A decorative vertical bar on the left side of the page, consisting of a dark blue bar and a lighter blue bar with a pointed top. Below the bars, there are several thin, dark blue lines that curve upwards and to the right, resembling stylized grass or reeds.

informatique

Décrire le comportement des systèmes à événements discrets (SED)

PCSI 2019-2020

Desvaux Melaine
CPGE lycée Brizeux, Quimper

Table des matières

Introduction	2
1. Modélisation des SED par diagramme d'état	3
1.1 Présentation du diagramme d'état.....	3
1.2 Etat et ses activités associées	4
1.3 Transition, événement, garde et effets associées.....	4
1.4 Evolution du diagramme d'état par franchissement des transitions	7
1.5 Pseudos-états.....	8
1.6 Démarche de modélisation par diagramme d'état	9
2. Modélisation des SED par diagramme d'état avec état composite	10
2.1 Notion de hiérarchisation.....	10
2.2 Etat composite orthogonal.....	10
2.3 Historique d'un état composite	11
3. Modélisation d'un SED par diagramme de séquence	12
3.1 Rappel du diagramme des cas d'utilisation : <i>Use case</i> ou <i>uc</i>	12
3.2 Diagramme de séquence : <i>Sequence Diagram</i> ou <i>sd</i>	12
Sources.....	13

Introduction

On a étudié plus tôt dans l'année la façon dont se comportaient les systèmes continus usuels (ordre 1, ordre 2...) lorsqu'ils étaient soumis à des entrées, elles aussi usuelles (échelon, sinusoïde...). Dans ce cours, on s'intéresse au comportement des systèmes à événement discrets que l'on notera SED.

Les systèmes à événements discrets (SED) se définissent par opposition aux systèmes continus dont l'évolution est continue dans le temps et peut être décrite par des équations différentielles. Dans un SED, le passage d'un état à un autre est déclenché par des événements ponctuels. Contrairement aux systèmes continus où les informations traitées sont de nature analogique, les systèmes à événements discrets manipulent des informations logiques ou numériques.

Les SED sont le plus souvent séquentiels, c'est-à-dire que la ou les sorties dépendent de la combinaison des entrées et de l'état précédent des sorties.

Exemple : télécommande de téléviseur

Un appui sur le bouton POWER entraîne soit l'allumage ou soit l'extinction du téléviseur.

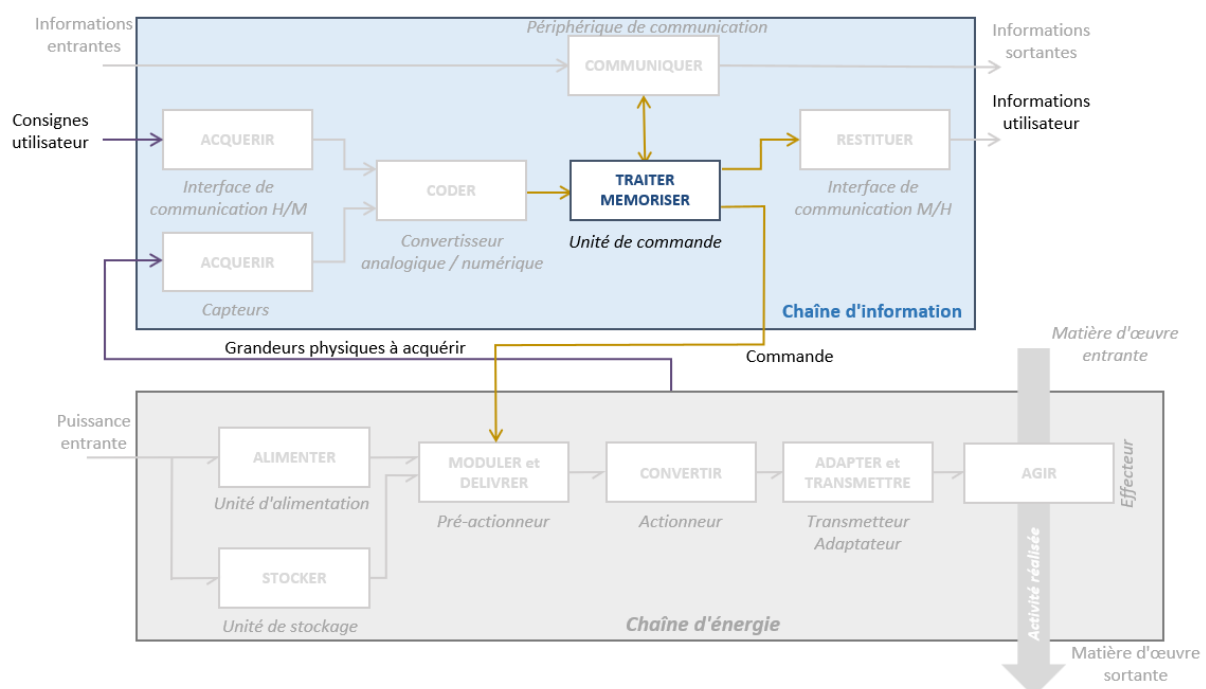
Ainsi, dans le cas d'un SED séquentiel :

- une même cause (même combinaison des entrées) peut produire des effets différents ;
- le temps peut être une cause déclenchante ;
- l'effet peut persister si la cause disparaît.

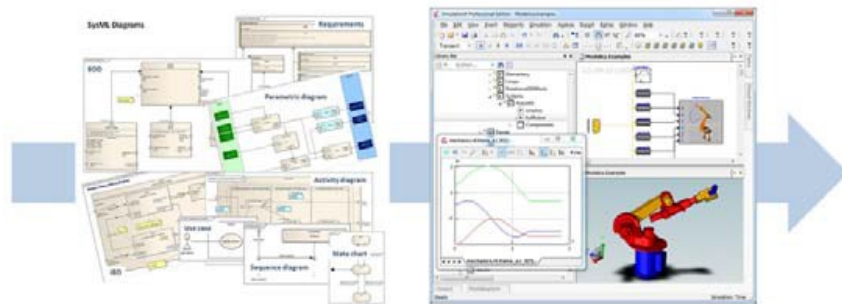


L'unité de commande de ces systèmes à événements discrets est très souvent réalisée à partir de circuits logiques et de calculateurs.

Elle est programmée à partir de l'analyse d'un cahier des charges avec comme objectif, d'aboutir à un fonctionnement du système conforme aux attentes de ses utilisateurs. Les **entrées** des SED sont des informations en provenance de l'IHM et des **capteurs** et les **sorties** sont des **commandes** vers les **IMH** et les **préactionneurs** des différentes chaînes fonctionnelles.



Comme pour toutes les autres étapes de conception d'un nouveau système, il se révèle plus efficace techniquement et économiquement de modéliser la structure du programme et de la simuler, avant tout choix de langage de programmation et réalisation du programme.



Elle peut être mise en oeuvre via le langage SysML et plus particulièrement grâce à l'utilisation des diagrammes d'état. Ceux sont ces différents diagrammes d'état que nous étudierons à travers ce cours.

1. Modélisation des SED par diagramme d'état

1.1 Présentation du diagramme d'état

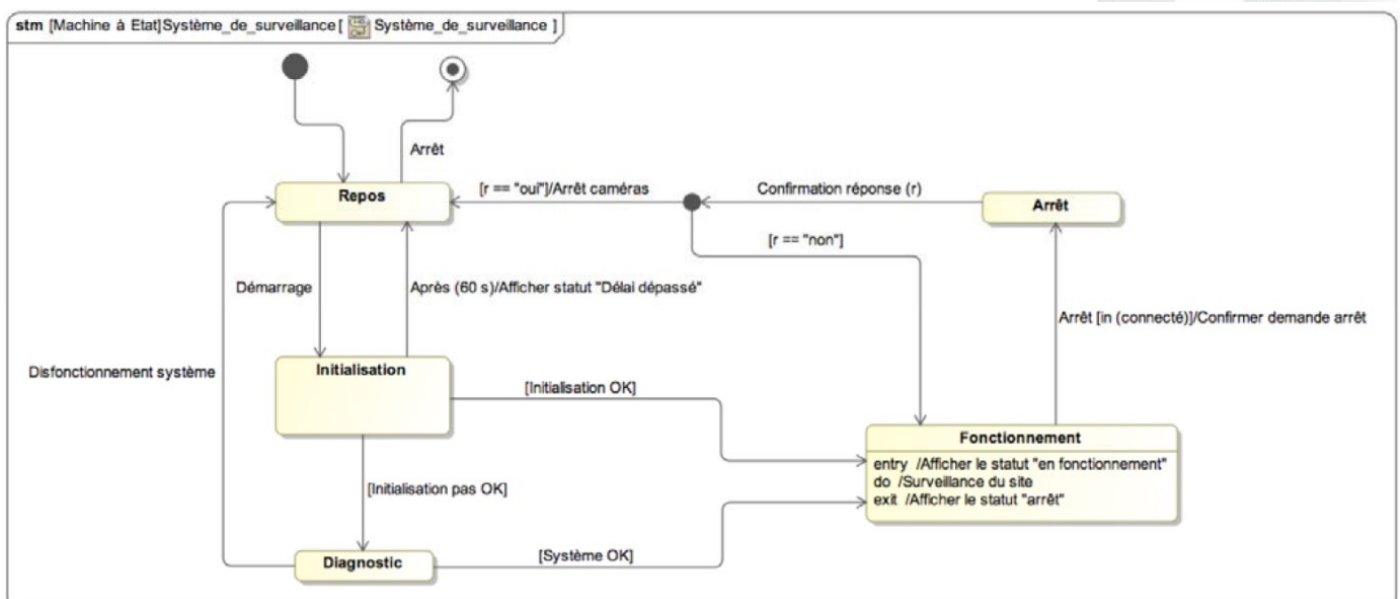
En langage SysML, un diagramme d'état (stm) est nécessairement associé à un bloc du diagramme de définitions de blocs (bdd) ou du diagramme de blocs internes (ibd). Ce bloc peut être le système, un sous-système ou un composant.

Le diagramme d'état décrit les **différents états pris par le bloc** ainsi que les **transitions possibles** entre ces différents états, et les **activités associées**.

Exemple : dispositif de vidéo surveillance

Le diagramme ci-dessous décrit le fonctionnement d'un système de vidéosurveillance. On y trouve :

- 5 états : *Repos*, *Initialisation*, *Diagnostic*, *Arrêt* et *Fonctionnement* ;
- des transitions entre les états, représentées par des flèches, et qui précisent sous quelles conditions le système passe d'un état à un autre.



On peut remarquer, avec cet exemple, que la représentation du comportement est en générale fonctionnelle dans les diagrammes d'état. Aucune information technique sur la manière dont sont transmises les informations, ni sur la façon dont sont réalisées les activités, n'est précisée.

1.2 Etat et ses activités associées

Un **état** (représenté graphiquement par un rectangle aux bords arrondis) modélise une **phase du fonctionnement** du système. Pendant cette période, l'état est dit **actif** et le système accomplit :

- une **simple activité** ;
- OU une **séquence d'activités** ;
- OU est **en attente**.

En dehors de cette période, l'état est dit **inactif**.

Par définition :

- il n'y a qu'un **seul état actif à chaque instant** ;
- un état possède un **titre unique** dans le diagramme.

Le lancement des activités à l'intérieur de l'état actif est organisé selon des mots réservés :

entry	Activité ayant une fin (ex. : allumer/éteindre un voyant), elle ne peut pas être interrompue . Elle est exécutée lors de l' activation de l'état .
do	Activités interruptibles . Elles sont exécutées dans l' ordre de leur écriture , à partir de l'instant où l'activité associée à entry est terminée.
exit	Activité ayant une fin , elle ne peut pas être interrompue . Elle est exécutée lors de la désactivation de l'état , juste avant la sortie de l'état

TITRE
Entry / activité1
Do / activité2
Exit / activité3

Remarques :

- les trois comportements **entry**, **do** et **exit** ne peuvent être **utilisés qu'une seule fois par état**, mais il est également possible de n'en **utiliser qu'une partie** (seulement **entry** par exemple) ;
- si **aucun mot** réservé n'est utilisé, cela correspond à un **do** ;
- un **état vide** (sans activité) indique un **état d'attente**.

1.3 Transition, événement, garde et effets associés

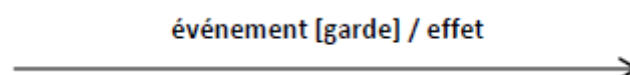
Une **transition** (représentée graphiquement par une flèche) modélise la possibilité d'un **passage** instantané d'un **état** vers un **autre**. On appelle **état source** l'état de départ d'une transition, et **état cible** l'état d'arrivée d'une transition.

La **transition** :

- n'a pas de durée ;
- n'est évaluée que si l'état source est actif.

Son **franchissement** est **conditionné** par des **événements** et des **conditions de garde**.

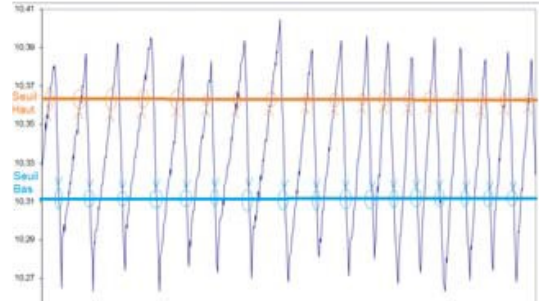
Ces événements et conditions de franchissement ainsi que l'éventuel effet associé à la transition sont indiqués le long de la flèche qui la symbolise suivant la notation :



Evènement

Un **évènement** correspond au **changement d'état** (supposé instantané) **0 ou 1** d'une **variable observée**. Il est **daté** dans le temps et est **traité instantanément** lors de son occurrence (apparition).

Exemples : appui sur un bouton, arrivée en fin de course d'un mécanisme, dépassement d'une valeur seuil...



Un **évènement** n'est **jamais mémorisé** et est donc **perdu** s'il ne mène à **aucune évolution** du diagramme d'état.

Il est possible d'utiliser des variables internes (compteurs ou horloge) pour spécifier un évènement :

<i>when</i> ($N=3$)	L'évènement apparaît lorsque la variable interne N devient égale à 3. Il permet par exemple d'utiliser un compteur.
<i>after</i> (T)	L'évènement apparaît après une durée T passée dans l'état d'amont. Il permet de réaliser une temporisation.
<i>at</i> (D)	L'évènement apparaît à la date D dans un référentiel de temps dont l'origine correspond généralement au démarrage du fonctionnement du système.

Si une **transition** n'a **pas d'évènement** spécifié, l'**évènement implicite** est la **fin des d'activités liées au do de l'état source** (les activités doivent donc avoir une fin).

Garde

La **garde** est une **condition de franchissement de la transition**. C'est une condition logique évaluée à l'instant de l'évènement (vrai ou faux, généralement notée 1 ou 0).

Contrairement à l'évènement qui, lui, est localisé dans le temps, la garde traduit une condition qui dure dans le temps et qui doit persister.

Exemple : Pour un bouton d'arrêt d'urgence :

- l'évènement est associé à l'instant où le bouton est enfoncé ;
- la garde est associée à l'état du bouton : enfoncé ou non.



Si une **garde** n'est **pas présente** le long d'une transition, elle est **considérée** comme toujours **vraie**.

NB : La syntaxe d'une condition de garde vérifiant si un état nommé TOTO est actif est : [in TOTO].

Equation logique

La condition logique évaluée pour la garde peut-être le résultat d'une combinaison de l'état de plusieurs grandeurs logiques, on parle alors du résultat d'une équation logique.

Dans ce type d'équation, on utilise des opérateurs logiques selon les règles de l'algèbre de BOOLE. Ces 4 opérateurs logiques sont : **OUI**, **NON**, **OU**, **ET**. Ils permettent de réaliser les opérations de base :

Entrées : a et b résultat : S			
OUI	notée $S = a$	qui donne à S la valeur 1 si et seulement si	$a = 1$
NON (appelé aussi « complément »)	notée $S = \bar{a}$		$a = 0$
OU (appelé aussi « somme logique »)	notée $S = a + b$		$a = 1$ OU $b = 1$
ET (appelé aussi « produit logique »)	notée $S = a \cdot b$		$a = 1$ ET $b = 1$

Quelques règles et propriétés de l'algèbre de Boole :

Propriétés de la somme logique : OU	
$a + 1 = 1$	
$a + 0 = a$	
$a + a = a$	
$a + \overline{a} = 1$	

Propriétés du produit logique : ET	
$a \cdot 1 = a$	
$a \cdot 0 = 0$	
$a \cdot a = a$	
$a \cdot \overline{a} = 0$	

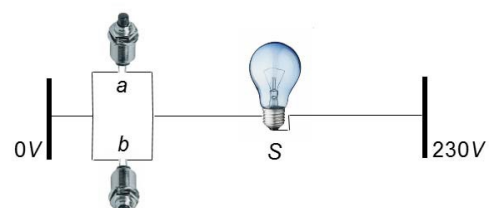
commutativité	associativité	distributivité
$a \cdot b = b \cdot a$	$a \cdot (b \cdot c) = (a \cdot b) \cdot c = a \cdot b \cdot c$	$a \cdot (b + c) = a \cdot b + a \cdot c$
$a + b = b + a$	$a + (b + c) = (a + b) + c = a + b + c$	$a + (b \cdot c) = (a + b) \cdot (a + c)$

Identité remarquables
$a + \bar{a} \cdot b = a + b$ $(a + b) \cdot (\bar{a} + c) = a \cdot c + \bar{a} \cdot b$

Théorèmes de De Morgan : $\overline{(a + b)} = \bar{a} \cdot \bar{b}$ et $\overline{(a \cdot b)} = \bar{a} + \bar{b}$

Toutes ces règles ne sont pas à apprendre par cœur mais plutôt à retrouver en faisant preuve de « logique ». On peut s'aider d'une représentation mentale de type schéma électrique dans laquelle:

- la variable dont on calcule l'état est une ampoule ;
- les variables qui interviennent dans l'équation sont des interrupteurs de type *bouton poussoir* ;
- l'opérateur logique OU correspond à un montage en parallèle ;
- l'opérateur logique ET correspond à un montage en série.



Représentation « électrique » de $S = a + b$

Effet

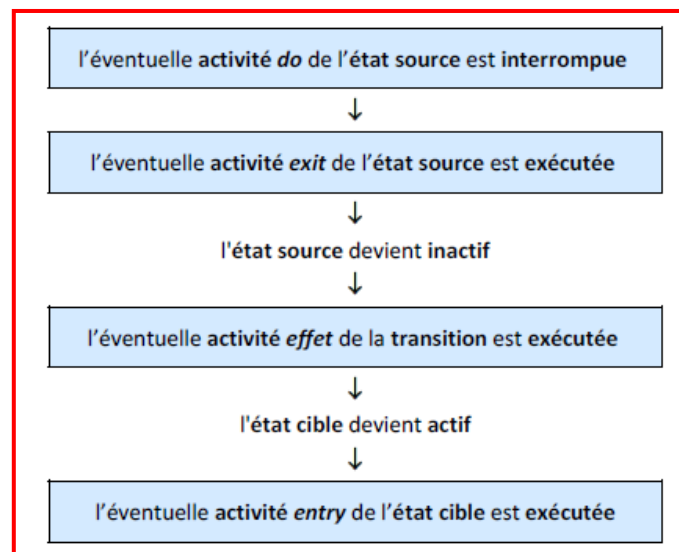
Un **effet** est une **activité** accomplie lorsque la **transition** est **franchie**. Les activités associées aux effets sont considérées instantanées. Une transition peut ne pas avoir d'effet associé.

1.4 Evolution du diagramme d'état par franchissement des transitions

Chronologie du franchissement d'une transition

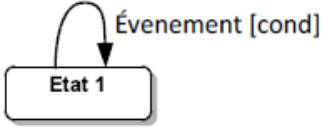
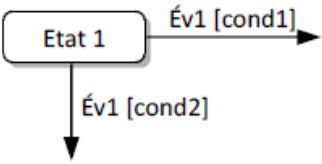
On peut formuler le principe de fonctionnement de base d'un diagramme d'état de la façon suivante : dès qu'un état est **actif**, la **transition** située **entre** cet **état source** et l'**état cible** est **franchissable**.

Lorsque l'événement apparaît, ET si la **condition** de garde est **vrai** :



Situations couramment rencontrées

	À l'occurrence (apparition) de l'événement, la transition est franchie, sans condition.
	À l'occurrence de l'événement, si la garde « cond » est vraie, la transition est franchie. Sinon, l'événement est « perdu » et il faut attendre une seconde occurrence pour éventuellement franchir la transition.
	Si la garde « cond » est vraie, la transition est : - immédiatement franchie s'il n'y a pas d'activité associée à l'état 1 ; - franchie dès la fin de l'éventuelle activité <i>do</i> de l'état 1. Cette fin d'activité constitue l'événement implicite de la transition. Il faut donc que cette activité do ait une fin (pas toujours respecté en pratique).
	La transition est dite automatique. Elle est : - immédiatement franchie s'il n'y a pas d'activité associée à l'état 1 ; - franchie dès la fin de l'éventuelle activité <i>do</i> de l'état 1. Cette fin d'activité constitue l'événement implicite de la transition. Il faut donc que cette activité do ait une fin (pas toujours respecté en pratique).

	Une transition réflexive entraîne une sortie de l'état puis un retour dans ce même état, avec appel des éventuelles activités <i>exit</i> et <i>entry</i>.
	Plusieurs transitions peuvent quitter un même état. Une seule d'entre elles doit être franchissable à un même instant. Les événements et/ou les conditions de garde doivent donc être exclusives.

1.5 Pseudos-états

Un **pseudo-état** est un état ne pouvant **pas** avoir d'**activité**.

Pseudo-état initial ●→

Unique et obligatoire, il est **activé** au **lancement** de la **machine à états** et marque le **début** de l'**exécution du diagramme d'état**. Il n'a aucune transition entrante.

Pseudo-état final →●

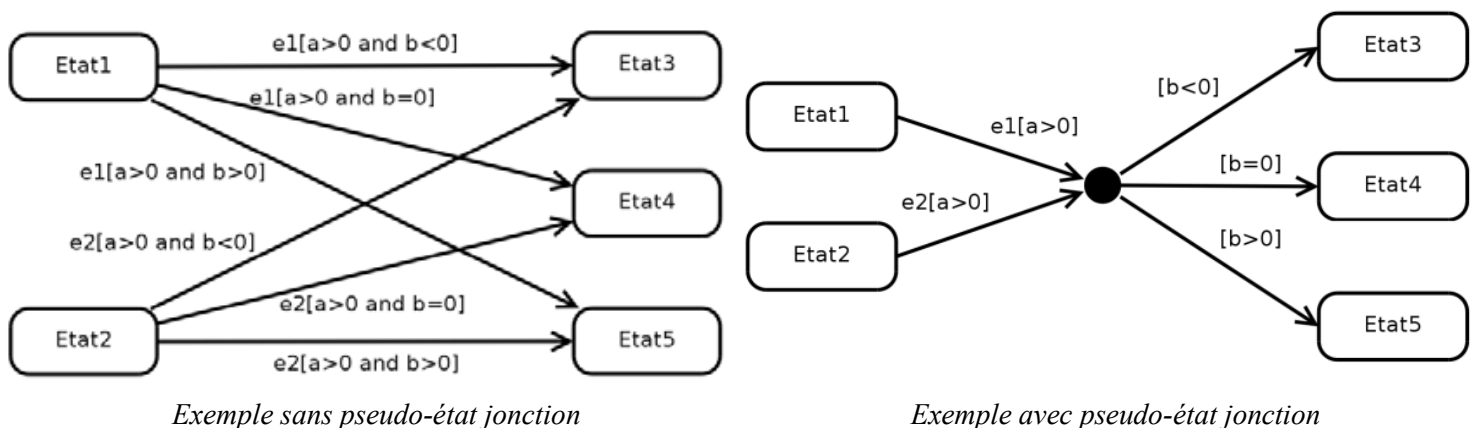
Optionnel, il signe la **fin de l'exécution du diagramme d'état**. Il n'a aucune transition sortante. Il peut y en avoir plusieurs car différents scénarios peuvent être possibles pour mettre fin à un diagramme d'état.

Pseudo-état jonction ●

Utilisé pour **regrouper** (« factoriser ») des **conditions de franchissement de transition**, en particulier des gardes communes à un événement. Il permet de partager des segments de transition et d'aboutir à une notation plus lisible des chemins alternatifs.

L'**évaluation** des **conditions de garde** en **aval** du pseudo-état est réalisée **avant** que ce pseudo-état ne soit atteint.

Exemples ci-dessous de 2 diagrammes d'état identiques : l'un sans pseudo-état jonction, et l'autre avec :



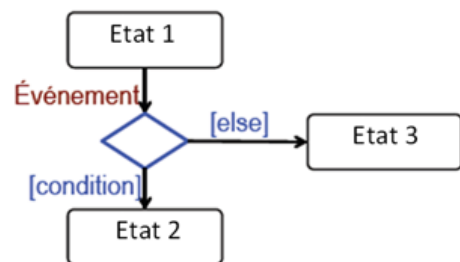
On passe de l'état 1 à l'état 3 si l'événement e1 apparait et que les conditions $a > 0$ et $b < 0$ sont vrais...

Pseudo-état décision

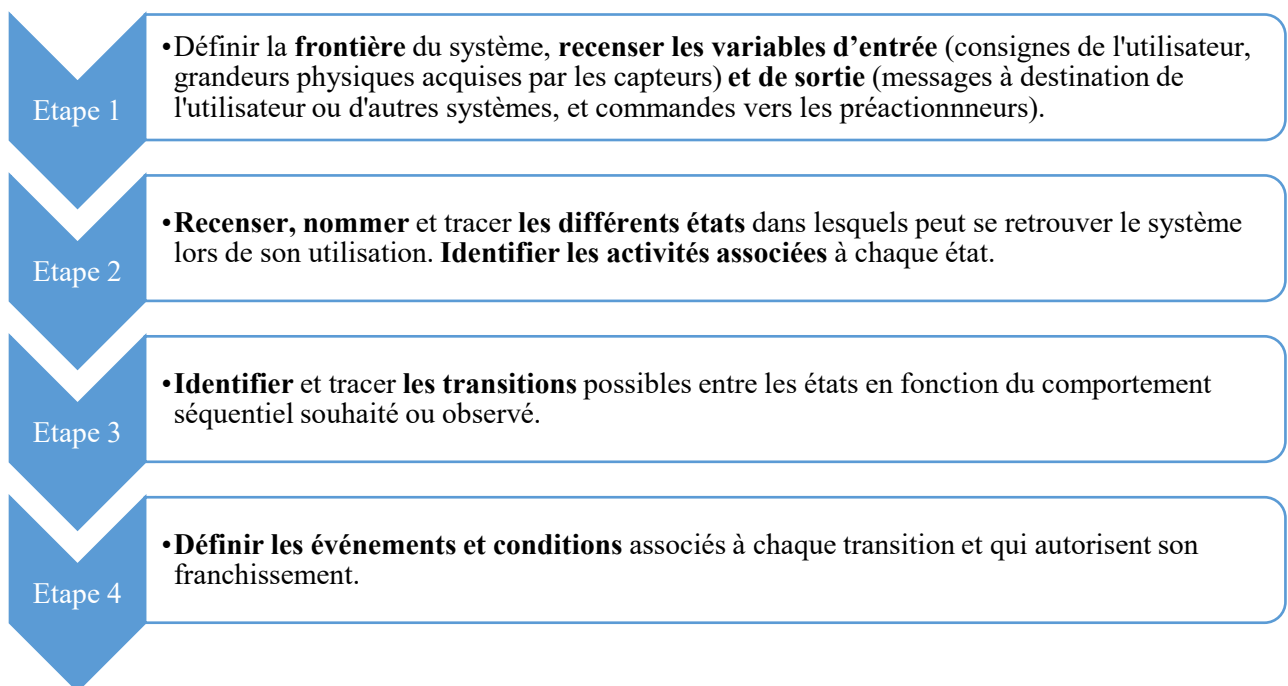
Utilisé pour une sélection ou une convergence de séquences exclusives. L'**évaluation** des **conditions de garde en aval** du pseudo-état est réalisée **au moment** où il est atteint. Les **conditions de garde** doivent être **exclusives**.

L'utilisation d'une clause *[else]* est recommandée après un pseudo-état décision, car elle garantit un modèle correct en englobant tout ce qui n'est pas décrit dans les autres expressions logiques et en assurant ainsi qu'au moins un segment en aval est franchissable.

Exemple ci-contre : dès que l'événement apparaît, le pseudo-état décision est atteint. Si la condition est vraie, c'est l'état 2 qui devient actif, sinon, c'est l'état 3.

**1.6 Démarche de modélisation par diagramme d'état**

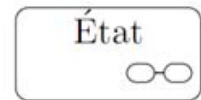
Une bonne modélisation du comportement séquentiel d'un système, en vue de programmer son unité de commande, nécessite de respecter la démarche suivante :



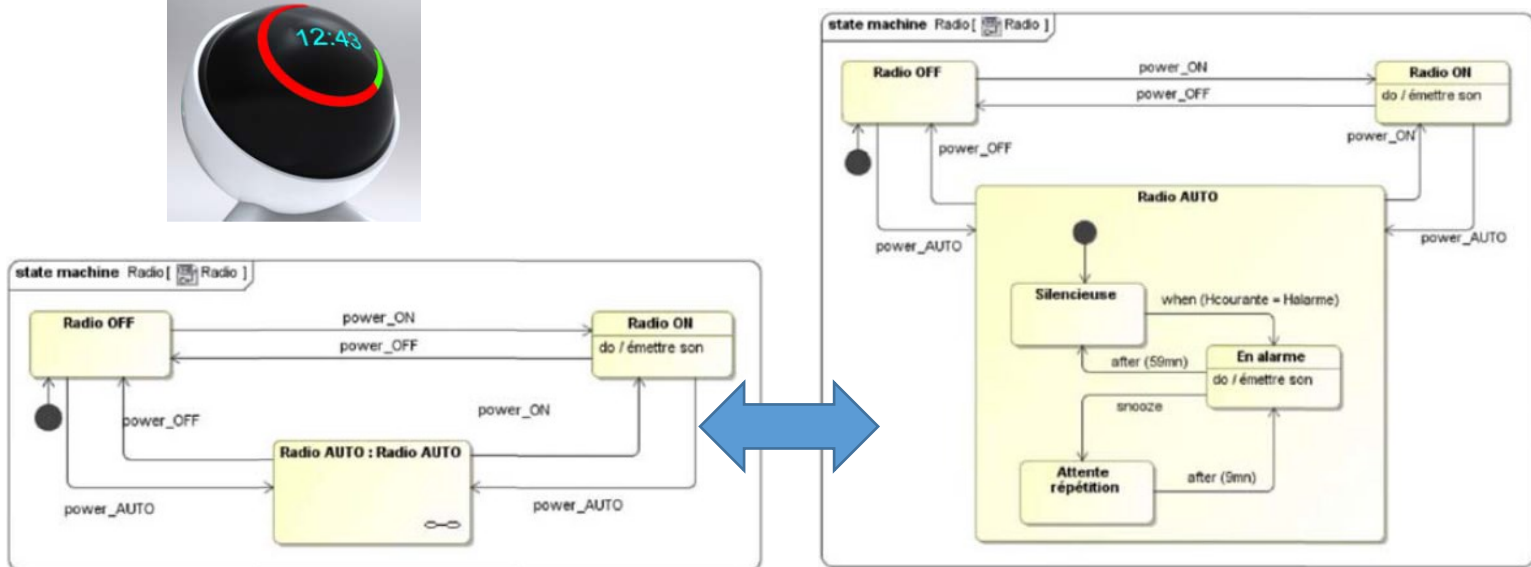
2. Modélisation des SED par diagramme d'état avec état composite

2.1 Notion de hiérarchisation

Un **état composite** décrit les **évolutions internes** d'un **état** à l'aide d'un **autre diagramme d'état**. Pour repérer un état composite, un signe symbolisant des lunettes est apposé sur l'état. Cette structure qui englobe plusieurs sous-états exclusifs considérés comme *hiérarchiquement inférieur* au diagramme principal, permet de rendre ce dernier plus lisible en entrant séparément dans le détail des évolutions internes du système.



Exemple : radio-réveil

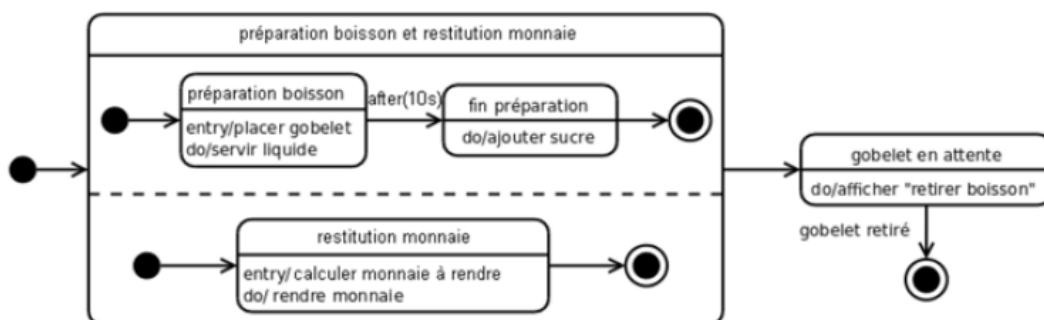


Une **transition** qui atteint la **bordure** d'un **état composite** est **équivalente** à une **transition** qui atteint l'état **initial** de sa **région interne**. Une **transition** qui **sort** de la **bordure** d'un **état composite** est **équivalente** à une **transition** qui sort de **tous les états** de sa **région interne**.

2.2 Etat composite orthogonal

Dans un **état composite orthogonal**, **plusieurs diagrammes d'états** peuvent **évoluer simultanément** dans des régions séparées par des pointillés. Les **différentes régions** de l'état orthogonal **fonctionnent en parallèle** sans **aucune influence** les uns sur les autres. Dans ce cas de figure plusieurs états sont actifs en même temps (un seul par région).

Exemple : distributeur de boissons

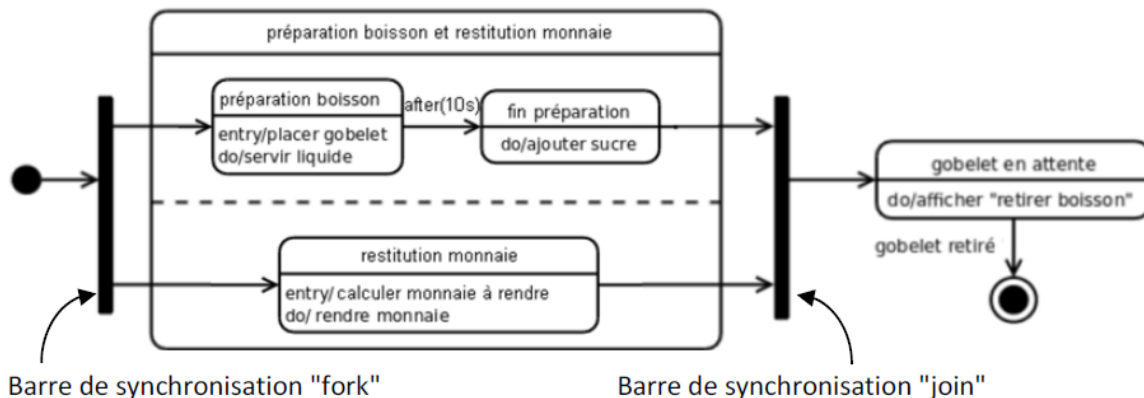


Une **transition** qui atteint la **bordure** d'un état **composite orthogonal** est **équivalente** à une **transition** qui atteint les **états initiaux** de **toutes** ses **régions**.

Toutes les **régions** d'un état **composite orthogonal** doivent **atteindre** leur **état final** pour que l'**état composite** soit considéré comme **terminé**. Ce n'est qu'à cette condition que la **transition** de **sortie** de l'**état composite** devient **franchissable**.

L'**activation** et la **sortie** d'un état **composite orthogonal** peuvent être également symbolisés par des **barres de synchronisation** *fork* et *join* qui fonctionnent par paire.

Exemple : distributeur de boissons : Le diagramme d'état ci-dessous décrit le même comportement que celui présenté précédemment.



Les **transitions**, nécessairement **automatiques**, qui **partent** d'une barre de synchronisation *fork* sont **franchies simultanément**.

La **transition** qui **part** d'une barre de synchronisation *join* n'est **franchissable** qu'après le **franchissement** de **toutes** les **transitions**, nécessairement **automatiques**, qui **convergent** vers cette barre.

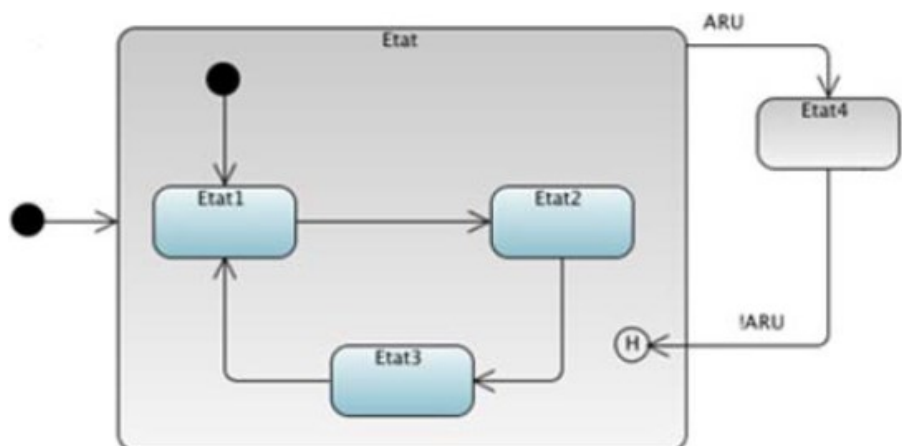
2.3 Historique d'un état composite

L'**état actif** au moment de la sortie d'un état composite peut être **mémorisé** par l'indication historique. Lors de la réactivation de l'état composite, celui-ci se **réactive à cet état**.



Exemple :

L'*historique* est utilisé ici pour permettre à un système de recommencer dans l'état où il était, après un appui sur l'arrêt d'urgence (ARU).



3. Modélisation d'un SED par diagramme de séquence

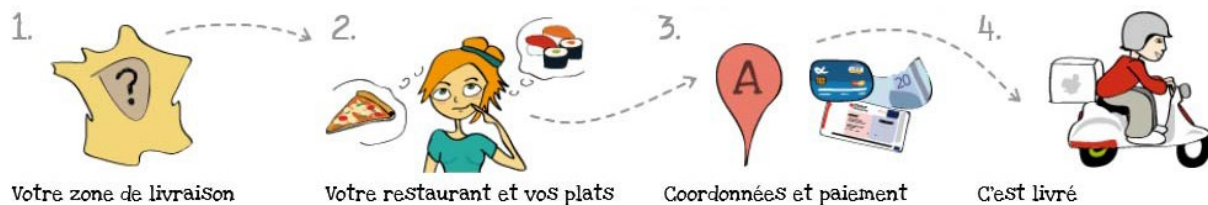
Un autre diagramme du langage SysML permet de décrire le comportement séquentiel d'un système. Il s'agit du diagramme de séquence que l'on définit pour les différents cas d'utilisations.

3.1 Rappel du diagramme des cas d'utilisation : *Use case* ou *uc*

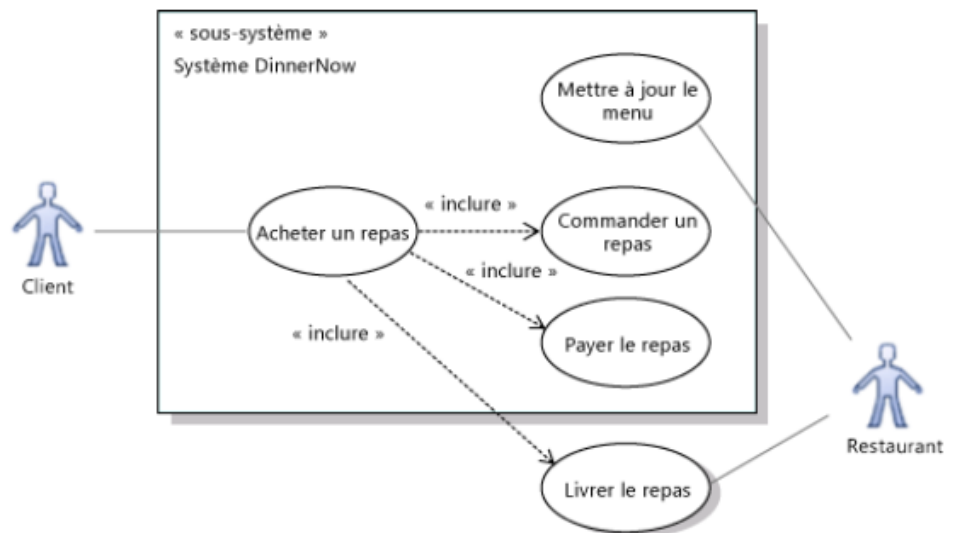
Le rôle d'un **diagramme des cas d'utilisation** est de spécifier les **acteurs** (placés à l'extérieur de la frontière d'étude) qui **utilisent** le **système** et la **manière** dont elles l'utilisent.

L'énoncé d'un cas d'utilisation est purement fonctionnel. Il est défini en termes de résultats attendus et est donc totalement indépendant des solutions technologiques choisies pour pouvoir le réaliser.

Exemple : système de vente de repas en ligne :



Le système est conçu pour permettre aux clients de sélectionner des éléments dans un menu, et aux restaurants proposant ce service de mettre le menu à jour.



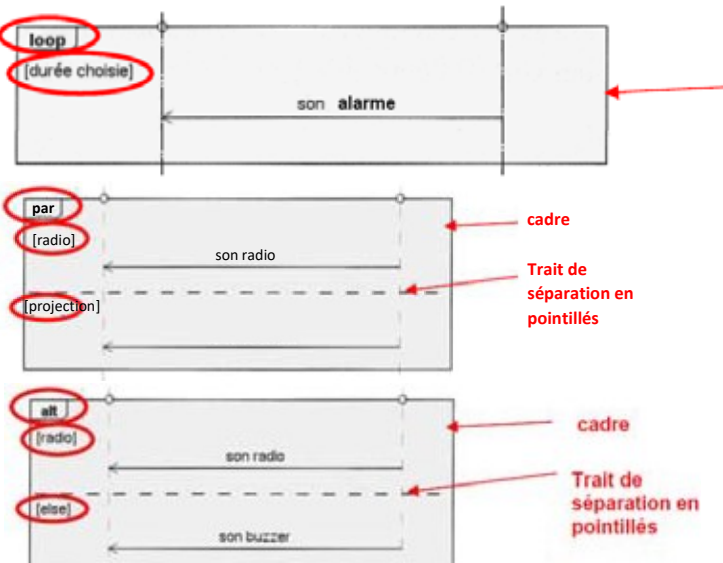
3.2 Diagramme de séquence : *Sequence Diagram* ou *sd*

Un diagramme de séquence est rattaché à un cas d'utilisation et décrit ce dernier en entier ou en partie, ce qui correspond à un scénario de fonctionnement possible, défini dans un cadre précis.

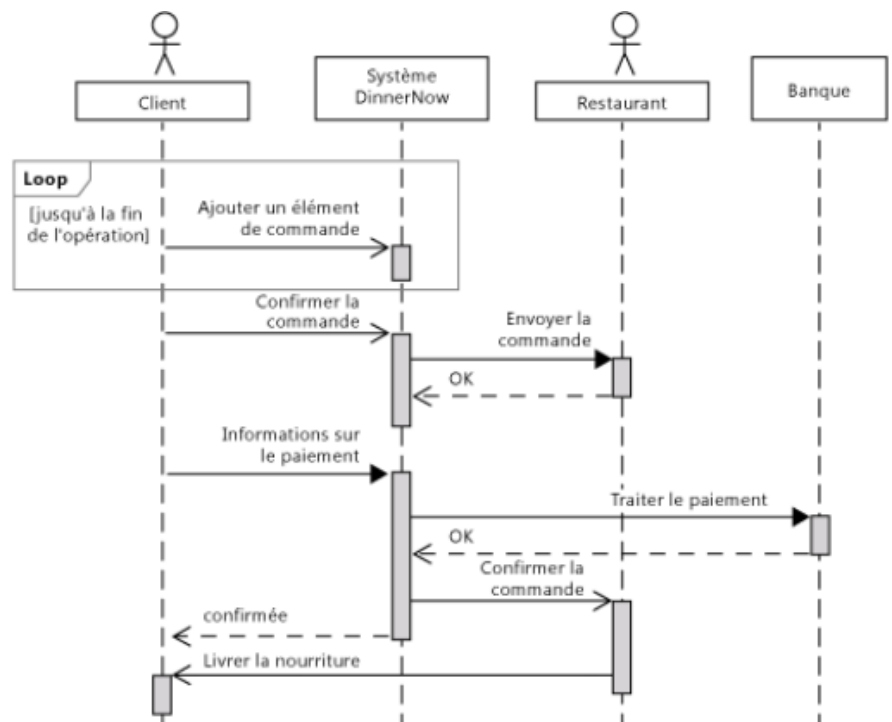
Il décrit, dans l'**ordre chronologique**, l'**enchaînement** des **interactions** (appelés messages) entre les **acteurs** du système ou entre des **composants** du système eux-mêmes.

Les principaux éléments sont :

Ligne de vie	Ligne verticale en pointillée. Une pour chaque élément dialoguant (acteur, système, sous-système ou composant).
Période d'activité	Bande verticale sur une ligne de vie. Optionnelle, elle facilite la lecture du diagramme.

Message	Flèche horizontale unidirectionnelle entre deux lignes de vie représentant un élément de communication. Elle déclenche une période d'activité (un comportement) chez le receveur du message, pour qui, l'arrivée d'un message est un événement. Ce message peut être : - <i>synchrone</i> : l'émetteur attend une réponse, son comportement est bloqué pendant l'attente ; - <i>asynchrone</i> : l'émetteur n'attend pas de réponse, son comportement continu ; - <i>une réponse</i>. Il est possible d'avoir un message réflexif correspondant à une interaction interne au composant.
Fragment	Cadre englobant une partie de la séquence loop <i>Boucle.</i> Le fragment est maintenu pendant une durée qui dépend de la condition de garde. par <i>Séquences en parallèle.</i> Les régions peuvent être exécutées simultanément. alt <i>Séquences alternatives conditionnées.</i> Seule la région dont la condition est vraie s'exécute. 

Exemple : système de vente de repas en ligne



Sources

J. Le Goff, S. Génouel, « Cours de Sciences Industrielles de l'Ingénieur CPGE 1^{ère} année », Pôle Chateaubriand Joliot-Curie, 2013