

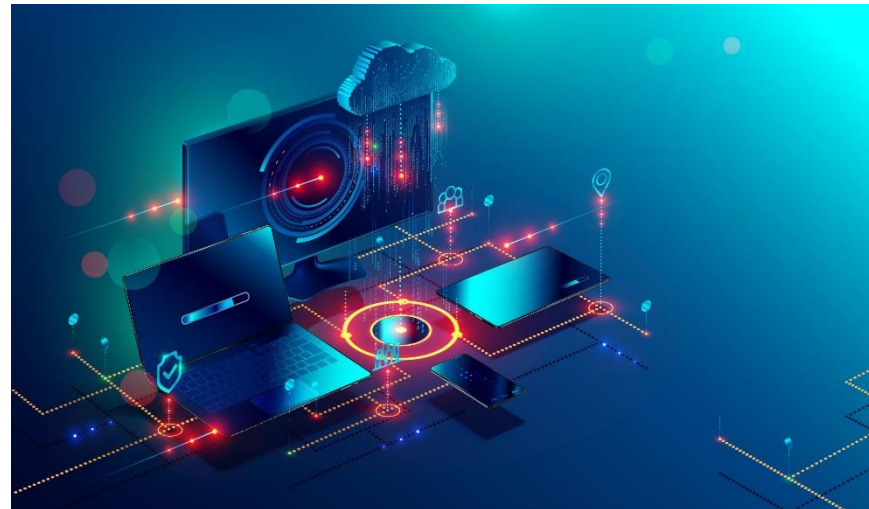
Analyse Orientée Objet

I. Introduction

II. Approche objet et système d'information.

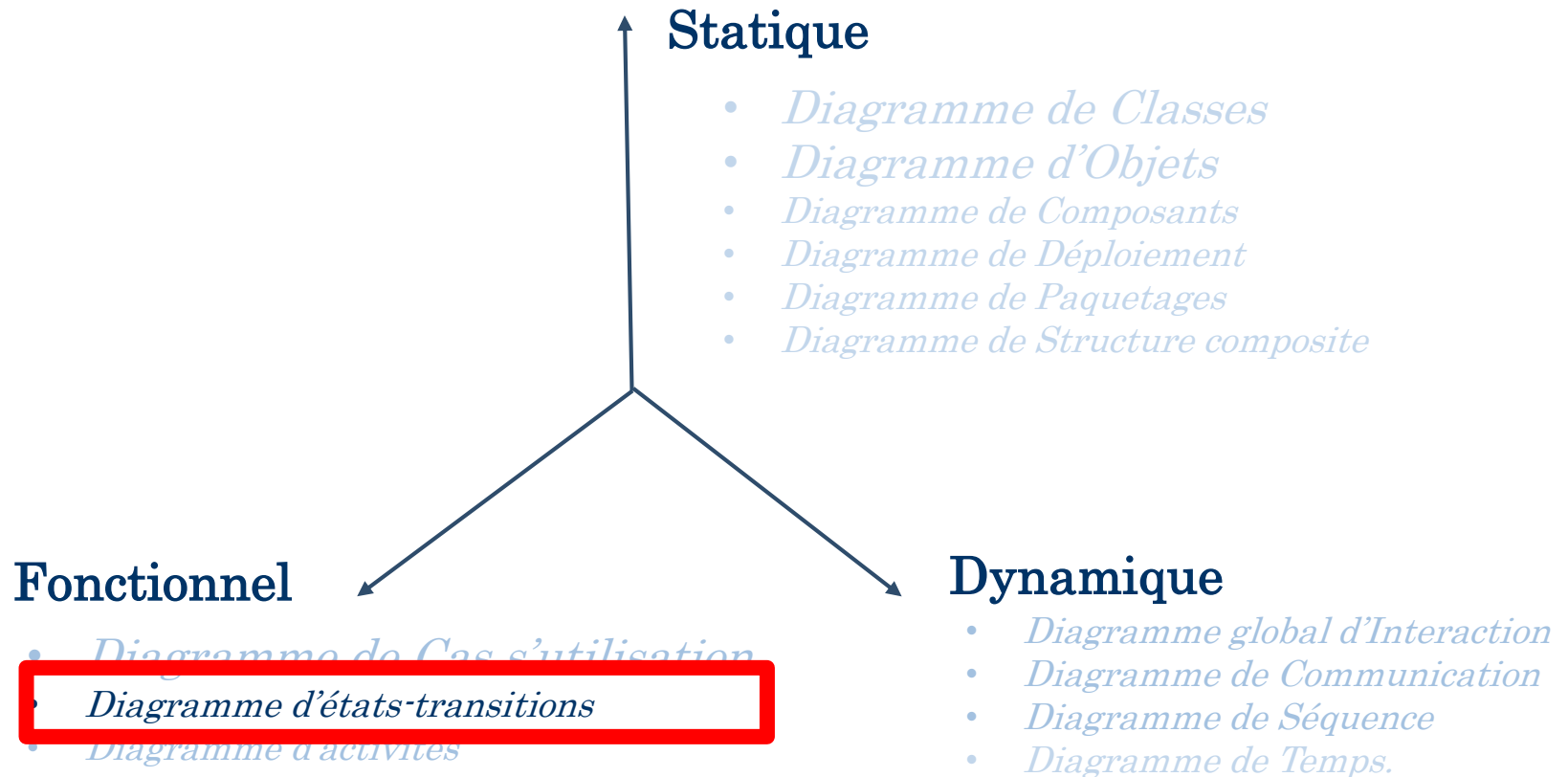
III. Principes Objet

IV. UML



IV. UML

3. Les diagrammes UML.



IV. UML

Diagramme d'états-transitions

1. Introduction.
2. Les éléments constitutifs d'un diagramme d'états-transitions.
3. Hiérarchie dans les machines d'états.
4. Exemples & Exercices.

IV. UML

Diagramme d'états-transitions

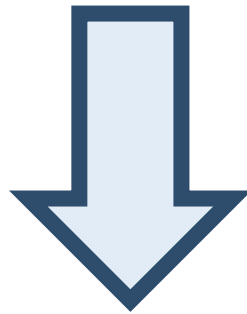
1. Introduction.
2. Les éléments constitutifs d'un diagramme d'états-transitions.
3. Hiérarchie dans les machines d'états.
4. Exemples & Exercices.

IV. UML

Diagramme d'états-transitions

1. Introduction.

- Le diagramme d'états-transitions d'UML permet de décrire le comportement interne d'un objet à l'aide d'un automate à états finis.



Qu'est ce qu'un automate à états finis?

IV. UML

Diagramme d'états-transitions

1. Introduction.

Qu'est ce qu'un automate à états finis?

Pour définir une *machine à états-finis* il faut d'abord définir les systèmes combinatoires et séquentiels



Les systèmes combinatoires et séquentiels.

IV. UML

Diagramme d'états-transitions

1. Introduction.

Les systèmes combinatoires et séquentiels?

- Dans un **système combinatoire**, le comportement d'une machine (ou d'un objet) dépend **uniquement** des **événements** qu'elle(il) reçoit;
=> la machine réagira toujours de la même manière à deux événements identiques.
- Dans un **système séquentiel**, le comportement d'une machine (ou d'un objet) dépend non seulement des événements qu'elle(il) reçoit mais aussi de ce qui s'est passé avant ces événements (son état).

IV. UML

Diagramme d'états-transitions

1. Introduction.

Les systèmes combinatoires et séquentiels?

Exemple:

Si on appuie sur la touche  (play/pause) d'un lecteur.

On ne pourra pas déterminer l'effet qu'aura l'appuie sur cette touche.

POURQUOI?

On ne connaît pas la situation (état) dans laquelle se trouvait le lecteur au préalable.

IV. UML

Diagramme d'états-transitions

1. Introduction.

On ne connaît pas la situation dans laquelle se trouvait le lecteur au préalable.

C'est-à-dire:

- *Si le lecteur est déjà en lecture ALORS la touche  provoque une pause.*
- *Si le lecteur est déjà à l'arrêt ALORS la touche  lance alors la lecture.*

IV. UML

Diagramme d'états-transitions

1. Introduction.

Conclusion:

Les **machines à états-finis** sont des machines qui ont un fonctionnement séquentiel.

Donc elles passent par un nombre de **situations (états)** limitées et clairement identifiées.

IV. UML

Diagramme d'états-transitions

1. Introduction.

Définition:

Le diagramme d'états-transitions (State Machine Diagram ou Statechart Diagram) permet de décrire le fonctionnement d'une machine (ou un objet) ayant un comportement séquentiel.

Ce diagramme présente l'ensemble des séquences possibles d'états et d'actions qu'une instance d'une classe (un objet) peut traiter au cours de son cycle de vie en *réaction à des événements* extérieurs.

Événements = signaux, invocations d'opération, etc.

IV. UML

Diagramme d'états-transitions

1. Introduction.

ENFIN

Un diagramme d'états-transitions est utilisé pour décrire le comportement type d'une instance quelconque d'une classe.



Remarque importante:

- Un diagramme d'états-transitions ne peut être associé qu'à une seule classe.
- Seules les classes ayant un cycle de vie **significatif** nécessitent le recours au diagramme d'états-transitions.

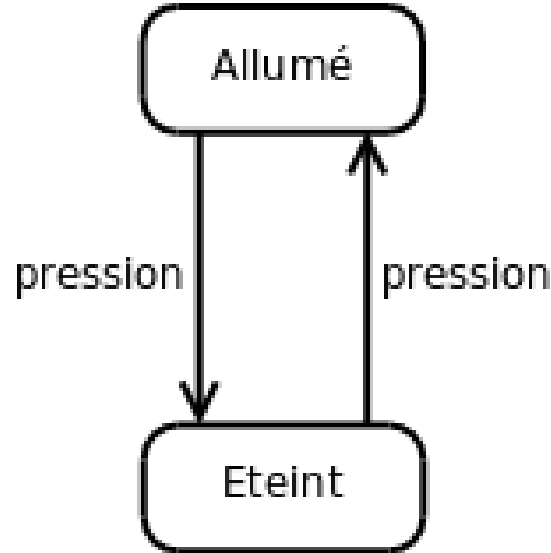
IV. UML

Diagramme d'états-transitions

En résumé

Un diagramme d'états-transitions est un graphe qui représente un *automate à états finis*.

L'automate à états finis est une machine dont le comportement des sorties ne dépend pas seulement de l'état de ses entrées, mais aussi d'un historique des sollicitations passées.



IV. UML

Diagramme d'états-transitions

1. Introduction.
2. Les éléments constitutifs d'un diagramme d'états-transitions.
3. Hiérarchie dans les machines d'états.
4. Exemples & Exercices.

IV. UML

Diagramme d'états-transitions

2. Les éléments constitutifs d'un diagramme d'états-transitions:

1. États.
2. Événements.
3. Transitions.
4. Points de choix.

IV. UML

Diagramme d'états-transitions

2. Les éléments constitutifs d'un diagramme d'états-transitions:
 1. États.
 2. Événements.
 3. Transitions.
 4. Points de choix.

IV. UML

Diagramme d'états-transitions

1. États.

Définition:

Un **état** est une **abstraction** d'un **moment** de la vie d'un objet pendant lequel il **satisfait** un ensemble de **conditions**.

- Un objet peut passer par une série d'états pendant sa durée de vie.
- Un état représente une période dans la vie d'un objet (instance d'une classe) pendant laquelle ce dernier attend un événement ou accomplit une activité.

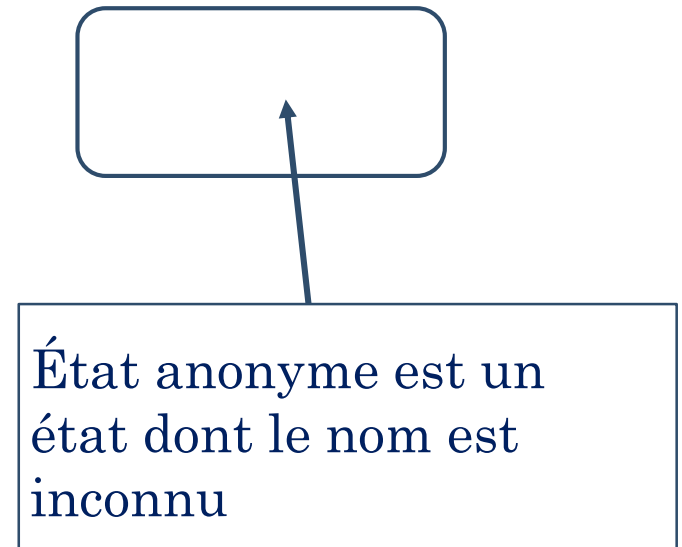
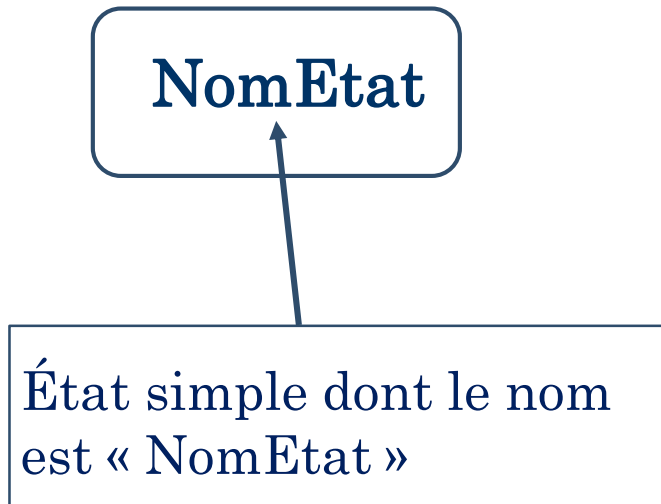
IV. UML

Diagramme d'états-transitions

1. États.

Un état est représenté par un rectangle aux coins arrondis.
On spécifie le nom de l'état dans le rectangle.
Ce nom est unique dans un diagramme d'états-transitions.

Représentation:



IV. UML

Diagramme d'états-transitions

1. États.

On retrouve dans un diagramme d'états-transitions deux autres états:

1. État initial:

C'est un pseudo état qui indique l'état de départ. C'est l'état dans lequel se trouve l'objet lors de sa création.

2. État final:

C'est un pseudo état qui indique que l'objet n'est plus nécessaire dans le système.

Représentation:



État initial est représenté par un point noir.



État final est représentée par un point noir entouré d'un cercle

Remarque importante:

*Tous les objets n'ont pas un état final.
Cas des objets permanents.*

IV. UML

Diagramme d'états-transitions

2. Les éléments constitutifs d'un diagramme d'états-transitions:
 1. États.
 2. Événements.
 3. Transitions.
 4. Points de choix.

IV. UML

Diagramme d'états-transitions

2. Événements.

Définition:

Un *événement* est un stimulus auquel l'objet doit répondre.

Quand un *événement* est reçu, une *transition* peut être déclenchée et faire basculer l'objet dans un nouvel état.

IV. UML

Diagramme d'états-transitions

2. Événements.

*Les événements peuvent être classés en plusieurs types explicites et implicites:
Signal, Appel, Changement et Temporel.*

IV. UML

Diagramme d'états-transitions

Événement de type signal (Signal)

- Un **signal** est un type de classeur destiné à véhiculer une **communication asynchrone** à sens unique entre deux objets.
- L'objet expéditeur crée et initialise une instance de signal et n'attend pas que le destinataire traite le signal pour poursuivre son déroulement.
- Un **objet** peut-être à la fois **expéditeur** et **destinataire**.
- Les signaux sont déclarés par la définition d'un classeur portant le stéréotype « **signal** » ne fournissant pas d'opération et dont les attributs sont interprétés comme des arguments.

Syntaxe du signal:

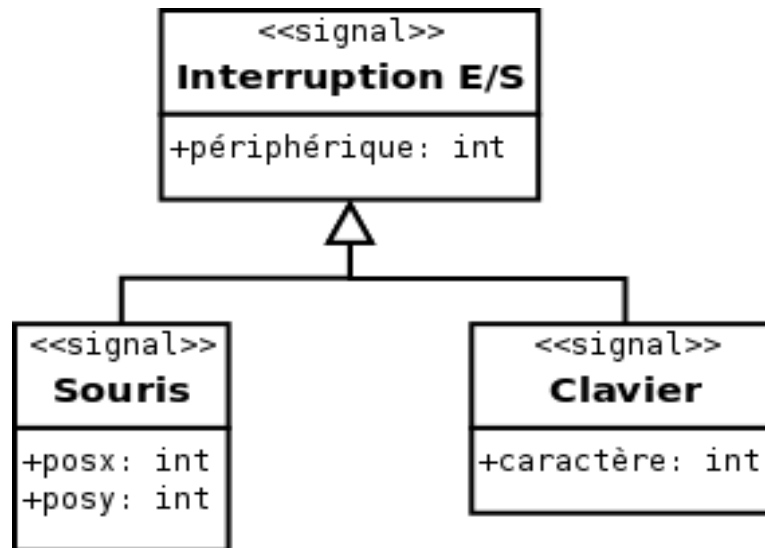
<nom_événement> ([<paramètre> : <type> [; <paramètre> : <type> ...]])

IV. UML

Diagramme d'états-transitions

Événement de type signal (Signal)

- Les signaux supportent la relation de généralisation. Ils héritent des attributs de leurs parents (héritage) et déclenchent des transitions correspondant au type du signal parent (polymorphisme).



IV. UML

Diagramme d'états-transitions

Événement d'appel (Call)

- Un événement d'appel représente la **réception** de l'**appel** d'une **opération** par un objet.
- Les paramètres de l'opération sont ceux de l'événement d'appel.
- Les événements d'appel sont des opérations déclarées dans les classes du diagramme de classes.

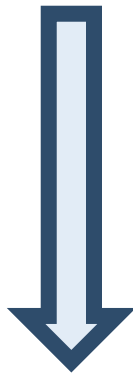
Syntaxe d'un événement d'appel:

<nom_événement> ([<paramètre> : <type> [; <paramètre> : <type> ...]])

IV. UML

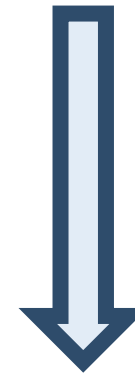
Diagramme d'états-transitions

Événement de type signal
(Signal)



*Une communication
asynchrone explicite et
nommée entre deux objets.*

Événement d'appel
(Call)



*l'invocation **synchrone** d'une
opération, avec un
mécanisme pour rendre
ensuite la main à l'émetteur.*

IV. UML

Diagramme d'états-transitions

Événement de changement (Change)

- Un **événement de changement** est généré par la satisfaction (passage de faux à vrai) d'une **expression booléenne** sur des valeurs d'attributs.
- Un événement de changement est évalué continuellement jusqu'à ce qu'il devienne vrai, et c'est à ce moment-là que la transition se déclenche.

Syntaxe de changement:

When (<condition_booléenne>)

IV. UML

Diagramme d'états-transitions

Événement temporel(after ou when)

- Les événements temporels sont générés par le passage du temps.
- Par défaut, le temps commence à s'écouler dès l'entrée dans l'état courant.
- Les événements temporels sont spécifiés:
 - De manière relative => temps écoulé.
 - De manière absolue => une date précise.

Syntaxe d'un événement temporel spécifié de manière relative:

After (<durée>)

Syntaxe d'un événement temporel spécifié de manière absolue:

When (date = <date>)

IV. UML

Diagramme d'états-transitions

2. Les éléments constitutifs d'un diagramme d'états-transitions:
 1. États.
 2. Événements.
 3. Transitions.
 4. Points de choix.

IV. UML

Diagramme d'états-transitions

3. Transitions:

1. Définition et syntaxe.
2. Transition externe.
3. Transition d'achèvement.
4. Transition interne.
5. Exemple.

IV. UML

Diagramme d'états-transitions

3. Transitions:

1. Définition et syntaxe.
2. Transition externe.
3. Transition d'achèvement.
4. Transition interne.
5. Exemple.

IV. UML

Diagramme d'états-transitions

3. Transitions.

Définition:

Une *transition* est une relation orientée entre deux états, à laquelle est attaché un *événement(déclencheur ou trigger)* et qui indique qu'un objet dans un état E1 passera dans un second état E2 si certaines conditions sont remplies.

La transition est représentée par une flèche entre deux états, accompagnée du nom de l'événement.

IV. UML

Diagramme d'états-transitions

3. Transitions.

Représentation:

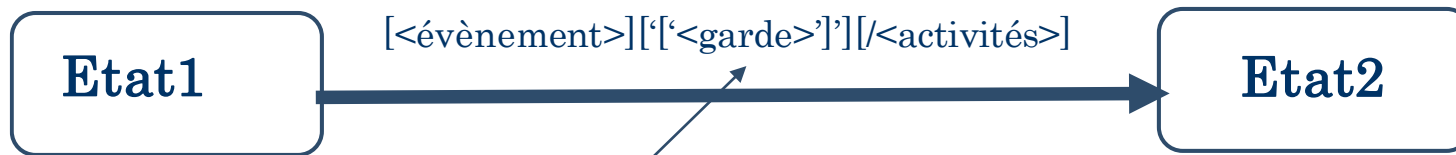


Vu au point précédent.

IV. UML

Diagramme d'états-transitions

3. Transitions.



Condition de garde:

Une condition de garde est une expression logique sur les **attributs de l'objet**, ainsi que sur les **paramètres de l'évènement** déclencheur. La condition de garde doit-être vraie pour que la transition se déclenche.

Remarque importante:

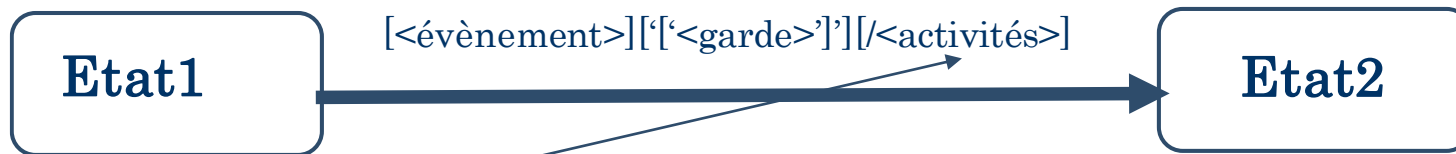
Une condition de garde est évaluée une fois que l'évènement déclencheur de la transition a lieu.

Un évènement de changement est réévalué en permanence jusqu'à ce qu'il devienne vrai.

IV. UML

Diagramme d'états-transitions

3. Transitions.



Activités:

*Si une **transition** se **déclenche**, son effet (`/<activité>`) dans la syntaxe s'exécute.*

Lorsque l'exécution de l'effet est terminée, l'état cible de la transition devient actif.

Une activité peut être:

- ✓ Une **opération primitive** comme une instruction d'assignation.
- ✓ L'envoi d'un signal,
- ✓ L'appel d'une opération,
- ✓ Une **liste d'activités**
- ✓ etc

IV. UML

Diagramme d'états-transitions

3. Transitions.

Exemple: Classe commande



L'événement « Expédition » assure la transition entre l'état En préparation et l'état En attente de la commande.

IV. UML

Diagramme d'états-transitions

3. Transitions:

1. Définition et syntaxe.
2. **Transition externe.**
3. Transition d'achèvement.
4. Transition interne.
5. Exemple.

IV. UML

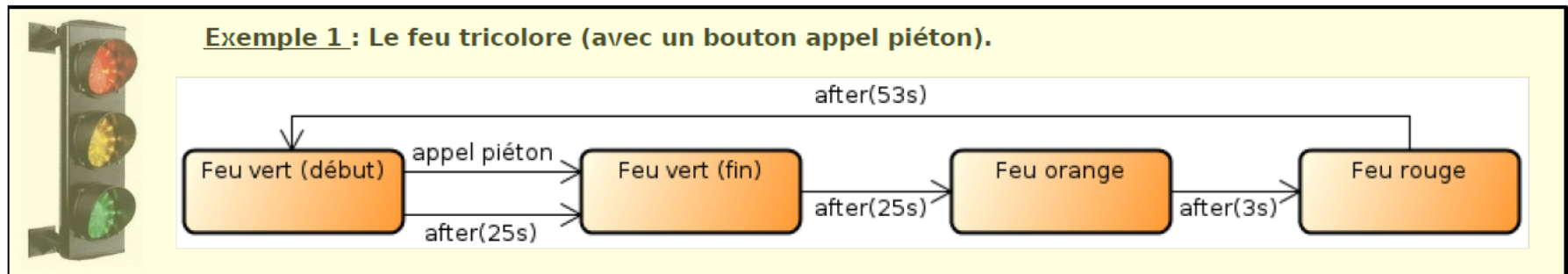
Diagramme d'états-transitions

2. Transition externe:



Une **transition externe** est une transition qui **modifie l'état** actif. Elle est représentée par une flèche allant de l'état source vers l'état cible.

Exemple:



IV. UML

Diagramme d'états-transitions

3. Transitions:

1. Définition et syntaxe.
2. Transition externe.
3. **Transition d'achèvement.**
4. Transition interne.
5. Exemple.

IV. UML

Diagramme d'états-transitions

3. Transition d'achèvement:

- *Une **transition d'achèvement** est une transition automatique, franchie quand l'activité associée à l'état source (y compris les états imbriqués) est achevée.*
- *Une **transition d'achèvement** est provoquée par l'achèvement de l'activité dans l'état source.*

Elle est dépourvue d'évènement.

- *Peut contenir une condition de garde qui est évaluée au moment où l'activité contenue dans l'état s'achève.*

IV. UML

Diagramme d'états-transitions

3. Transitions:

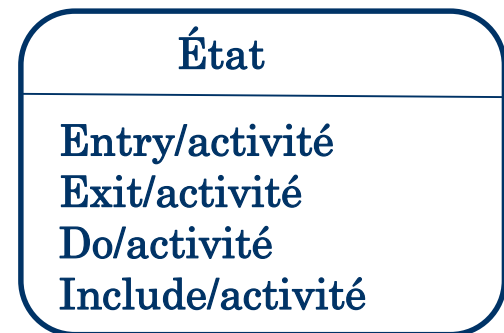
1. Définition et syntaxe.
2. Transition externe.
3. Transition d'achèvement.
4. **Transition interne.**
5. Exemple

IV. UML

Diagramme d'états-transitions

4. Transition interne:

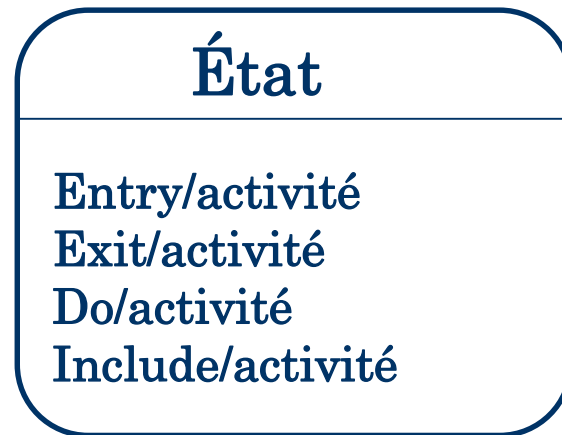
- *Les règles de déclenchement d'une **transition interne** sont les mêmes que pour une transition externe excepté qu'une transition interne ne possède pas d'état cible et que l'état actif reste le même à la suite de son déclenchement.*
- *La syntaxe d'une transition interne est la même qu'une transition classique.*
- *Par contre, les transitions internes sont spécifiées dans un compartiment de leur état associé.*



IV. UML

Diagramme d'états-transitions

4. Transition interne:



- *Les **transitions internes** possèdent des **noms d'événement prédéfinis** correspondant à des déclencheurs.*
- *Ces mots réservés prennent la place du nom de l'événement dans la syntaxe d'une transition interne.*

IV. UML

Diagramme d'états-transitions

4. Transition interne:

Entry: Permet de spécifier une activité qui s'accomplit quand on entre dans l'état.

Exit: Permet de spécifier une activité qui s'accomplit quand on sort de l'état.

Do: Une activité *Do* commence dès que l'activité *entry* est terminée. Lorsque l'activité *entry* est terminée, une transaction d'achèvement peut être déclenchée après l'exécution de l'activité *Exit*. Si une transaction se déclenche pendant que l'activité *Do* est en cours, cette dernière est interrompue et l'activité *exit* de l'état s'exécute.

Include: Permet d'invoquer un sous-diagramme d'états-transitions.

État

Entry/activité

Exit/activité

Do/activité

Include/activité

IV. UML

Diagramme d'états-transitions

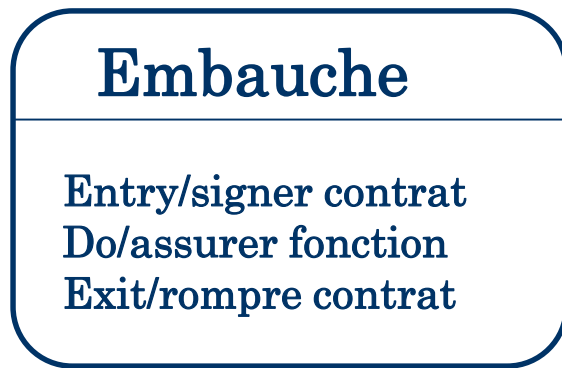
3. Transitions:

1. Définition et syntaxe.
2. Transition externe.
3. Transition d'achèvement.
4. Transition interne.
5. Exemple.

IV. UML

Diagramme d'états-transitions

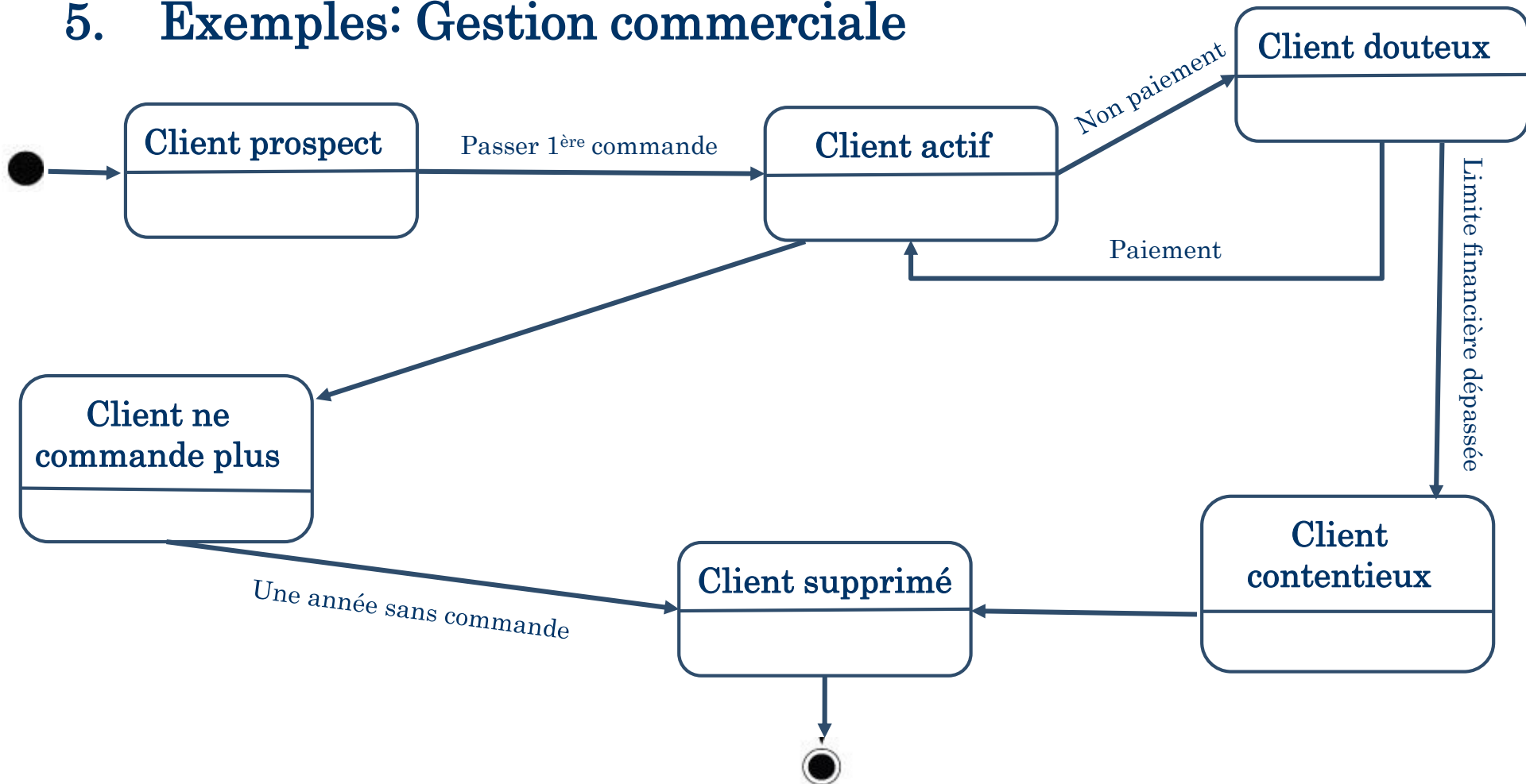
5. Exemples:



IV. UML

Diagramme d'états-transitions

5. Exemples: Gestion commerciale



IV. UML

Diagramme d'états-transitions

2. Les éléments constitutifs d'un diagramme d'états-transitions:
 1. États.
 2. Événements.
 3. Transitions.
 4. Points de choix.

IV. UML

Diagramme d'états-transitions

4. Points de choix:

Il est possible de représenter des **alternatives** pour le franchissement d'une transition.

On utilise pour cela des **pseudo-états** particuliers:

- ✓ Les **points de jonction** (représentés par un cercle plein).
- ✓ Les **points de décision** (représentés par un losange).

IV. UML

Diagramme d'états-transitions

4. Points de choix:

Points de jonction:

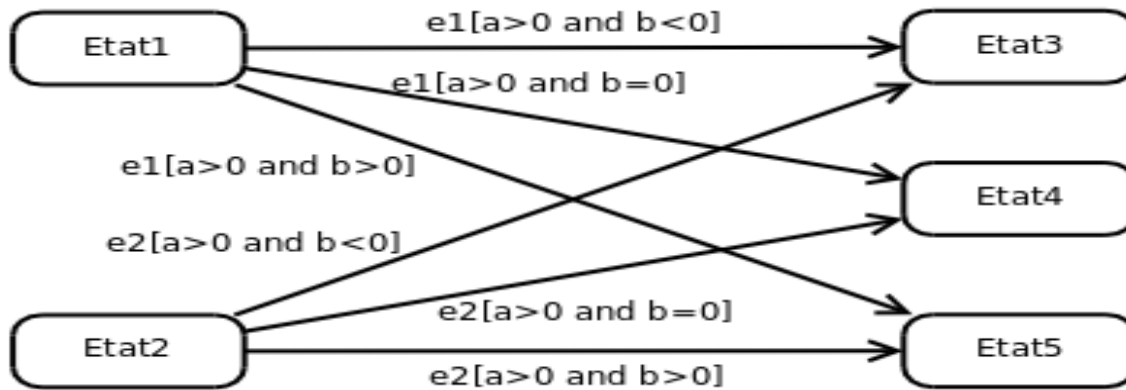


Les **points de jonction** ne sont qu'un élément graphique permettant de regrouper plusieurs segments de transition de façon à rendre le schéma plus lisible.

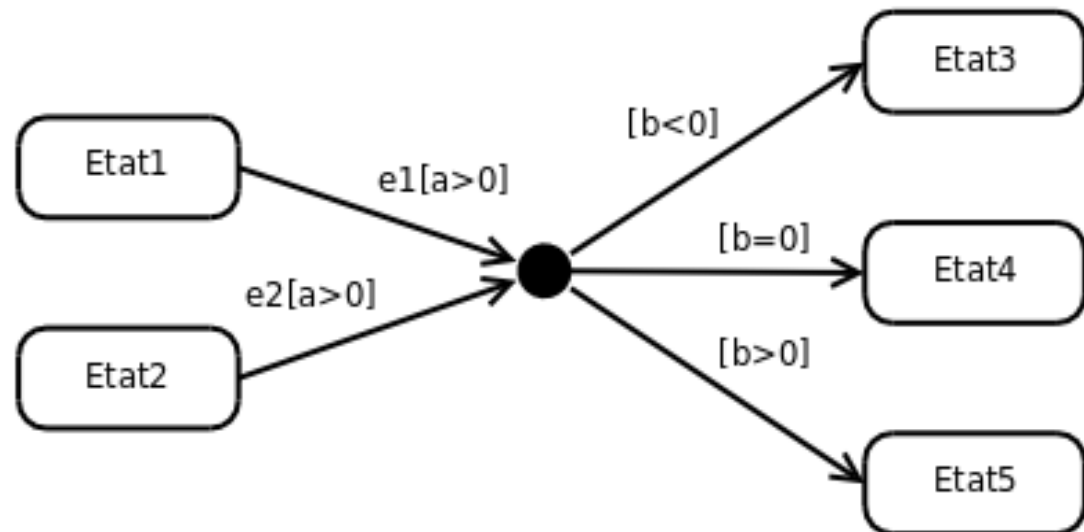
IV. UML

Diagramme d'états-transitions

4. Points de choix: Points de jonction:



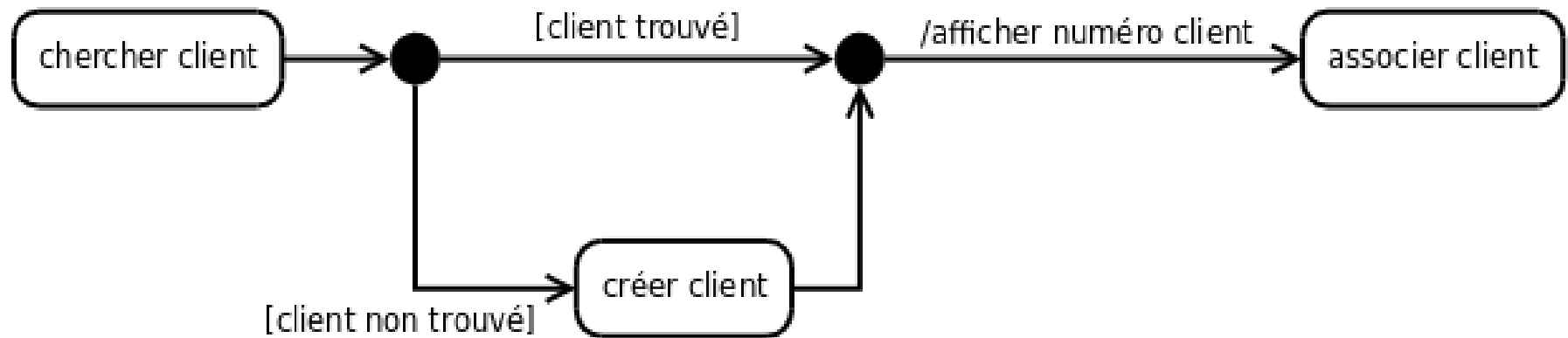
Avec points de jonction



IV. UML

Diagramme d'états-transitions

4. Points de choix: Points de jonction:



On peut utiliser les **points de jonction** pour représenter une **alternative**.

IV. UML

Diagramme d'états-transitions

4. Points de choix: Points de jonction:



- Les **points de jonction** sont un artefact graphique (un pseudo-état) qui permet de partager des segments de transition. L'objectif étant d'aboutir à une notation plus compacte ou plus lisible des chemins alternatifs.
- Un **point de jonction** peut avoir plusieurs segments de transition entrante et plusieurs segments de transition sortante.
- Un **point de jonction** n'est pas un état qui peut être actif au cours d'un laps de temps fini.
- Lorsqu'un chemin passant par un **point de jonction** est emprunté (donc lorsque la transition associée est déclenchée), toutes les gardes le long de ce chemin doivent s'évaluer à VRAI dès le franchissement du premier segment.

IV. UML

Diagramme d'états-transitions

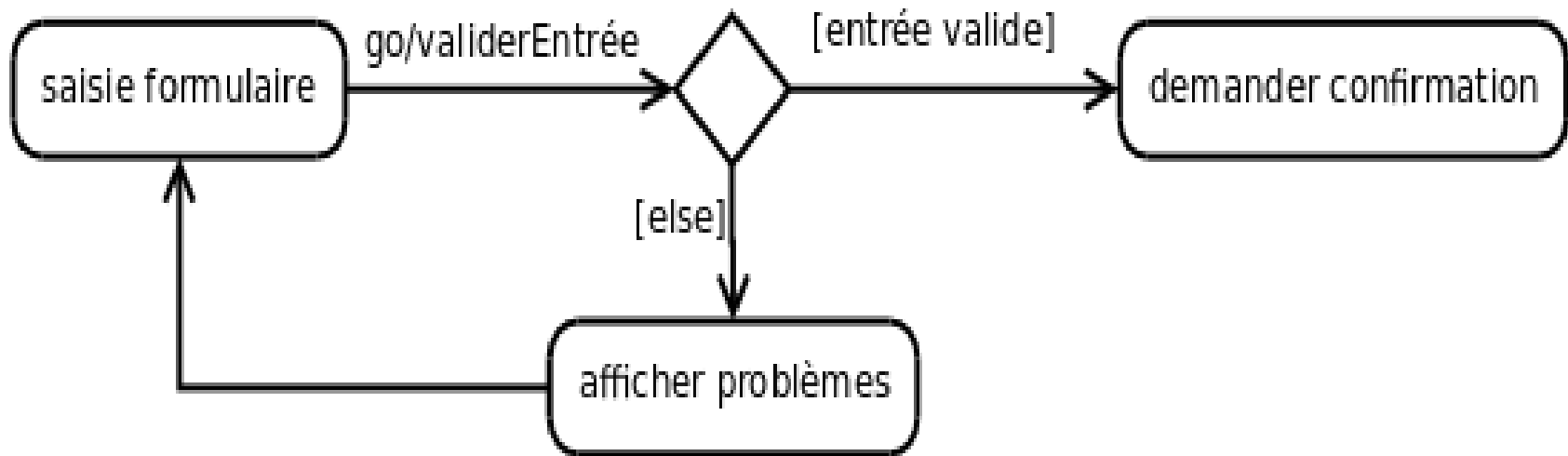
4. Points de choix: Points de décision:

- Un **point de décision** est un **point de choix** dit « **dynamique** », représenté par un losange.
- Un **point de décision** possède **une entrée** et au moins **deux sorties**.
- Contrairement au un point de jonction, les **gardes** situées après le point de décision sont évaluées ***au moment où il est atteint***.
- Cela permet de **baser le choix** sur les **résultats** obtenus en franchissant le segment **avant le point de choix**.

IV. UML

Diagramme d'états-transitions

4. Points de choix: Points de décision:

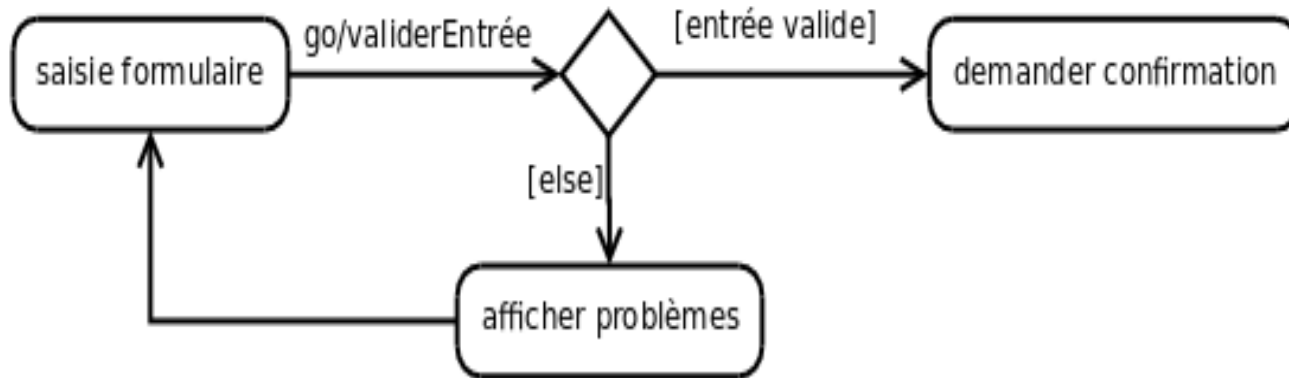


Un point de décision permet de simuler un choix.
SI-ALORS-SINON, qui entraîne deux états différents

IV. UML

Diagramme d'états-transitions

4. Points de choix: Points de décision:



Remarque importante:

Quand le point de décision est atteint, et si aucun segment en aval n'est franchissable, c'est que le modèle est mal formé.

*On peut alors utiliser une **garde particulière** notée **[else]** qui sera positionnée sur un des segments en aval du point de choix.*

IV. UML

Diagramme d'états-transitions

1. Introduction.
2. Les éléments constitutifs d'un diagramme d'états-transitions.
3. Hiérarchie dans les machines d'états.
4. Exemples & Exercices.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

1. État composite.
2. État historique.
3. Gestion de la concurrence.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

1. État composite.

- Certains **états** sont **complexes** et correspondent à la réalisation de **plusieurs activités** (séquentielles ou simultanées) qui ne pourront pas être définis par des transitions internes.
- Il peut alors être intéressant de décomposer les états complexes en sous-états.



IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

1. État composite.

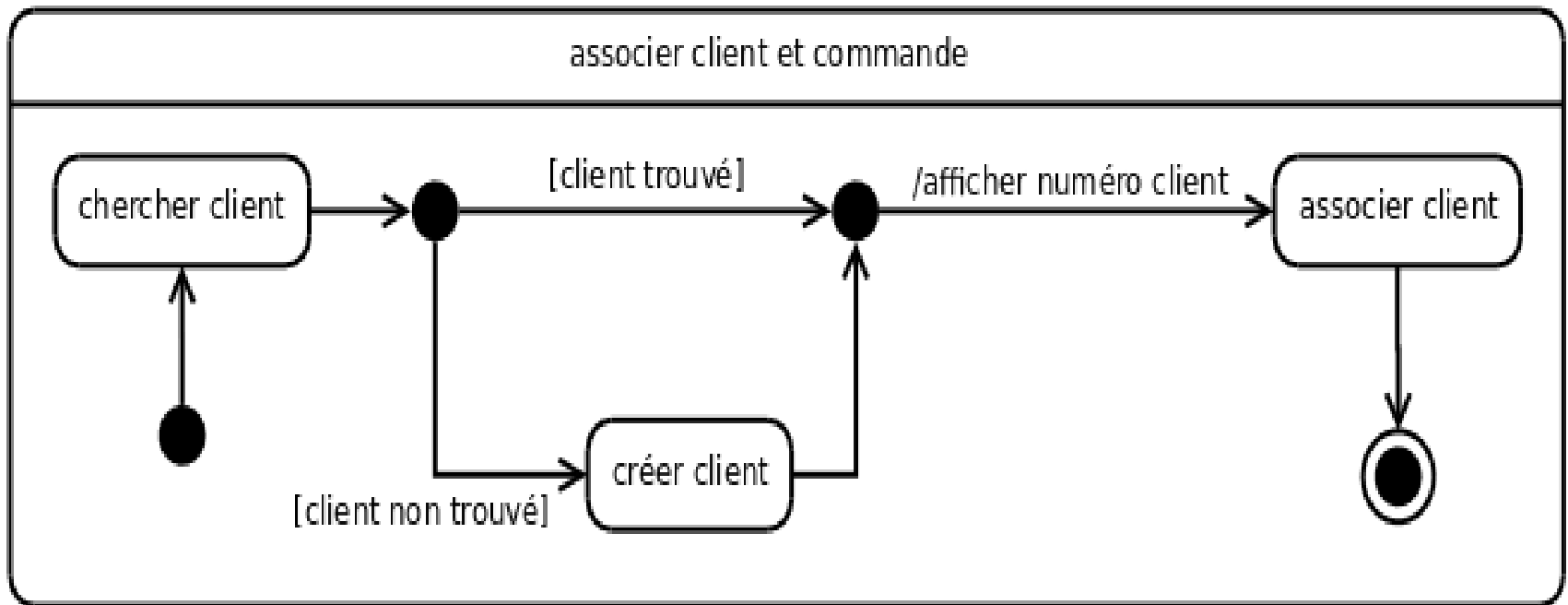
- Un état **composite** par opposition à un état dit « **simple** », est graphiquement **décomposé** en deux ou **plusieurs sous-états**.
- Un état composite est représenté par les deux compartiments de **nom et d'actions** internes habituelles, et par un compartiment contenant le **sous-diagramme**.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

1. État composite.

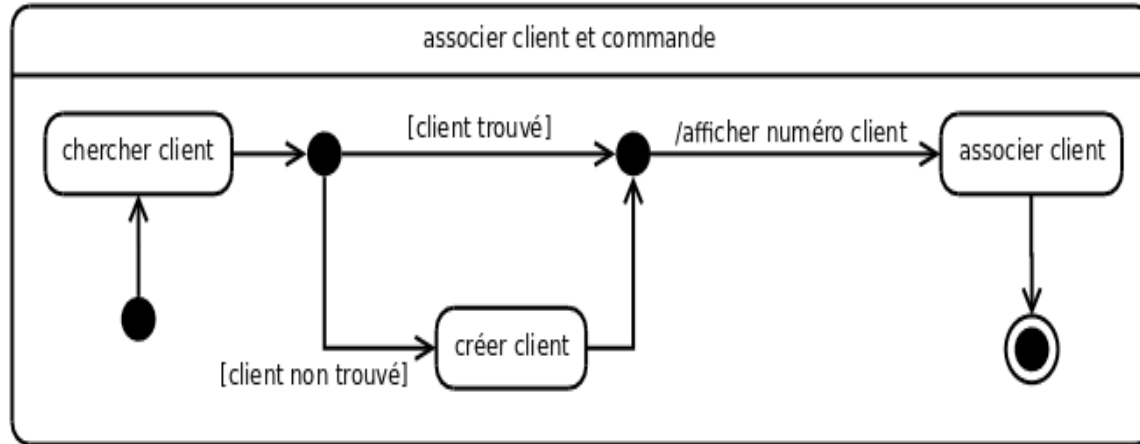


Exemple d'état composite modélisant l'association d'une commande à un client

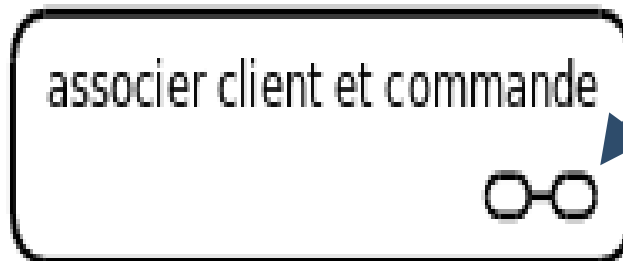
IV. UML

Diagramme d'états-transitions

1. État composite.



Représentation abrégée d'un état composite



IV. UML

Diagramme d'états-transitions

1. État composite.



Représentation abrégée d'un état composite

Il n'est pas nécessaire de représenter les sous-états à chaque utilisation de l'état englobant.

Une représentation permet d'**indiquer** qu'un état est composite et que sa définition est donnée sur un **autre diagramme**.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

1. État composite.

- Dès qu'un **état composite** est **actif**, il **active** son **sous-état initial**.
- Si un état composite est raccroché à une **transition automatique**, elle est franchie lorsque nous atteignons le **sous-état final** de l'état composite.
- Si une des **transitions externes** attenantes à l'état composite est **franchissable** alors elle est franchie et **tous les sous-états** deviennent **inactifs**.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.



1. État composite.
2. État historique.
3. Gestion de la concurrence.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

2. État historique.

- Nous avons vu que lorsqu'une transition raccrochée à un état composite est franchissable alors elle est franchie même si les activités en cours (sous-états) ne sont pas terminées.
- Si nous désirons qu'un **état composite reprenne son activité** à l'endroit **où il s'était interrompu** lors de sa précédente activation, il faut lui définir un **nouveau sous-état d'entrée** que nous appelons **état historique** et que nous désignons par 
ou 

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

2. État historique.

- (H) Pour reprendre au début du sous-état du plus haut niveau dans lequel nous nous étions arrêté.
- (H*) Pour reprendre au début du sous état dans lequel nous nous étions arrêté, quelque soit son niveau d'imbrication. (historique profond).

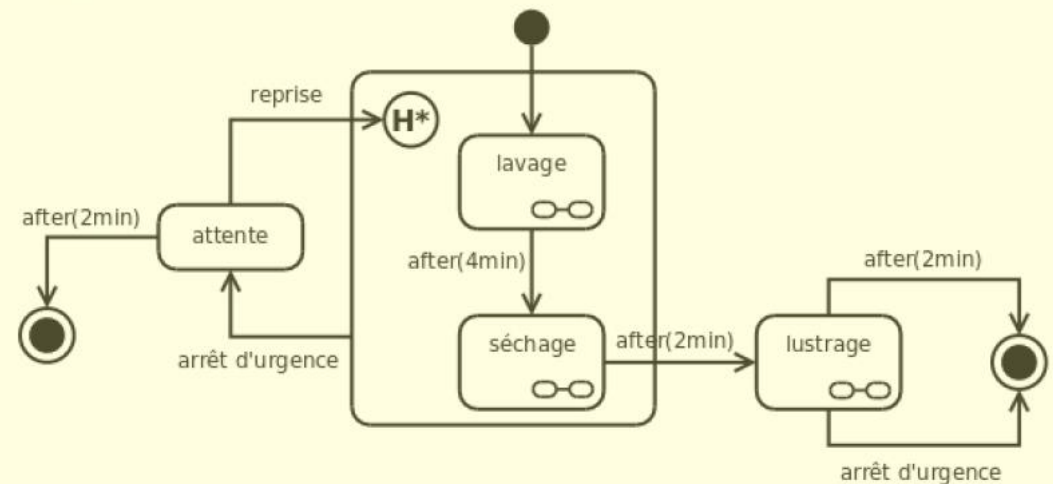
IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

2. État historique: Exemple.

Exemple : un système de lavage automatique de voiture.

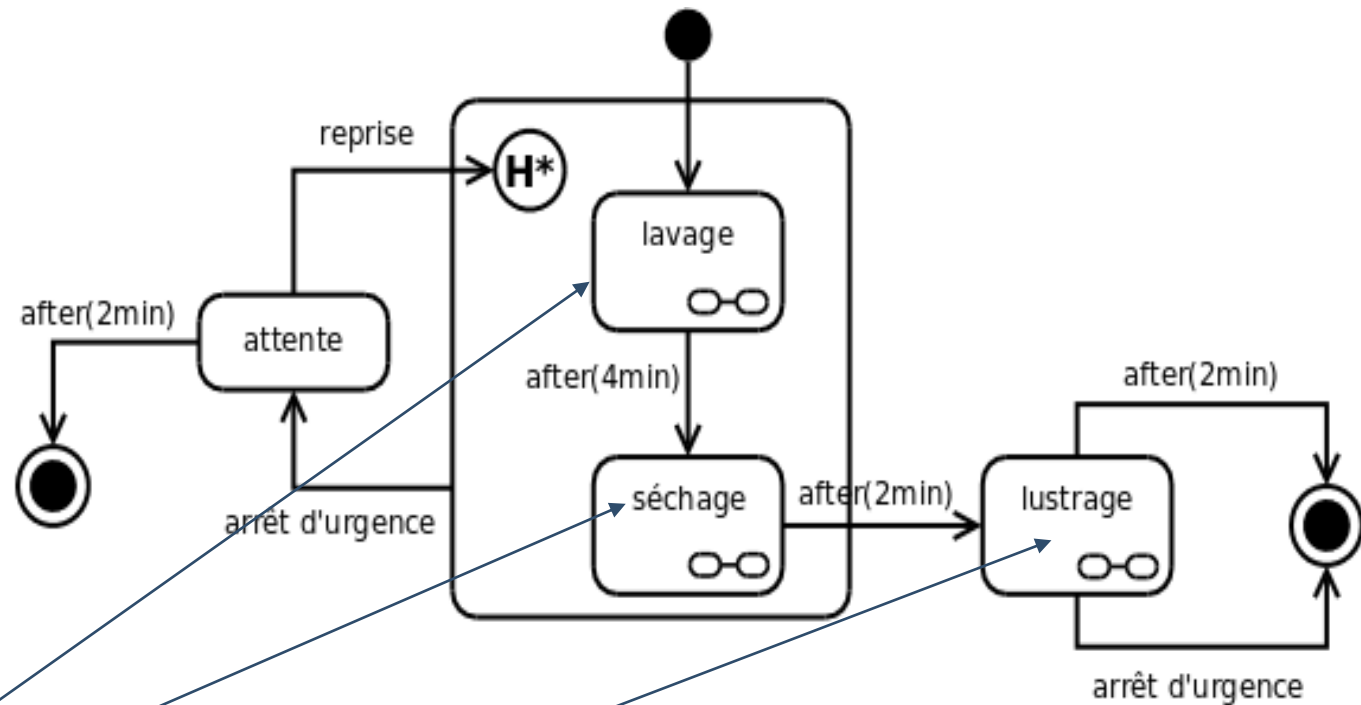


IV. UML

Diagramme d'états-transitions

Exemple:

Diagramme d'états-transitions modélisant le lavage automatique d'une voiture.



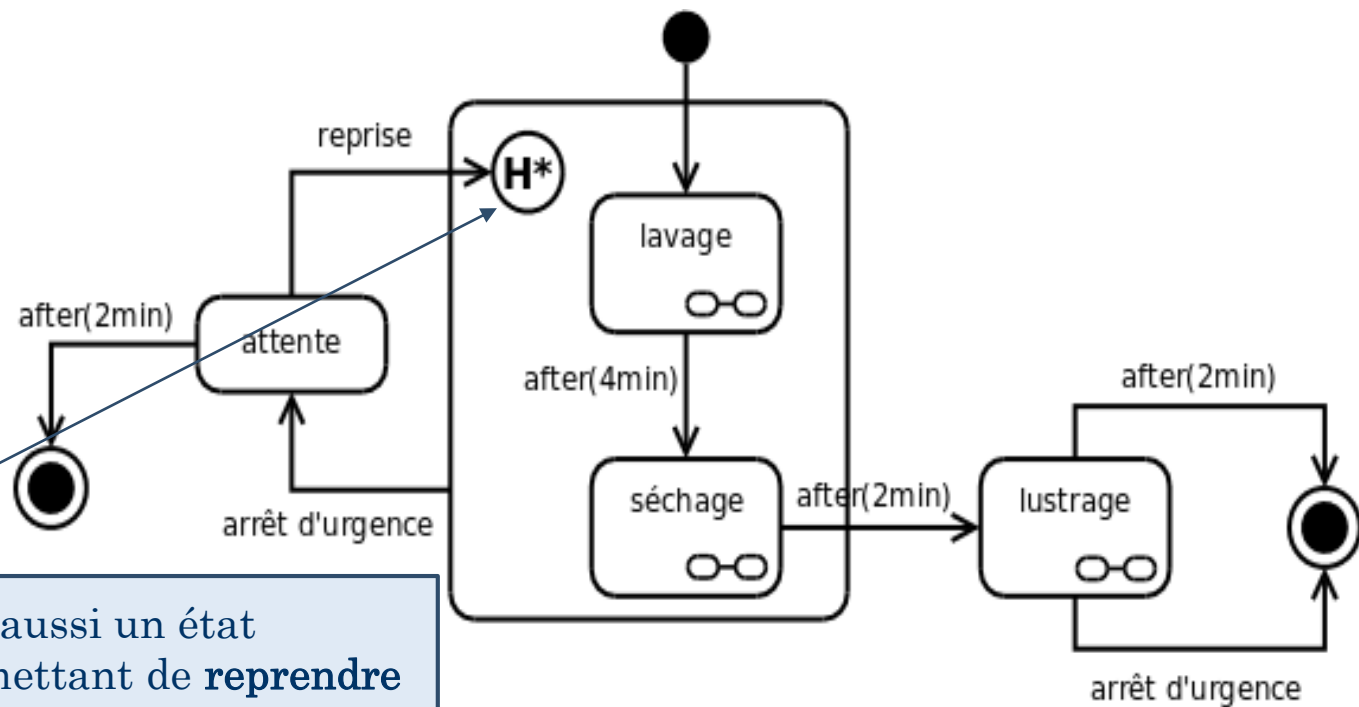
Les états *Lavage*, *Séchage* et *Lustrage* sont des états **composites** définis sur trois autres diagrammes d'états-transitions.

IV. UML

Diagramme d'états-transitions

Exemple:

Diagramme d'états-transitions modélisant le lavage automatique d'une voiture.

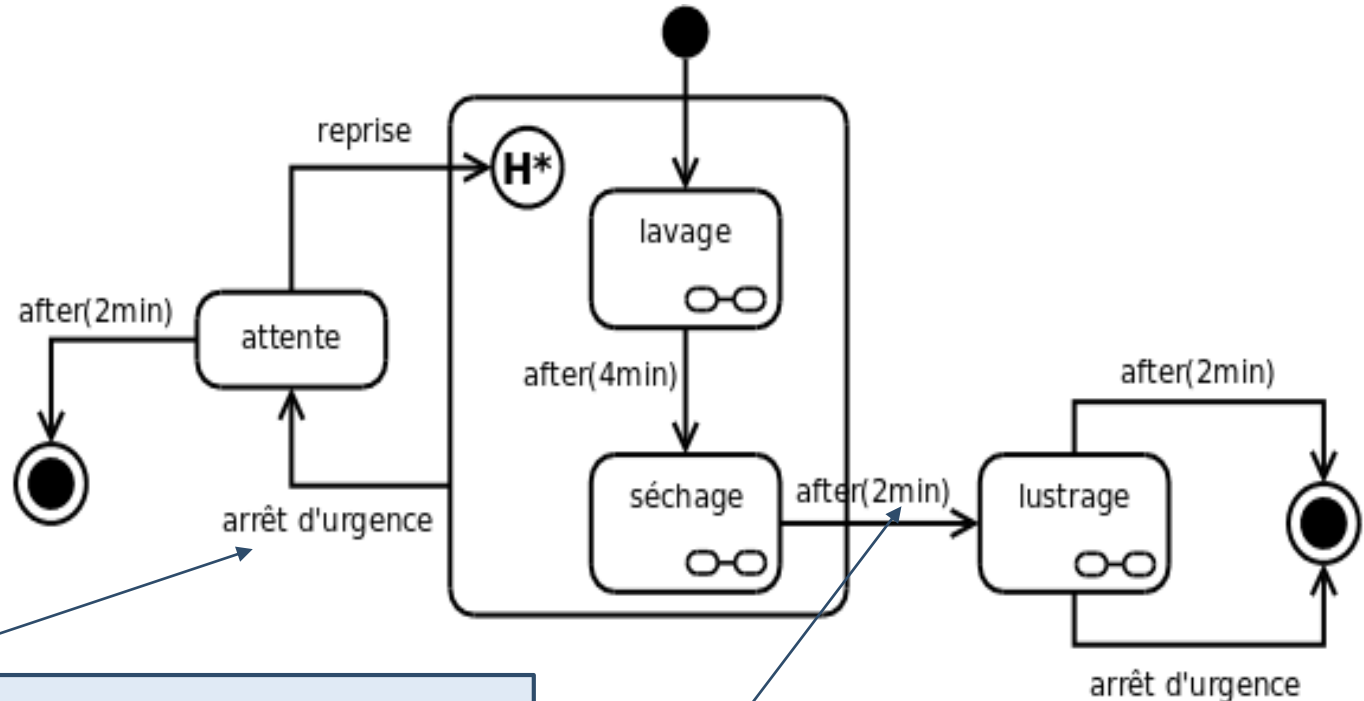


Ce diagramme possède aussi un état **historique** profond permettant de **reprendre** le programme de *lavage* ou *séchage* d'une voiture à l'endroit où il **était arrivé** avant d'être interrompu.

IV. UML

Diagramme d'états-transitions

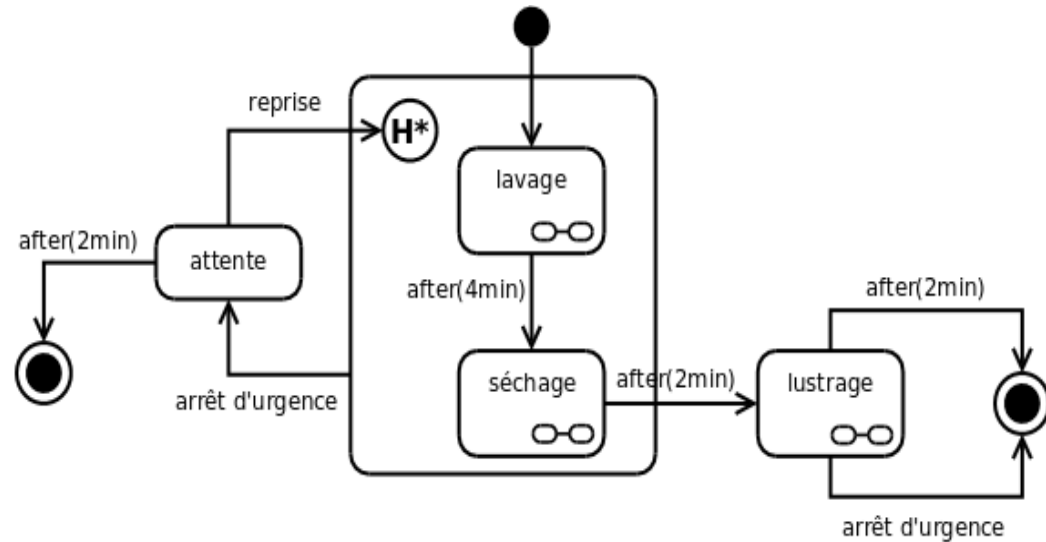
Exemple: Diagramme d'états-transitions modélisant le lavage automatique d'une voiture.



En phase de *lavage* ou de *séchage*, le client peut appuyer sur le bouton d'arrêt d'urgence. S'il appuie, la machine se met en attente. Il a alors deux minutes pour reprendre le lavage ou le séchage, **exactement** où le programme a été interrompu.

IV. UML

Diagramme d'états-transitions



Remarque importante:

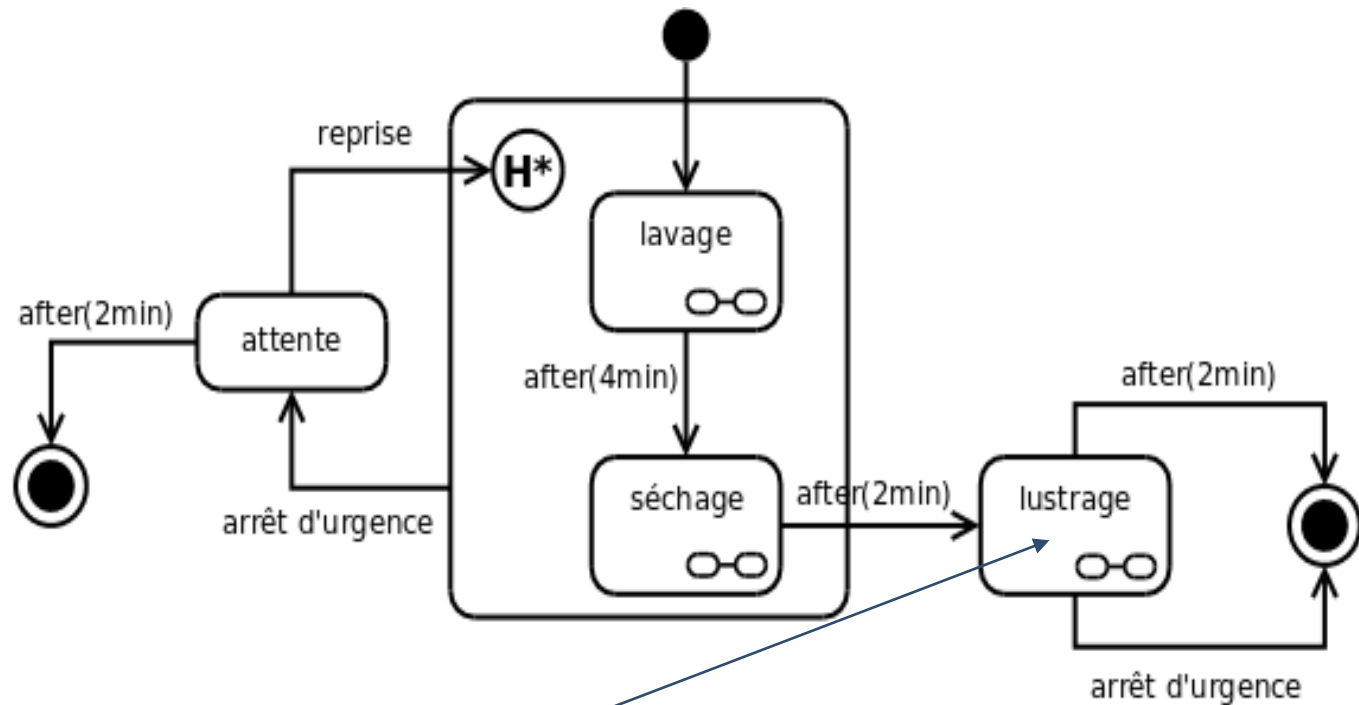
- H*** Le programme reprend au niveau du dernier sous-état actif des états de lavage ou de séchage.
- H** Si l'état avait été un état historique plat, c'est toute la séquence de lavage ou de séchage qui aurait commencé.

IV. UML

Diagramme d'états-transitions

Exemple:

Diagramme d'états-transitions modélisant le lavage automatique d'une voiture.



En phase de *lustrage*, le client peut aussi **interrompre** la machine, mais dans ce cas, la machine s'arrêtera définitivement.

IV. UML

Diagramme d'états-transitions

3. Hiérarchie dans les machines d'états.

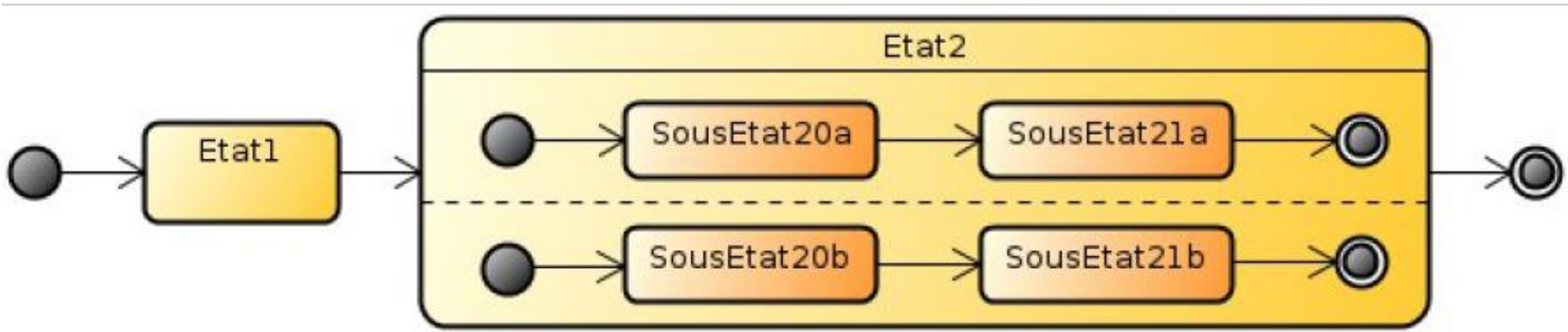
1. État composite.
2. État historique.
3. Gestion de la concurrence.

IV. UML

Diagramme d'états-transitions

3. Gestion de la concurrence.

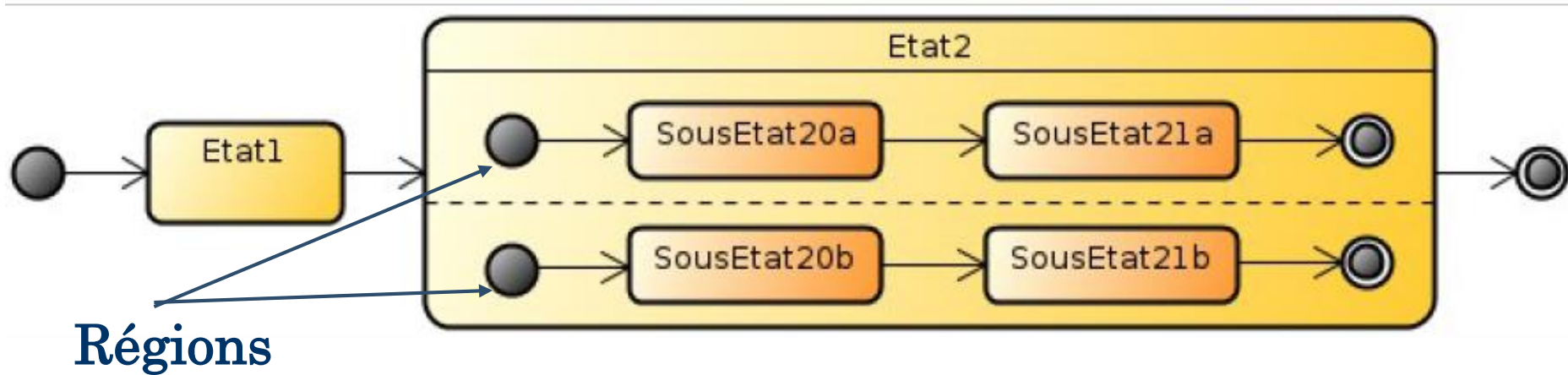
- Les diagrammes états-transitions permettent de décrire les **mécanismes concurrents** grâce à l'utilisation d'**états orthogonaux**.
- Un **état orthogonal** est un état qui possède **plusieurs régions** (séparées par des pointillés horizontaux) qui évoluent **simultanément et en parallèle**.
- Chaque **région** représente un **flot d'exécution**, elle peut posséder un **état initial** et un **état final**.



IV. UML

Diagramme d'états-transitions

3. Gestion de la concurrence.



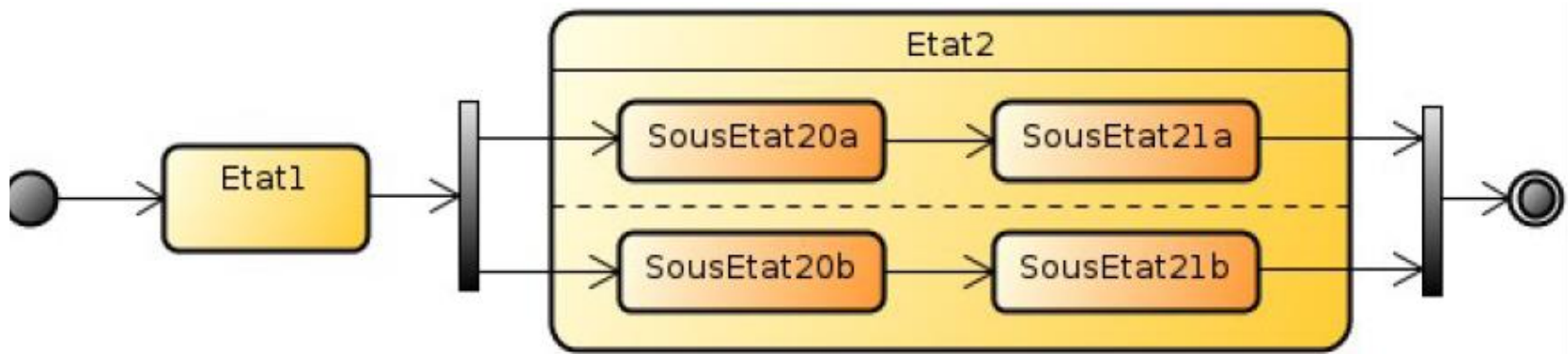
- Lorsque l'état2 est activé, l'activation des **états initiaux** de **chaque région** est **exécutée**.
- Si la **transition externe** qui permet de sortir de l'états2 est une transition automatique, elle est franchie uniquement lorsque **toutes les régions** ont leur **état final actif**.

IV. UML

Diagramme d'états-transitions

3. Gestion de la concurrence.

- Il est aussi possible de représenter des **flots d'exécutions parallèles** en utilisant la notation des transitions concurrentes (barres épaisses)



IV. UML

Diagramme d'états-transitions

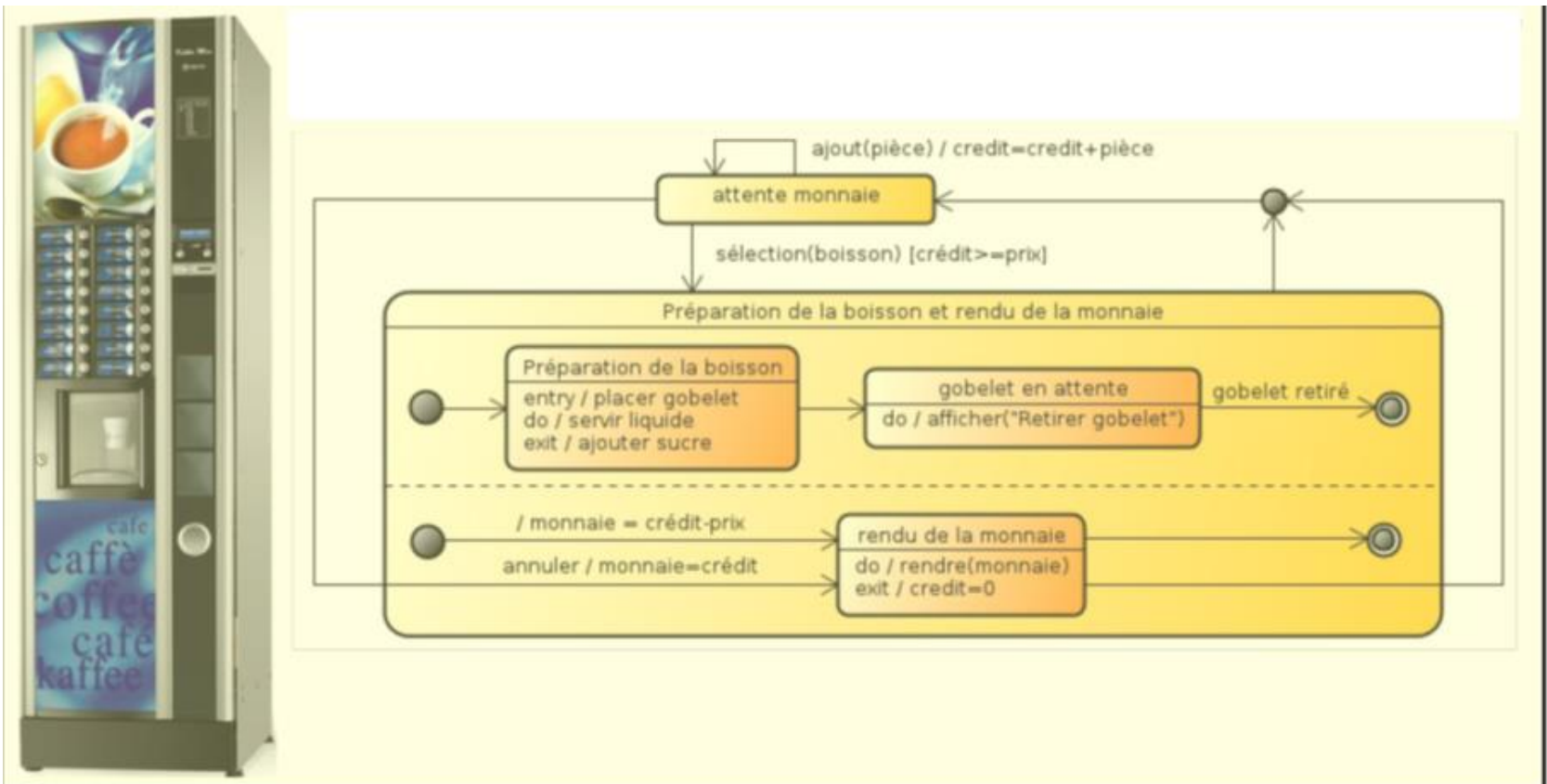
1. Introduction.
2. Les éléments constitutifs d'un diagramme d'états-transitions.
3. Hiérarchie dans les machines d'états.
4. **Exemples & Exercices.**

IV. UML

Diagramme d'états-transitions

4. Exemples.

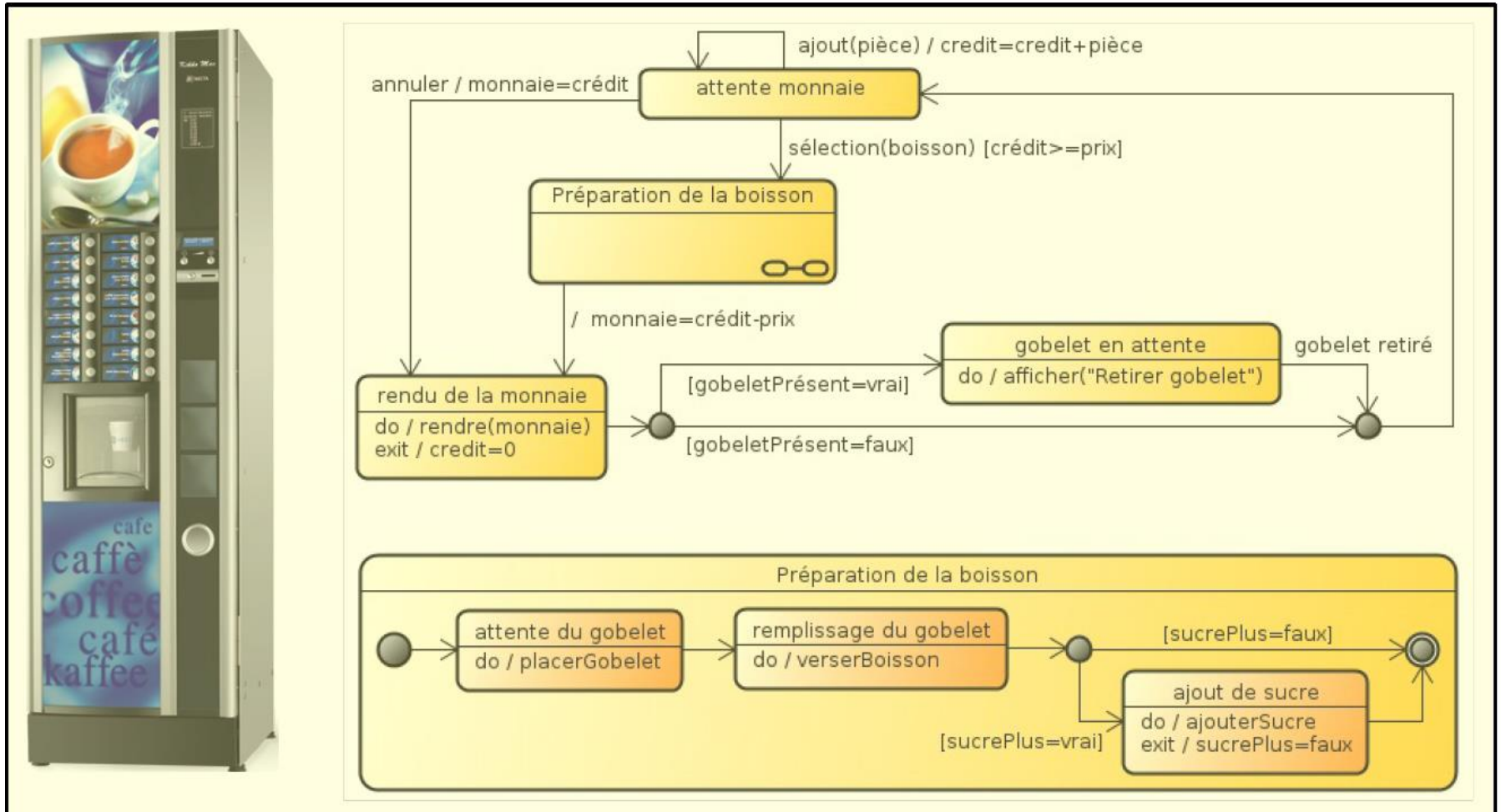
Soit un distributeur automatique de boisson, si nous considérons que nous pouvons rendre la monnaie en même temps que se prépare la boisson, nous obtenons le diagramme d'états-transitions suivant:



IV. UML

Diagramme d'états-transitions

4. Exemples.



IV. UML

Diagramme d'états-transitions

4. Exercices.

On considère une boîte de vitesse automatique de voiture. La boîte au démarrage est au point mort. La marche arrière ainsi que la position parking peuvent être enclenchées à partir du point mort. La première marche avant peut également être enclenchée à partir du point mort.

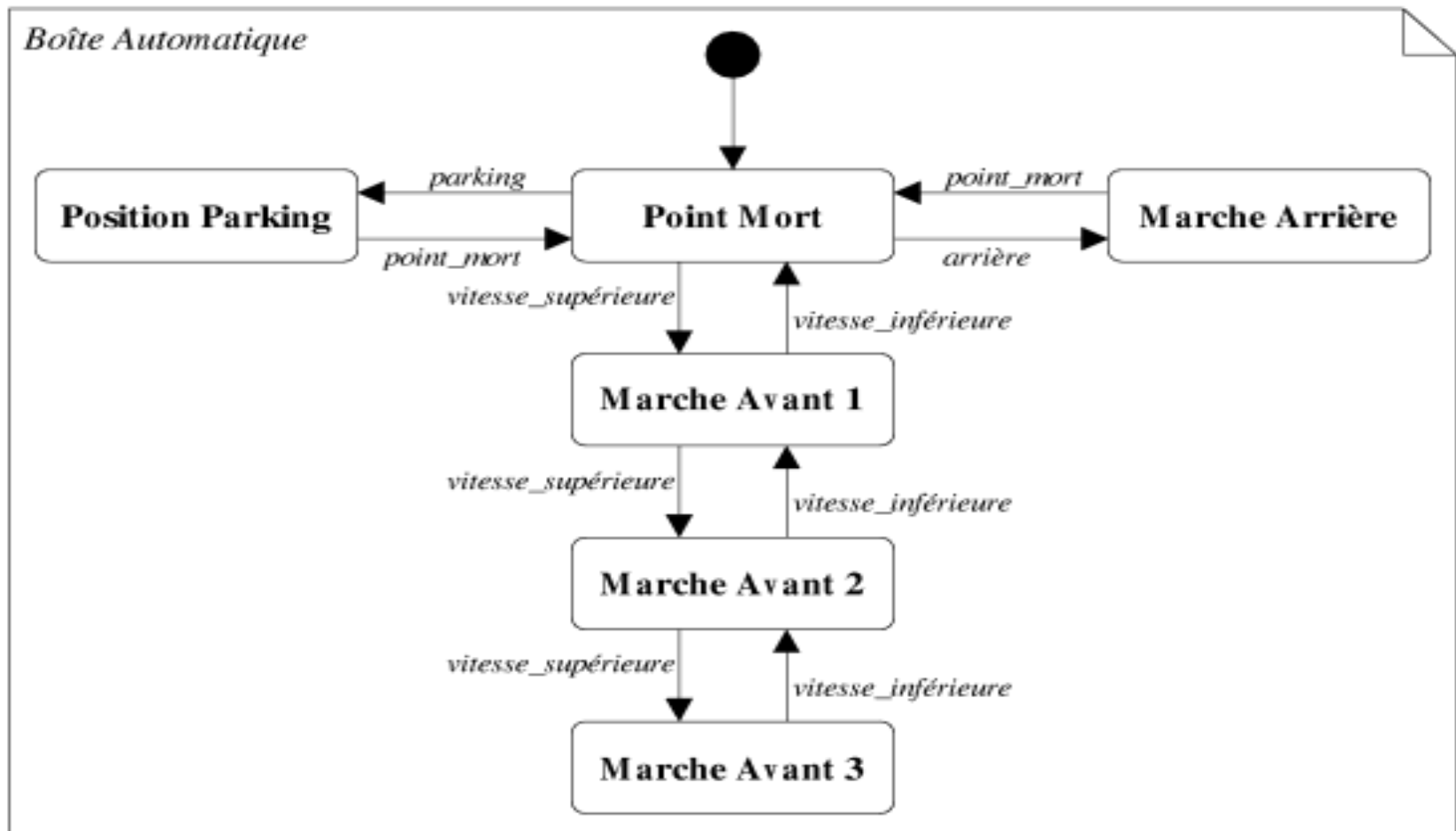
En revanche, les autres marches avant, la seconde et la troisième, sont enclenchées en séquence: 1-> 2 -> 3 pour une accélération, et 3 -> 2 -> 1 pour une décélération.

Seules la marche arrière, la position parking et la première marche avant peuvent être armées directement du point mort.

IV. UML

Diagramme d'états-transitions

4. Exercices.



IV. UML

Diagramme d'états-transitions

4. Exercices.

On désire modéliser le mécanisme d'une montre digitale. Une montre digitale simple comporte un affichage et deux boutons de réglage. On considère pour l'instant la montre avec mode de fonctionnement (affichage et réglage). Le mode réglage possède deux sous-modes (réglage des minutes et réglage des heures). Le bouton A est utilisé pour changer de mode, ce qui s'effectue de manière cyclique:

Affichage -> réglage minutes -> réglage heures -> affichage

IV. UML

Diagramme d'états-transitions

4. Exercices.

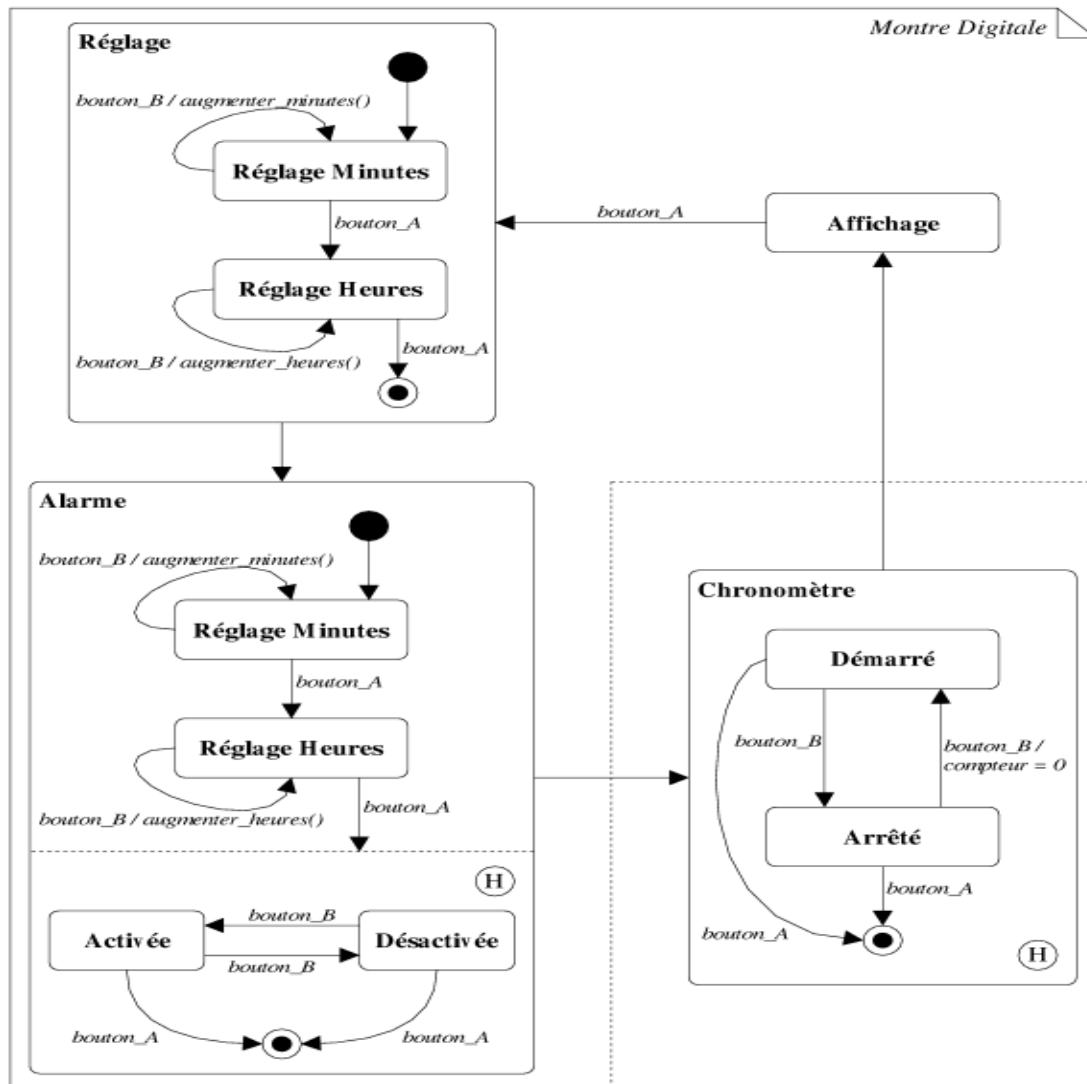
Dans les deux sous-modes de réglage, le bouton B permet d'augmenter d'une minute ou d'une heure chaque fois qu'il est appuyé. On ajoute les modes chronomètre et alarme à la montre. L'alarme se programme avec le bouton B (de la même manière que le réglage simple de la montre). Le chronomètre est lancé et stoppé également avec le bouton B. Le passage d'un mode à l'autre d'effectue toujours avec le bouton A:

Affichage -> réglage -> alarme -> chronomètre -> affichage

Le chronomètre fonctionne en parallèle avec les autres modes, et l'alarme possède un état interne (activée ou désactivée), indépendant des autres états, qui se règle avec le bouton B.

IV. UML

Diagramme d'états-transitions



IV. UML

Diagramme d'états-transitions

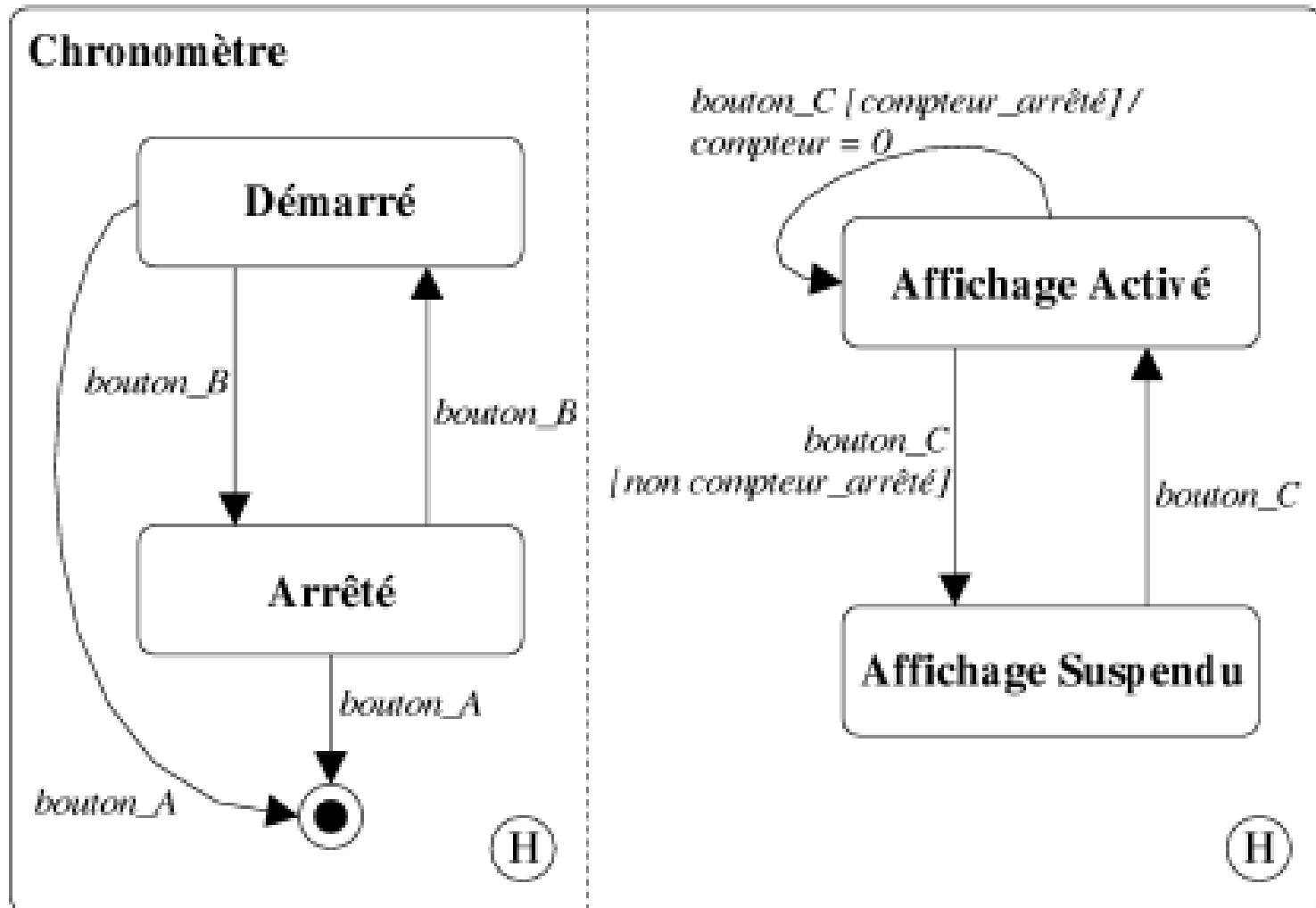
4. Exercices.

On ajoute un bouton C à la montre pour étendre les fonctionnalités du chronomètre.

Le bouton B sert alors à la mise en route, à l'arrêt et à la reprise du compteur. Le bouton C permet de suspendre ou de reprendre l'affichage, il remet également le compteur à zéro si le chronomètre est arrêté

IV. UML

Diagramme d'états-transitions



IV. UML

Diagramme d'états-transitions

**Enoncés dans mon centre de
ressources**

Bibliographie

1. Benoît CHARROUX, Aomar OSMANI, Yann THIERRY-MIEG. UML2 Pratique de la modélisation. 3^{ème} édition. PEARSON.
2. Laurent AUDIBERT. UML2 de l'apprentissage à la pratique. 2^{ème} édition. ELLIPSES.
3. Christian SOUTOU. Modélisation des bases de données (UML et les modèles entité-association). 3^{ème} édition. EYROLLES.
4. Chantal MORLEY, Jean HUGUES, Bernard LEBLANC. 4^{ème} édition. UML 2 pour l'analyse d'un système d'information. DUNOD.
5. Hugues BERSINI. L'orienté objet. 3^{ème} édition. EYROLLES.
6. Laurent DEBRAUWER, Fien VAN DER HEYDE. UML 2.5. 4^{ème} édition. ENI Editions.
7. Jean-Luc HAINAUT. Bases de données concepts, utilisation et développement. DUNOD.
8. Gilles ROY. Conception de bases de données avec UML. Presses de l'université du Québec.
9. Craig LARMAN. UML2 et les design patterns. 3^{ème} édition. PEARSON Education.
10. Frank BARBIER. UML 2 et MDE. DUNOD.
11. Laurent DEBRAUWER, Naouel KARAM. UML 2 entraînez-vous à la modélisation. Seconde édition. ENI Editions.
12. Corine COSTA. Cours Projets et bureau d'études.