Patrons

D'estat

Patró State → Patró de comportament

El patró d'estat també és coneix amb el nom de Objectes per estat.

S'usa per una aplicació que:

- Quan el comportament d'un objecte depèn de l'estat i s'ha de canviar en temps d'execució.
- Quan les operacions tenen llargues sentencies condicionals amb múltiples branques que depenen de l'estat del objecte.
- Com a màquina d'estats

Exemples

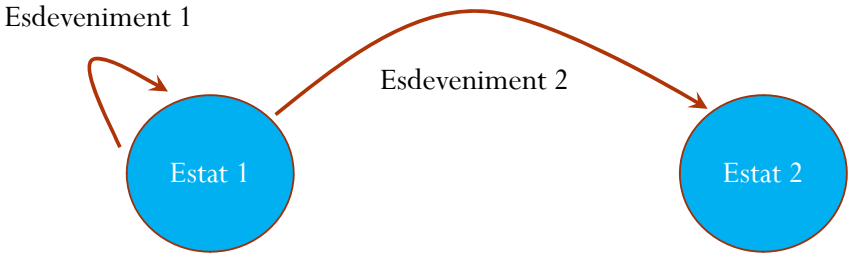
- En una connexió TCP on tindrem els estats de connectat, desconnectat i inicialitzar la connexió
- es troba molt sovint en les eines dels editors de dibuixos i en els editors de text, per exemple quan es selecciona si es vol escriure en un full A4 o en un sobre, etc.

Motivació:

- Canviar el comportament depenent de l'estat.
- Quan volem que un objecte canviï el seu comportament, segons canvia el seu estat, es presenta el problema de la complexitat de codi.

Exemple: Màquina d'estats finits (FSM)

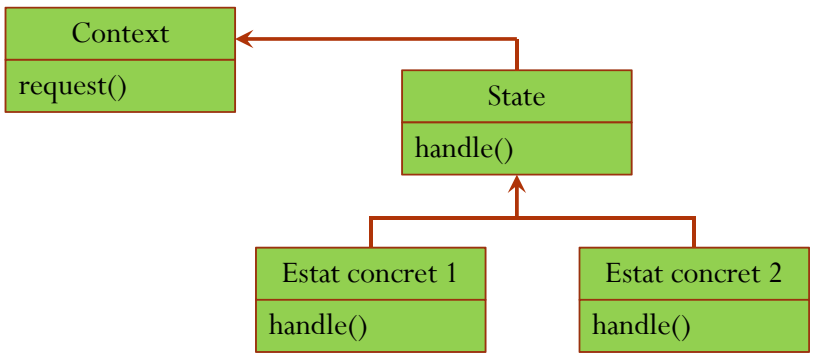
- Els processos d'Erlang es poden utilitzar per implementar màquines d'estat finits.
- Un FMS és un model que consisteix en un nombre finit d'estats i esdeveniments. Depenent de l'esdeveniment i de l'estat actual del FMS es produiran un conjunt d'accions i una transició a un nou estat.



- Cada estat es representa com una funció d'una cua recursiva, i cada esdeveniment es representa com un missatge entrant.
- Quan es rep un missatge es compara amb una clàusula de recepció
  - si es compleix s'executen un conjunt d'accions i
  - Pot haver-hi una transició d'estat mitjançant la crida la funció corresponent del nou estat.

Exemple: Màquina d'estats finits — Patró d'estat

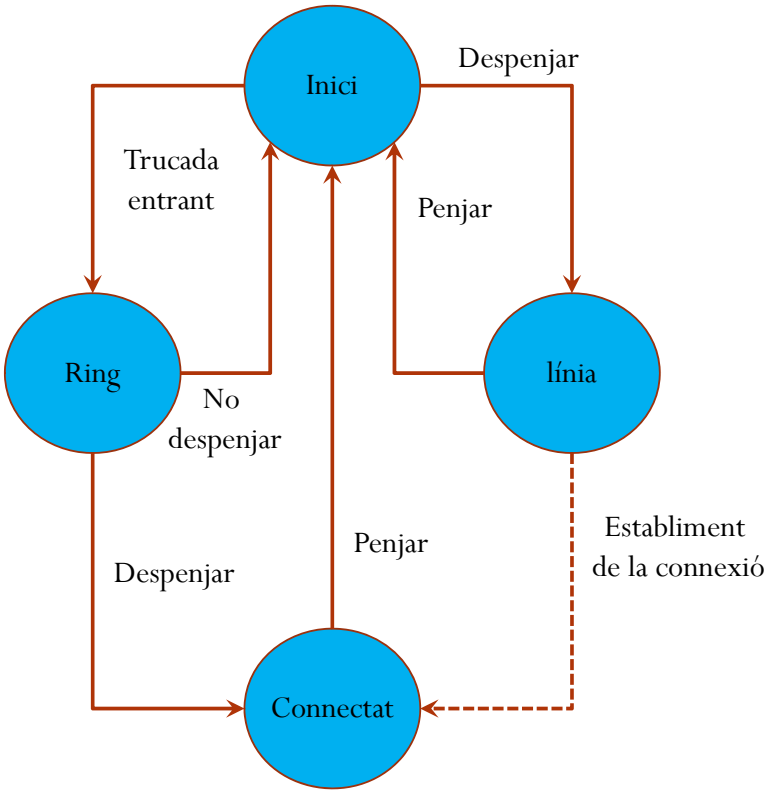
- Estructura:



- Participants:
  - Context: deneix una interfcie pels clients i mante una instancia d'un estat concret que defineix l'estat actual del procés "Context".
  - State: defineix una interfcie per encapsular el comportament associat amb estat particular del context.
  - Concrete State: implementa un comportament associat amb l'estat del context.

Exemple: Telèfon fixa

- Es vol modelar un telèfon d'una línia fixa amb una màquina d'estats finits



Exemple: Telèfon fixa. Solució

- S'ha de fer un procés per a cadascun dels estats

Inici

```
idle() ->
    receive
        {Number, incoming} ->
            start_ringing(),
            ringing(Number);
        off_hook ->
            start_tone(),
            dial()
    end.
```

Trucada entrant

Funció per fer sonar el telèfon

Salt a l'estat de Ring

Despenjar el telèfon

Funció per activar el to

Salt a l'estat de línia

Exemple: Telèfon fixa. Solució

- S'ha de fer un procés per a cadascun dels estats

Ring

```
ringing(Number) ->
    receive
        {Number, other_on_hook} ->
            stop_ringing(),
            idle();
        {Number, off_hook} ->
            stop_ringing(),
            connected(Number)
    end.
```

No despenjar el telèfon
Tall de la línia, …

Funció per aturar el so del telèfon

Salt a l'estat inicial

Despenjar el telèfon

Salt a l'estat de connectat

Exemple: Telèfon fixa. Solució

Línia

```
dial() ->
    receive
        {Number, other_state} ->
            connected(Number);
        on_hook ->
            idle()
    end.
```

Quan s'ha marcat el numero i s'estableix la trucada

Salta a l'estat connectar

Penjar el telèfon

Connectat

```
connected(Number) ->
    receive
        on_hook ->
            idle()
    end.
```

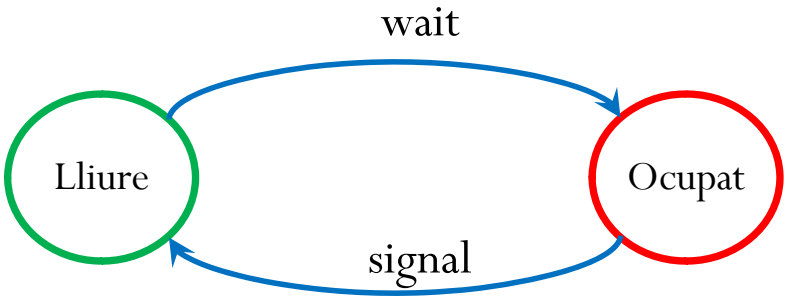
Penjar el telèfon

```
start_ringing() -> ...
start_tone()    -> ...
stop_ringing()  -> ...
```

Exemple: Semàfor amb exclusió mútua. Proposta

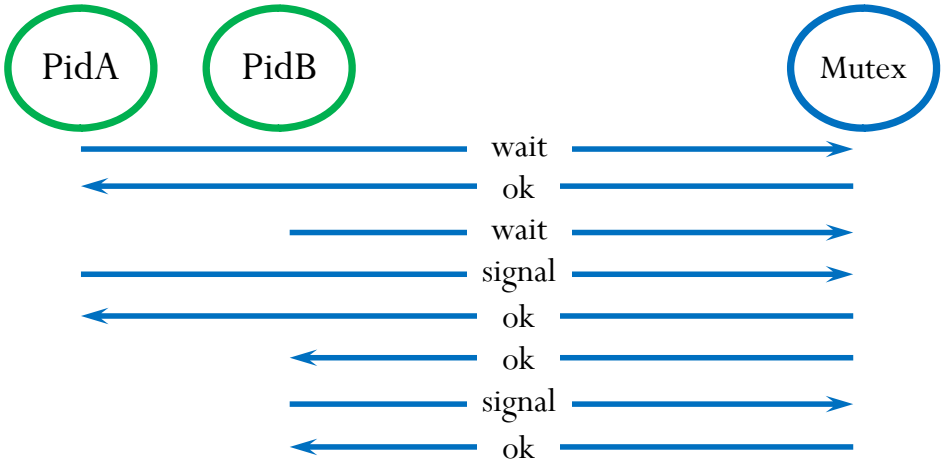
- Un semàfor és un procés que 'serializa' l'accés a un recurs en particular, garantint l'exclusió mútua.
- El semàfor està pensat per llenguatges que treballen amb la memòria compartida.
- Es poden utilitzar com a mecanisme general per gestionar qualsevol recurs, no només la memòria.
- Suposem que només pot haver-hi un procés utilitzant un servidor de fitxers al mateix temps, per garantir que no hi ha dos processos llegint o escrivint simultàniament.

Exemple: Semàfor amb exclusió mútua. Proposta



- Abans de fer qualsevol crida al servidor de fitxers, el procés que vol accedir al fitxer crida a la funció mutex:wait(), imposant un bloqueig al servidor.
- Quan el procés acabi la gestió dels fitxer, crida a la funció mutex:signal(), i elimina el bloqueig.

Exemple: Semàfor amb exclusió mútua. Proposta



- Si un procés PidB crida mutex:wait () quan el semàfor està ocupat amb PidA,
- PidB queda en una clàusula de recepció fins que PidA cridi a signal/0.
- El semàfor quedarà disponible, i el procés espera el primer missatge de la cua de missatges, PibB, i podrà accedir al servidor de fitxers.

Exemple: Semàfor amb exclusió mútua. Solució

```
-module(mutex).
-export([start/0, stop/0]).
-export([wait/0, signal/0]).
-export([init/0]).

start() ->
    register(mutex, spawn(?MODULE, init, [])).

stop() ->
    mutex ! stop.

free() ->
    receive
        {wait, Pid} ->
            Pid ! Ok,
            busy(Pid);
        stop ->
            terminate()
    end.
```

```
signal() ->
    mutex ! {signal, self()},
    ok.

init() ->
    free().
```

```
wait() ->
    mutex ! {wait, self()},
    receive
        ok ->
            ok
    end.

busy(Pid) ->
    receive
        {signal, Pid} ->
            free()
    end.
```

```
terminate() ->
    receive
        {wait, Pid} ->
            exit(Pid, kill),
            terminate()
    after
        0 -> ok
    end.
```