Petri nets

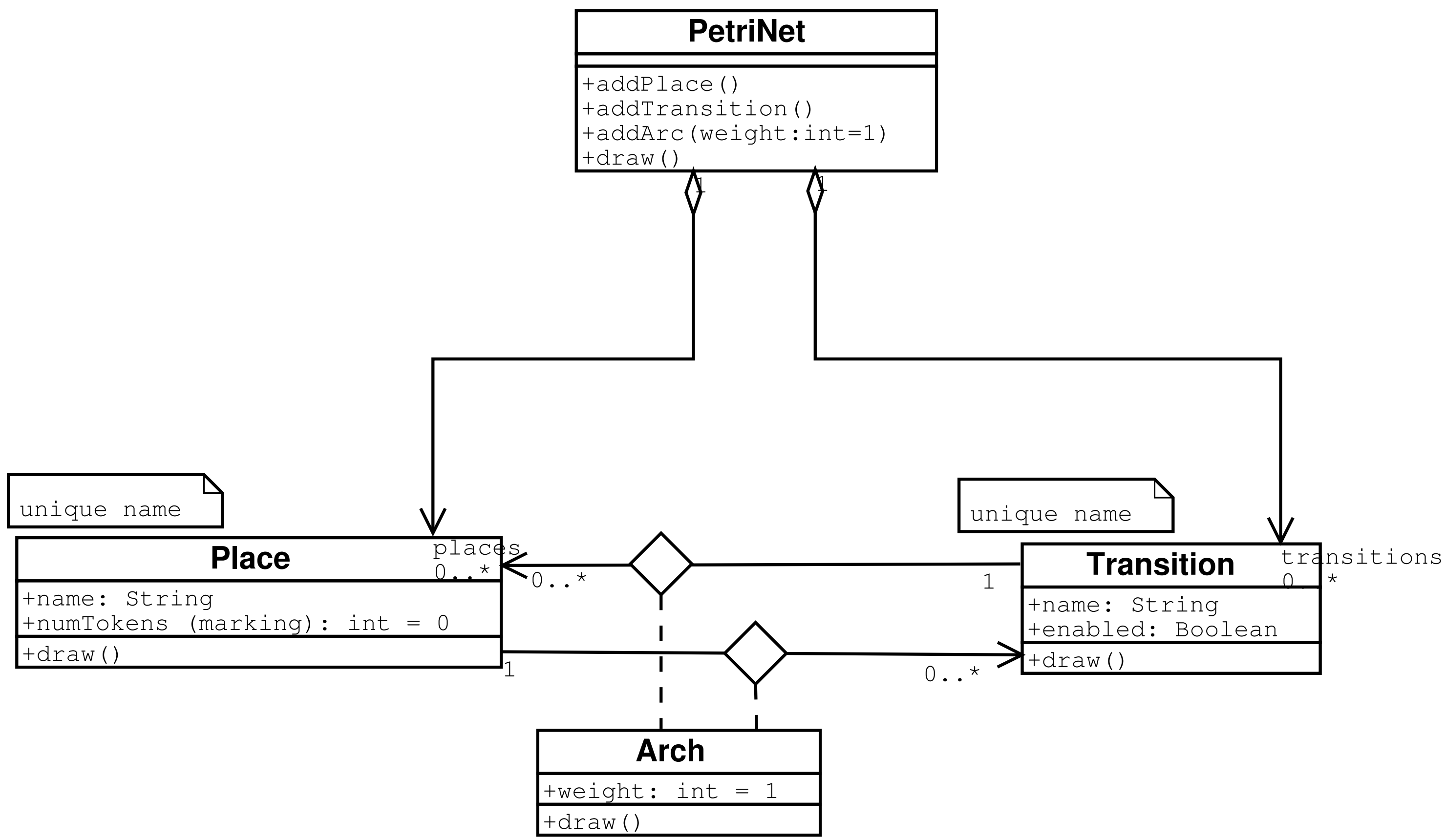
- Formalism similar to FSA
- Graphical notation
- C.A. Petri 1960s
- Additions to FSA:
  - Explicitly (graphically) represent when event is enabled
    $\rightarrow$ describe control logic
  - Elegant notation of concurrency
  - Express non-determinism

Petri net notation and definition (no dynamics)

$$(P, T, A, w)$$

- $P = \{p_1, p_2, \ldots\}$ is a finite set of *places*
- $T = \{t_1, t_2, \ldots\}$ is a finite set of *transitions*
- $A \subseteq (P \times T) \cup (T \times P)$ is a set of *arcs*
- $w : A \to \mathbb{N}$ is a *weight function*

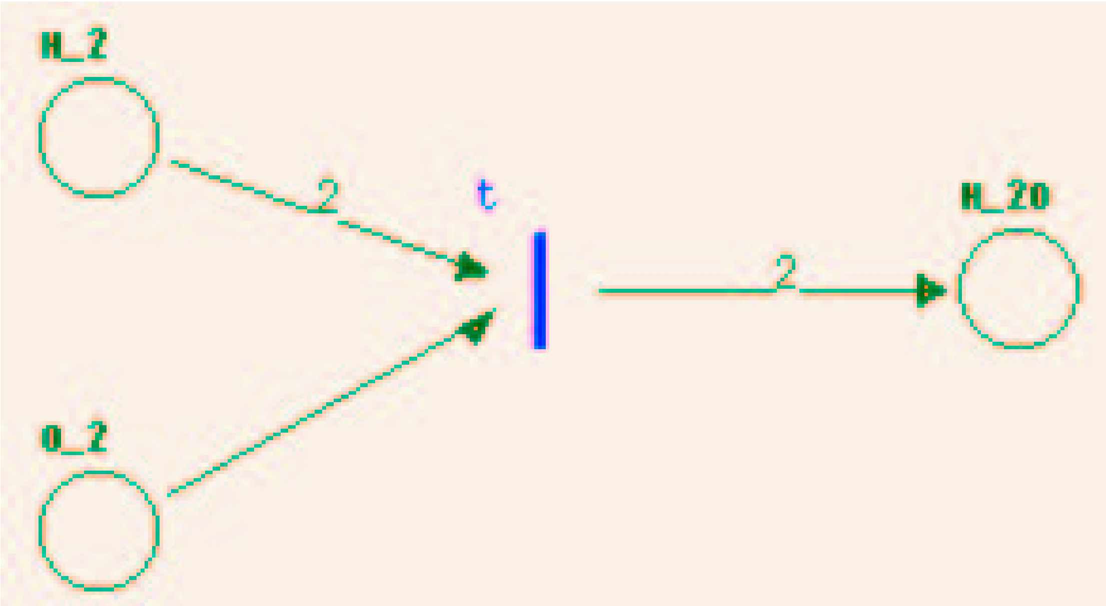
Class Diagram meta-model of Petri nets



Derived Entities

- $I(t_j) = \{p_i : (p_i, t_j) \in A\}$ set of *input places* to transition t_j
  ($\equiv$ conditions for transition)
- $O(t_j) = \{p_i : (t_j, p_i) \in A\}$ set of *output places* from transition t_j
  ($\equiv$ affected by transition)
- Transitions $\equiv$ events
- similarly: input- and output-transitions for p_i
- graphical representation: *Petri net graph* (multigraph)

Example Petri net



- $P = \{H_2, O_2, H_2O\}$
- $T = \{t\}$
- $A = \{(H_2, t), (O_2, t), (t, H_2O)\}$
- $w((H_2, t)) = 2, w((O_2, t)) = 1, w((t, H_2O)) = 2$

Introducing State: Petri net Markings

- Conditions met ? Use *tokens* in places
- Token assignment $\equiv$ *marking* x

$$x : P \rightarrow \mathbb{N}$$

- A marked Petri net

$$(P, T, A, w, x_0)$$

x_0 is the *initial marking*

- The *state* **x** of a marked Petri net

$$\mathbf{x} = [x(p_1), x(p_2), \ldots, x(p_n)]$$

Number of tokens need not be bounded (cfr. State Automata states).

State Space of Marked Petri net

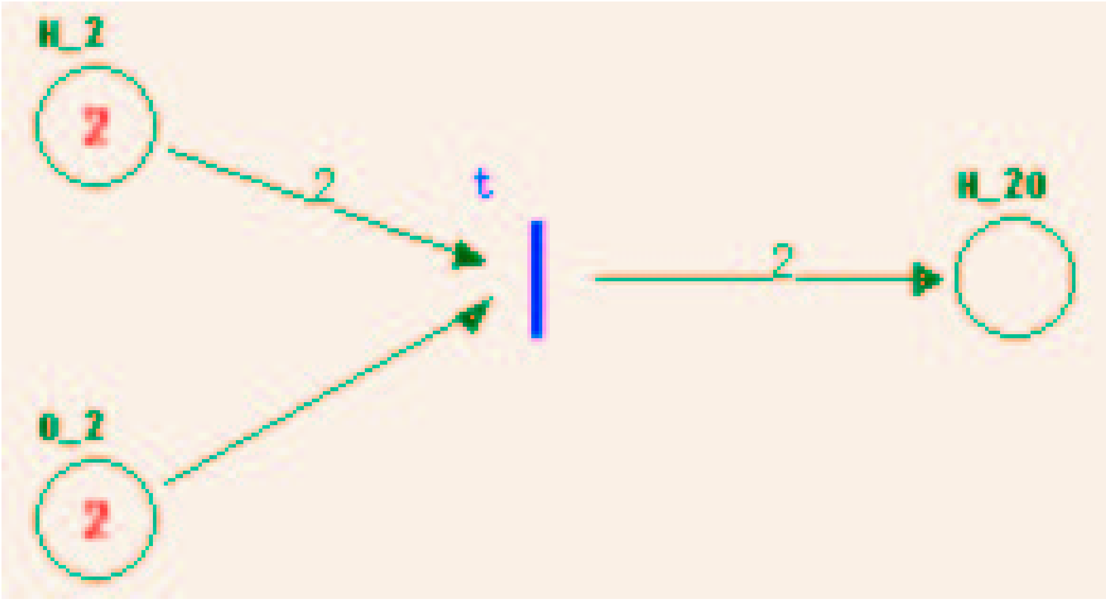
- All n-dimensional vectors of nonnegative integer markings

$$X = \mathbb{N}^n$$

- Transition $t_j \in T$ is *enabled* if

$$x(p_i) \geq w(p_i, t_j), \forall p_i \in I(t_j)$$

Example with marking, enabled



Petri Net Dynamics

State Transition Function f of marked Petri net (P,T,A,w,x_0)

$$f : \mathbb{N}^n \times T \rightarrow \mathbb{N}^n$$

is defined for transition $t_j \in T$ if and only if

$$x(p_i) \geq w(p_i,t_j), \forall p_i \in I(t_j)$$

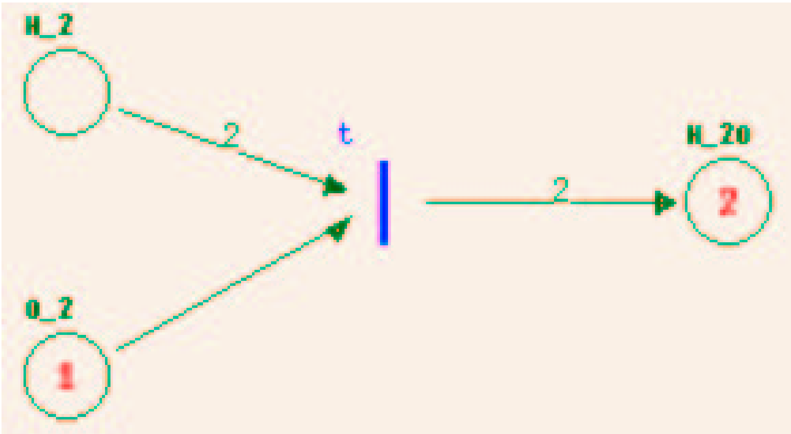
If $f(\mathbf{x},t_j)$ is defined, set $\mathbf{x}' = f(\mathbf{x},t_j)$ where

$$x'(p_i) = x(p_i) - w(p_i,t_j) + w(t_j,p_i)$$

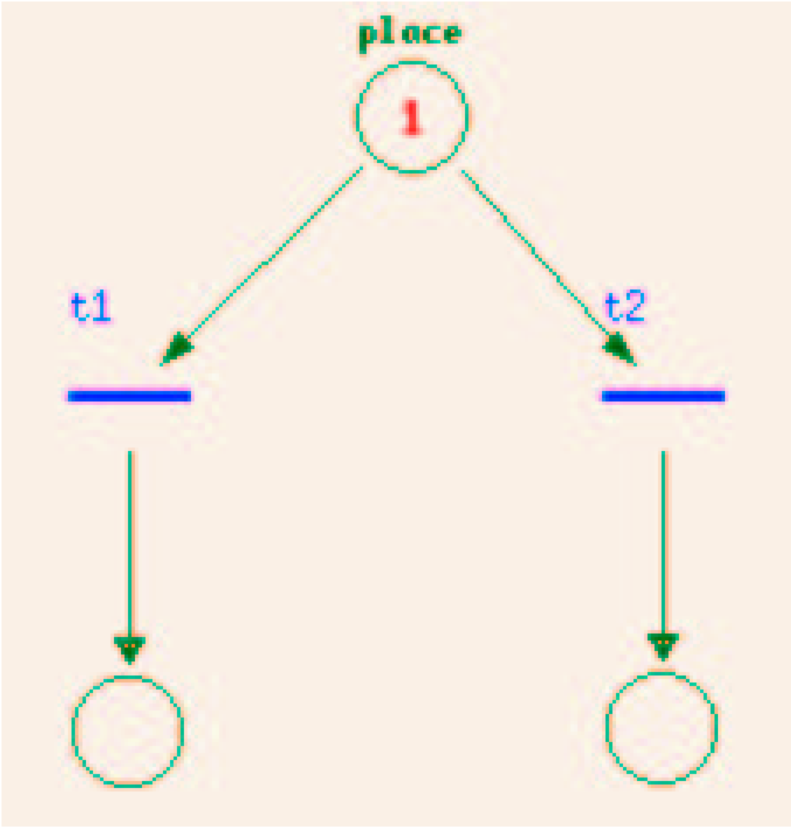
- State transition function f based on *structure* of Petri net
- Number of tokens *need not be conserved* (but can)

Example "firing"

- Use PNS tool http://www.ee.uwa.edu.au/ braunl/pns/
- Select Sequential Manual execution
- Transition: $[2, 2, 0] \rightarrow [0, 1, 2]$



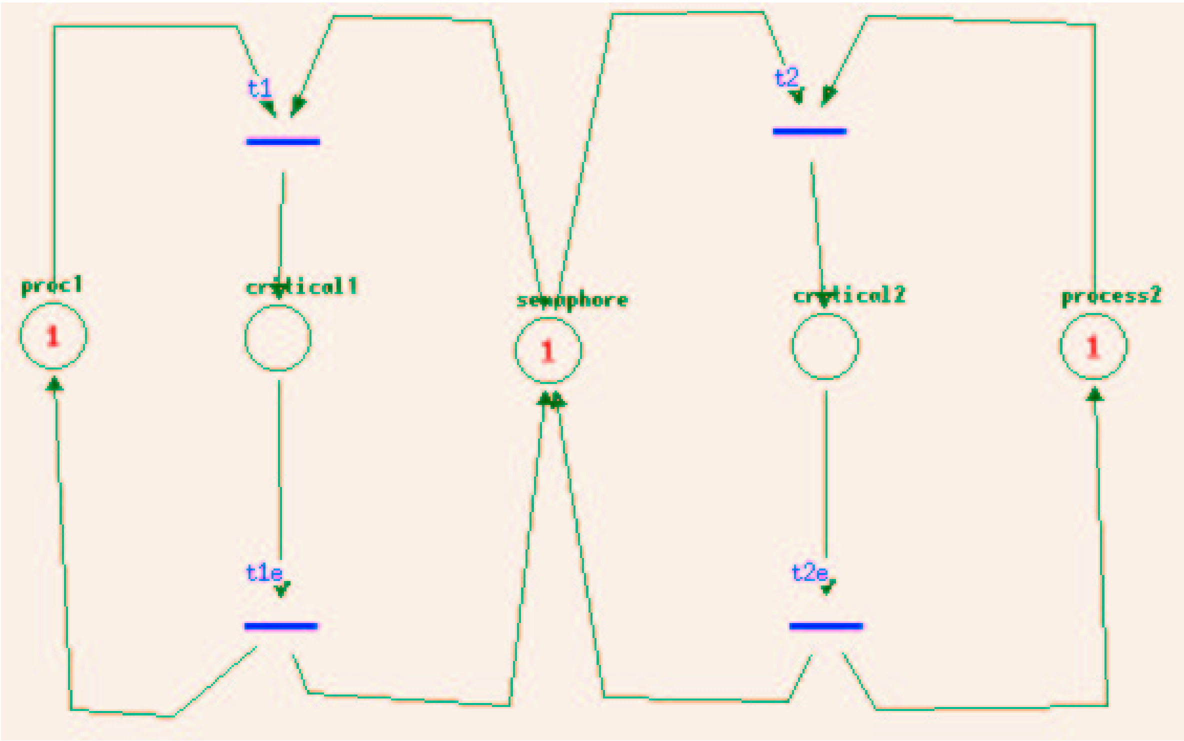
Conflict, choice, decision



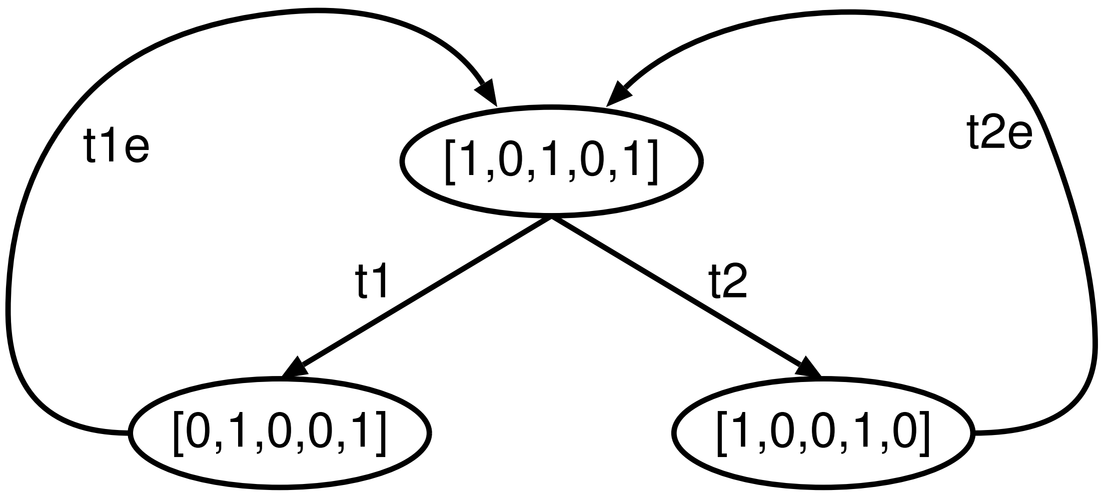
Semantics

- *sequential* vs. *parallel*
- Handle nondeterminism:
  1. User choice
  2. Priorities
  3. Probabilities (Monte Carlo)
  4. Reachability Graph (enumerate all choices)

Application: Critical Section



Reachability Graph



Representing a Petri net as a State Machine

Construct Reachability Graph

- Reachability Graph is State Machine
- States are tuples $(p_1, p_2, \ldots, p_n)$
- Events correspond to t_i firing
- May be infinite

Representing a State Machine as a Petri net

1. no output
2. with output

$\Rightarrow$ automatic (though inefficient) transformation

Modular Composition: Communication Protocol

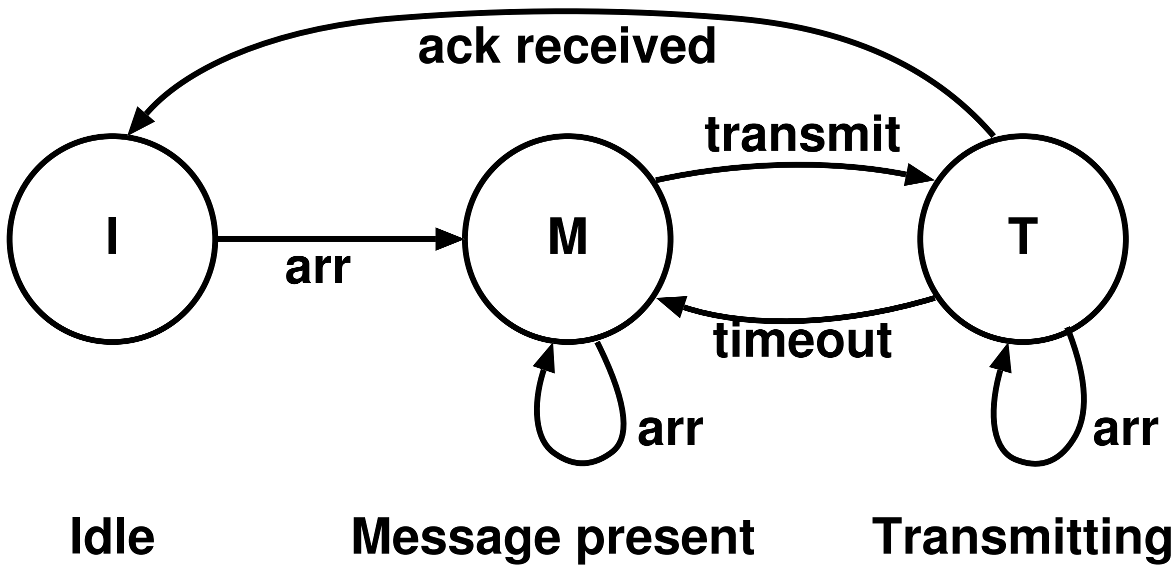
Build incrementally:

1. Single transmitter: FSA vs. Petri net
2. Two transmitters competing for channel

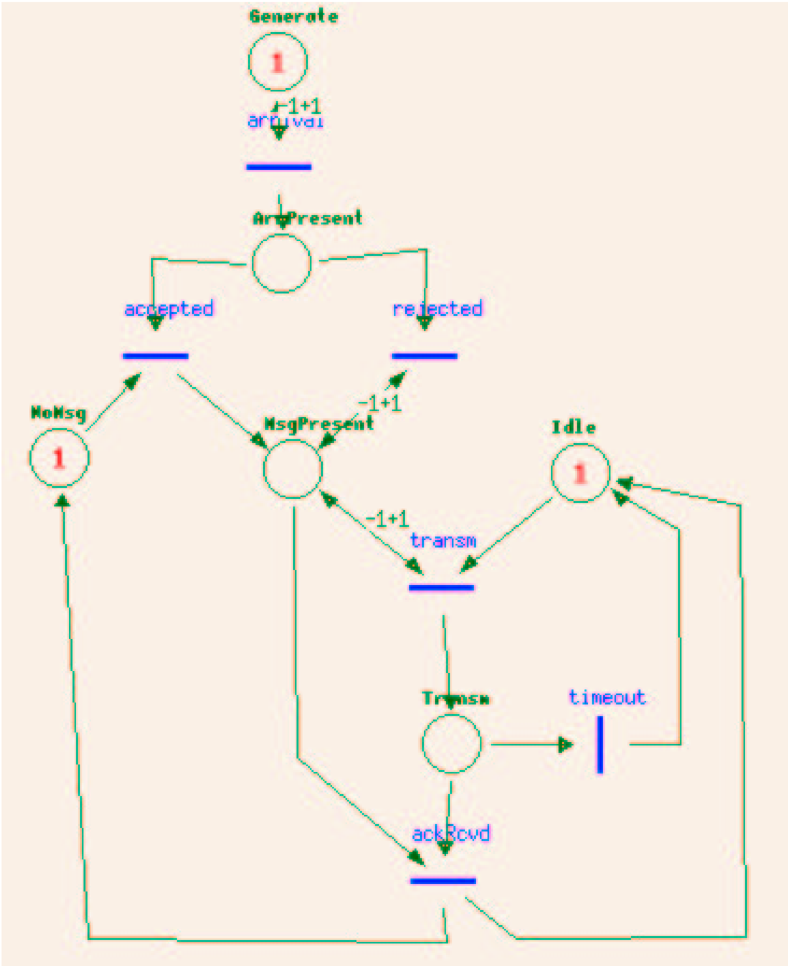
Pros/Cons of Petri net models (depends on goals !):

- Petri net is more complex than FSA for single transmitter
- More insight
- Incremental modelling
- Modular modelling
- Intuitive modelling of concurrency

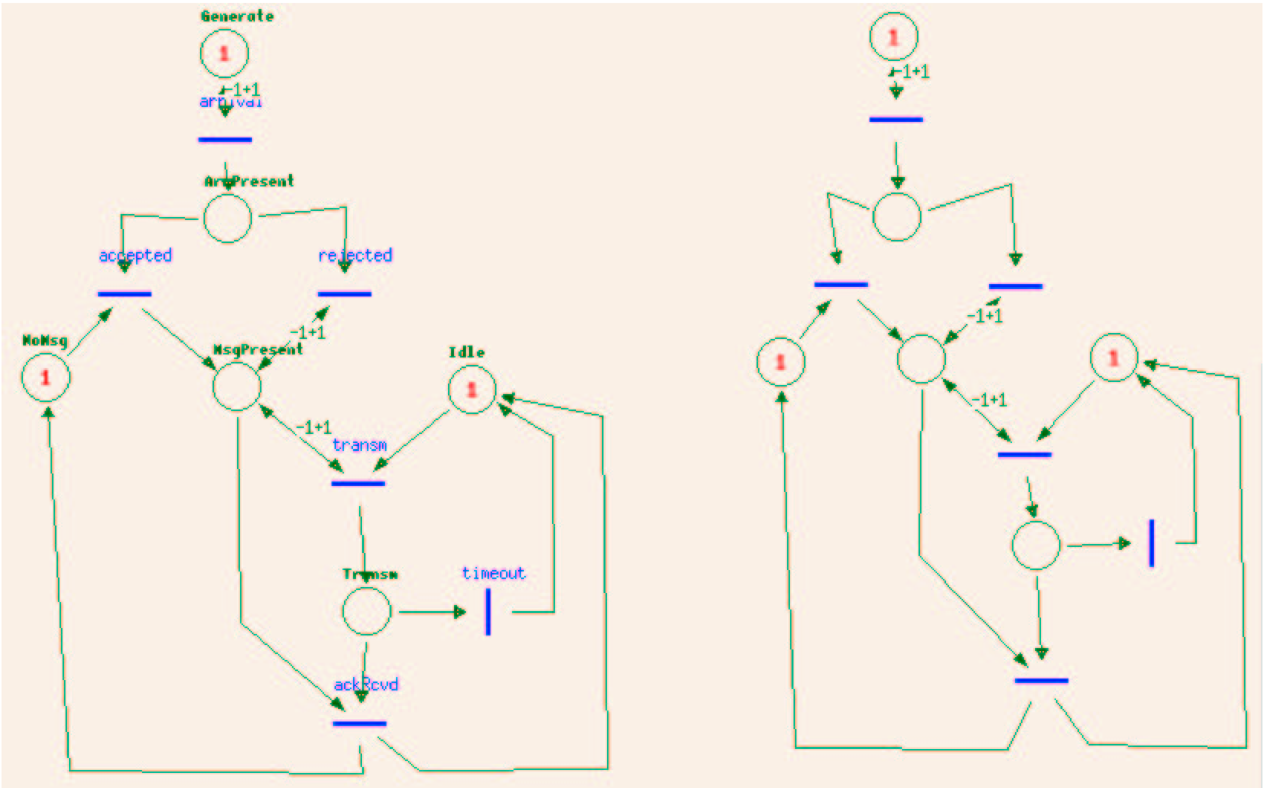
Single Transmitter FSA



Single Transmitter Petri net



Concurrent, Non-interacting Transmitters



Concurrent, Interacting Transmitters