HCI: Dialog Design Using Petri Nets

Learning Objective

- In the previous lecture, we discussed about State-Charts, a formalism suitable for dialog design, which is potentially more expressive than State Transition Networks (STNs)
- In this lecture, we shall discuss a powerful formalism for dialog design, namely the (classical) Petri Nets

(Classical) Petri Net (PN)

- The formalism was first proposed by Carl Adam Petri (1962, PhD thesis)

- It is a simple model of dynamic behavior
  - Just four elements are used to represent behavior: **places**, **transitions, arcs and tokens**
  - Graphical and mathematical description for easy understanding
  - Formal semantics allow for analysis of the behavior

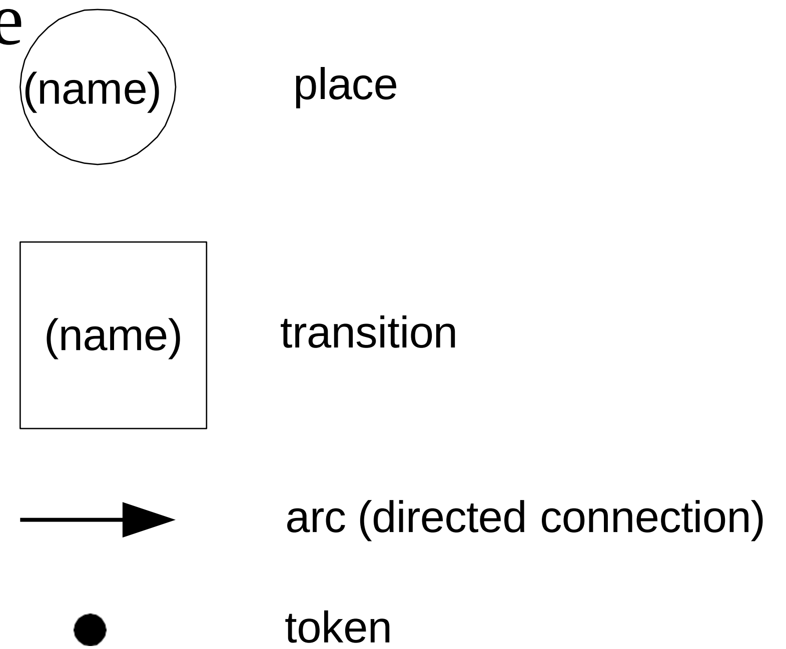
Elements of PN

- Place: used to represent passive elements of the reactive system
- Transition: used to represent active elements of the reactive system
- Arc: used to represent causal relations
- Token: elements subject to change

The state (space) of a process/system is modeled by places and tokens and state transitions are modeled by transitions

Elements of PN: Notation

- A place is represented by a circle
- Transitions are represented by squares/rectangles
- Arcs are represented by arrows
- Tokens are represented by small filled circles



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
  - A **collection of objects**, for example a truck with products, a warehouse with parts, or an address file
  - An **indicator of a state**, for example the indicator of the state in which a process is, or the state of an object
  - An **indicator of a condition**: the presence of a token indicates whether a certain condition is fulfilled

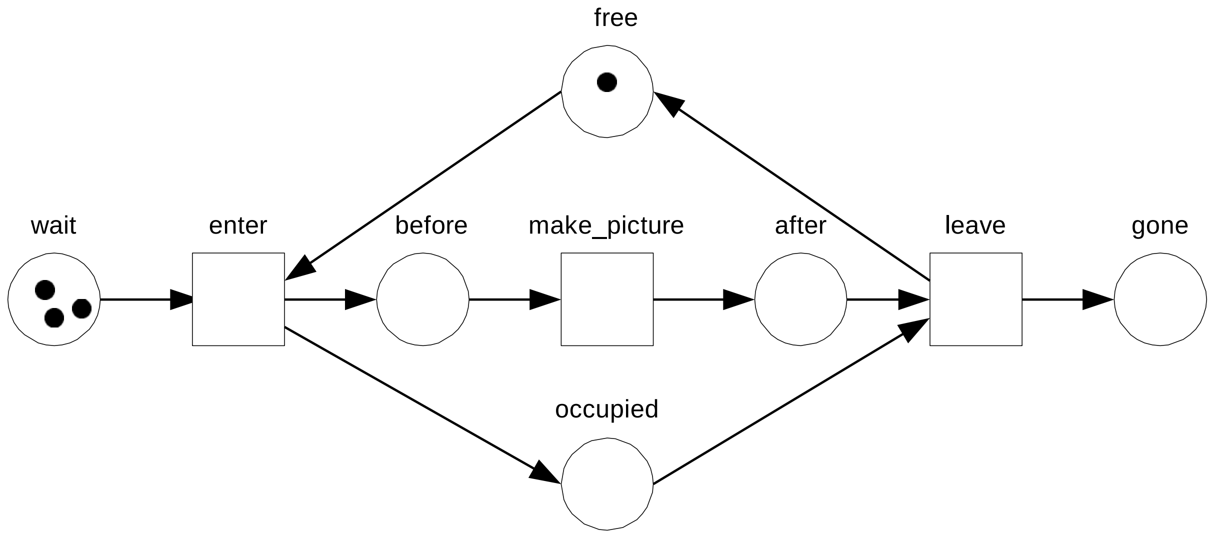
Role of a Place

- A place in a PN can represent the following
  - 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 a warehouse, 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

- A transition can be used to represent things such as
  - An **event** (e.g., starting an operation, 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

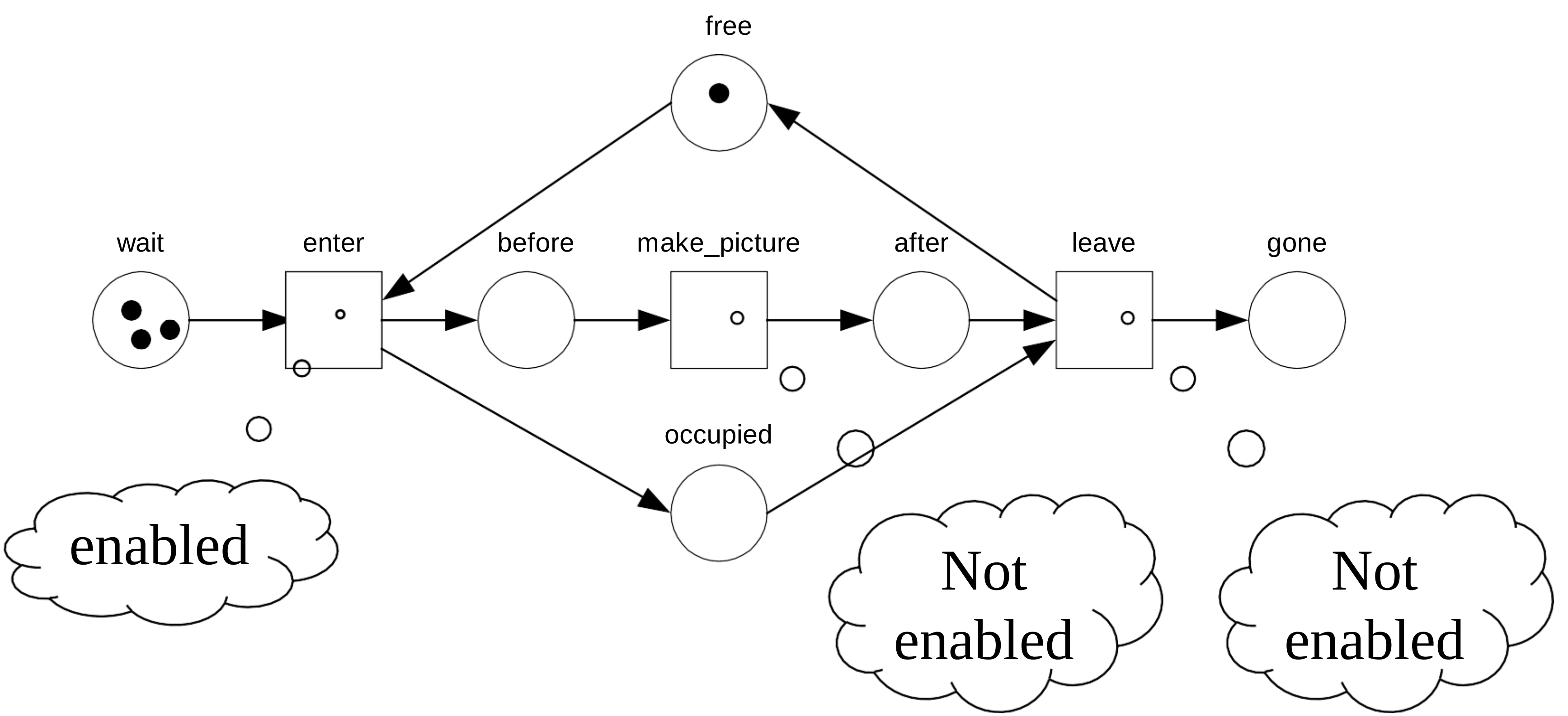
PN Construction Rules



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

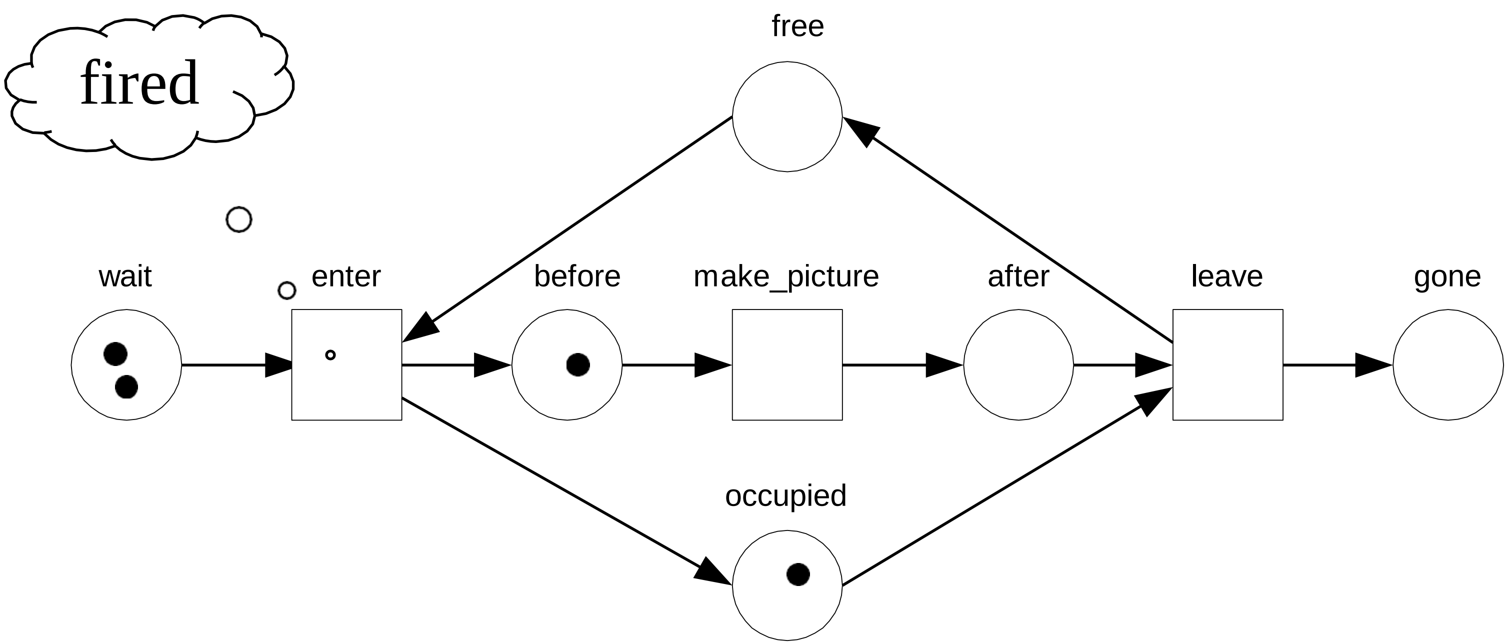
Enabled

- 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

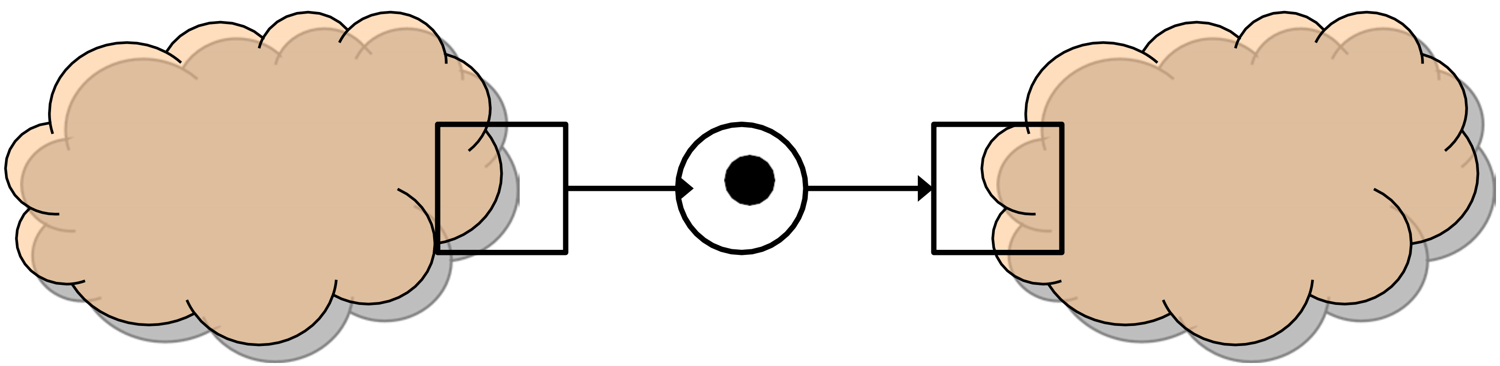


Remarks

- Firing is **atomic** (i.e., it always completes after start)
- Non-determinism: multiple transitions may be enabled, but only one fires at a time
- The **state** of the reactive system is represented by the distribution of tokens over places (also referred to as **marking**)

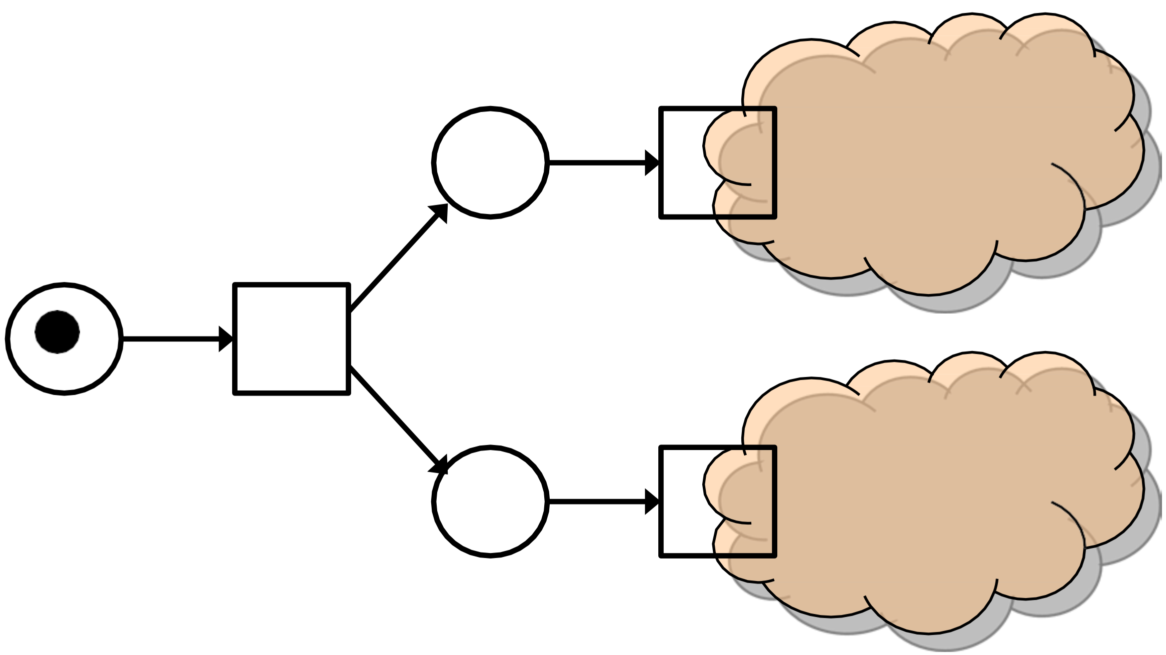
Typical PN Structures

- **Causality**, i.e., one part of the PN is caused by the other part



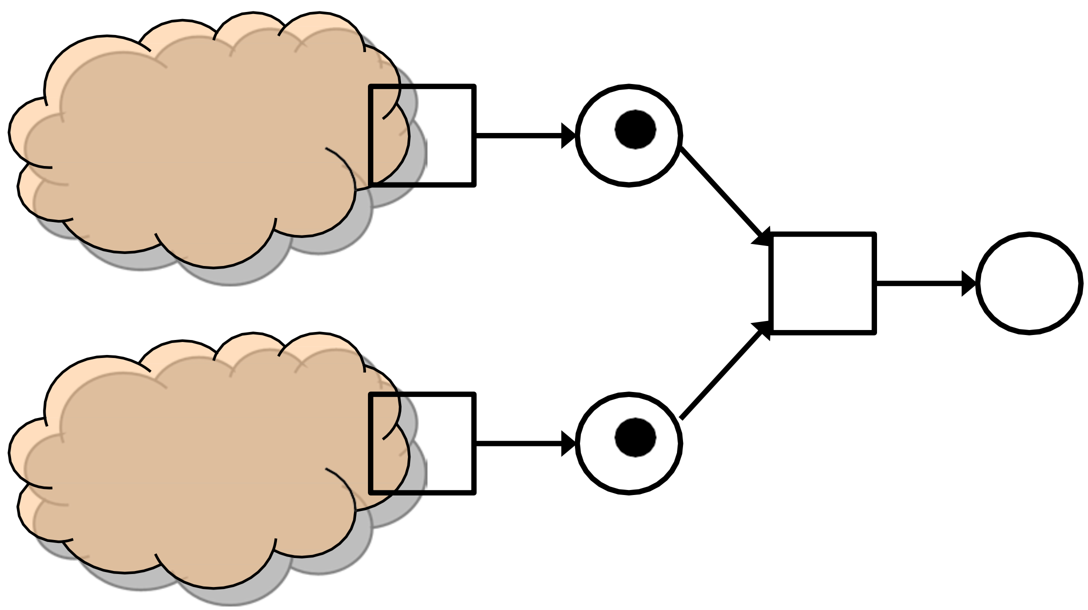
Typical PN Structures

- **Parallelism (AND-split)**, i.e., two parts of the PN can be activated at the same time



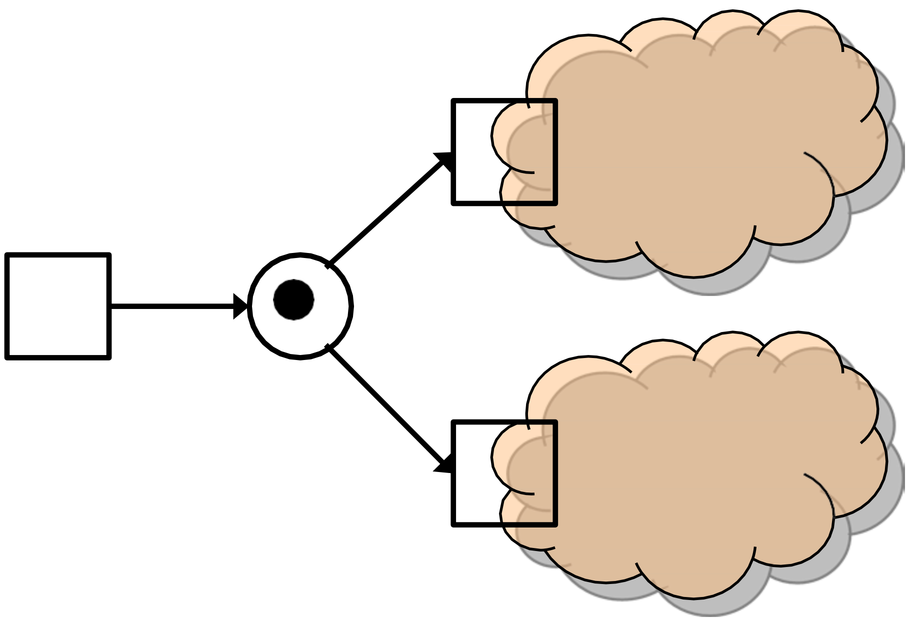
Typical PN Structures

- **Parallelism (AND-join)**, i.e., two parts of the PN must be active at the same time, or enable further firings



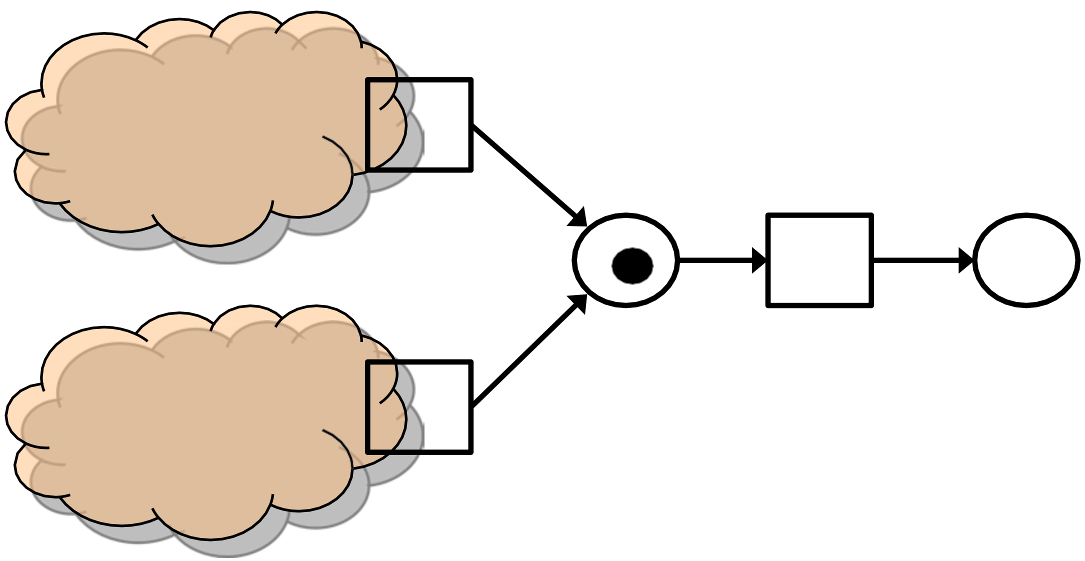
Typical PN Structures

- **Choice (XOR-split)**, i.e., either of the two sub nets of a PN can be activated



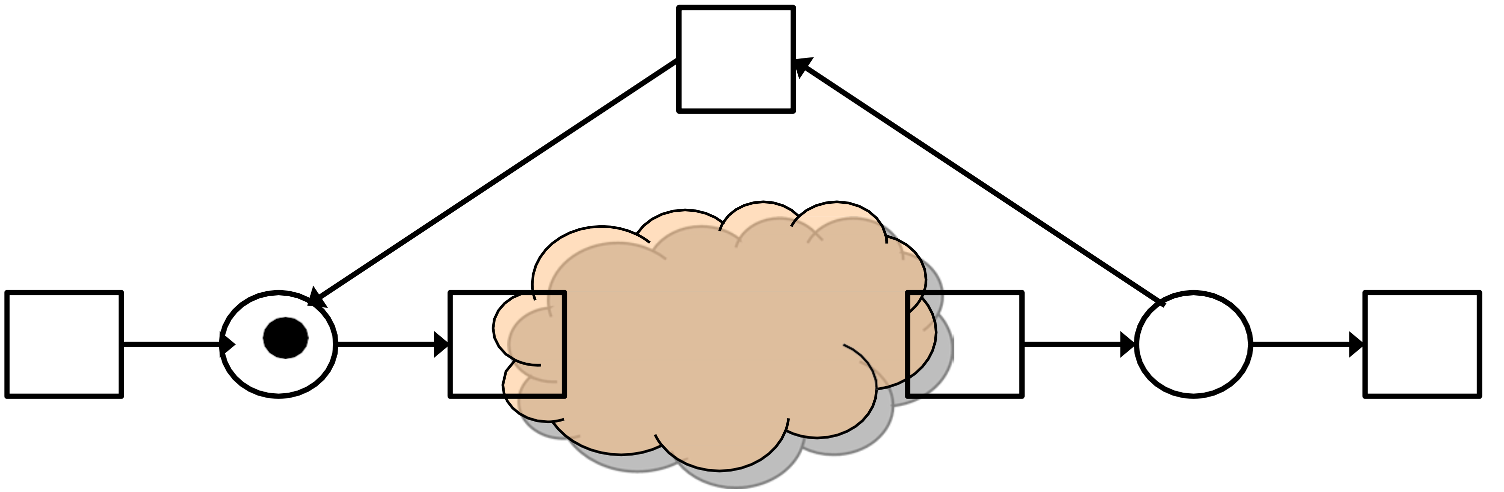
Typical PN Structures

- **Choice (XOR-join)**, i.e., either of the two sub nets of a PN is an enabler



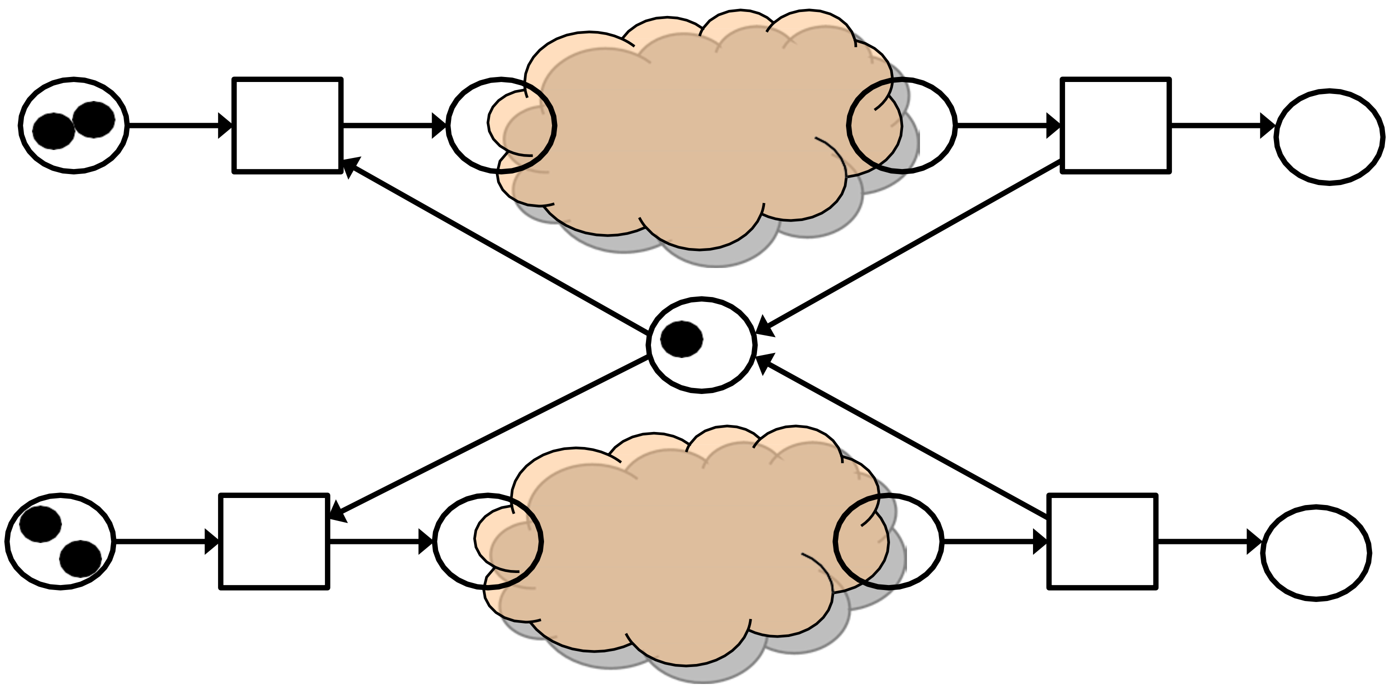
Typical PN Structures

- **Iteration (1 or more times)**, i.e., the firing iterates at least once



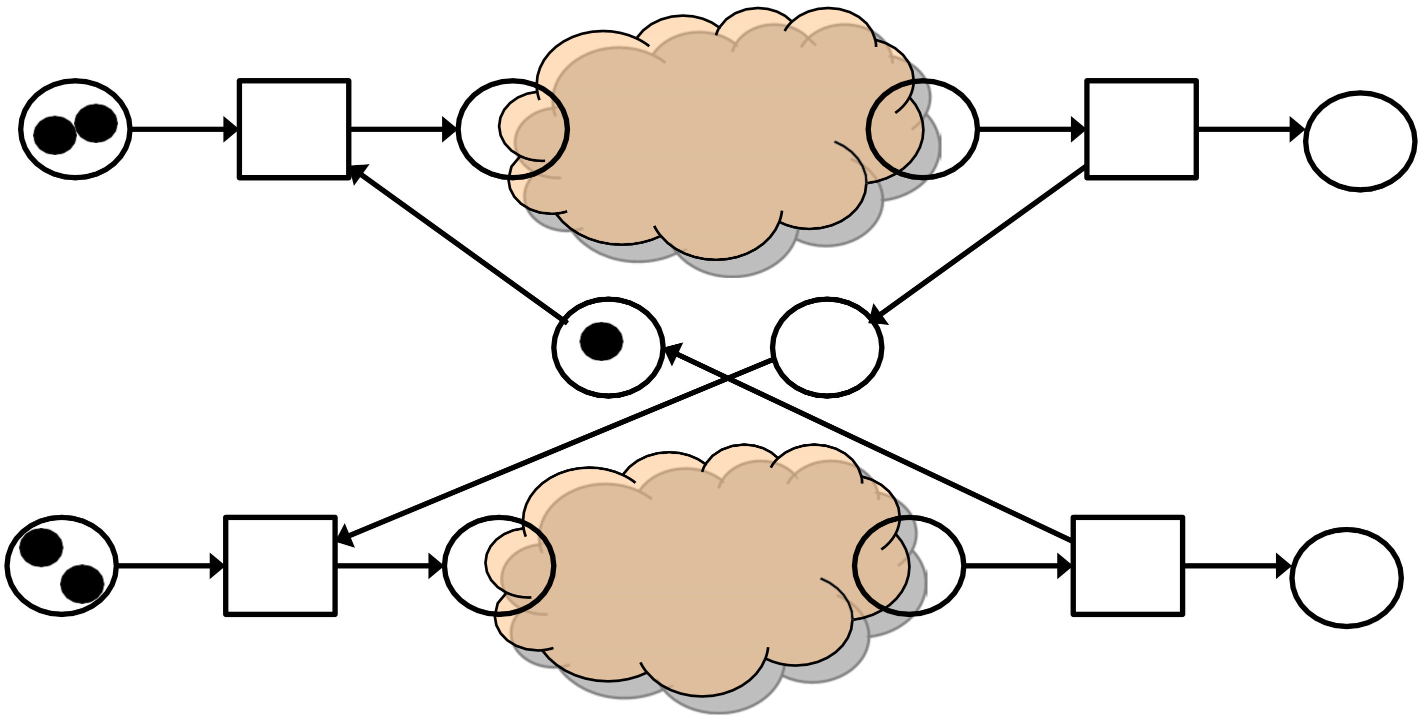
Typical PN Structures

- **Mutual exclusion**, i.e., only one of the sub nets should be active at a time



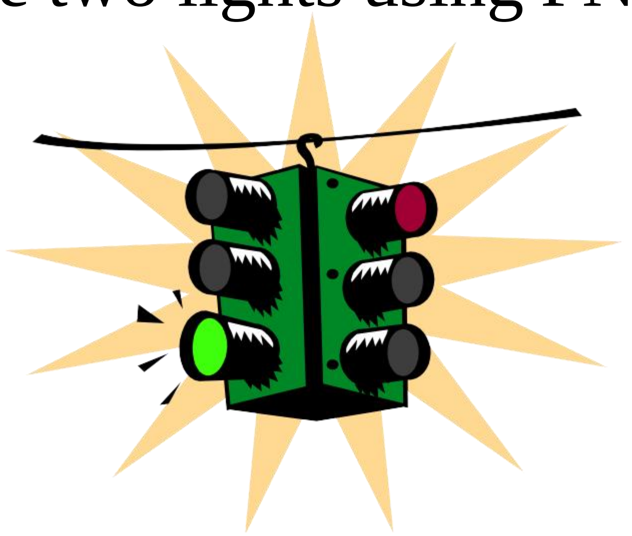
Typical PN Structures

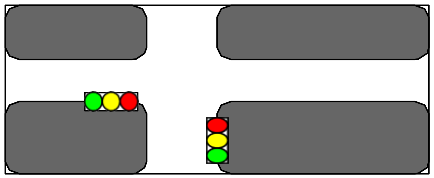
- **Alternating**, i.e., the sub nets of a PN should be alternatively activated



Example: Two Traffic Lights

- Let us illustrate the idea with an example. Suppose there are two traffic lights at a road junction. How we can model the behavior of these two lights using PN?





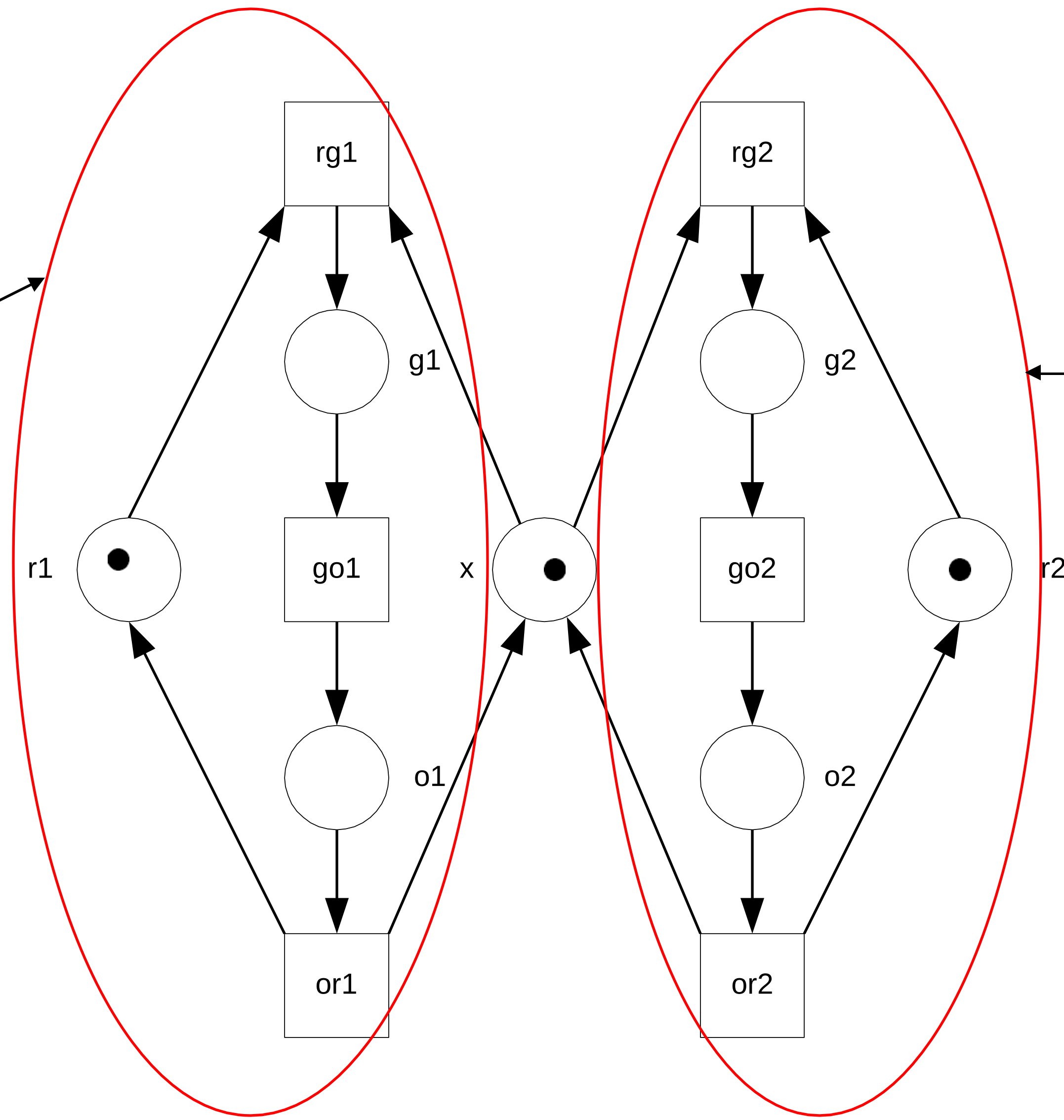
Example: Two Traffic Lights

- The characteristics of the combined system (of two lights)
  - They are mutually exclusive
  - They should alternate
- We can use the typical structures to model the behavior

Example: Two Traffic Lights

One light

(r=red,
g=green,
o=orange)



The other light

(r=red,
g=green,
o=orange)

Summary

- Look closely in the example how the elements of a PN are used to model the behavior of the system
- In the next lecture, we shall discuss with an example the usefulness of formal dialog representation
- Also we shall discuss about the properties we check with the formalisms and how?