LYCÉE LA MARTINIÈRE MONPLAISIR LYON
SCIENCES INDUSTRIELLES POUR L'INGÉNIEUR
CLASSE PRÉPARATOIRE M.P.S.I.
ANNÉE 2018 - 2019

C9 : MODÉLISATION DE LA CHAINE D'INFORMATION DES SYSTÈMES

C9-2 - Modélisation des systèmes à évènements discrets

18 Juin 2019

Table des matières

I Introduction **1**
1 Modélisation de la commande des systèmes . . . 1
2 Modélisation à l'aide de l'outil SysML . . . 1
II Diagramme d'états **2**
1 Introduction . . . 2
2 États . . . 4
a) Définition . . . 4
b) État composite . . . 4
c) État initial, état final . . . 5
3 Transition . . . 5
a) Définition . . . 5
b) Transition externe . . . 6
c) Transition interne . . . 6
d) Transition propre . . . 7
e) Événement . . . 7
f) Condition de garde . . . 8
g) Effet d'une transition . . . 9
4 Concurrence . . . 10
a) États orthogonaux . . . 10
b) Transitions complexes ou concurrentes . . . 10
5 Points de choix . . . 11
a) Point de jonction . . . 11
b) Point de décision . . . 12
6 État historique . . . 13
III Diagramme d'activités **14**
1 Définition . . . 14
2 Exemples d'applications . . . 15

Compétences

- **Analyser :**
  - Appréhender les analyses fonctionnelles et structurelles : systèmes à évènements discrets
- **Modéliser :**
  - Identifier et caractériser les grandeurs physiques : flux d'information
  - Proposer un modèle de connaissance et de comportement : systèmes à évènements discrets, chronogramme, structures algorithmiques
  - Valider un modèle : grandeurs influentes d'un modèle
- **Expérimenter :**

- Mettre en oeuvre un protocole expérimental : systèmes logiques à évènements discrets

- **Concevoir :** Architecture fonctionnelle et structurelle : système logique, systèmes à évènements discrets, Structures algorithmiques

I. Introduction

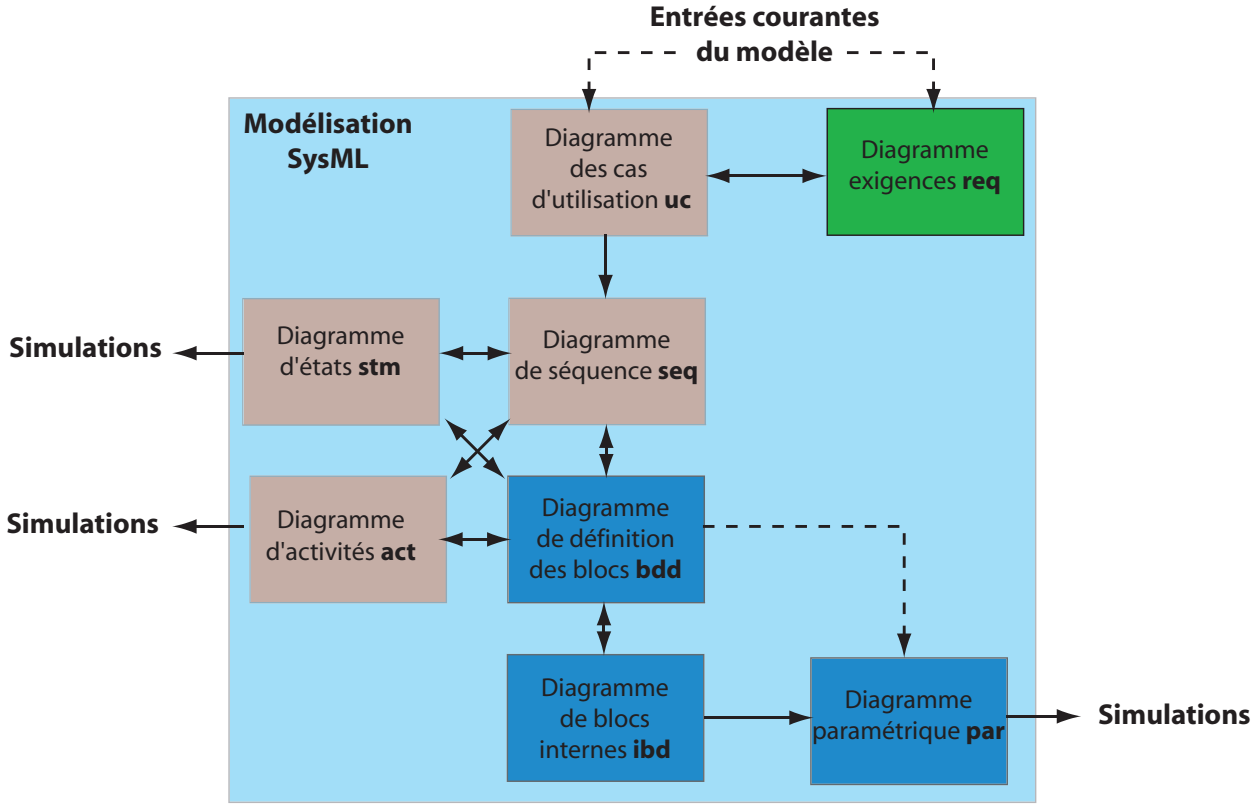
1 Modélisation de la commande des systèmes

Le chapitre précédent à permis de donner les bases de l'**étude des systèmes numériques** avec la manipulation des informations binaires. Ces informations sont **traitées** par des **cartes de commandes** qui sont généralement insérées dans des **micro-contrôleur**. Il faut maintenant faire le lien entre le **cahier des charges** et le **fonctionnement attendu du système** en mettant en relation toutes les informations issus de commandes ou de mesures numériques. Dans ce chapitre nous développerons les bases de lecture des **diagrammes comportementaux d'états et d'activités** nécessaires pour la commande et la simulation du fonctionnement d'un **système complexe**.

2 Modélisation à l'aide de l'outil SysML

L'outil SysML est un ensemble de diagrammes permettant de passer des **exigences d'un cahier des charges** à la simulation d'un système complexe dans le but d'en optimiser la conception et ainsi réduire les **écarts entre performances réelles et les performances attendues** d'un système.

Rappelons sur la figure suivante la structure de l'**outil SysML**.



On note bien que deux diagrammes comportementaux qui apparaissent dans ce diagramme n'ont pas encore été traité cette année : **le diagramme d'états et le diagramme d'activités**.

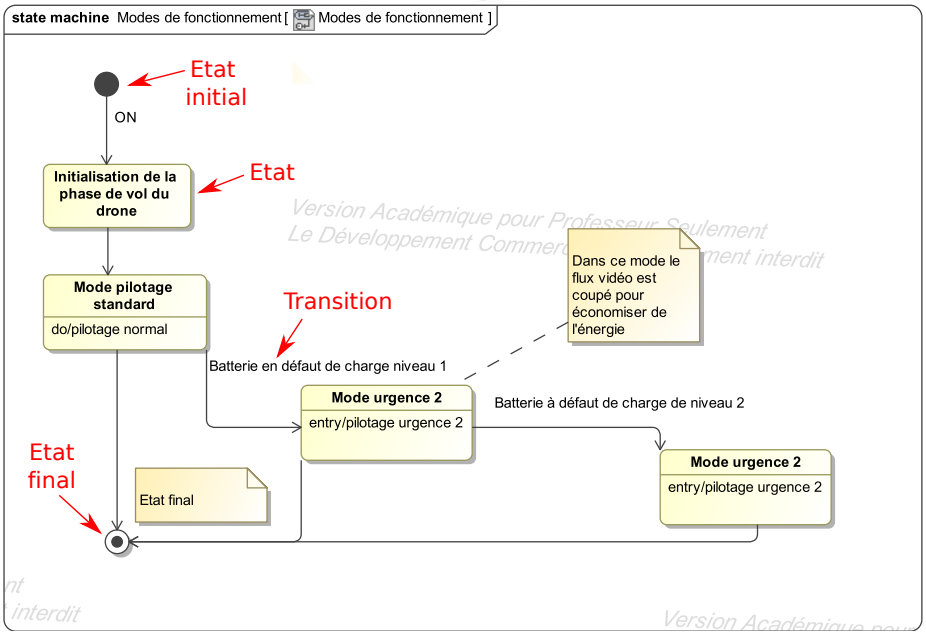
Ces deux diagrammes sont en lien avec le diagramme de définition des blocs puisqu'ils permettent de modéliser le comportement d'un bloc.

II. Diagramme d'états

1 Introduction

Le diagramme de d'état permet de modéliser le comportement d'un système ou d'une partie d'un système On parlera alors d'**objet**. Il peut être complété par un diagramme d'activité qui sera présenté brièvement en dernière partie de ce chapitre.

Exemple 1 : ***Diagramme d'états des modes de fonctionnement d'un drone de cinéma***



Ce diagramme permet de modéliser le passage d'un mode de fonctionnement à un autre du drone. Il est composé :

- d'états ;
- de transition,

Dans les parties suivantes nous détaillerons les différentes syntaxes existantes pour ce diagramme vis-à-vis de ces deux entités.

Définition 1 : *Diagramme d'état*

Le **diagramme d'états** est un diagramme comportemental appelé *State Machine Diagram* (**stm**) dans le langage SysML. Il est **rattaché à un bloc** (système, sous-système ou composant). Il permet de modéliser les différents états pris par le bloc en fonction d'évènements. Les éléments graphiques utilisés dans ce diagramme sont principalement :

- des rectangles aux coins arrondis pour les états,
- des flèches orientés de l'état de départ à l'état cible pour les transitions,
- des occurrences attachées à la transition spécifient les conditions de franchissement.

Remarque 1 :

Chaque bloc ne conduit pas forcément à un diagramme d'états car cela n'est pas toujours possibles.

2 États

a) Définition

Définition 2 : *États*

Un **état** représente une situation d'une durée finie durant laquelle un système ou sous-système (on parlera alors d'**objet**) :

- exécute une activité,
- satisfait à une certaine condition,
- ou bien est en attente d'un évènement.

Remarque 2 : *Règle de syntaxe*

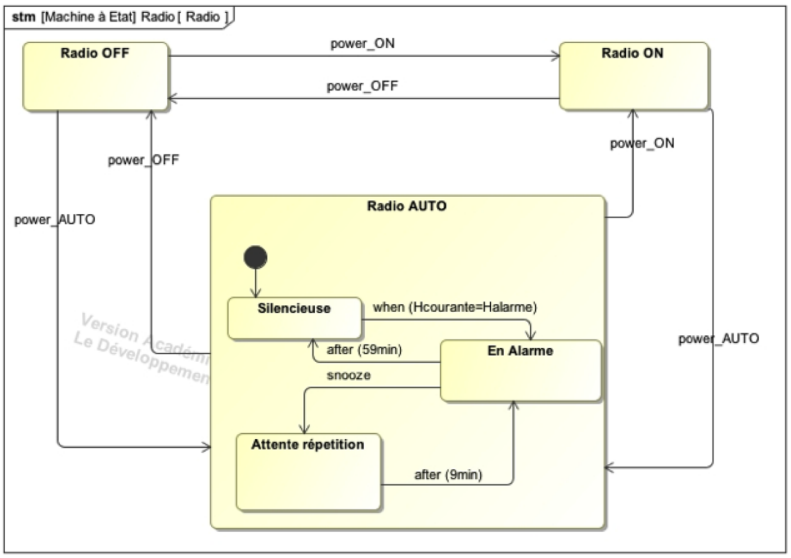
Un état peut être partitionné en plusieurs compartiments séparés par une ligne horizontale. Le premier compartiment contient le nom de l'état et les autres peuvent recevoir des transitions internes, ou des sous-états quand il s'agit d'un état composite. Dans le cas d'un état simple, on peut omettre toute barre de séparation.

b) État composite

Définition 3 : *État composite*

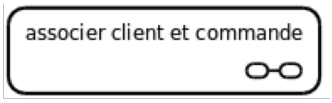
Un **état composite** peut contenir ou envelopper des sous-états.

Exemple 2 : *Diagramme d'état d'un radio-réveil*



Remarque 3 :

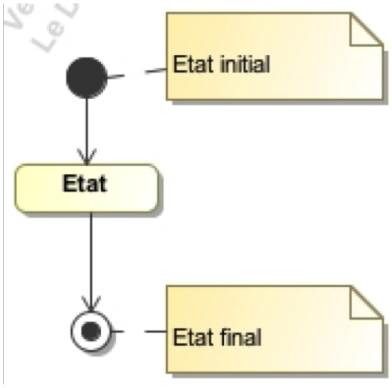
On peut représenter de manière abrégé un état composite. Ceci permet d'indiquer qu'un état est composite et que sa définition est donnée sur un autre diagramme.



c) État initial, état final

Définition 4 : *État initial, état final*

- Un **état initial** est un pseudo état qui indique l'état de départ par défaut, lorsque le diagramme d'états ou l'état enveloppant est invoqué.
- Un **état final** est un pseudo état qui indique que le diagramme d'états-transitions, ou l'état enveloppant est terminé.



3 Transition

a) Définition

Définition 5 : *Transition*

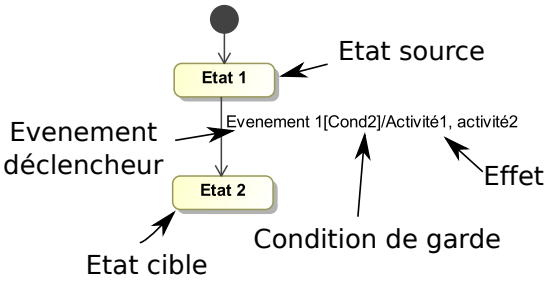
Une transition représente le passage d'un état à un autre.

b) Transition externe

Définition 6 : *Transition externe*

Une **transition externe** décrit la réaction d'un objet lorsqu'un évènement se produit. En règle général une transition possède :

- un état source : *Etat 1*,
- un état cible : *Etat 2*,
- un évènement déclencheur : *Evenement 1*,
- une condition de garde : *Cond2*,
- un effet : *activité 1* puis *activité 2*;



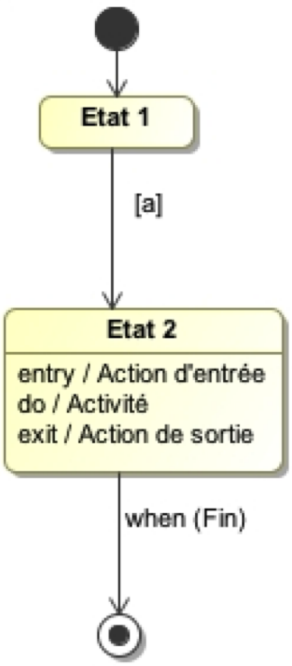
Syntaxe : évènement [garde] /effet.
Elle est représentée par une ligne terminée par une flèche. Elle relie un état source à un état cible.
Une transition ne peut être franchie que si son état source est actif.

c) Transition interne

Définition 7 : *Transition interne*

Une transition interne est notée à l'intérieur d'un état. Elle est précédée de l'un des mots clé suivants :

- **entry/action d'entrée** : entraine la réalisation d'une action à l'entrée de l'état,
- **do/Activité** : Entraine la réalisation d'une activité lorsque l'état est actif,
- **On évènement/activité** : entraine la réalisation d'une activité lorsqu'un évènement se réalise,
- **exit/Action de sortie** : entraine la réalisation d'une action à la sortie de l'état.

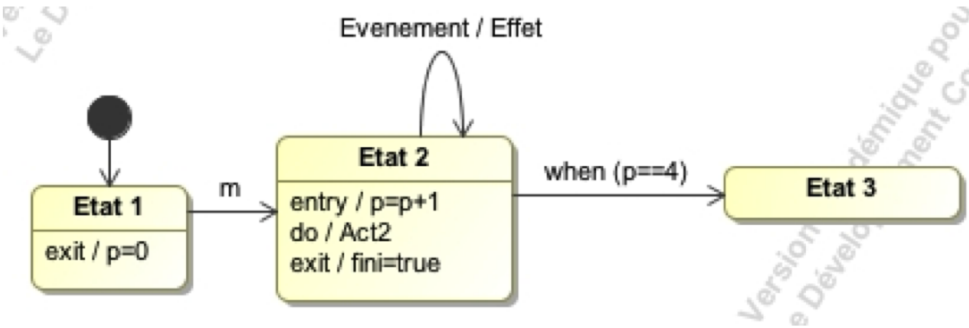


d) Transition propre

Définition 8 : *Transition propre*

Une transition propre concerne un état propre. Elle relie donc un état à lui-même.

- Dans l'exemple ci-dessous, lorsque **Etat2** est actif, si **Evenement** se produit, alors **Effet** (activité de courte durée) est déclenchée puis **Etat2** est de nouveau actif.
- Chaque fois que **Evenement** se produit, la réactivation de **Etat2** fait que **p** est incrémenté.
- Le fait de sortir et d'entrer dans **Etat2** peut avoir des effets néfastes comme l'interruption puis le redémarrage de **Act2**.



e) Événement

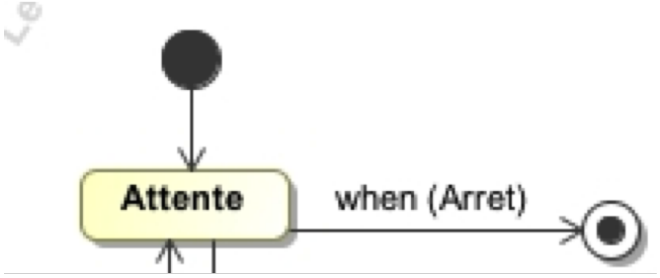
Définition 9 : *Évènement*

Un **événement** se produit pendant l'exécution d'un système. Les diagrammes d'états permettent de spécifier les réactions d'une partie du système à des événements discrets.

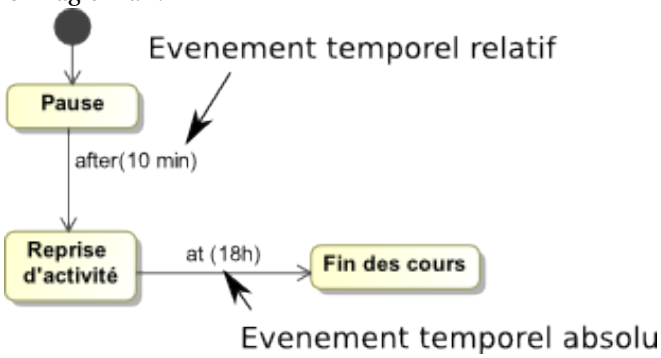
- Un événement se produit à un instant précis et est dépourvu de durée.
- Quand un évènement est reçu, une transition peut être déclenchée et faire basculer l'objet dans un nouvel état.
- On peut diviser les événements en plusieurs types explicites et implicites : signal, appel, changement et temporel.

Propriété 1 : *Types d'évènements*

- Un **événement de type signal** est destiné explicitement à véhiculer une communication asynchrone à sens unique entre deux objets. L'expéditeur crée et initialise explicitement une instance de signal et l'envoi à un objet. Il n'attend pas que le destinataire traite le signal pour poursuivre son déroulement. La réception d'un signal est un événement pour le destinataire. Par exemple l'appui sur un bouton provoque un signal.
- Un **événement de type changement** est généré par satisfaction (passage de faux à vrai d'une expression booléenne sur des valeurs d'attributs. La syntaxe est la suivante : **when** (< **condition_booléenne** >) Un évènement de changement est évalué continuellement jusqu'à ce qu'il devienne vrai et c'est à ce moment là que la transition se déclenche.



- Les **événement temporels** sont générés par le passage du temps. Ils peuvent être spécifiés de manière absolue (date précise) ou relative (temps écoulé). Par défaut le temps commence à s'écouler dès l'entrée dans l'état courant.
  - Un événement temporel spécifié de manière relative : **after** (< **durée** >),
  - Un événement temporel spécifié de manière absolue : **when** (< **date** >) ou **at** (< **date** >) dans le logiciel MagicDraw.



f) Condition de garde

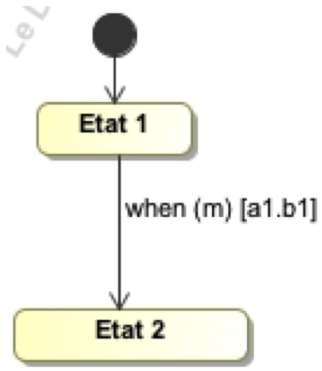
Définition 10 : *Condition de garde*

Une **condition de garde** est représentée entre crochet : $[garde]$
Il s'agit d'une expression logique sur les attributs de l'objet associé au diagramme d'états ainsi que sur les paramètres de l'événement déclencheur.

- La condition est évaluée uniquement lorsque l'événement déclencheur se produit.
- Si l'expression est fausse à ce moment là, la transition ne se déclenche pas.
- Si elle est vrai, la transition se déclenche et ses effets se produisent.

Exemple 3 :

Par exemple, ici :
- Lorsque l'événement de changement **m** se produit,
- si l'expression booléenne de la condition de garde $a_1 \cdot b_1$ est vrai, alors
- l'objet passe de l'**Etat1** à l'**Etat2**.



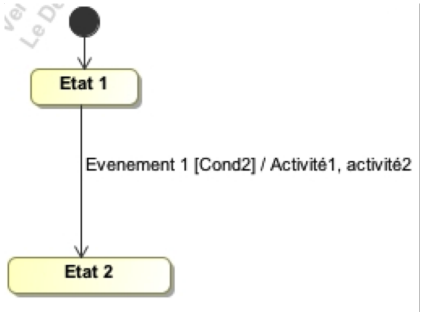
g) Effet d'une transition

Lorsqu'une transition se déclenche, son effet spécifié par "/**activité**" dans la syntaxe, s'exécute. Il s'agit généralement d'une activité qui 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.

La façon de spécifier l'activité à réaliser est laissée libre (langage naturel ou pseudo-code). Lorsque l'exécution de l'effet est terminée, l'état cible de la transition devient actif.

Exemple 4 :

- Quand **Evenement1** se produit, si **Cond2** est vrai, alors l'effet **Activité1** est déclenché.
- Quand **Activité1** est terminée, **Activité2** est déclenchée.
- Quand **Activité2**, l'objet passe à **Etat2**.



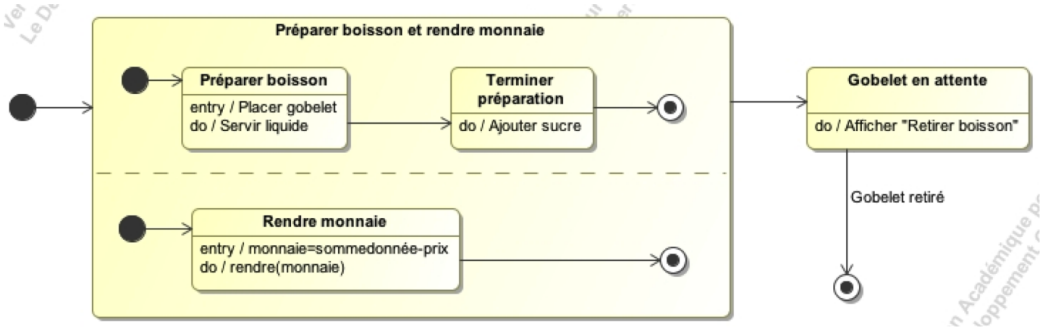
4 Concurrence

a) États orthogonaux

Définition 11 : *Etats orthogonaux*

Un **état orthogonal** est un état composite comportant plus d'une région représentant chacune d'elle un flot d'exécution. Ces états permettent de décrire efficacement les mécanismes concurrents.
Chaque région peut posséder un état initial et 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 concurrents.
Toutes les régions concurrentes d'un état composite orthogonal doivent atteindre leur état final pour que l'état composite soit considéré comme terminé.
Graphiquement dans un état orthogonal, les différentes régions sont séparées par un trait horizontal en pointillé allant du bord gauche au bord droit de l'état composite.

Exemple 5 :



b) Transitions complexes ou concurrentes

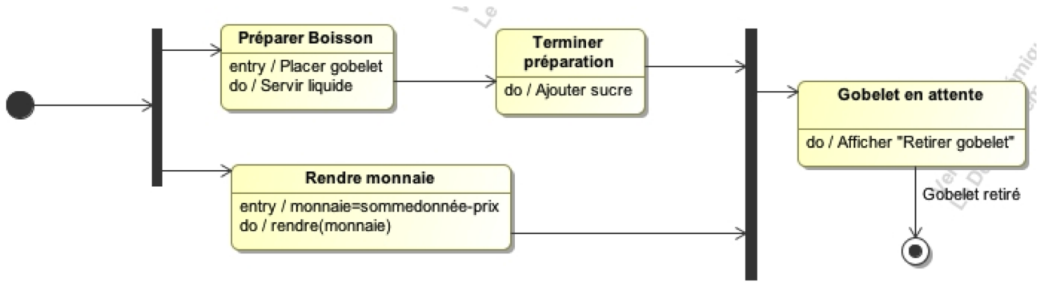
Définition 12 : *Transitions complexes ou concurrentes*

Les **transitions complexes ou concurrentes** sont représentées par une barre épaisse et peuvent éventuellement être nommées.

- Dès le franchissement de la bifurcation les états immédiatement suivants sont activés.
- L'union ne peut être franchies que lorsque toutes les activités des états immédiatement précédents sont terminées.

Exemple 6 :

Sur le diagramme ci-dessous l'état orthogonal *préparer boisson et rendre monnaie* n'apparaît pas pour alléger la représentation, car la notion de concurrence est clairement apparente de par l'utilisation de transitions complexes.



5 Points de choix

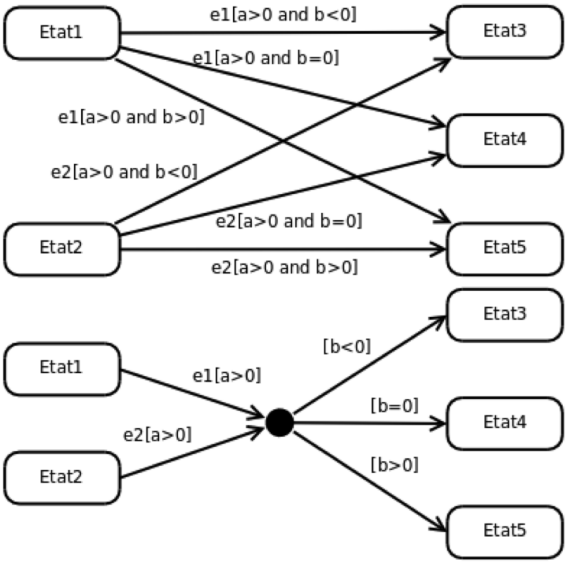
a) Point de jonction

Définition 13 : *Point de jonction*

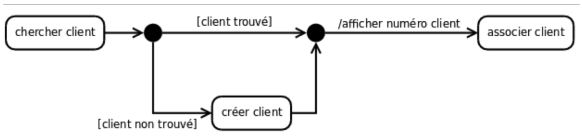
Un **point de jonction** permet de partager des segments de transition. L'objectif étant d'aboutir à une notation plus compacte ou plus lisible des chemins alternatifs. Un point de jonction peut avoir plusieurs segements de transition entrante et plusieurs segments de transition sortante. Par contre il ne peut avoir d'activité interne ni des transitions sortantes dotées de déclencheurs d'événements. Il ne s'agit pas d'un état qui peut être actif au cours d'un laps de temps fini. Toutes les gardes le long de ce chemin doivent s'évaluer à vrai dès le franchissement du premier segment.

Exemple 7 :

La figure suivante montre bien la meilleure lisibilité avec l'utilisation d'un point de jonction. Les deux diagrammes sont équivalents.



La figure suivante montre l'intérêt des points de jonctions lors d'un branchement conditionnel.

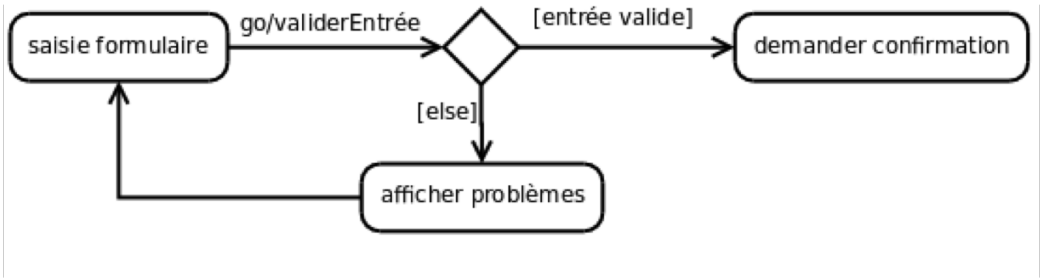


b) Point de décision

Définition 14 : ***Point de décision***

Un **point de décision** possède une entrée et au moins deux sorties. Contrairement à 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 des résultats obtenus en franchissant le segment avant le point de choix. Si, quand le point de décision est atteint, aucun segment en aval n'est franchissable, c'est que le modèle est mal formé.
Il est possible d'utiliser une garde particulière notée **[else]**, sur un des segments en aval d'un point de choix. ce segment n'est franchissable que si les gardes des autres segments sont toutes fausses. L'utilisation d'une clause **[else]** est recommandée après un point de décision car elle garantit un modèle bien formé.

Exemple 8 :



6 État historique

Définition 15 : *État historique*

Un **état historique** est un pseudo-état qualifié d'*état historique plat,* qui mémorise le dernier sous-état actif d'un état composite. Graphiquement il est représenté par un cercle contenant un **H**.

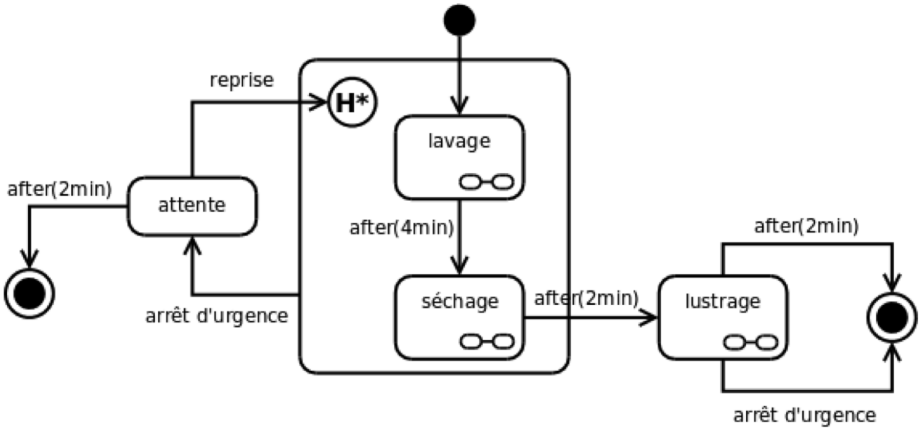
- Une transition ayant pour cible l'état historique est équivalente à une transition qui a pour cible le dernier état visité de l'état englobant.
- Un état historique peut avoir une transition sortante non étiquetée indiquant l'état à exécuter si la région n'a pas encore été visitée.

ll est également possible de définir un *état historique profond* représenté graphiquement par un cercle contenant un **H***. Il permet d'atteindre le dernier état visité dans la région, quel que soit son niveau d'imbrication, alors que l'état historique plat limite l'accès aux états de son niveau d'imbrication.

Exemple 9 :

La figure suivante montre un diagramme d'état modélisant le lavage automatique d'une voiture.

- Les états de lavage, séchage, et lustrage sont des états composites définis sur trois autres diagrammes d'états non représentés ici.
- En phase de lavage ou de séchage, le client peut appuyer sur le bouton d'arrêt d'urgence qui provoque le mise en attente de la machine.
- Il a alors deux minutes pour reprendre le lavage ou le lustrage exactement là où le programme a été interrompu : c'est à dire au niveau du dernier sous-état actif des états de lavage ou de lustrage (état historique profond).
- Si l'état avait été un état historique plat, c'est toute la séquence de lavage ou de lustrage qui aurait recommencé.
- En phase de lustrage, le client peut aussi interrompre la machine. Mais la machine s'arrêtera définitivement.



III. Diagramme d'activités

1 Définition

Définition 16 : *Diagramme d'activité*

Le **diagramme d'activité** est un diagramme comportemental appelé *Activity Diagram* (**act**) dans le langage SysML. Il permet de modéliser le déroulement d'un processus sous la forme d'une activité correspondant à la décomposition séquentielle d'actions aussi appelées tâches. Les éléments graphiques utilisés dans ce diagramme sont principalement :

- des rectangles aux coins arrondis pour les états,
- des flèches orientés de l'état de départ à l'état cible pour les transitions,
- des occurrences attachées à la transition spécifient les conditions de franchissement.

Remarque 4 :

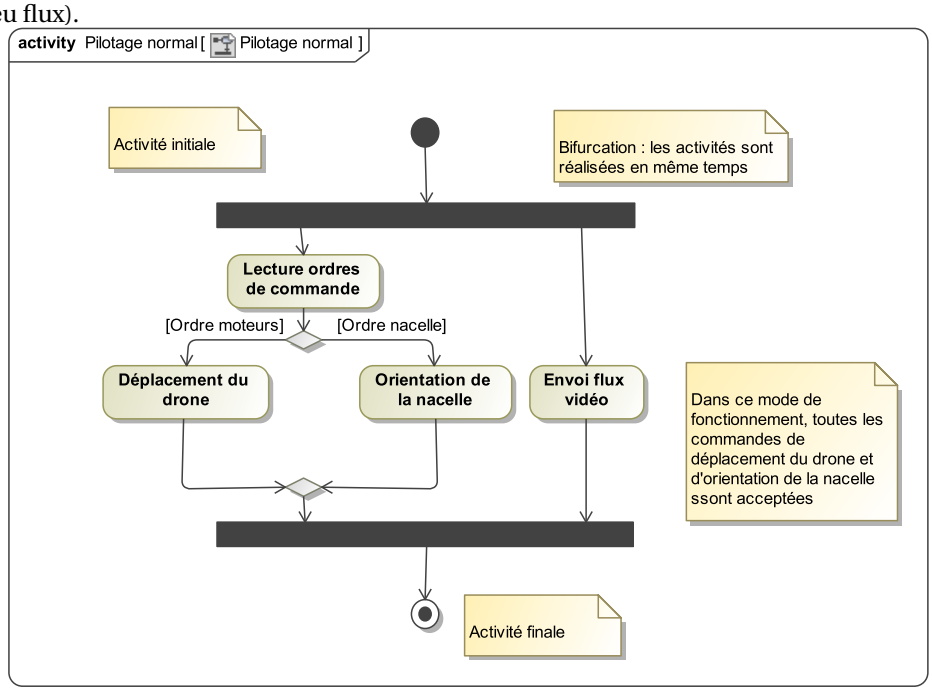
- Ce diagramme ne possède aucun événement associé aux transitions entre actions : la fin d'une action implique automatiquement le passage à la suivante, donc dans un ordre déterminé d'actions menant à un résultat. Lorsque le processus est enclenché il va à son terme selon un ordre précis.
- Ce diagramme permet aussi de représenter des **algorigrammes**, c'est à dire un flux de contrôle.
- Ce diagramme ne figure pas explicitement dans le programme.

2 Exemples d'applications

A l'aide de ce diagramme, on peut décrire plus en détail ce qui se passe dans chaque mode .

Exemple 10 :

- **Diagramme d'activité du mode de pilotage normal :** il correspond à un pilotage complet du drone avec en parallèle l'envoi du flux vidéo (les deux barres noirs marquent le début et la fin de l'envoi des deu flux).



- **Diagramme d'activité du mode d'urgence 2 :** il correspond à la procédure d'atterrissage automatique. La note précise les limitations de mode.