An Agent-based DES Supported Modeling Framework

Content

- **Enterprise Control: context**
- **The Proposed Control Architecture**
- **Supervisory level**
- **Agent level**
- **The modeling framework**
- **Case study**
- **Conclusions**

Enterprise Organisation Paradigm

- **Modern Enterprise**:
  - **frictionless, seamless, integrated, adaptive, efficient networking organization,**
  - **dynamically formed to create sustainable value**
  - **through the delivery of integrated Product-Service-Organisation (PSO) configurations providing benefits to customers**

Modelling Paradigm

Complex Adaptive System - main features:

- characteristics of **Large Scale Systems** (LSS)
- a given set of **operation states**
- composed by **subsystems networked by a given set of interdependences**
- it has an environmental **responsive behavior**
- **different possibilities** of subsystem failures

Control Approach

Intelligent distributed control systems - from mixing together several concepts:

- the **hierarchical approach** in modelling, design and operation
- the **agent-based approach** for the dynamic reconfiguration of the control system.

Hierarchical control

- **Goal**:
  - Optimal process functioning policy.
- **Features**:
  - Process model necessary
  - Evaluation of all possible behaviors of the controlled system
- **Disadvantage**:
  - Lack of robustness

Agent based architecture

- **Goal**:
  - Highly robust control structure with respect to resource break-downs and order cancellation.
- **Features**:
  - Different categories of partial models
  - Agents with possible conflicting objectives
- **Disadvantage**:
  - Lack of global optimality in functioning

Control Approach

- **Intelligent distributed control systems** are consequently composed by
- intelligent autonomous and negociating modules – (intelligent agents) –
- supported by a flexible thus consistent modelling formalism as the complex Discrete Event Systems and hybrid systems.

Supervision architecture

- structure & functioning -

Generic solution:

- heterarchical organization
- stepwise decomposition of goals through heterarchy
- reconfigurable links between modules
- implementation of the IPDI Principle - optimal combination of algorithmic and intelligent problem-solving techniques

Generic supervisory module

- constraints & requirements -

characteristics of **Large Scale Systems**

necessity of qualitative models

large state-space of reachable solutions

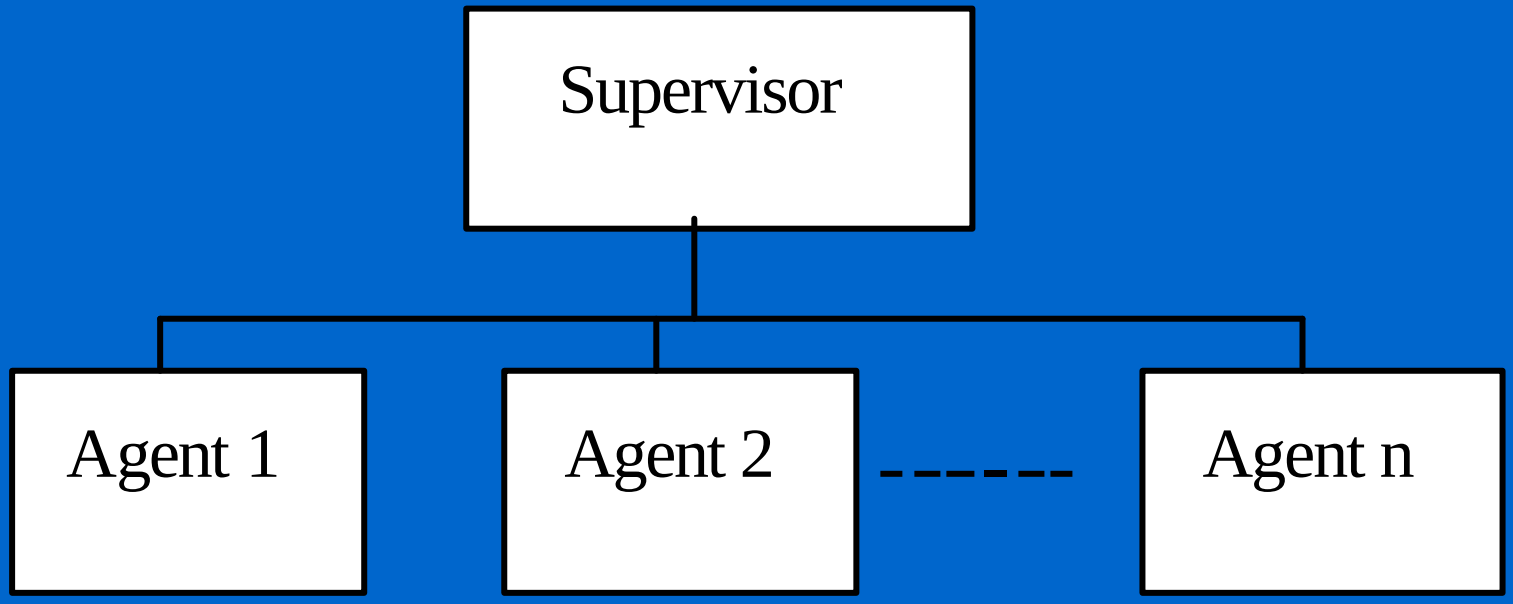
necessity of heuristic solving rules

environmental **responsive behavior**

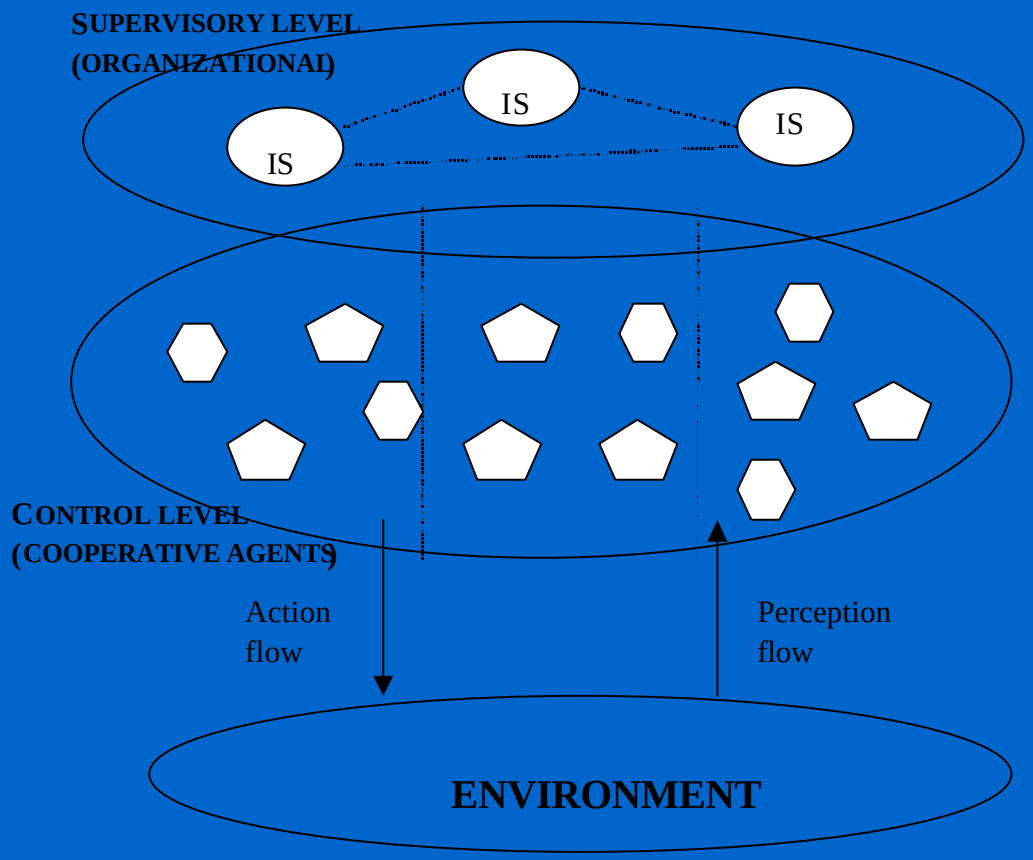
solving time is limited

The Control Architecture

- basic principle -



The Supervised Control Architecture



Controller - Supervisor units

- functioning assumptions -

- **The *scheduling problem*** :
- **Given**:
  - models of resources: process model
  - local goals: functioning specifications models
- **Requested:**
  - a control policy allowing the accomplishment of local goals by available resources

Controller - Supervisor units

- functioning assumptions -

- One cannot *a priori* know if a set of functioning specifications could be accomplished by a given subsystem. The **supervisor** must decide if a given problem could (or not) to be solved.

- Diagnosis of perturbation occurred in the functioning of the supervised system is considered to be always solved. The **main focus** of supervision module is on the **synthesis of a new control policy.**

Agent level

- **Types**:
  - Resource agents (**servers**).
  - Product agents (**clients**).
- **Negotiation procedure** - “Centered on the client”.

Supervisory level (1)

Off-line

- Combines the internal model of agents into a global closed-loop model.
- Synthesizes all the possible control policies.
  (the maximally permissive supervisor - in DES sense)
- Selects the optimal control policy according to a given optimality criterion (ex: client deadlines)
- Sends preliminary indications to agents

Supervisory level (2)

On-line

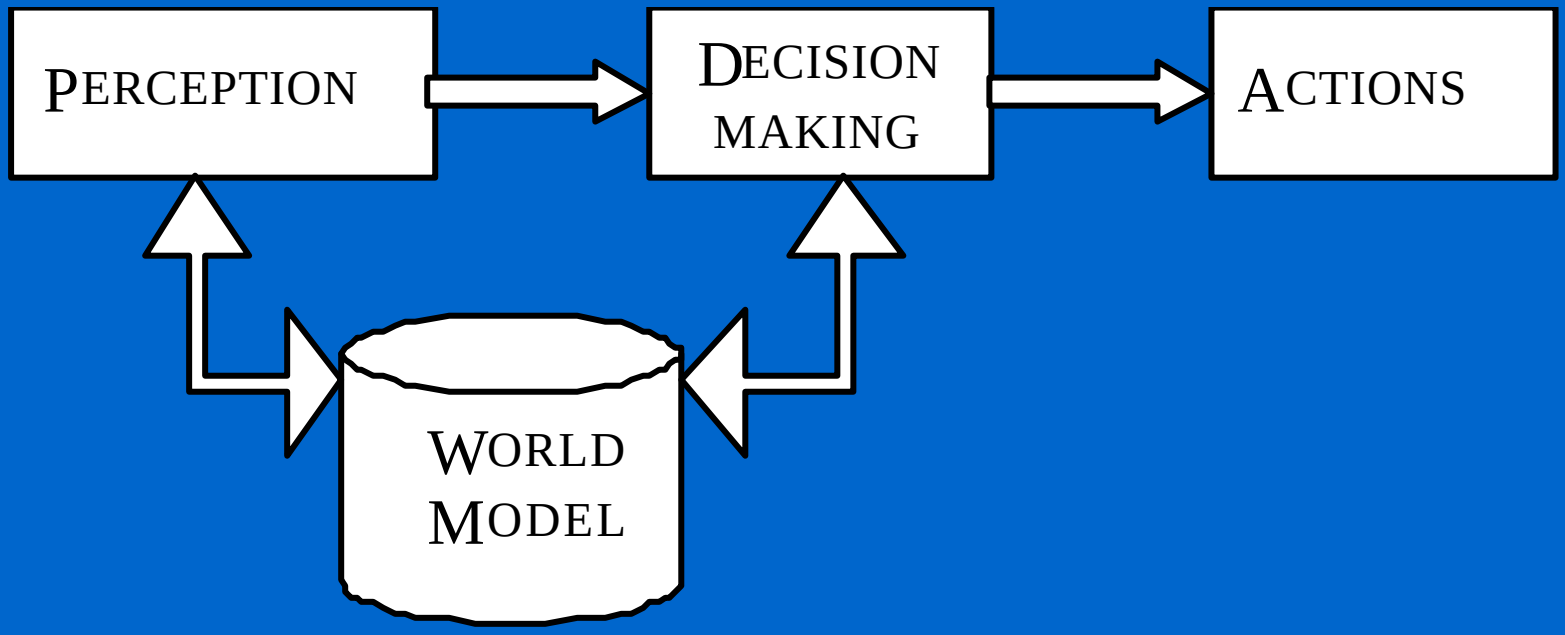
- **Monitors** agents behavior.
- **Advises** agents in their decisions according to the optimal control policy.
- **Breakdown situation** - agents became autonomous.
  - Result: non-optimal behavior for controlled system.
  - Specificity: the supervisor is still able to on-line advise agents.

The Control Architecture

Main properties:

- **Autonomy** – at agent level.
- **Capacity of collaboration**:
  - horizontally - between agents
  - vertically - between agents and supervisors
- **Adaptativity** *versus* environmental changes.

Generic agent structure



Agents basic features

- capacity of **generalization** (extracting the generic features of a problem – behavioral scheme)
- **heuristic (rule-based) search** of solutions
- **auto-evaluation** or value judgement of a planned action
- capacity of **learning** – based on the previous experience

Agent model

- **the environment** – represented dually by its model and its perception as feedback for actions
- **the goals** – decomposable in tasks
- **the methods** for executing tasks and for decomposing goals
- **the actions** on the environment as the results of tasks

Server agents

- **Components**:
  - the list of processing tasks;
  - the (up-datable) current state of the resource.
- **Information**:
  - the history of negotiations with resource agents and their basic characteristics;
  - the (up-datable) evaluation function value;
  - information about agents implied in in-course negotiations.

Client agents

- **Components**:
  - all the possible workflows for achieving its goal
- **Information**:
  - limits of cost functions;
  - deadlines;
  - the actual state;
  - the current value of its cost function;
  - the details of the in-course negotiations.

Modeling Support

- some requirements -

- to reflect the typical behavior of the process from real-time control point of view (actions, conditions, events)
- to allow modeling at different complexity/ hierarchical levels
- to allow a modular approach
- to integrate time (eventually undeterministic)
- to allow qualitative/ quantitative performance evaluation

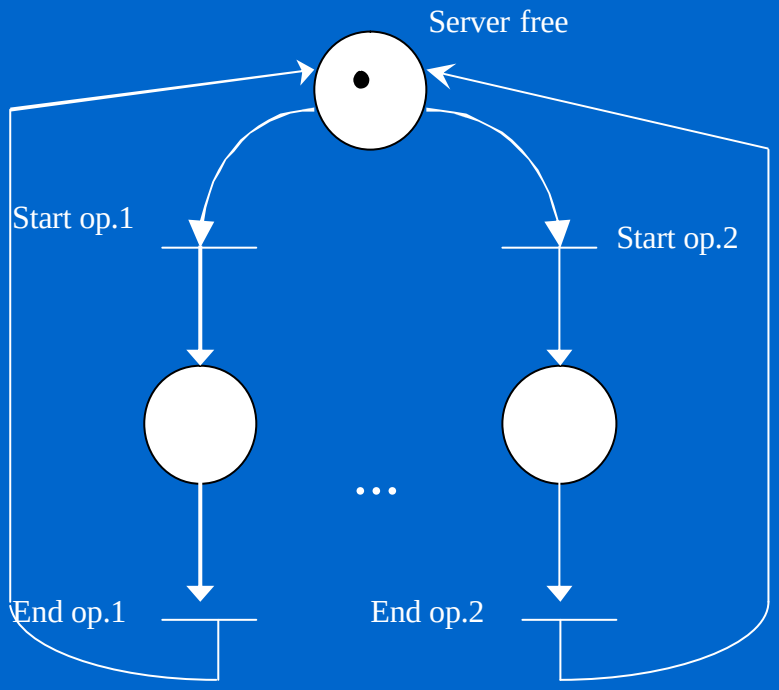
Modeling framework

Discrete Event Systems (DES)

- Categories of events:
  - **start** of an operation (transmitted by the controlling equipment) - controllable;
  - (normal) **end** of an operation (transmitted by the system) - uncontrollable;
  - **operation failure** - uncontrollable.

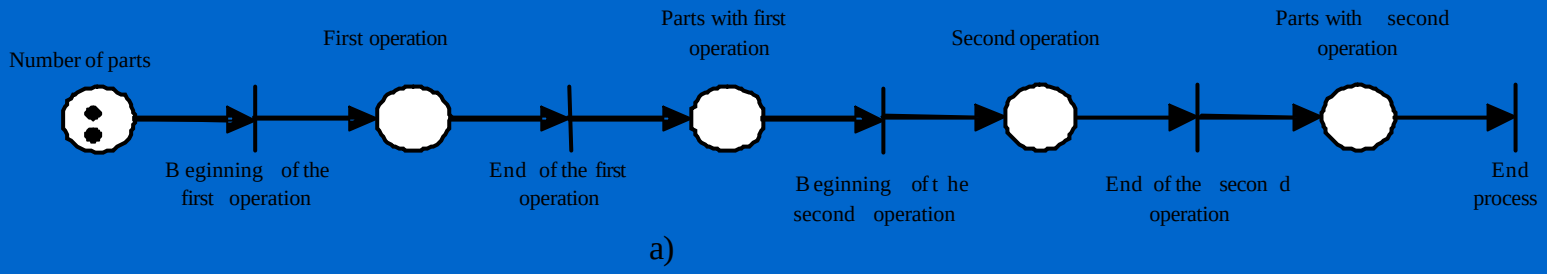
Generic (simplified) model

Server Agent

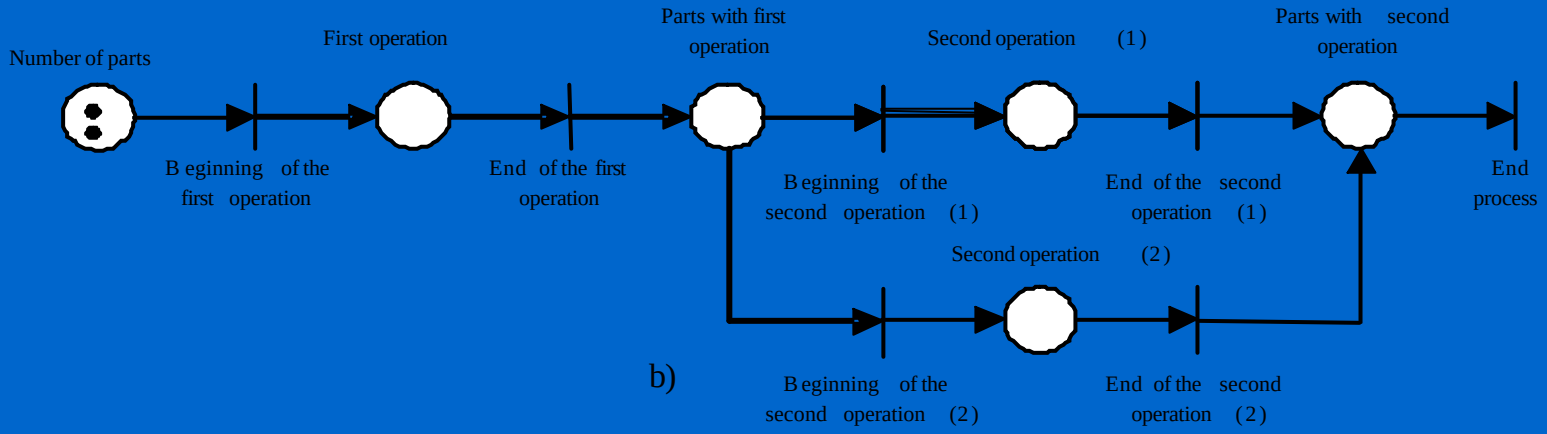


Generic (simplified) model -

Client Agent

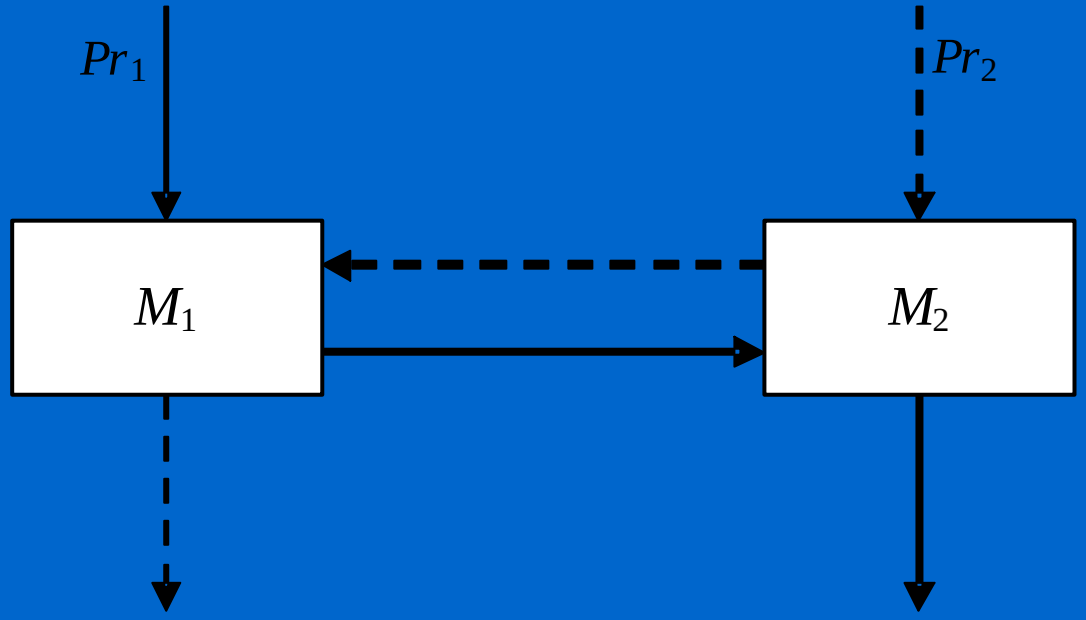


a)

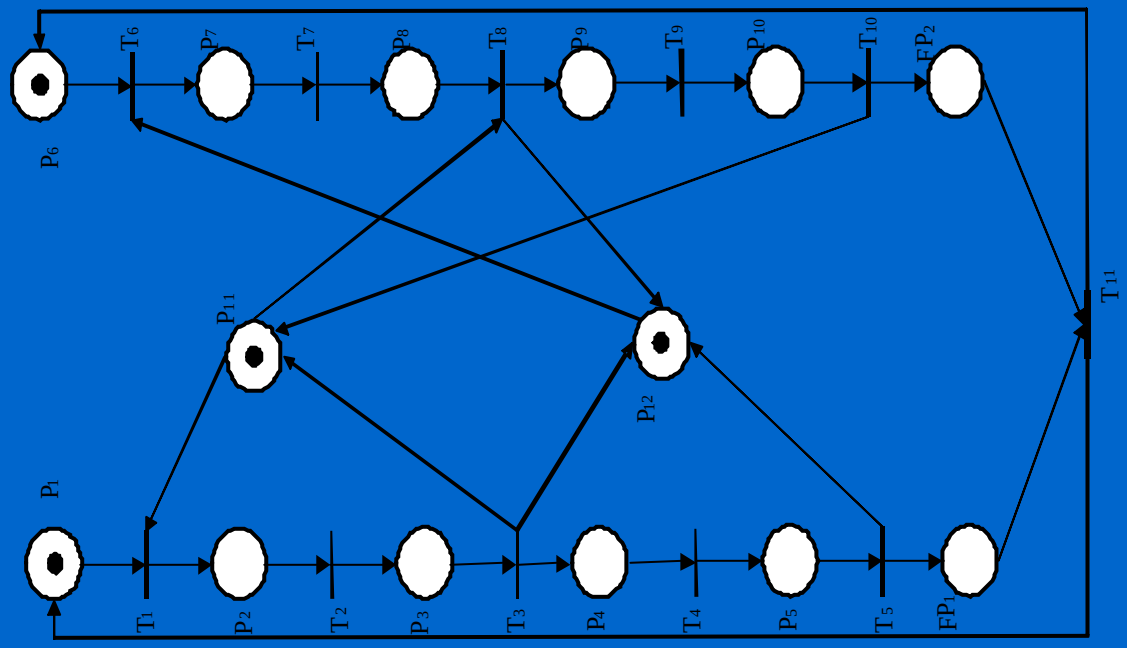


b)

Modelling Sample



Global model - 4 agents



Supervisory synthesis

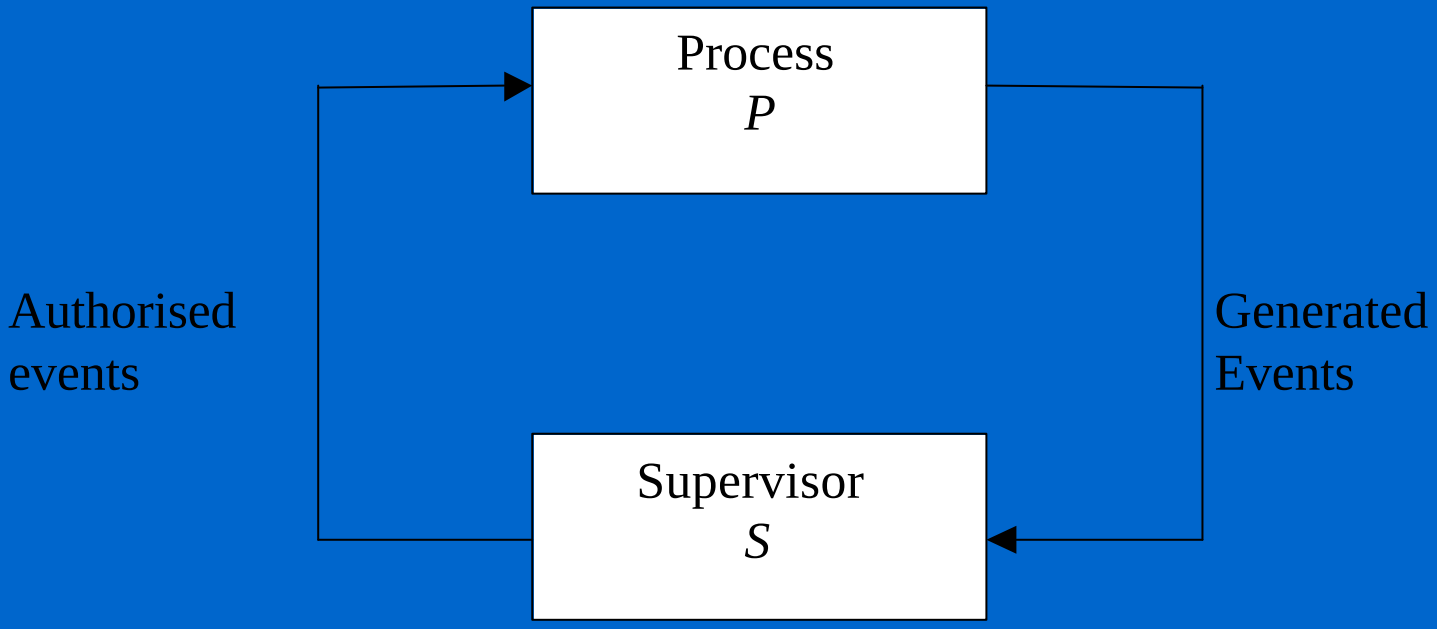
- **Objective**:

  Synthesizing the DES supervisor for the global model, eliminating blocking and forbidden states

- **Methodologie(s)**:

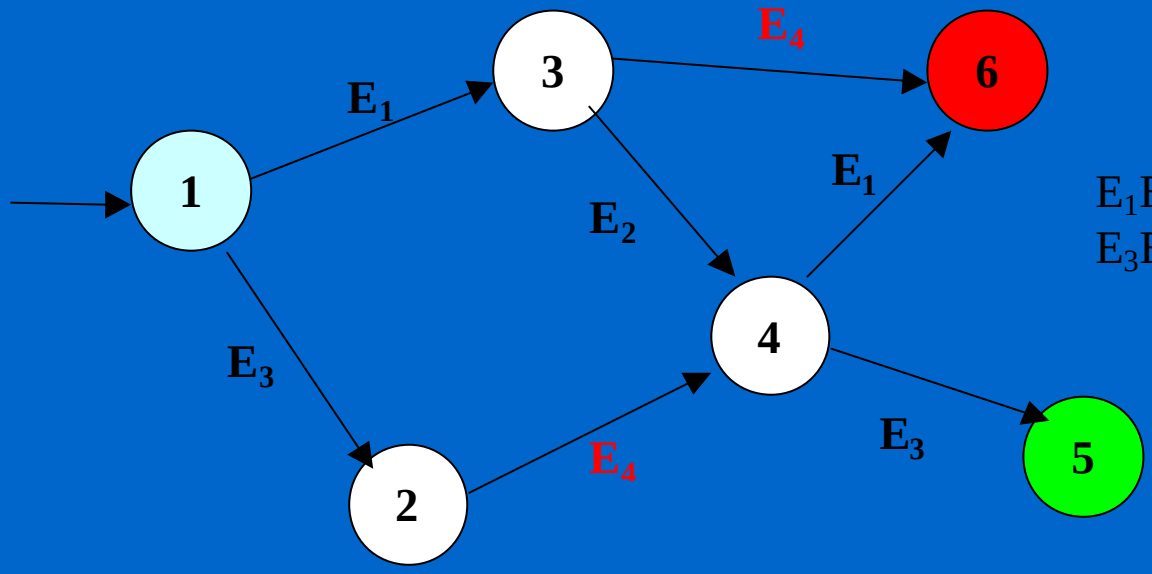
  - Kumar algorithm (maximally permissive supervisor)
  - Yamalidou & Moody method (based on marking invariant)

DES Supervision



The feedback loop of supervisory control

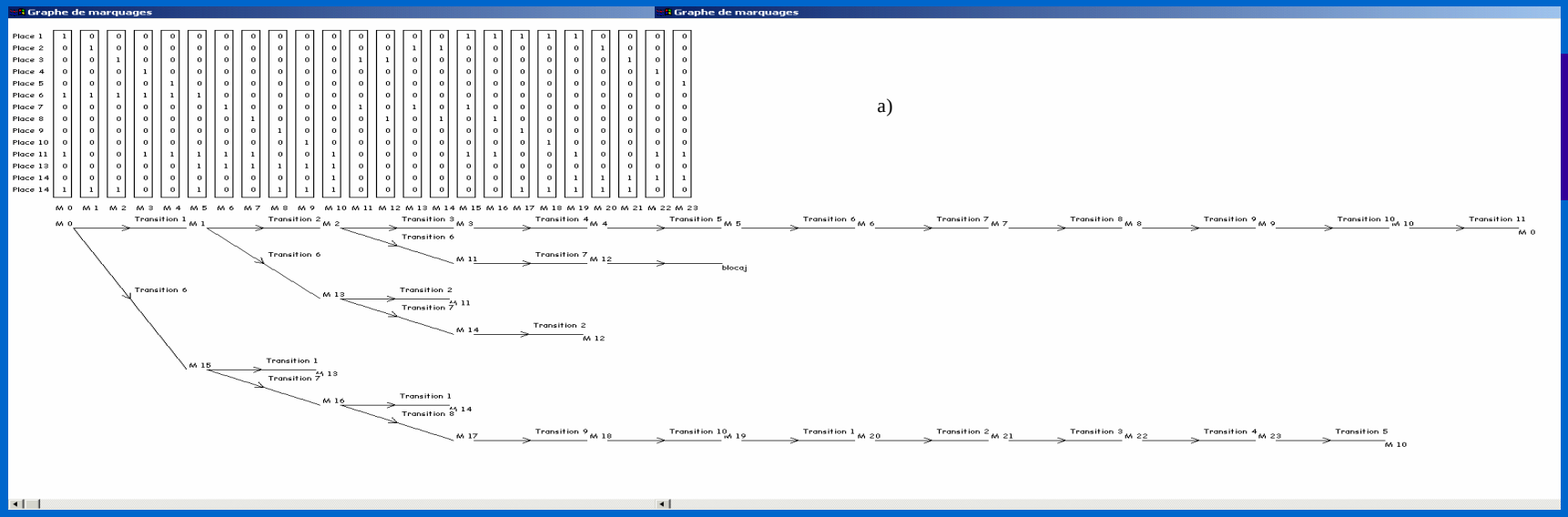
Dealing with uncontrollability

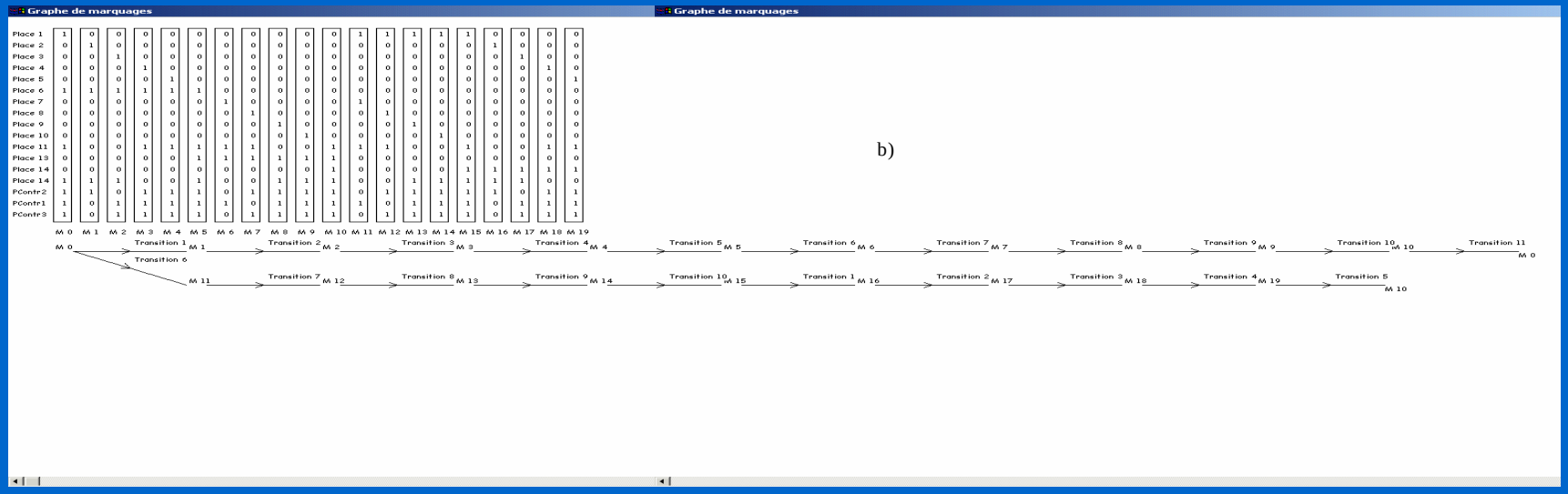


$E_1E_2E_3$ - uncontrollable
$E_3E_4E_3$

Kumar ’s algorithm

- Given: **Process** P and **functioning specifications** Spec
- Algorithm:
  1. Obtaining the global model (P x Spec)
  2. Finding forbidden states
  3. Finding weakly forbidden states
  4. Obtaining the maximally permissive supervisor





The state space:
a) complete b) desirable

Yamalidou & Moody

$$\sum_{i=1}^{n} l_i \mu_i \; ? \; ß \qquad (1)$$

Where specifications are given in terms of **forbidden states**, that are not verifying type (1) conditions

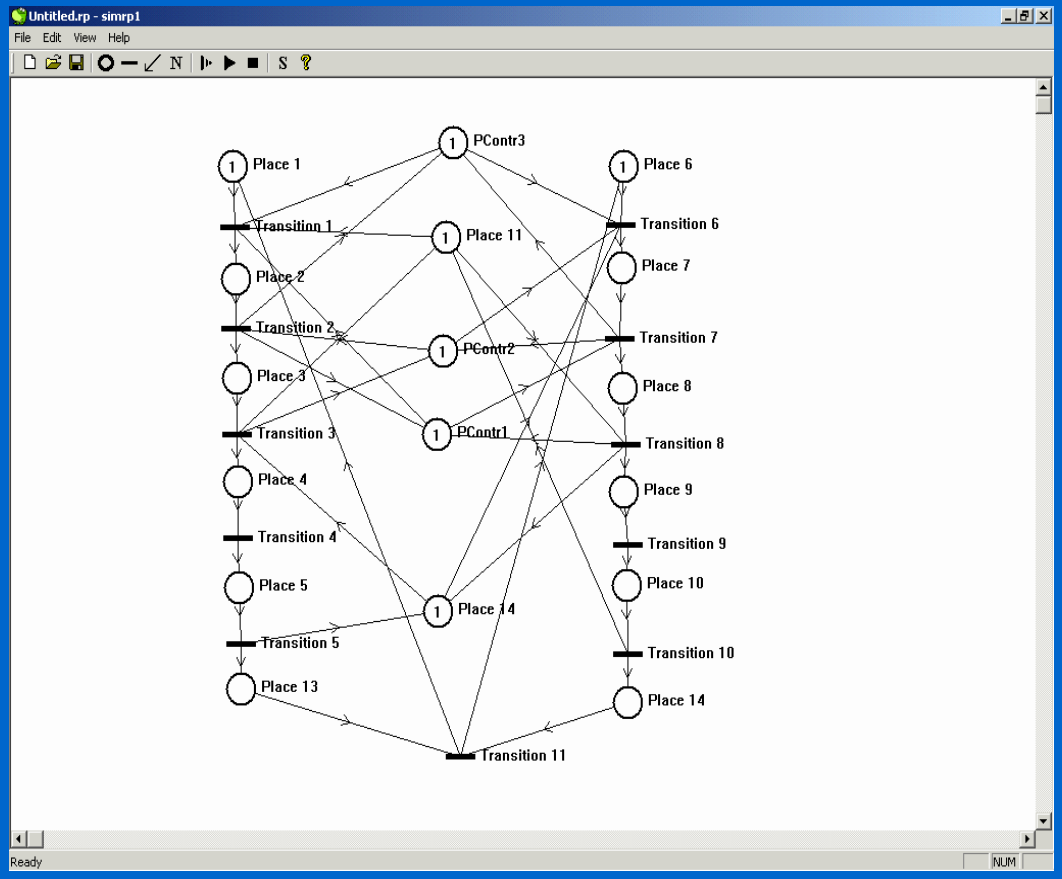
Supervisory synthesis

Steps:

1. Use of the Kumar ’s algorithm on the coverability tree of the global model – maximally permissive supervisor

2. Application of a modified version of the Yamalidou & Moody method – supervisory places forcing the desired behavior

Supervisor for the example



Research perspectives

- Generalizing the algorithm for several classes of Petri Net models
- Implementing communication modules for agents
- Extending the procedure for timed and temporal Petri Nets models - quantitative optimization