


Cours n°7 : Diagramme d'états-transitions

1) Rôle du diagramme états-transitions (State Machine) :

Remarque préliminaire : Systèmes combinatoires et séquentiels

- **Dans un système combinatoire**, le comportement d'une machine (ou d'un objet) dépend uniquement des événements qu'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'il reçoit mais aussi de ce qui s'est passé avant ces événements.

Par exemple : Nous ne pouvons pas déterminer l'effet qu'aura l'appui sur la touche  d'un lecteur MP3 si nous ne connaissons pas la situation dans laquelle il se trouvait au préalable (S'il est déjà en lecture, l'appui sur la touche provoque la pause et s'il est à l'arrêt ou en pause l'appui lance alors la lecture).

Les machines ayant un fonctionnement séquentiel passent par un nombre de situations (états) limitées et clairement identifiées. Nous disons alors que ce sont des **machines à états finis**.

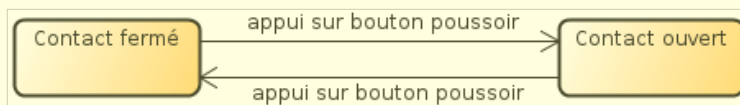
Le diagramme états-transitions (**State Machine Diagram** ou **Statechart Diagram**) fait parti des diagrammes comportementaux. Son rôle, est de décrire le fonctionnement d'une machine (ou d'un objet) ayant un comportement séquentiel.

Il représente les différents états (situations) dans lesquels peut se trouver la machine (ou l'objet) ainsi que la façon dont cette machine (ou l'objet) passe d'un état à l'autre en réponse à des événements.

Ex : Le télérupteur

Le télérupteur est un appareil qui permet de commander des points d'éclairage à partir d'autant d'endroits que nous désirons via des boutons poussoirs (Un appui sur l'un des boutons poussoirs provoque le changement d'état du contact qui alimente les lampes).

Le télérupteur possède deux états possibles (contact fermé ou contact ouvert). L'appui sur l'un des boutons poussoirs ne permet pas de déterminer l'état du contact suite à cet événement. Si le contact était fermé, il s'ouvrira et s'il était ouvert, il se fermera. Un même événement (l'appui sur un BP) provoque des réactions différentes en fonction de l'état du télérupteur au moment de l'événement (état antérieur).



Un diagramme d'états-transition est toujours associé à une et une seule classe.

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 pendant laquelle ce dernier attend un événement ou accomplit une activité. L'état de l'objet correspond donc à un moment de son cycle de vie. Pendant qu'il se trouve dans un état, un objet peut se contenter d'attendre un signal provenant d'autres objets. Il est alors inactif. Il peut également être actif et réaliser une activité. Une activité est l'exécution d'une série de méthodes et d'interactions avec d'autres objets.

L'ensemble des états du cycle de vie d'un objet contient un état initial. Celui-ci correspond à l'état de l'objet juste après sa création (défini par l'un des constructeurs de l'objet). Il peut également contenir un ou plusieurs états finaux. Ceux-ci correspondent à une phase de destruction de l'objet. Il arrive également qu'il n'y ait pas d'état final car un objet peut ne jamais être détruit (dans le cas des services notamment).

2) Représentation du diagramme états-transitions :

2-1) Etat :

§ **Un état** est une situation stable qui possède une certaine durée pendant laquelle un objet exécute une activité ou attend un événement.



- Un état est représenté par un rectangle arrondi contenant son nom.
- L'état initial est représenté par un point noir (état dans lequel il se trouve lors de sa création). ●
- Un état final correspond à une étape où l'objet n'est plus nécessaire dans le système et où il est détruit. Tous les objets n'ont pas d'état final. C'est notamment le cas des objets permanents dans le système. L'état final se représente par un point entouré d'un cercle. ○

2-2) Evénement:

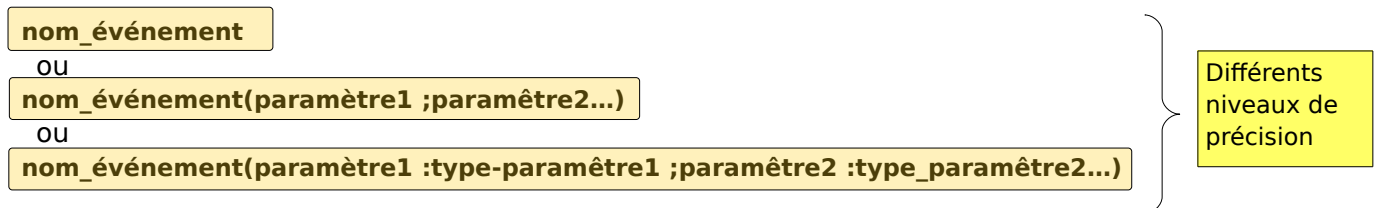
§ **Un événement** est un fait qui déclenche le changement d'état, qui fait donc passer un objet d'un état à un autre état (désactivation d'un état et activation de l'état suivant). Il est lié à la réception d'un signal (d'un message) par l'objet, demandé généralement par un autre objet.

Un événement se produit à un instant précis et est dépourvu de durée. Quand un événement est reçu, une transition peut être déclenchée et faire basculer l'objet dans un nouvel état.

Il existe quatre types d'événements :

• **Événement de type signal (signal event) :**

Correspond à la réception d'un signal **asynchrone** émit par un autre objet ou par un acteur. Le signal est très souvent associé à la gestion événementielle d'une structure complexe comme une IHM. Un événement de type signal peut être porteur de paramètres et s'écrit comme suit :



Les signaux sont déclarés dans le diagramme de classes de la même manière qu'une classe mais en ajoutant le stéréotype <<signal>> au dessus du nom. Un signal peut aussi être déclaré de façon textuelle.



• **Événement appel d'opération (call event) :**

Appel d'une méthode de l'objet courant par un autre objet ou par un acteur. Il possède la même syntaxe que l'événement de type signal. La méthode est aussi déclarée dans le diagramme de classe, mais à l'intérieur de la classe (pas dans un classeur stéréotypé <<signal>>).

• **Événement de changement (change event) :**

Un événement de changement d'état est généré par la satisfaction (vrai ou faux) d'une expression booléenne sur des valeurs d'attributs. Il s'agit d'une manière déclarative d'attendre qu'une condition particulière soit satisfaite. **L'expression est testée en permanence.**

La **syntaxe** est la suivante :

when(expression booléenne)

• **Événement temporel (time event) :**

Les événements temporels sont générés par l'écoulement du temps. Ils sont spécifiés soit de manière absolue (date précise), soit de manière relative (temps écoulé). Par défaut, le temps commence à s'écouler dès l'entrée dans l'état courant.

- **de manière absolue** (déclenchement à une date précise)

Syntaxe : **when(date= « expression précise d'une date »)** ex : **when(date=17/12/2010)**

- **de manière relative** (déclenchement après une certaine durée passée dans l'état actuel).

Syntaxe : **after(« expression d'une durée »)** ex : **after(10secondes)**

2-3) Transitions - transition externe :

Une transition définit la réponse d'un objet à l'arrivée d'un événement. Elle indique qu'un objet qui se trouve dans un état peut « transiter » vers un autre état en exécutant éventuellement certaines activités, si un événement déclencheur se produit et qu'une condition de garde est vérifiée.

§ **Syntaxe d'une transition externe :** Une transition externe est une transition qui modifie l'état actif. Il s'agit du type de transition le plus répandu. Une transition entre deux états est représentée par un trait droit fléché allant de l'état source vers l'état cible. L'événement qui détermine le franchissement de la transition est indiqué sous forme de texte. Si la transition est automatique, aucun événement n'est spécifié.



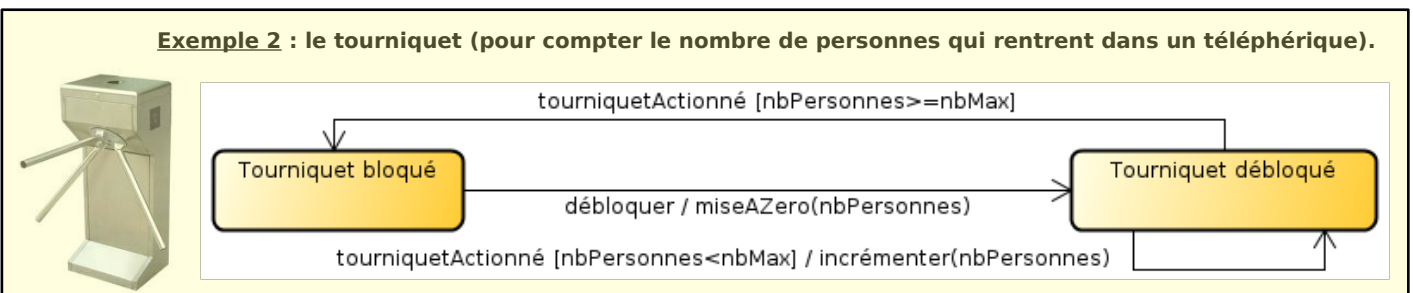
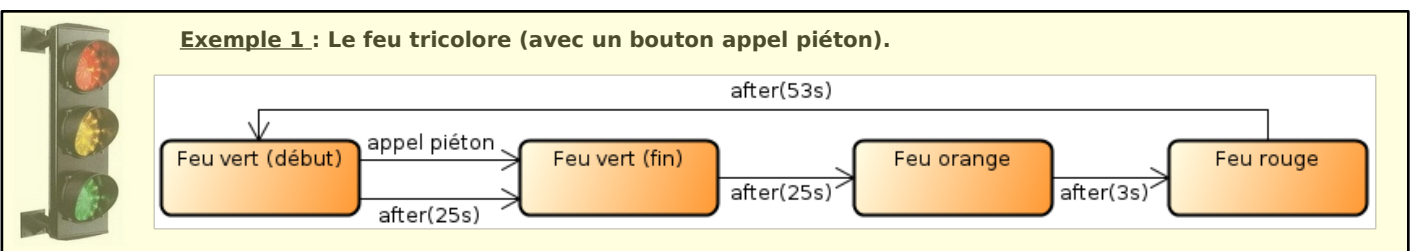
Une transition n'a pas de durée, elle est franchie instantanément (on dit qu'elle est déclenchée).

La structure de l'événement correspondant aux détails de la transition sont précisés au moyen de la syntaxe suivante : **déclencheur [garde] / effet** (chacun de ces éléments est facultatif)

- **déclencheur :** **nom de l'événement** qui provoque le franchissement de la transition. La syntaxe de l'événement est celui que nous venons de décrire dans la section précédente.
- **garde :** **condition de garde [entre crochets]**. Il est possible d'associer une condition à une transition, qui est alors appelée condition de garde. Pour que la transition soit franchie, il faut que la condition soit remplie en plus de la réception de l'événement associé, si celui-ci existe. C'est une expression booléenne (sur les attributs de l'objet ou les paramètres de l'événement déclencheur) qui doit être vraie pour que la transition soit déclenchée. La condition de garde est évaluée **uniquement** lorsque l'événement déclencheur se produit (contrairement à celle de l'événement de changement qui est testée en permanence).
- **effet :** spécifie une **activité** à effectuer lors du déclenchement (**nom de l'activité précédés de /**). Une activité est une série d'actions. Une action consiste à affecter une valeur à un attribut, créer ou détruire un objet, effectuer une opération (lancer une méthode), envoyer un signal à un autre objet ou à soi-même (Si c'est le cas, le nom du signal est précédé de ^), etc.

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

Remarque : Une transition sans nom d'événement est appelée **transition automatique**, elle est déclenchée lorsque l'activité de l'état source est terminée.

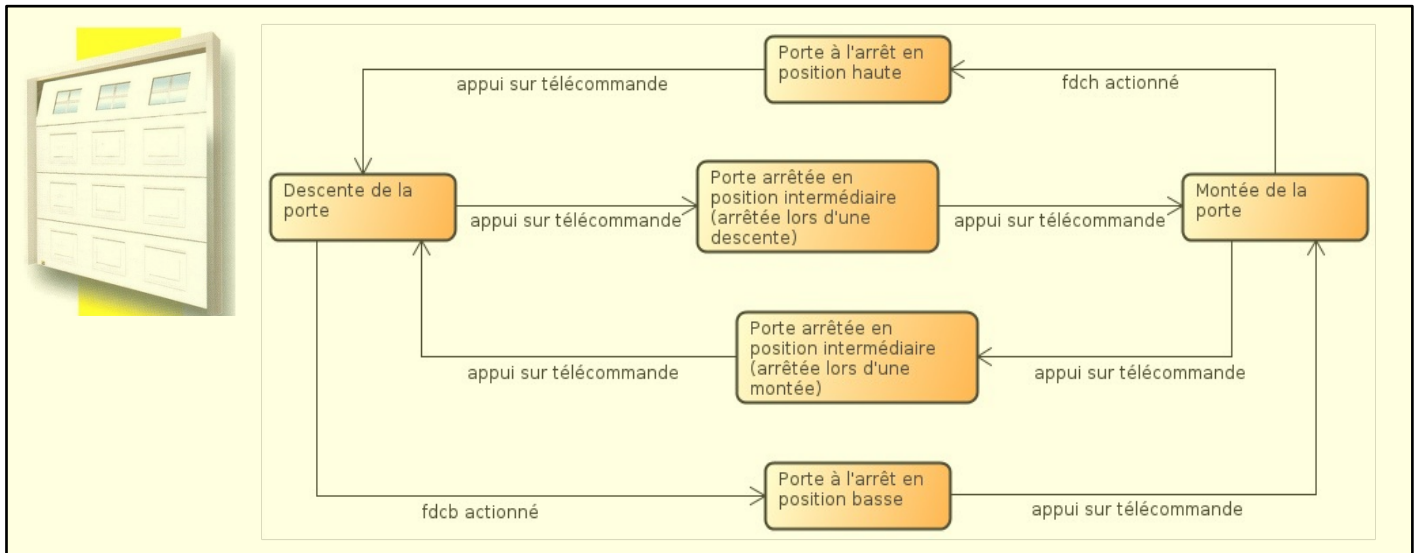


Exemple 3 : La porte de garage motorisée.

2 capteurs **fdch** (fin de course haut) et **fdcb** (fin de course bas) permettent de savoir si la porte est en position haute ou basse. L'appuie sur le bouton de la télécommande provoque :

- La montée de la porte si elle est à l'arrêt en position basse ou en position intermédiaire après une descente.
- La descente de la porte si elle est à l'arrêt en position haute ou en position intermédiaire après une montée.
- L'arrêt de la porte si elle est en mouvement.

L'action sur **fdch** ou **fdcb** provoque l'arrêt de la montée ou de la descente.



2-4) Transition interne :

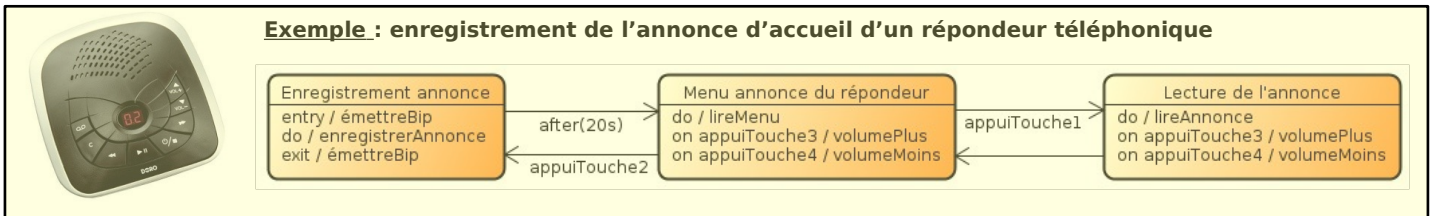
Les états que nous avons vus jusqu'à maintenant étaient relativement simples, ils n'effectuaient qu'une seule activité. Nous pouvons avoir des états qui effectuent plusieurs activités successivement ou en parallèle. L'enchaînement de ces activités à l'intérieur d'un même état peut être spécifié grâce à des transitions internes. La syntaxe d'une transition interne est la même que celle d'une transition externe :

déclencheur [garde] / effet

UML définit des mots clé correspondant à des événements particuliers :

- **entry** : définit l'activité à exécuter lors de l'entrée dans l'état.
- **exit** : définit l'activité à exécuter lors de la sortie de l'état.
- **do** : définit l'activité à exécuter dès que celle définie par **entry** est terminée.
- **on** : (optionnel) définit l'activité à exécuter à chaque fois que nous avons un événement particulier.

Les transitions internes s'écrivent à l'intérieur de l'état, séparé du nom de l'état par un trait.



2-5) Action et activité :

- Une **action** consiste à envoyer un signal, à faire appel à une méthode, à affecter une valeur à un attribut...
- Une **activité** est une série d'actions.

Il existe deux types d'activité :

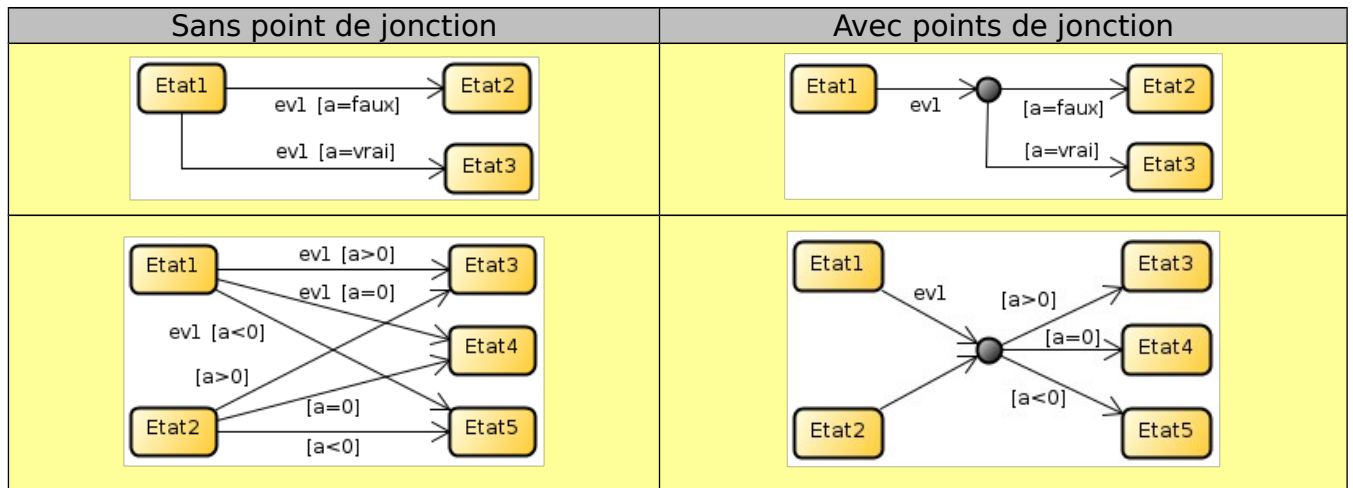
- **Les activités atomiques :**
Ce sont des activités qui n'ont pas de durée. Elles ne peuvent pas être interrompues (ex : affecter une valeur à un attribut). *Elles peuvent être associées à des états ou à des transitions.*
- **Les activités continus (ou durables) :**
Ce sont des activités qui ont une certaine durée, elles peuvent être par conséquent interrompues. *Elles ne peuvent être associées qu'à des états.*

2-6) Point de choix :

Il est possible de représenter des alternatives pour le franchissement d'une transition en utilisant des pseudo-états particuliers : les points de jonction et les points de décision.

a) Point de jonction :

Les points de jonction sont représentés par un cercle plein : ● Ils permettent à plusieurs transitions d'avoir une partie commune en partageant des segments de transition. L'utilisation de points de jonction a pour but de rendre la notation des transitions alternatives plus lisible.



b) Point de décision :

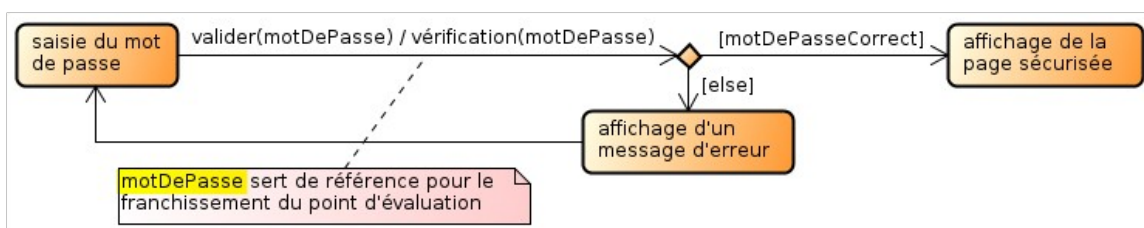
Les points de décision sont représentés par des losanges :

Ils ont un fonctionnement similaire à celui du point de jonction mise à part que nous pouvons sortir de l'état d'origine dès que le segment avant le point de décision est franchissable (même si aucun des segments après le point de décision n'est franchissable).

Le choix du segment à franchir derrière le point de décision se fait au moment de l'arrivée sur le point de décision. Cela permet aux segments qui sont derrière le point de décision d'avoir une condition de garde qui dépend d'éléments qui sont définis par une activité effectuée lors du franchissement du segment de transition avant le point de décision.

Remarque importante : Lorsque nous arrivons sur un point de décision, il faut qu'un des segments de transition qui suit, soit franchissable (une transition n'ayant pas de durée, nous ne pouvons pas rester à attendre quelque chose au niveau du point de décision). Pour éviter ce problème nous pouvons ajouter un segment avec la garde **[else]** qui est automatiquement franchie si aucune des gardes des autres segments n'est vraie.

Un segment de transition derrière un point de décision ne peut pas comporter d'événement.



2-7) État composite :

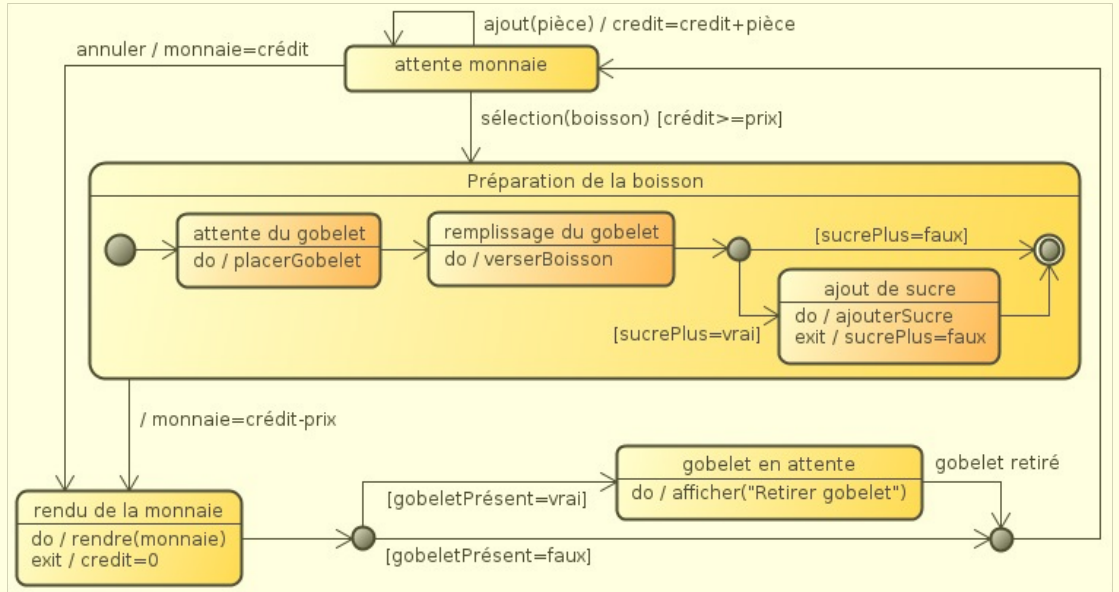
Certains états sont complexes et correspondent à la réalisation de plusieurs activités (séquentielles ou simultanées) qui ne pourront pas être définis par des transitions internes. *Il peut alors être intéressant de le décomposer en sous-états. On parle alors d'état composite.*



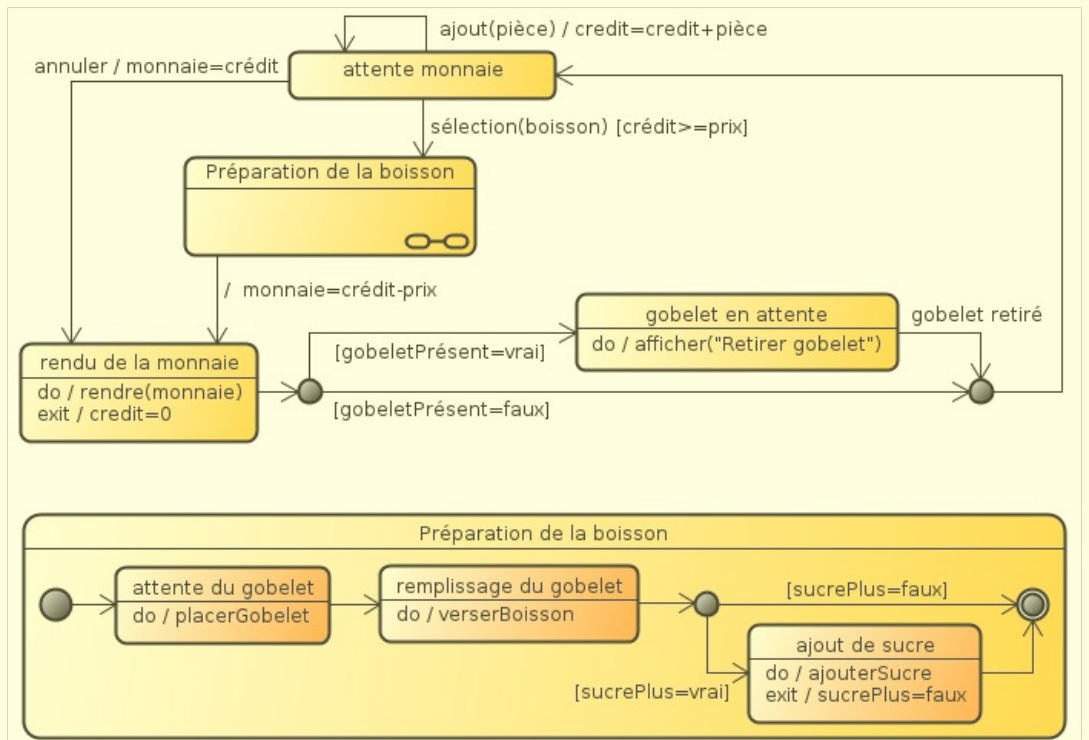
- Dès qu'un état composite est actif, il active son sous état initial.
- Si un état composite est raccordé à une transition automatique, elle est franchie lorsque nous atteignons le sous état final de l'état composite.
- Si une des transitions externes attenantes à l'état composite est franchissable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



Exemple : le distributeur automatique de boisson.



Afin d'éviter de surcharger le diagramme d'états-transition, il est possible de placer le symbole (**o-o**) à l'intérieur de l'état composite et de spécifier ensuite son contenu ailleurs dans une autre page.



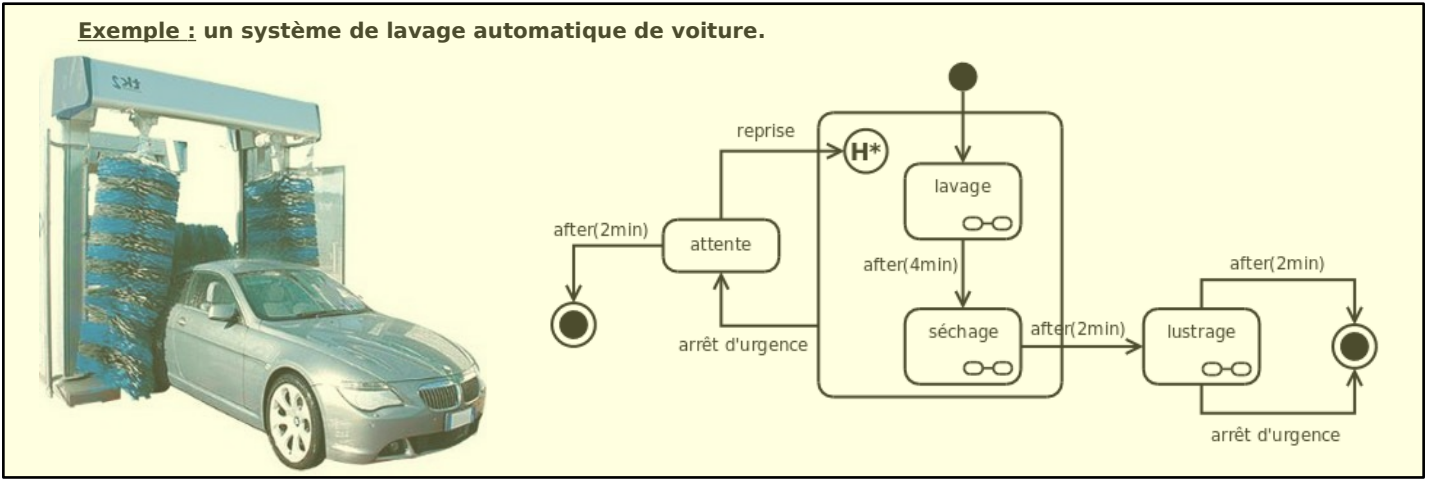
2-8) État historique :

Nous avons vu que lorsqu'une transition raccordée 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 **(H)** ou **(H*)** :

- **(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).

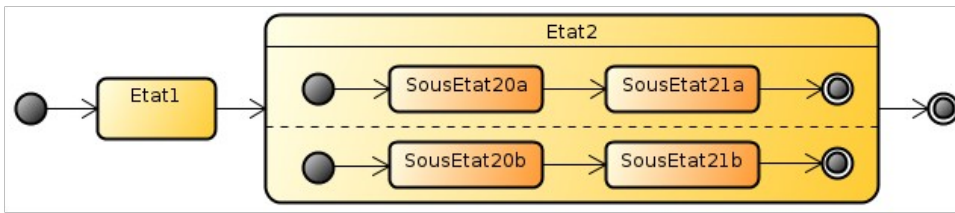
Exemple : un système de lavage automatique de voiture.



2-9) État orthogonal :

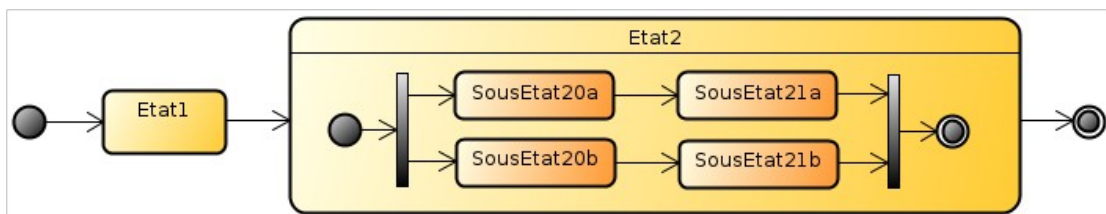
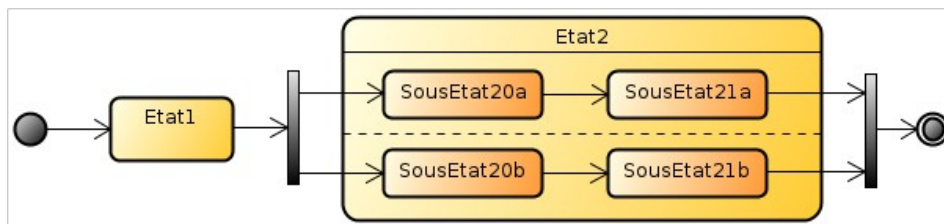
Les diagrammes états-transitions permettent de décrire efficacement les mécanismes concurrents grâce à l'utilisation d'états orthogonaux. Un état orthogonal est un état qui possède plusieurs régions (séparées par des pointillés horizontaux) qui évoluent simultanément et en parallèles. Chaque région représente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

Une transition qui atteint la bordure d'un état composite orthogonal est équivalente à une transition qui atteint les états initiaux de toutes ses régions concurrentes. Toutes les régions concurrentes d'un état composite orthogonal doivent attendre leur état final pour que l'état composite soit considéré comme terminé.



- 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'état2 est une transition automatique, elle est franchie uniquement lorsque toutes les régions ont leur état final actif.

Le tracé précédent est très clair, mais il est aussi possible de représenter des flots d'exécutions parallèles en utilisant la notation des transitions concurrentes (barres épaisses) :





Exemple : Dans l'exemple du distributeur automatique de boisson que nous avons vu précédemment, si nous considérons que nous pouvons rendre la monnaie en même que se prépare la boisson, nous obtenons le diagramme d'états-transitions suivant :

