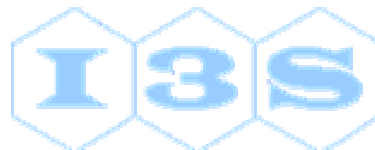


# Les Objets + le Synchrone dans les systèmes temps réel

Charles André - Projet SPORTS



# Le projet SPORTS

Synchronous

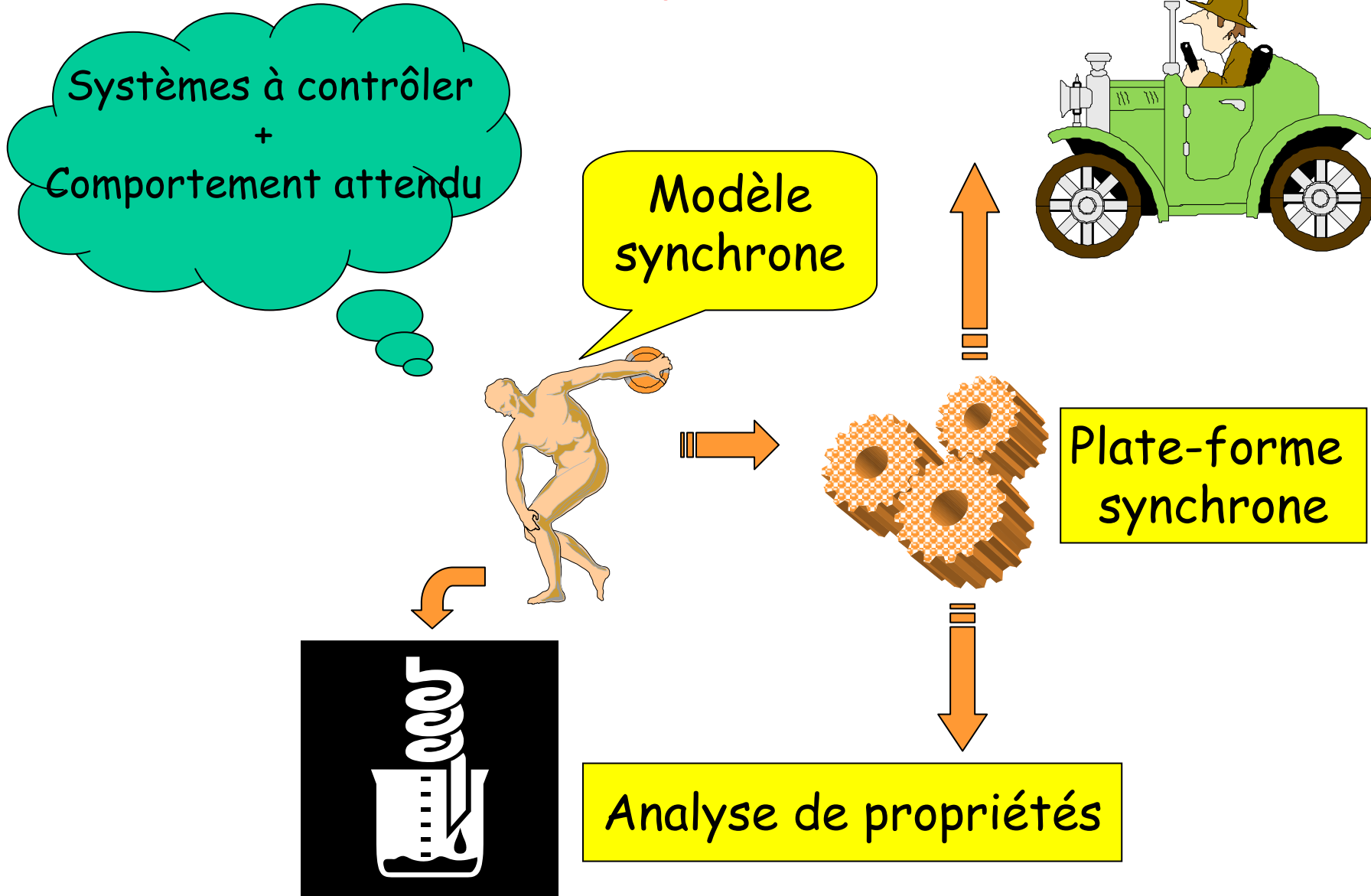
Programming Of

Real-Time | Reactive

Systems

Web site: <http://www.i3s.unice.fr/~map/WEBSPORTS>

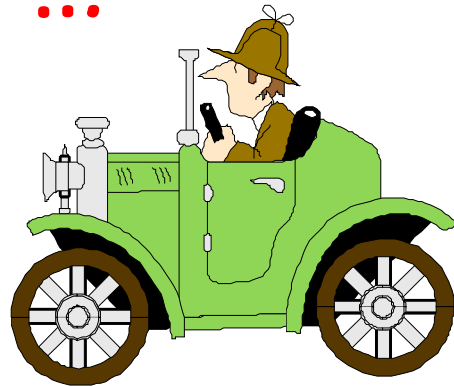
# Nos objectifs ...



# Nos contributions ...

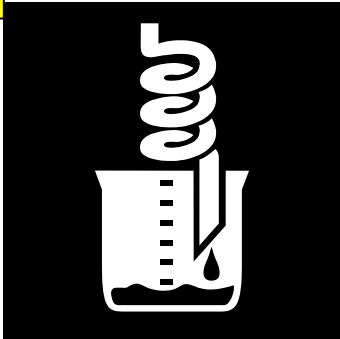
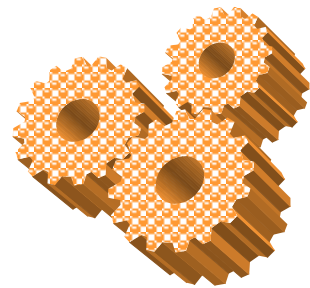
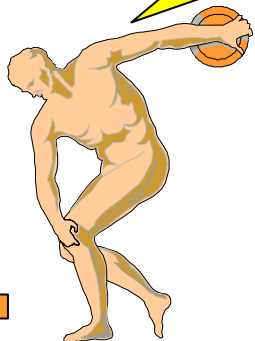
Systemes à contrôler  
+  
Comportement attendu

SyncCharts  
S-Grafcet  
SEP, SIB



Machines  
d'exécution

Expression  
des  
propriétés



Garanties de  
propriétés

Evaluation de  
performance

# Objets et systèmes T.R

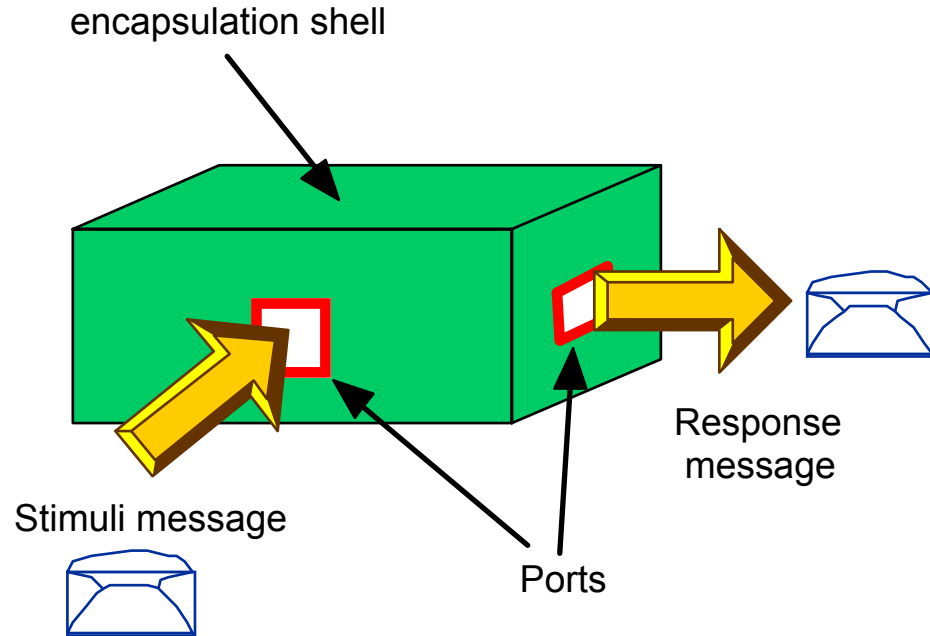
- **ROOM** 1994 (**R**eal-Time **O**bject-**O**riented **M**odeling) B. Selic, G. Gullekson, P. Ward
- **UML** (Profile for concurrency and distribution) 1998 Bran Selic, Jim Rumbaugh
- **Rational Rose RealTime**
- **ROPES** (Rapid Object-oriented Process for Embedded Systems) 1999 B.P. Douglass

# Objets actifs (1)

- Like passive objects they encapsulate data. However, they can also **initiate control activity**, which they do by having **their own encapsulated thread of execution**.
- To support both asynchronous and synchronous **communication** with other active objects, each active object has a **message queue** or mailbox.
- On message reception, the **object executes** whatever code is appropriate to process the message.

*Garth Gullekson.*

# Objets actifs (2)

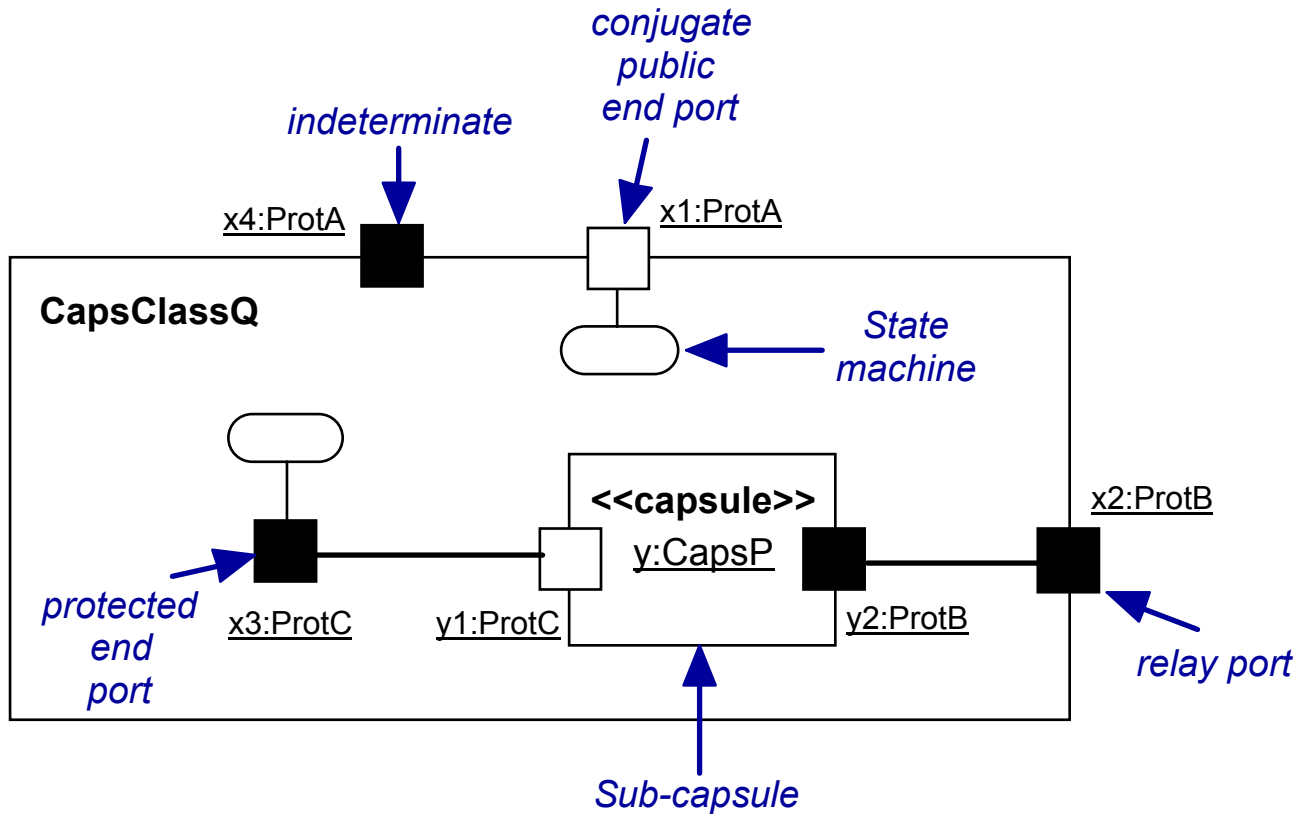


# ROOM

- **Actors** → **capsules** = complex physical, possibly distributed architectural objects that interact with their surroundings through one or more signal-based boundary objects
- **Ports** = physical part of the implementation of a capsule that mediates the interaction of the capsule with the outside world. Each port plays a **role** in a collaboration. A **protocol** captures the complex semantics of these interactions.
- **Bindings** → **Connectors** = abstract views of signal-based communication channels that interconnect two or more ports.



# UML stereotypes



# Objets synchrones

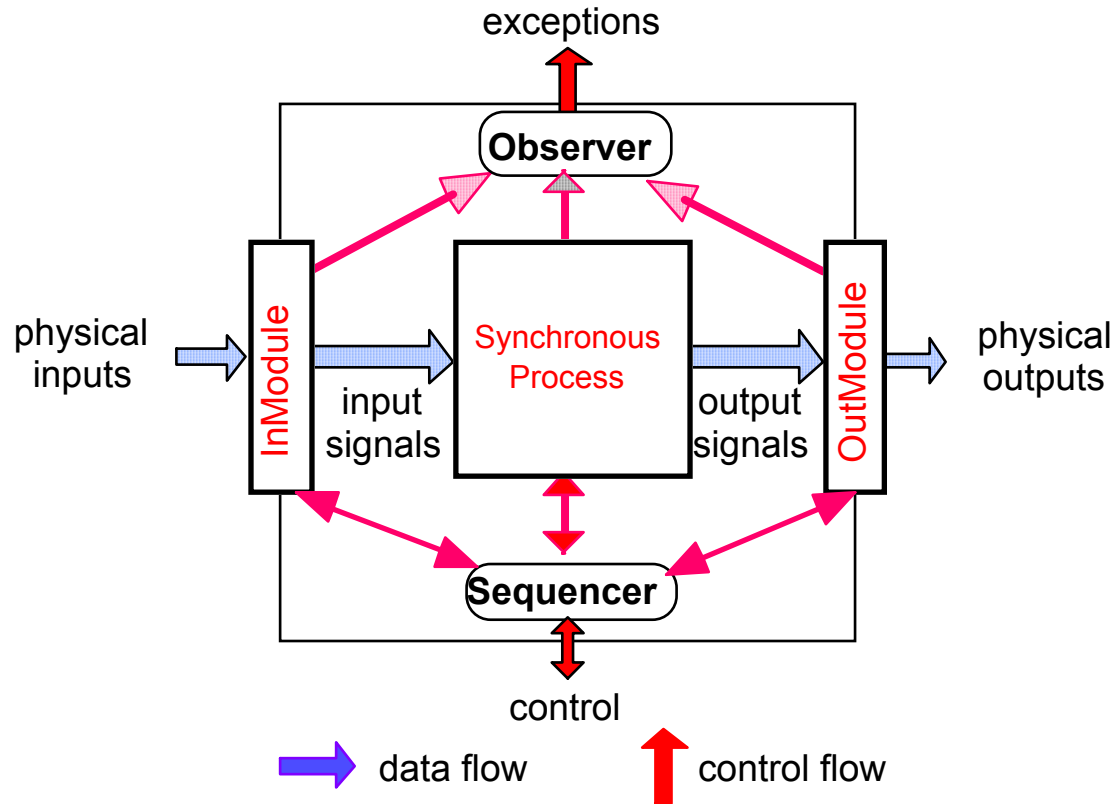
- F. Boulanger. *Intégration de Modules Synchrones dans la Programmation par Objets*. Paris XI, 1993.

## Programmes synchrones → objets

- (Abstract) Class Reactive\_Machine
- Derived: Class Synchronous
- Instances of Synchronous = Synchronous Objects
- Methods:
  - Begin\_of\_instant
  - Activate
  - End\_of\_instant
  - List\_of\_signals
  - ...
- Class library

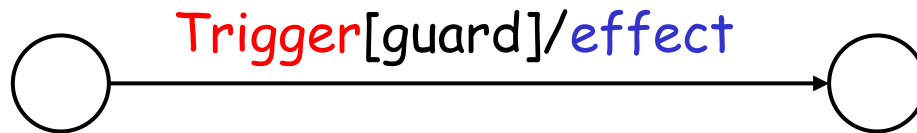
# Machines d'exécution

C. André, H. Boufaïed, *Execution Machine for Synchronous Languages*, IDPT 2000, Dallas (TX), June, 2000

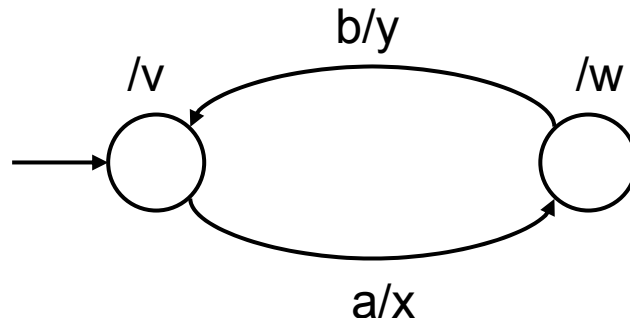


# Les SyncCharts (1)


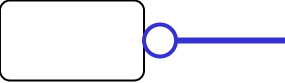
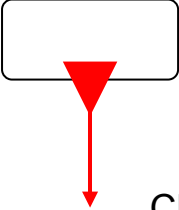
- Representation and Analysis of Reactive Behaviors : A Synchronous Approach. C.André *IEEE-SMC, CESA'96, pp 19-29, Lille (F) July 1996*
- Formalisme graphique **synchrone** inspiré Statecharts, compatible Esterel
- **Etats/Transitions**

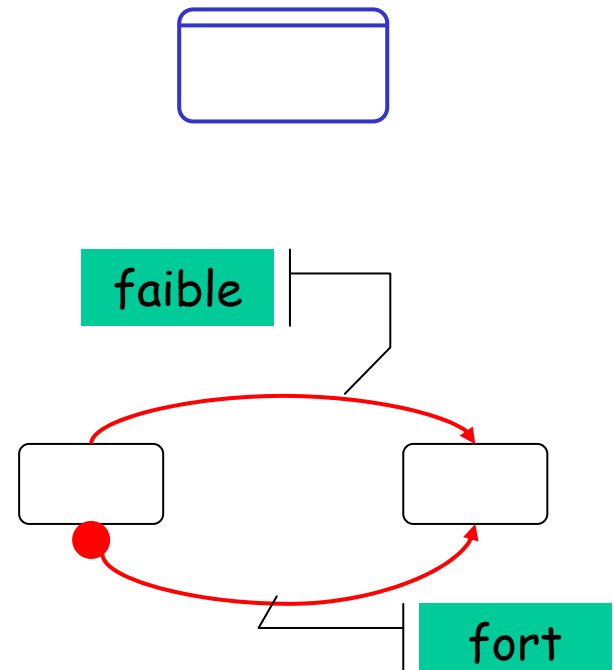


- Graphe d'états: effets associés états ou transitions



# Les SyncCharts (2)

- Notion d'instant :
  - Occurrences strictement futures ou présentes (#)
  - Possibilités d'états transitoires
- Hiérarchie : macro-états
- Parallélisme : 
- Prémptions :
  - Suspension 
  - Avortements :
  - Terminaison normale 



# Les SyncCharts (3)

- **Communication** par diffusion instantanée
- Possibilité de cacher des signaux
- Comportement **déterministe**
- **Sémantique** : algèbre de processus synchrones entièrement compatible avec celle d'Esterel (les syncCharts supportent l'insertion de code Esterel)

# Tk Button: Default Bindings

Tk automatically creates class bindings for buttons that give them default behavior:

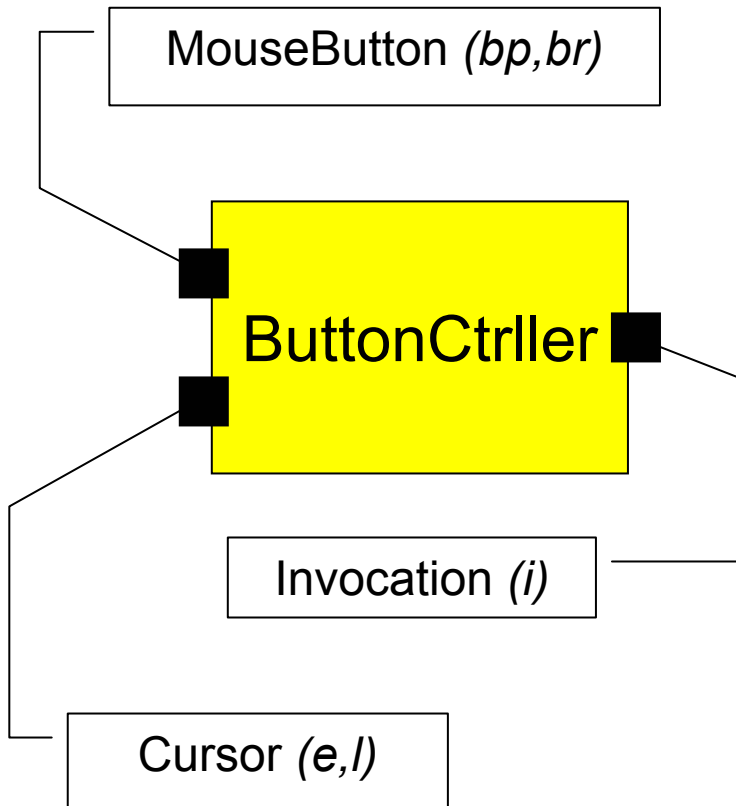
- [1] A button **activates** whenever the mouse passes over it and **deactivates** whenever the mouse leaves the button. Under Windows, this binding is only active when mouse button 1 has been pressed over the button.
- [2] A **button's relief** is changed to sunken whenever mouse button 1 is pressed over the button, and the relief is restored to its original value when button 1 is later released.

# Tk Button (continued)

- [3] If mouse button 1 is pressed over a button and later released over the button, the button is *invoked*. However, if the mouse is not over the button when button 1 is released, then no invocation occurs.
- [4] When a button has the input focus, the space key causes the button to be invoked.



# Contrôleur de Bouton Tk



bp: button pressed

br: button released

e: enter

l: leave

i: invoke

---

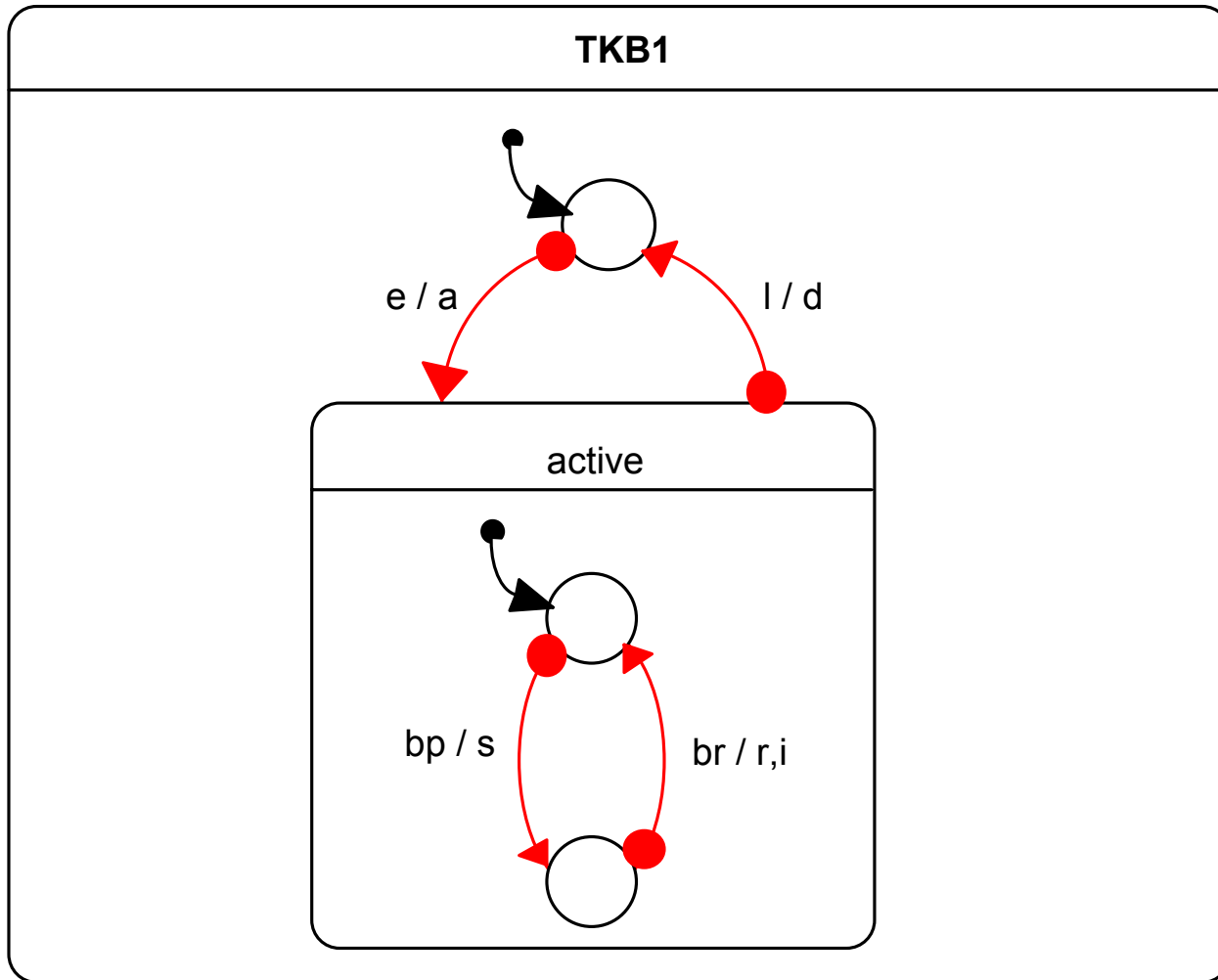
a: activate

d: de-activate

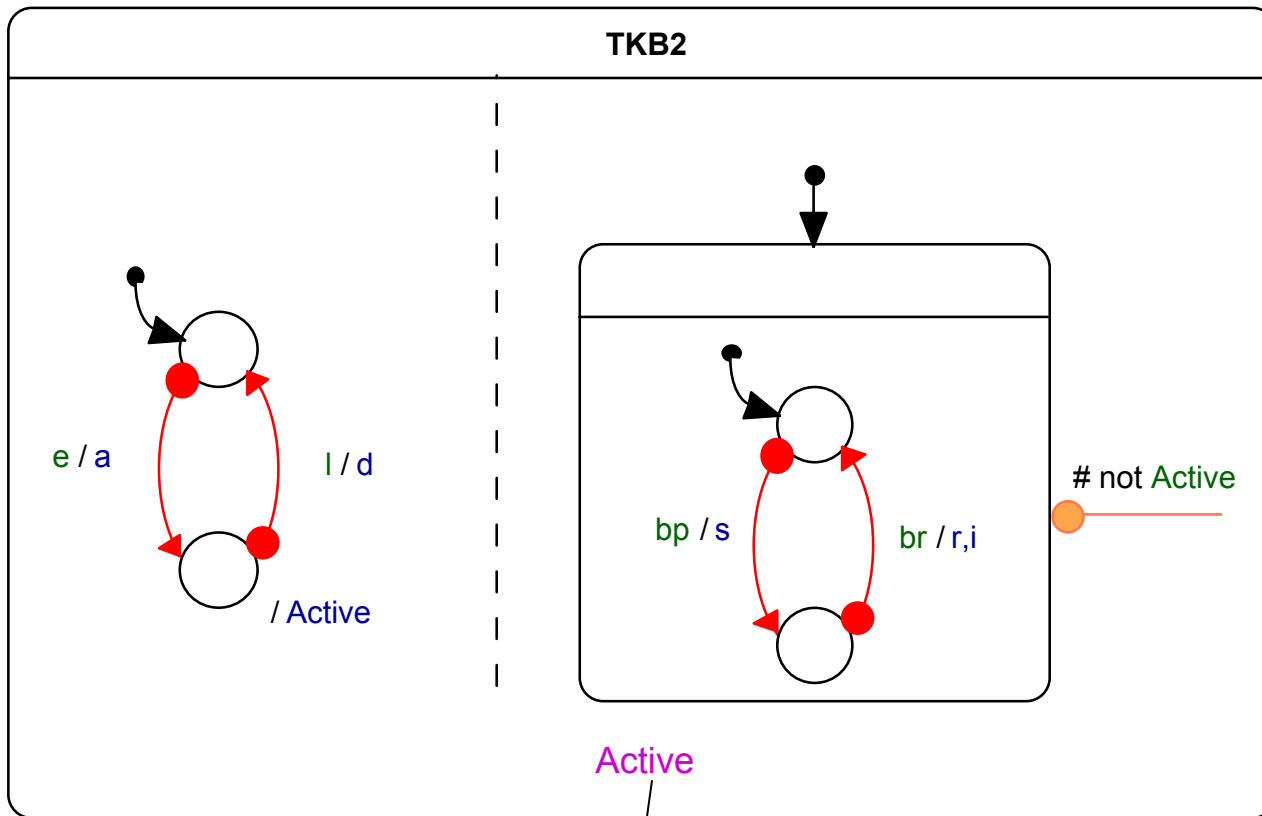
r: raise

s: sink

# Solution naïve



# Séparation + Histoire

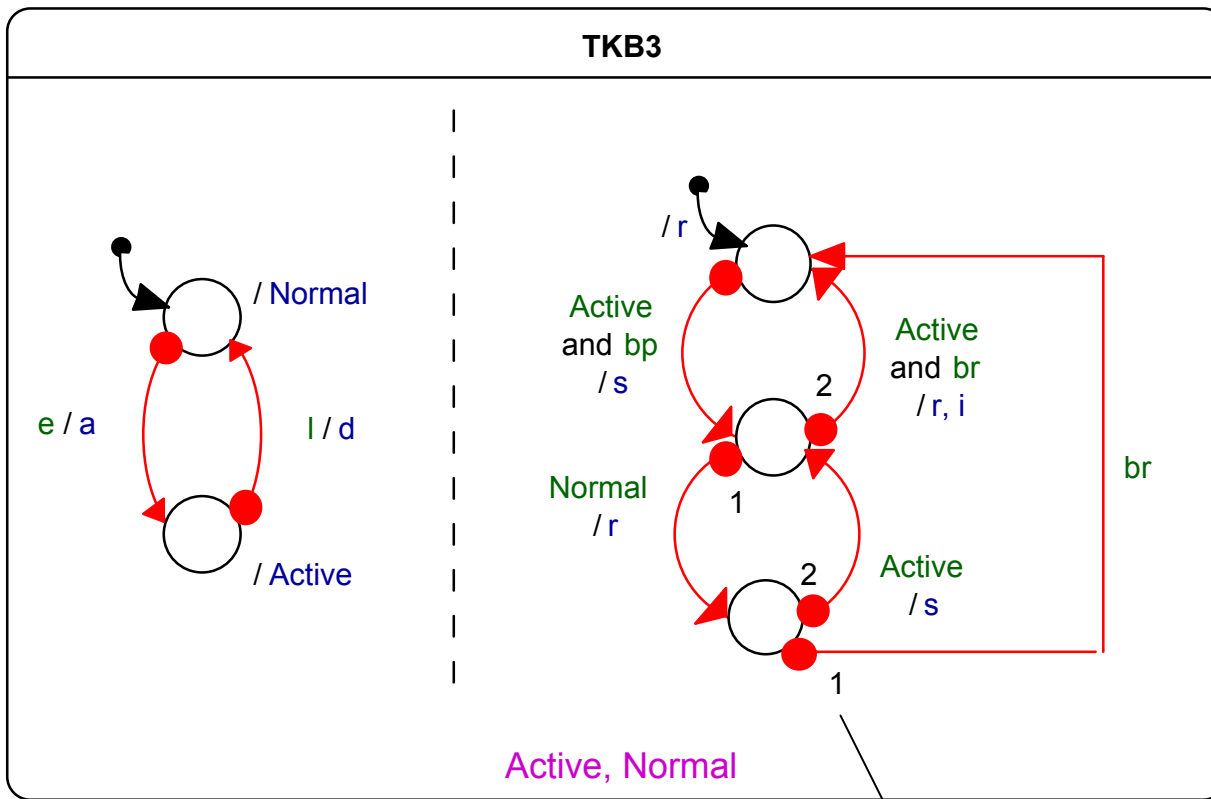


*Relatif au Port  
Cursor*

*Relatif au Port  
MouseButton*

**Couplage  
(signaux locaux)**

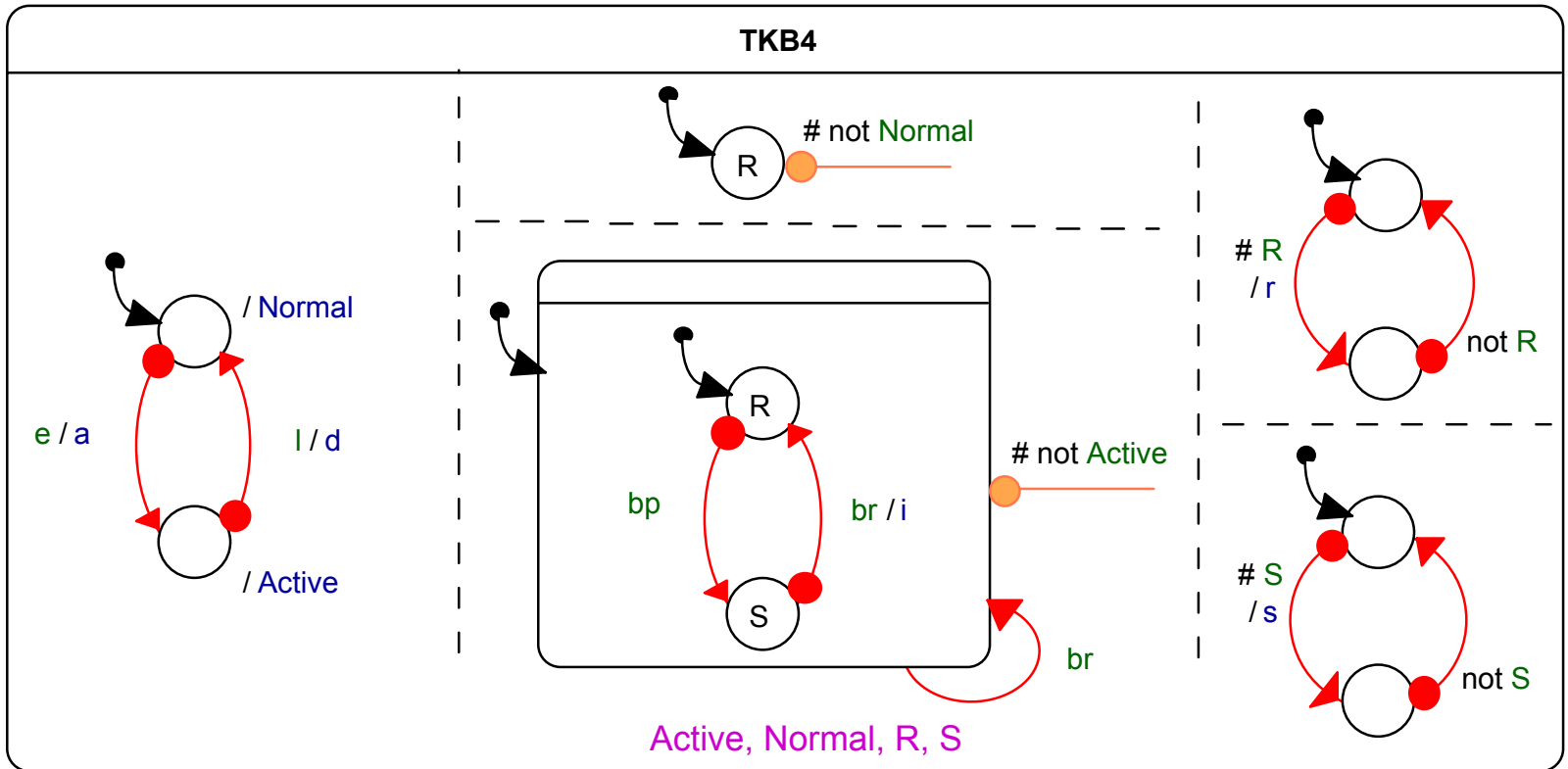
# Solution (1)



relation  $e \# l \# bp \# br$

priorité

# Solution (2)



R = Raised; S = Sunken

relation e # l # bp # br

# S.I.B

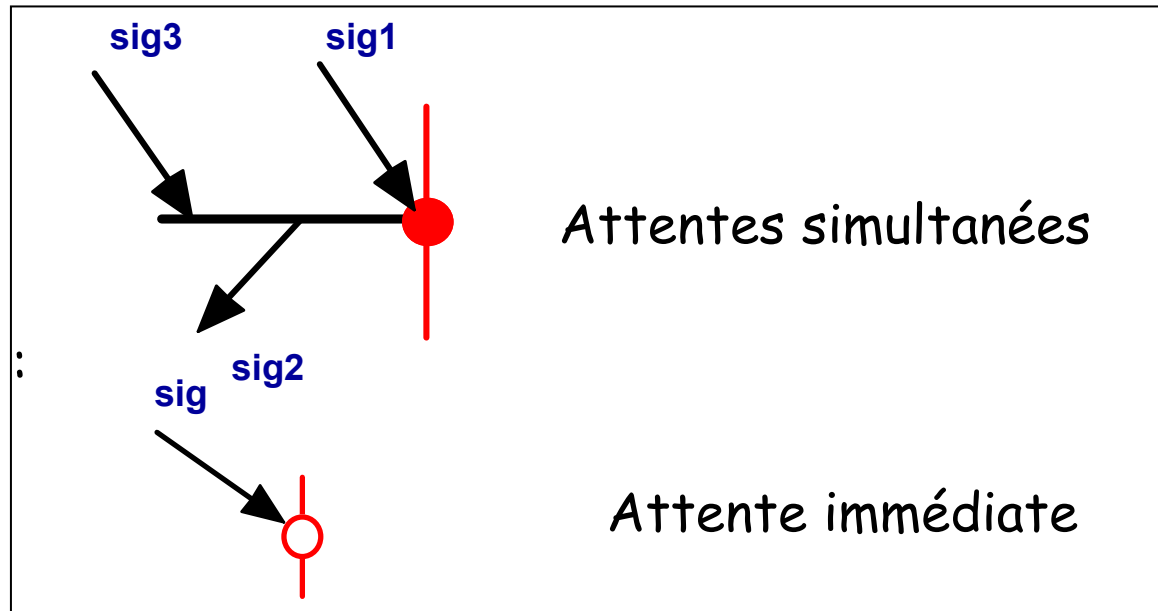
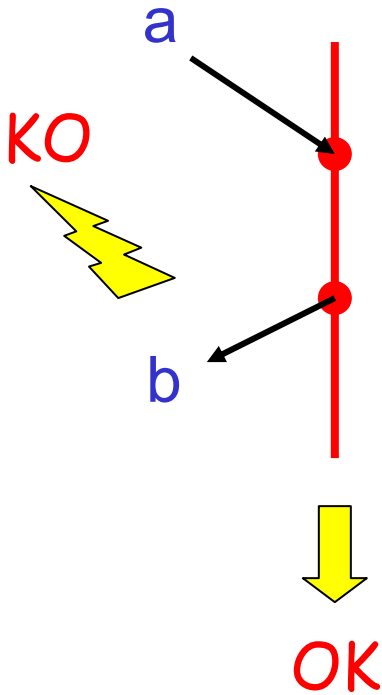
- Synchronous Interface Behavior
  - Observed: un ensemble de signaux observés
  - Timebase: un ensemble de bases de temps
  - Une évolution attendue
- Un sib est une boîte réactive qui **observe** des signaux et **réagit** en conséquence :
  - En terminant avec **OK** (satisfaction)
  - En terminant avec **KO** (non conforme)
  - En attendant la réaction suivante (pause)

# Eléments de base : Expect

Observed = {a,b,c}

Attendre la prochaine occurrence de **a** (input)  
puis

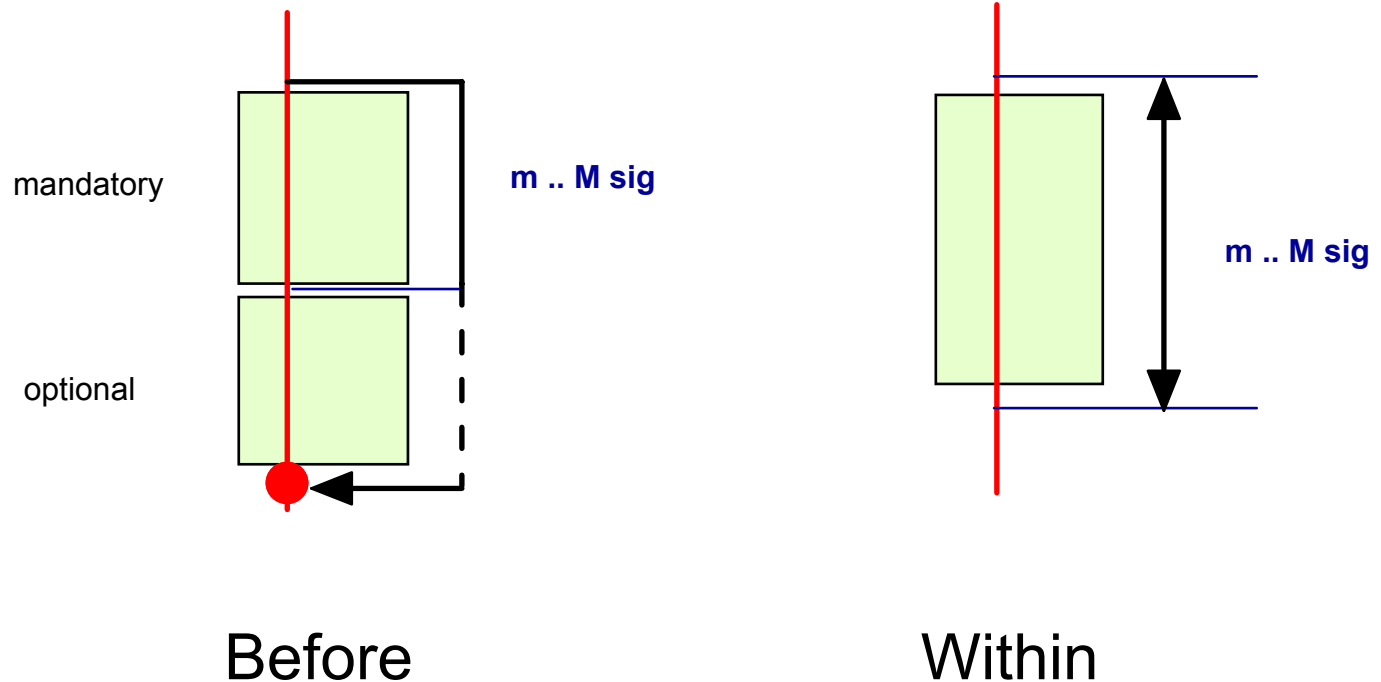
Attendre la prochaine occurrence de **b** (output)



Propres au synchrone :

# Prise en compte du temps

## Temps multiforme





# Autres blocs

- Les SIB sont inspirés des **MSC** (Message Sequence Charts) + synchrone
- **Autres constructions** :
  - Parallèle, alternatives, itération bornée, et deux formes de préemption (upto, watchdog)
- **Sémantique** : algèbre de processus synchrones
- **Introduction maîtrisée du non-déterminisme** (intervalles, séquences optionnelles)

# Utilisation des SIB

- **Principe** : 1 sib  $\rightarrow$  1 module Esterel  
Les traductions de sibs composées avec le contrôleur conçu.
- **Satisfaction de scénario** :
  - 1 famille de scénarii  $\rightarrow$  1 sib
  - Montrer que **OK** est possiblement émis
- **Propriété de sûreté** :
  - 1 contraposée de propriété  $\rightarrow$  1 sib
  - Montrer que **OK** n'est jamais émis

# Perspectives

- SyncCharts :
  - Proto universitaire disponible
  - Version commerciale : Esterel Studio
- SIB :
  - Proto universitaire en développement
- UML :
  - Diagrammes de classes conservés
  - Modèle dynamique → syncCharts
  - Diagrammes de séquence → SIB
- UML + synchrone =
  - Support de description et de validation
  - A la recherche d'une méthodologie