Module M3301
Partie 2 - Méthodologie de construction du logiciel
Modélisation et construction des logiciels complexes

J. Christian Attiogbé

Université de Nantes

UNIVERSITÉ DE NANTES

iut NANTES Université de Nantes

Plan du cours

Introduction

Voici un modèle ! ici, sous forme d'un **automate**.

Avez-vous une intuition de ce que c'est (ce qu'il fait) ?

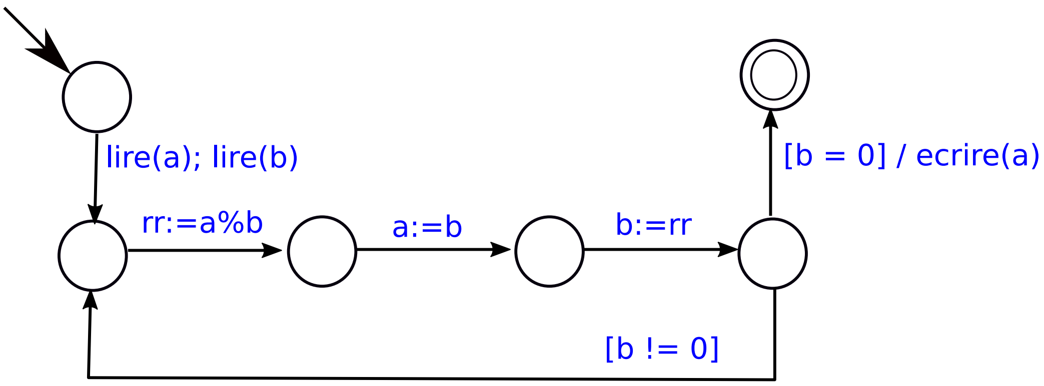


Figure: Exemple de modèle

Introduction

Avez-vous une intuition de ce que c'est (ce qu'il fait) ?

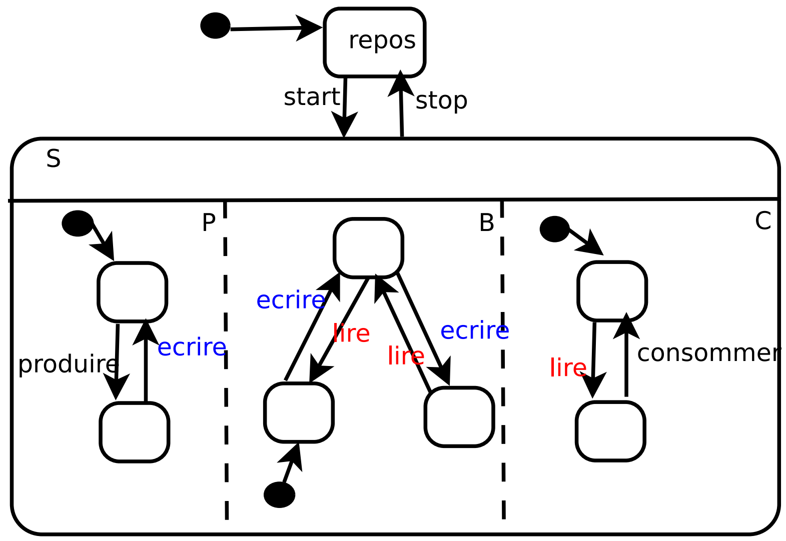


Figure: Exemple de modèle

Notions de bases: état, transition, comportement

Le comportement d'un système (quel qu'il soit), peut être vu comme une suite des différents états du système.
Les changements d'états ou transitions entre états peuvent être internes ou dus à des événements externes au système.

Une **suite d'états décrit un comportement**, donc des transitions entre des états.

Notations : états, transitions

Etat initial, état final, transition (étiquetée avec `alpha`) entre états

Modélisation

En modélisation et en construction de logiciels en particulier, on prévoit le comportement des logiciels. On fait pour cela des **modèles**.
On modélise ainsi ce qu'on va construire (aussi bien les les données que les traitements).
On peut ainsi prédire les comportements corrects et incorrects (*bugs*) ; il faut pour cela que les modèles soient (mathématiquement) les plus précis possibles.

Un modèle simule mathématiquement le (comportement d'un) système. En ce sens un automate à états sert de modèle d'un système ; il permet d'en simuler le comportement.

Automates : définition

Automates à états

Un automate à états est défini formellement par un n-uplet : $\langle S, A, \delta, S_0, S_f \rangle$

- Ensemble d'**états** fini: $S = \{S_0, \cdots, S_f, \cdots\}$
- Un alphabet d'**actions** ou d'étiquettes : A
- Une relation de **transition** δ définie sur $(S \times A)$ et S : $S \times A \leftrightarrow S$
- Un **état initial** S_0 (élément de S)
- Un ou des **états finaux** S_f (éléments de S)

Automate déterministe

L'automate est déterministe lorsque la relation de transition est une fonction plutôt qu'une relation, auquel cas il est non déterministe.

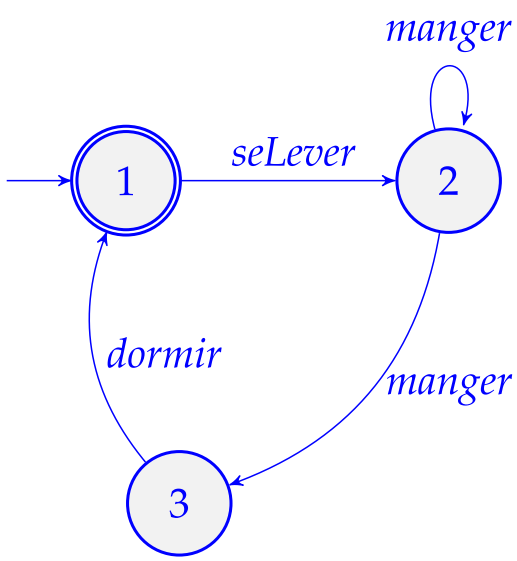
Notions

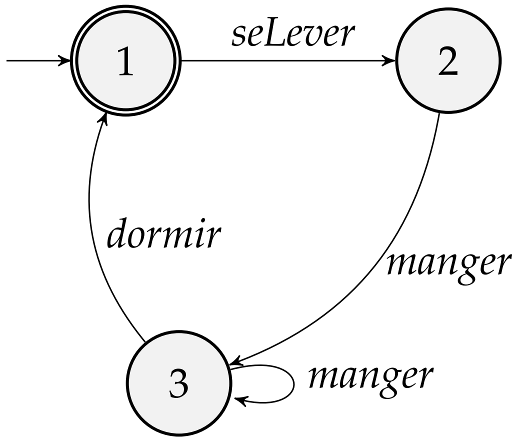
- **Alphabet** d'actions ensemble des actions effectuées par un système
- **Espace d'états** : ensemble des états (possibles) d'un système
- **Relation de transition, Fonction de transition**
- **Graphe d'états**

Terminologie

- Diagramme Etat/Transition (*State/Transition Diagram*)
- Automate (*Automaton, Automata*)
- Machine à états finis (*Finite State Machine*)
- Système de transitions (étiquetées) *Labelled Transition System*
- Espace d'états (*state space)*

Exemple d'automates non-déterministe et déterministe

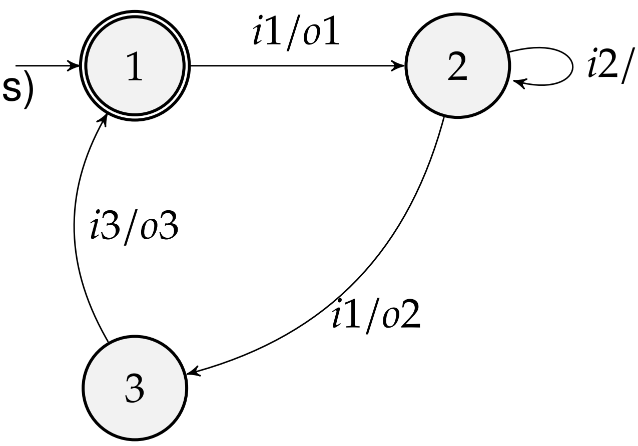




Machine à états de Mealy

Un automate de Mealy est défini par un n-uplet
$\langle S, A = In \cup Out, \delta, \delta_o, S_0, S_f \rangle$

- Ensemble d'états : S
- Ensemble d'entrées : In
  (événements d'entrées, conditions)
- Ensemble de sorties : Out
  (actions de sortie, traitements)
- Un état initial : S_0
- Deux relations :
  - transition $\delta : S \times In \rightarrow S$
  - sortie $\delta_o : S \times In \rightarrow Out$
- Etat final S_f



Machines de Mealy : notation

- Notation des transitions : $Ss \quad —i/o\rightarrow \quad St$
- si l'entrée i est reçue alors que le système est dans l'état Ss, la sortie o est produite et le nouvel état du système est St
- i est aussi appelé le déclencheur (*trigger*).

La notation des transitions est étendue de la façon suivante : les transitions entre états ont la forme evt [garde] / actions*.
evt est l'entrée reçue/lue.
[garde] : garde est une condition sur l'état du système après l'événement evt.
action* est une suite de 0 ou plusieurs actions.

Exemple d'automate de Mealy

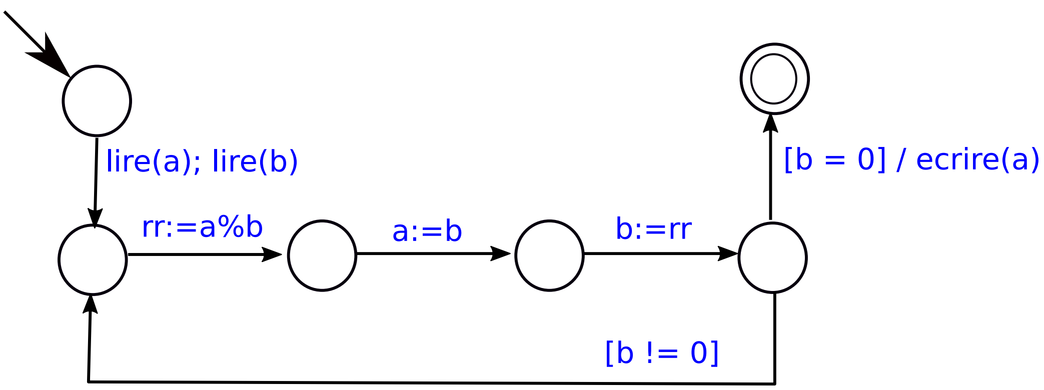


Figure: Exemple de modèle

Caractéristiques d'une machine de Mealy

- **non hiérarchique**
- un état dénote l'état complet du système
- le système est dans **un seul état à la fois**
- une **transition est atomique** ; elle ne peut être décomposée.

Généralisation des machines à états

Il y a plusieurs modèles et familles de modèles à états. Plusieurs langages ou formalismes de modélisation utilisent ces modèles à états.

- **System Design Language (SDL)** (normalisé par la *International Telecommunication Union* (ITU), 1988, très utilisé en Télécoms et ailleurs)
- **Statecharts** de David Harel (hiérarchiques) 1987, utilisé pas exemple dans papyrus, ...
- **Réseaux de Petri** de Carl Adam Petri (1962, 1969)
- **Algèbres de processus**, telles que *Calculus of Communicating Systems* (CCS, 1980) de R. Milner, *Communicating Sequential Processes* (CSP, 1978) de T. Hoare. C'est la famille la plus expressive des modèles à états.

Bibliographie

- Théorie des automates et langages formels
  Michel Rigo
  `http://www.discmath.ulg.ac.be/cours/main_autom.pdf`
- Compilateurs : Principes, techniques et outils
  Alfred AHO, Ravi SETHI et Jeffrey ULLMAN.
  Dunod, 2000. (vers anglaise *Compilers, Addison Wesley, 1986*)
- Notes de révision : Automates et langages
  Benjamin MONMEGE et Sylvain SCHMITZ
  `http://www.lsv.fr/~schmitz/teach/2013_agreg/notes-r62.pdf`