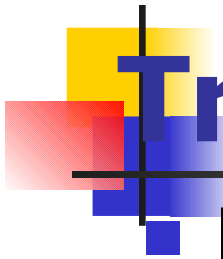
Complaints handling (3)

- First, the complaint is evaluated by a complaint manager of department C. Evaluation shows that either further processing is needed or not. Note that this decision does not depend on the form handling. If no further processing is required and the form is handled, the complaint is archived. If further processing is required, an employee of the complaints department executes the task 'process complaint' (this is the actual processing where certain actions are proposed if needed). For the actual processing of the complaint, the report resulting from the form handling is used. Note that the report can be empty. The result of task ´process complaint´ is checked by a complaint manager. If the result is not OK, task ´process complaint´ is executed again. This is repeated until the result is acceptable. If the result is accepted, an employee of the department C executes the proposed actions. After this the processed complaint is archived by an employee of department L.

Travel agency

- Consider a fragment of the process of booking trips involving six steps: *register*, (booking of) *flight*, (booking of *hotel*, (booking of) *car*, *pay*, and *cancel*.
- The process starts with task *register* and ends with *pay* or *cancel*.
- The tasks *flight*, *hotel* and *car* may succeed or fail.

Let us consider a number of variants…

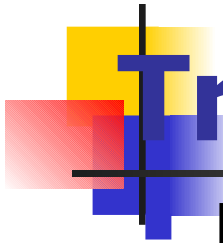


Travel agency: Variant 1

- Every trip involves a flight, hotel and car and these are booked in parallel. If all three succeed, the payment follows. Otherwise task cancel is executed. Cancel is delayed until all three bookings succeed/fail and does not withdraw work.

Travel agency: Variant 2

- Every trip involves a flight, hotel and car and these are booked in parallel. If all three succeed, the payment follows. Otherwise task cancel is executed. *Task cancel can be executed the moment the first task fails and withdraws work-items waiting for remaining booking tasks.*



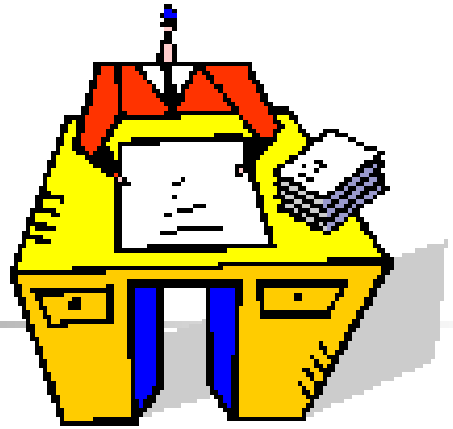
Travel agency: Variant 3

Every trip may involve a flight, hotel and/or car and these are booked in parallel. A trip should involve *at least a flight, hotel or car but may be any combination of the three bookings*, e.g., a flight and car but not a hotel. If all bookings succeed, the payment follows. Otherwise task cancel is executed. Task cancel can be executed the moment the first task fails and withdraws work-items waiting for remaining booking tasks.

Travel agency: Variant 4

Every trip may involve a flight, hotel and/or car and these are booked in parallel. A trip should involve at least a flight, hotel or car but may be any combination of the three bookings, e.g., a flight and car but not a hotel. If all bookings succeed, the payment follows. Otherwise task cancel is executed. Task cancel can be executed the moment the first task fails and withdraws work items waiting for remaining booking tasks. *All bookings that have completed successfully need to be compensated.*

Complaints handling



- Each year travel agency Y has to process a lot of complaints (about 10.000). There is a special department for the processing of complaints (department C). There is also an internal department called logistics (department L) which takes care of the registration of incoming complaints and the archiving of processed complaints. The following procedure is used to handle these complaints.

Complaints handling (2)

- An employee of department L first registers every incoming complaint. After registration a form is sent to the customer with questions about the nature of the complaint. This is done by an employee of department C. There are two possibilities: the customer returns the form within two weeks or not. If the form is returned, it is processed automatically resulting in a report which can be used for the actual processing of the complaint. If the form is not returned on time, a time-out occurs resulting in an empty report. Note that this does not necessarily mean that the complaint is discarded. After registration, i.e., in parallel with the form handling, the preparation for the actual processing is started.

Complaints handling (3)

- First, the complaint is evaluated by a complaint manager of department C. Evaluation shows that either further processing is needed or not. Note that this decision does not depend on the form handling. If no further processing is required and the form is handled, the complaint is archived. If further processing is required, an employee of the complaints department executes the task 'process complaint' (this is the actual processing where certain actions are proposed if needed). For the actual processing of the complaint, the report resulting from the form handling is used. Note that the report can be empty. The result of task ´process complaint´ is checked by a complaint manager. If the result is not OK, task ´process complaint´ is executed again. This is repeated until the result is acceptable. If the result is accepted, an employee of the department C executes the proposed actions. After this the processed complaint is archived by an employee of department L.

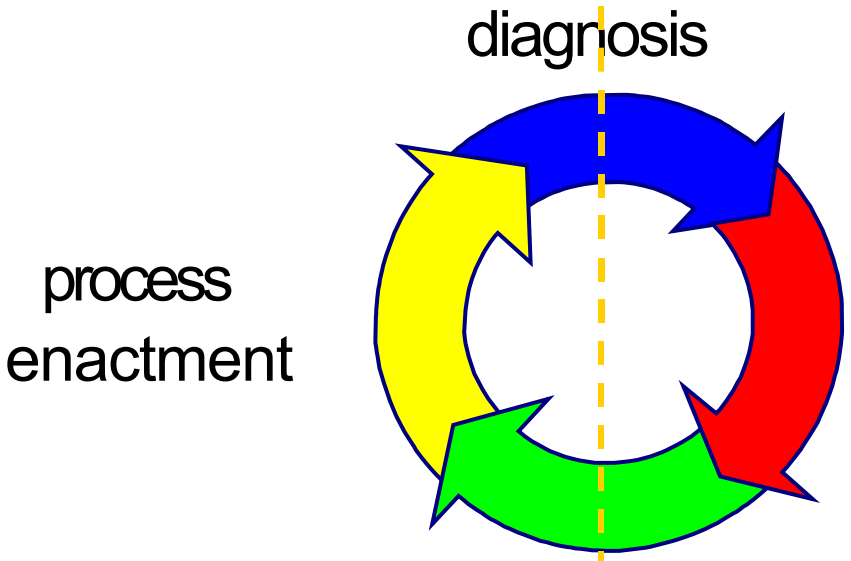
Analysis of workflows:

Verification, validation, and performance analysis.

Design-time and run-time questions

Run-time

Design-time



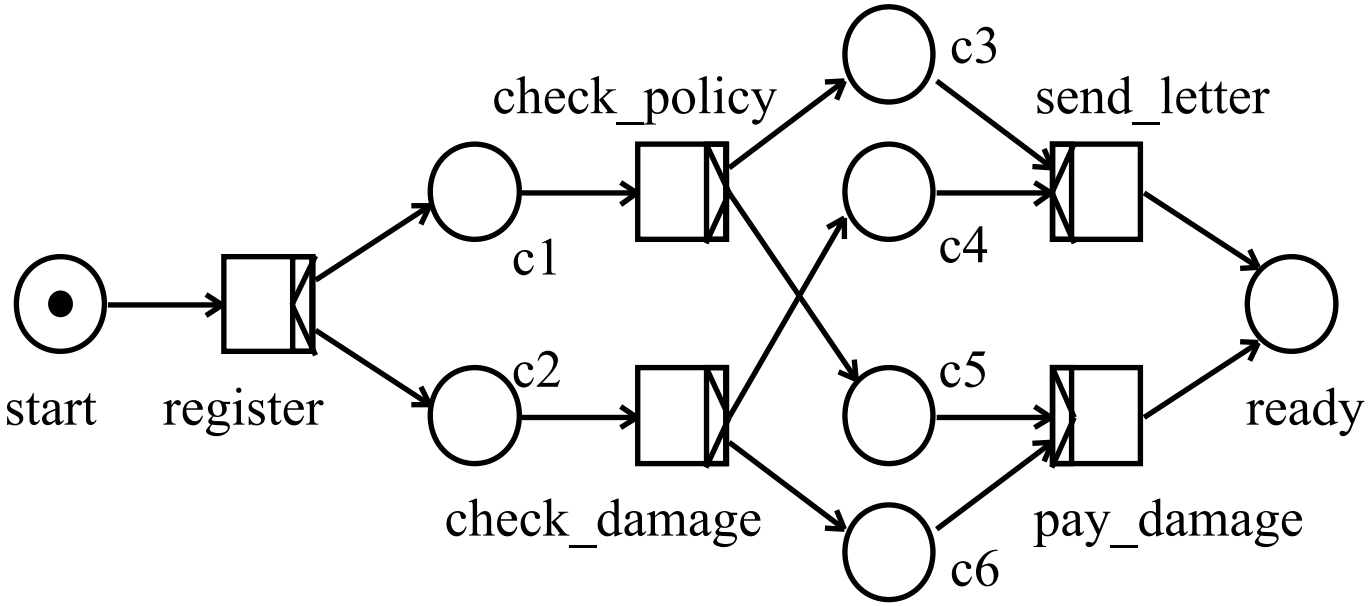
- ***process mining***

- ***verification***
  validation
  performance analysis

Techniques to analyze workflows (design-time)

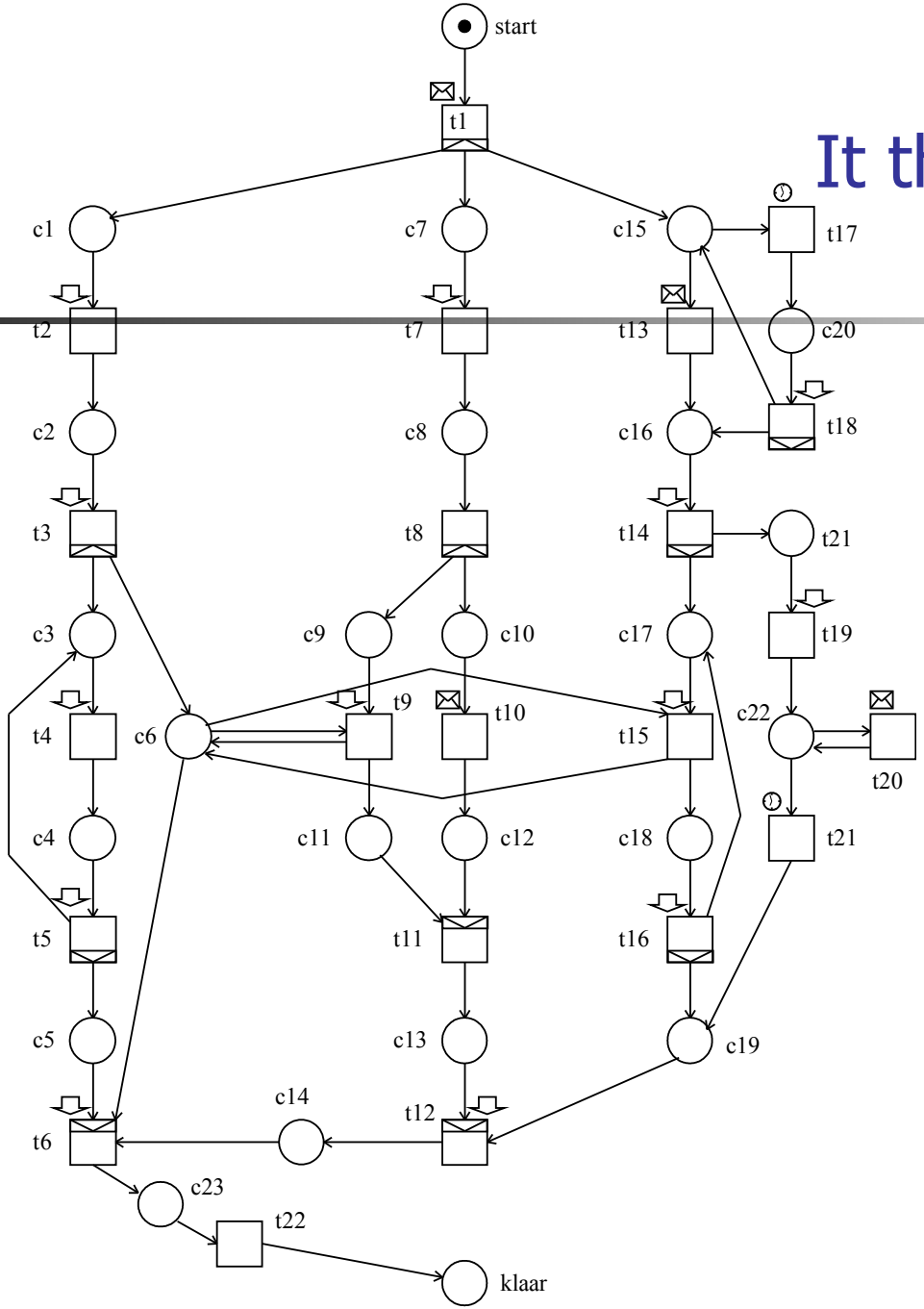
- **Validation** is concerned with the **relation between the model and reality**.
- **Verification** is typically used to answer **qualitative questions**
  - Is there a deadlock possible?
  - It is possible to successfully handle a specific case?
  - Will all cases terminate eventually?
  - It is possible to execute two tasks in any order?
- **Performance analysis** is typically used to answer **quantitative questions**
  - How many cases can be handled in one hour?
  - What is the average flow time?
  - How many extra resources are required?
  - How many cases are handled within 2 days?

Verification:
analysis techniques can be used to avoid logical errors.

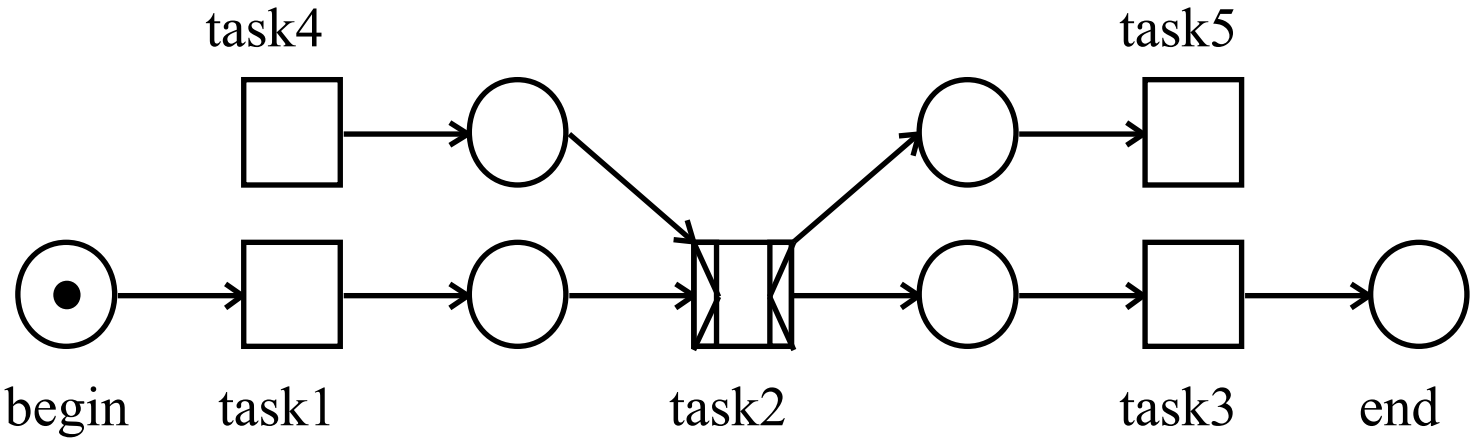


Is this a correct workflow?
If not, how to correct it?

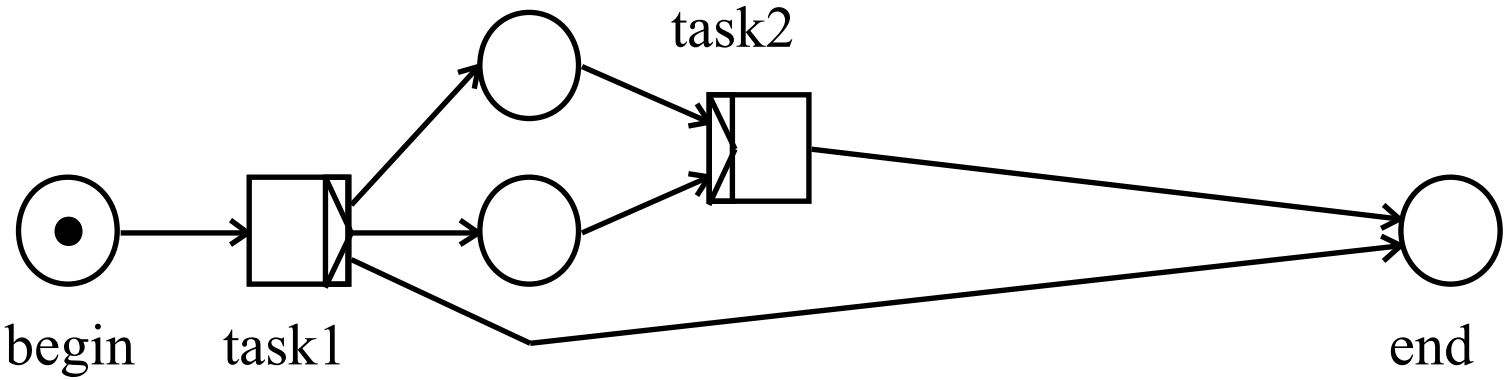
It this process correct?



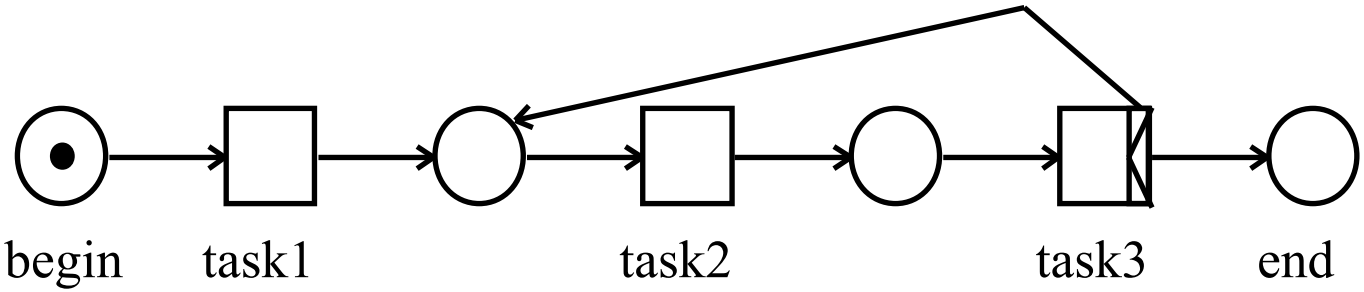
Error 1: dangling tasks



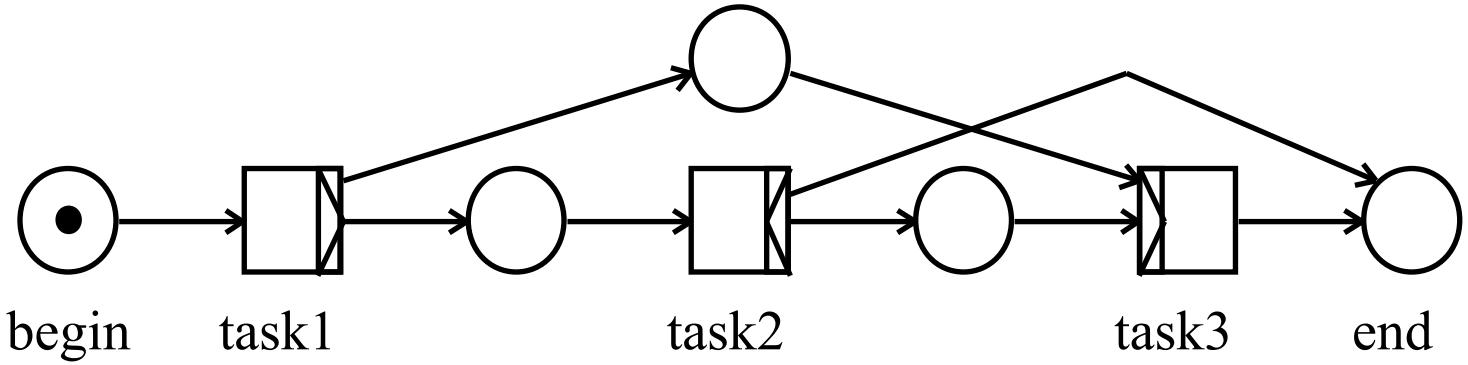
Error 2: deadlock (task2)



Error 3: unbounded and never-ending



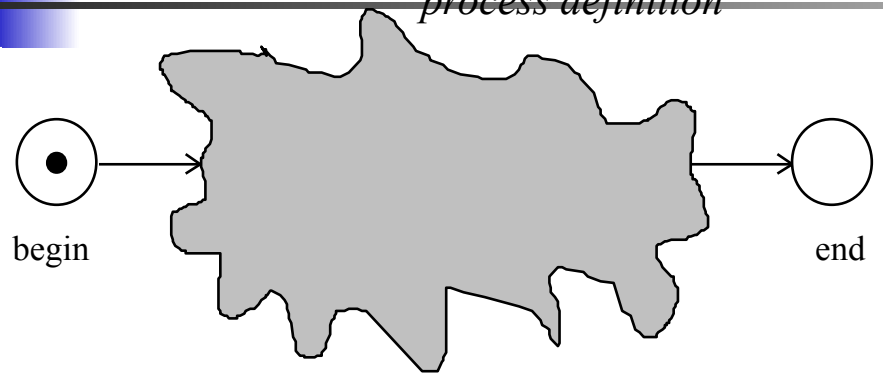
Error 4: deadlock before or after termination



Good structuring: well balanced operators

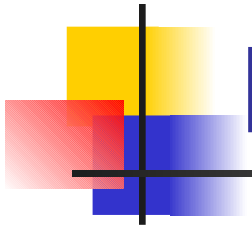
- AND-split ..... AND-join
-
-
- Explicit OR-split ..... OR-join

Soundness property



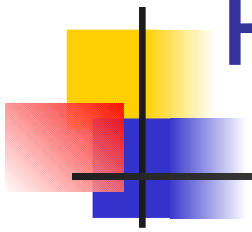
Eventually the case terminates and the moment it terminates all references have been removed.

- The soundness property corresponds to two standard Petri-net properties (liveness and boundedness).
- Standard Petri-net-based tools can be used.
- For (almost) free-choice nets this can be checked in polynomial time!



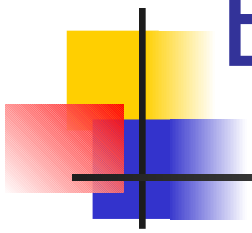
Reachability graph

- The reachability graph of a Petri net is the part of the transition system reachable from the initial state in graph-like notation.
- The reachability graph can be calculated as follows:
  1. Let X be the set containing just the initial state and let Y be the empty set.
  2. Take an element x of X and add this to Y. Calculate all states reachable for x by firing some enabled transition. Each successor state that is not in Y is added to X.
  3. If X is empty stop, otherwise goto 2.



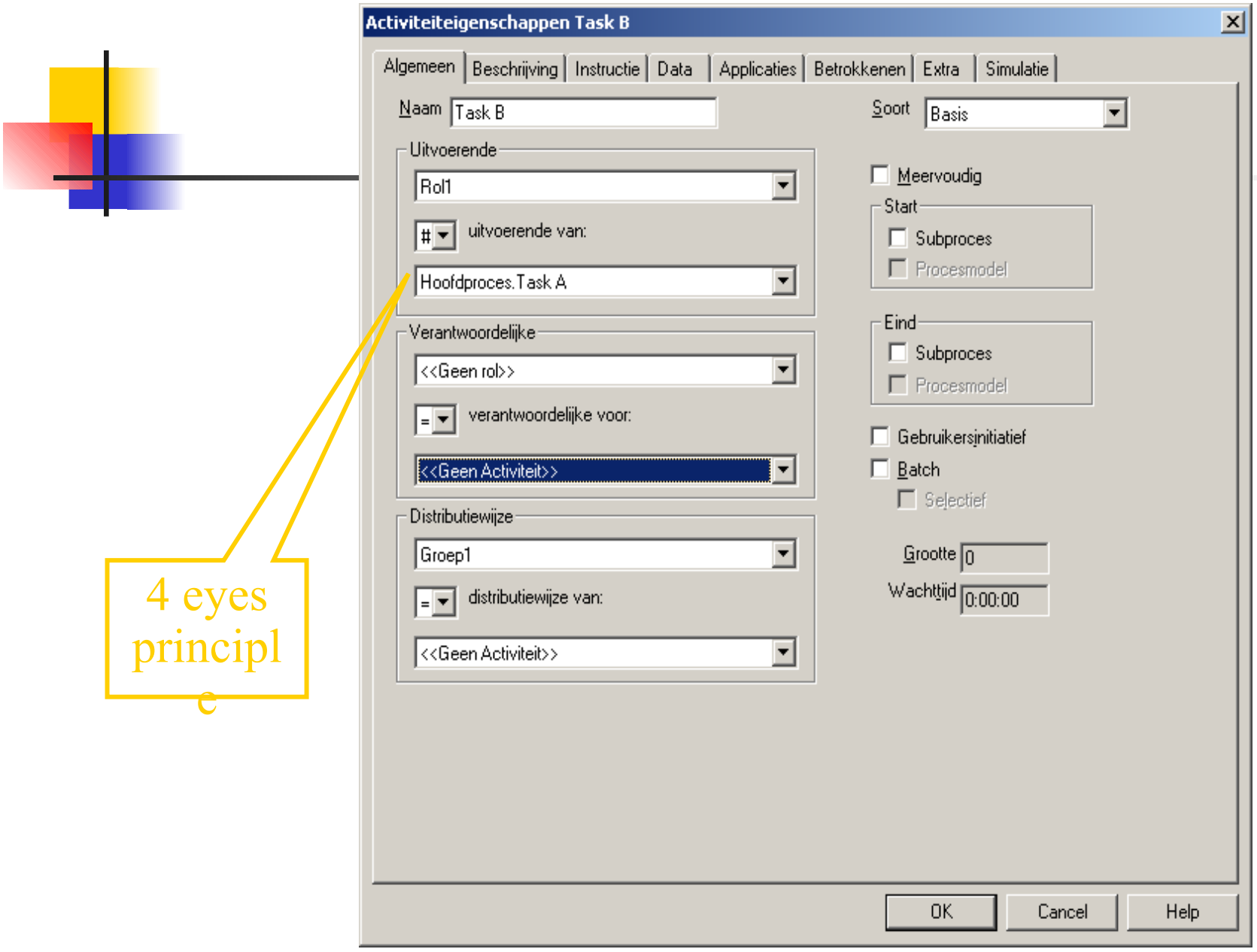
Performance analysis

- Throughput, waiting and service time
- Occupation rates
- Techniques:
  - Simulation
  - Queuing theory
  - Markovian analysis

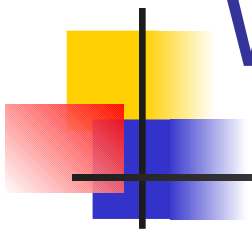


Example:

- sequential

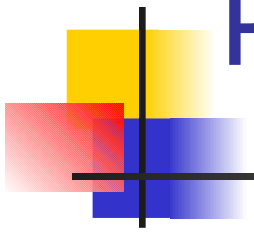

Activiteiteigenschappen Task B
Algemeen
Beschrijving
Instructie
Data
Applicaties
Betrokkenen
Extra
Simulatie
Naam
Task B
Soort
Basis
Uitvoerende
Rol1
uitvoerende van:
Hoofdproces.Task A
Meervoudig
Start
Subproces
Procesmodel
Eind
Subproces
Procesmodel
Verantwoordelijke
<<Geen rol>>
verantwoordelijke voor:
<<Geen Activiteit>>
Gebruikersinitiatief
Batch
Selectief
Distributiewijze
Groep1
distributiewijze van:
<<Geen Activiteit>>
Grootte
0
Wachttijd
0:00:00
OK
Cancel
Help
4 eyes principle

Workflow Patterns



- http://www.workflowpatterns.com
- 30 patterns

Pattern 1(Sequence)



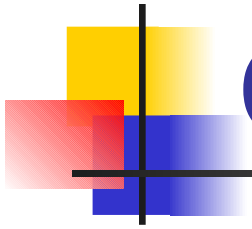
- Description An activity in a workflow process is enabled after the completion of another activity in the same process.
- Synonyms Sequential routing, serial routing

Assumptions so far ...

- Eventually every work-item is executed by a single resource.
- Every resource is working on one activity at the same time.

Some observations:

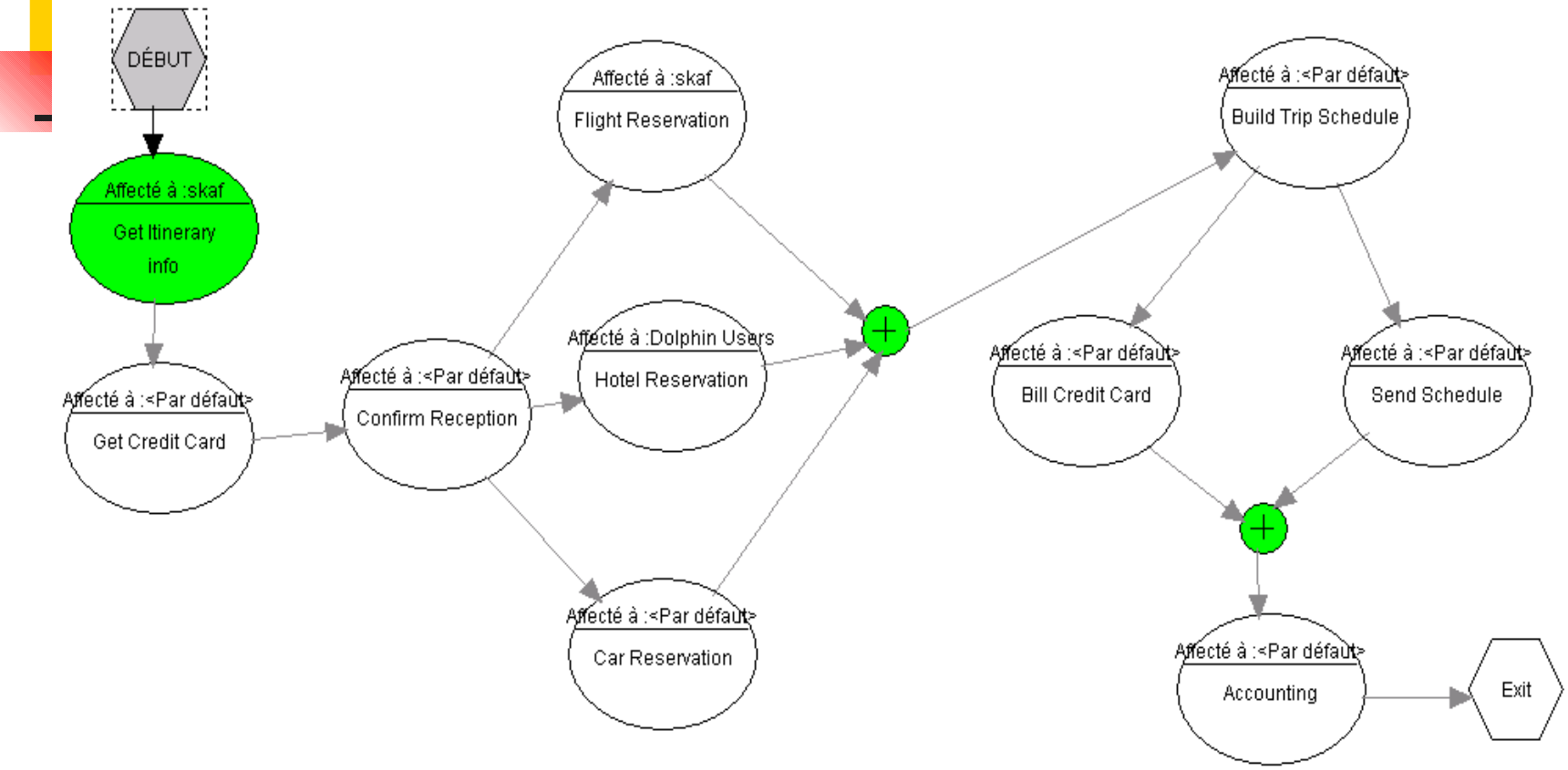
- There may be a need to further limit the set of resources (e.g., the 4 eyes principle), i.e., we need to be able to *specify further constraints*.
- There may be many resources that have the right role/group combination, i.e., *work distribution* is needed.
- There may be many work items that can be executed by the same resource at a given point in time, i.e., *work items need to be ordered*.

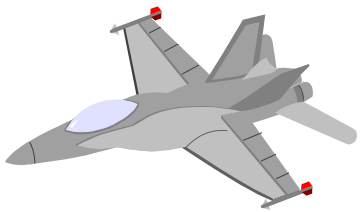


Open Issues

- Flexibility and exceptions handling
- **Transactional Workflow**
- Fault Tolerance
- Workflow Mining
- People Workflow

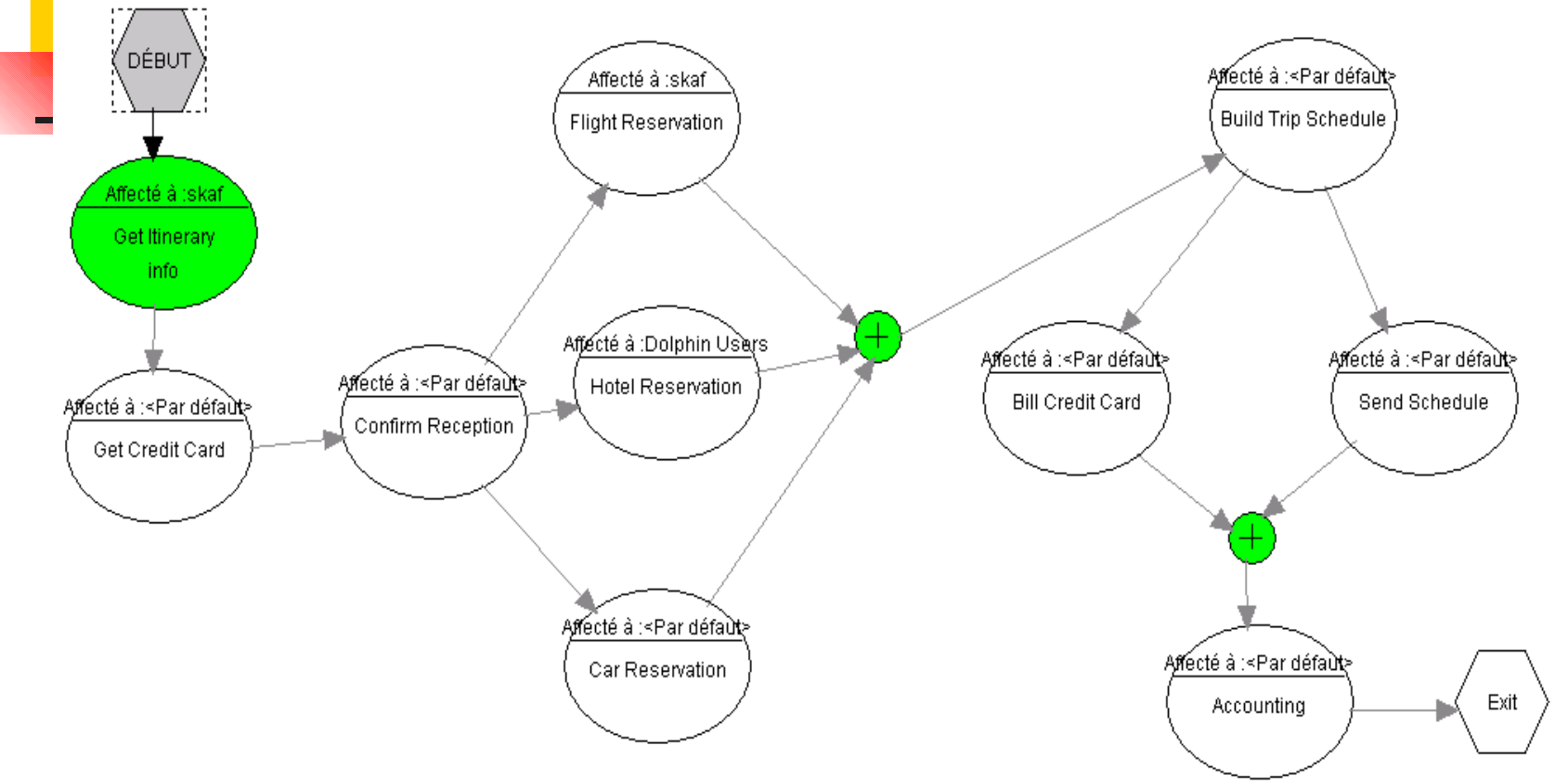
Why Transactional Workflow ?

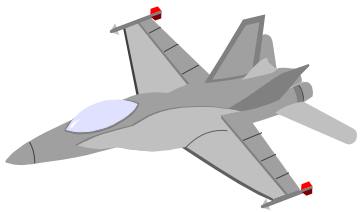




Parts of Travel reservation of “traveluck”, Inc

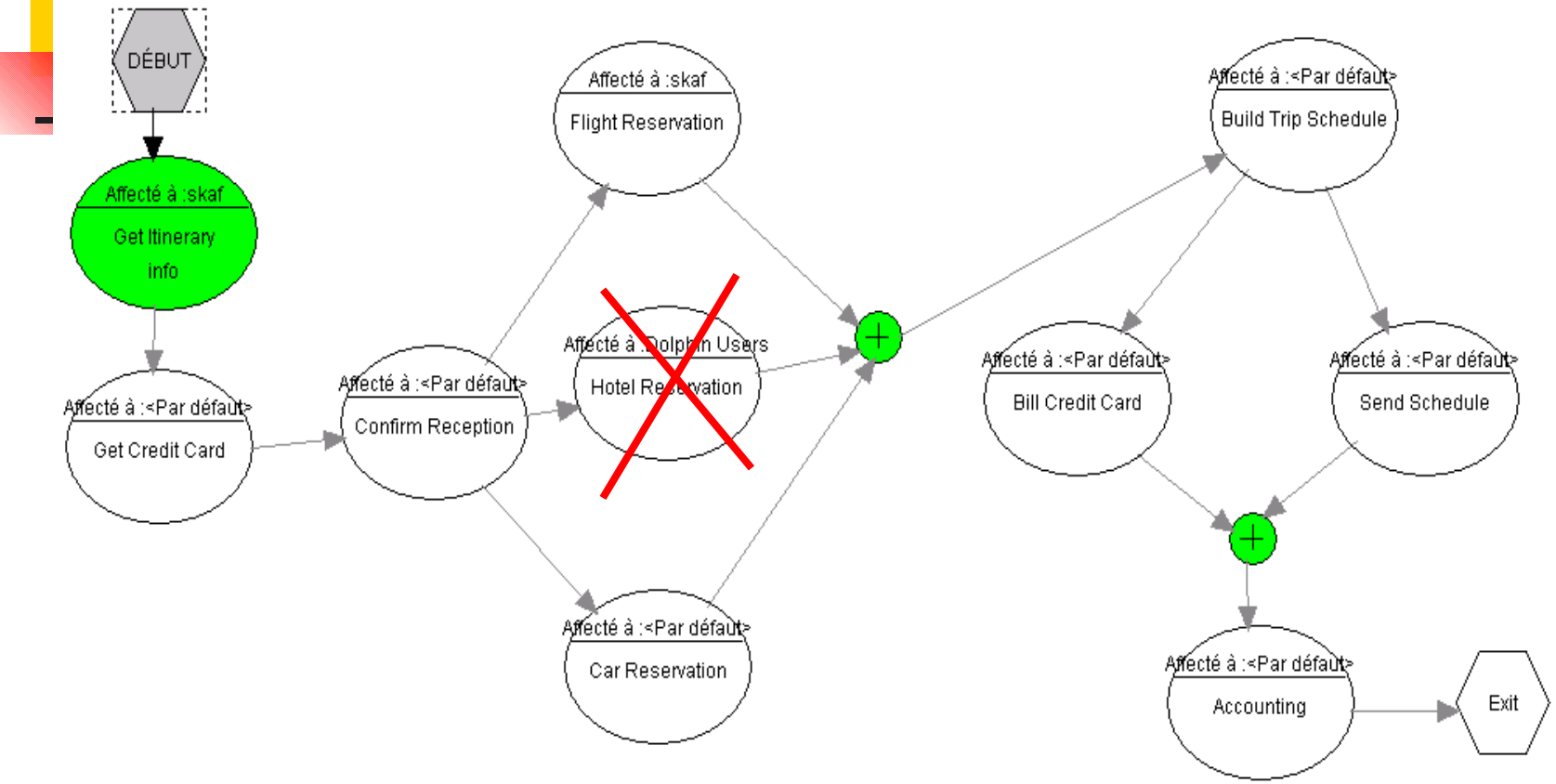
Why Transactional Workflow ?

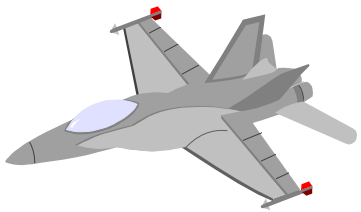




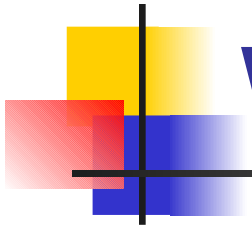
Data Sharing Semantics

Why Transactional Workflow ?





Failure Semantics



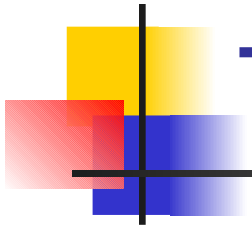
Why Workflow Technology

- To streamline, coordinate and monitor an organizational process involving human and automated tasks spread across multiple enterprises with heterogeneous (existing and new) computing environment.
- Some products focus on one aspects of workflow technology, primarily to reduce paper work or coordinate activities among humans:
  - document management, e-mail routing, etc...

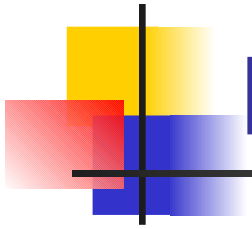
Benefits of Workflow Technology

- For customers:
  - Gain improved insight into the processing of their work cases.
  - Response time,
  - Quality of services and accuracy of status are enhanced.
- At enterprise level
  - Organize, schedule, control and monitor process
  - Improve productivity
  - Reduce paper work…
  - reinforces security and confidentiality ..
  - The image of the enterprise is improved

Benefits of Workflow Technology

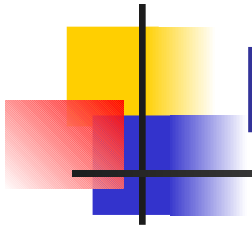


- For the enterprise staff
  - Employee has more accurate view of his tasks
  - External pressures for increased efficiency
  - No need to spend time and effort to control the organization of work cases...
  - Internal pressure for increased effectiveness
  - Desire by workers for more reward and less stress !!
- For the line management (supervisors)
  - just in time information to make decision to improve tasks !!



Benefits of Workflow Technology

- For the top management
  - formally describe the processes of an enterprise
  - improve process
  - Workflow also helps to quickly educate new or temporary actors.
- In general:
  - Support on-line data entry, Support data exchange and transactions across independent enterprises (EDI)
  - In most general form, workflow technology can be used to support programming-in-the-large..



Be Careful !

- Workflow must be flexible…
- Must not kill creativity of people..
- Must be accepted by people..
- No successful story of Wf from 1970 to 1980 because of people rejection (don’t take in consideration social and organization setting) …
- Workflows are **people** systems