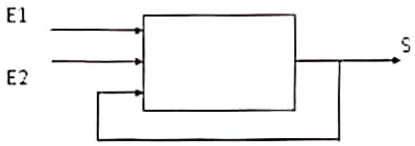
Cycle 9 – Modéliser, prévoir et vérifier des performances des systèmes combinatoires et séquentiels

1. Présentation

1.1 Représentation d'un système à logique séquentielle à « n » entrées et « p » sorties

Définition : système séquentiel

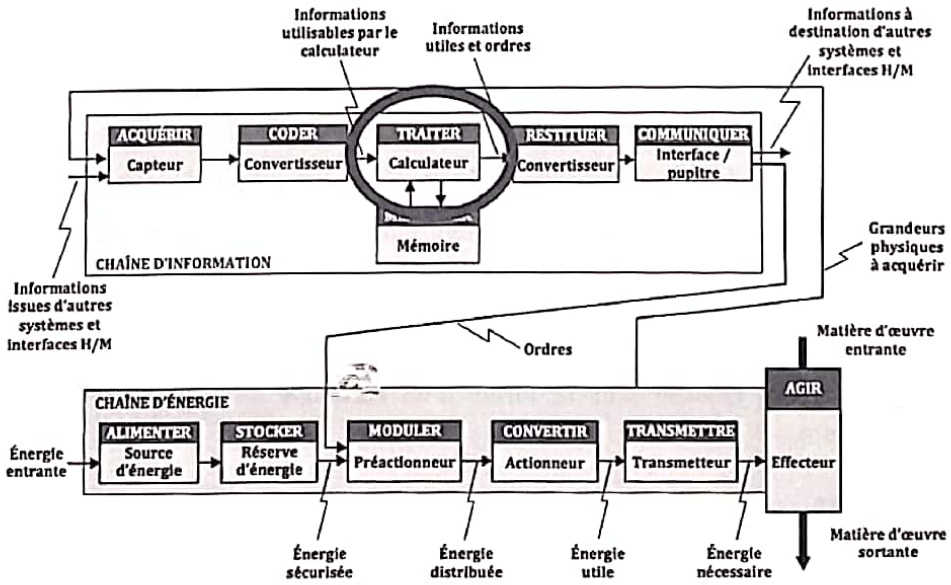
Un système est dit séquentiel, lorsque la ou les sorties dépendent de la combinaison des entrées et de l'état précédent des sorties : $S = f(E1, E2, S, t)$



Remarques :

- 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.

1.2 Situation dans la chaine d'information



1.3 Utilisation de la logique séquentielle

Le comportement des systèmes automatisés est très souvent décrit par la logique séquentielle. La synthèse d'un graphe d'états permet de réaliser la partie commande d'un système. Un autre outil au programme de CPGE est le diagramme de séquence. Ces deux diagrammes seront étudiés en utilisant les normes du langage SysML.

Exemple :

Pour une machine à café, les états peuvent être :

1. Attente de pièce ;
2. Descendre le gobelet ;
3. Verser la poudre à café ;
4. Verser l'eau chaude ;
5. Indiquer que le café est prêt.

2. Description du fonctionnement séquentiel par graphe d'états

Dans une machine à états, la loi d'évolution des états n'est évidemment pas aléatoire. Cette loi est soigneusement choisie par le créateur de la machine afin que celle-ci remplisse la fonction précise.

Le graphe d'états, comme son nom l'indique, représente graphiquement l'évolution des états d'une machine d'états.

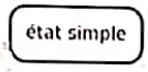
Remarque :
Le graphe d'états, comme l'algorigramme, est un outil graphique permettant de modéliser le comportement séquentiel, en termes de déroulement d'actions temporelles.
Mais il peut aussi servir à programmer les composants réalisant la fonction "Traiter" de la chaîne d'information (microcontrôleur, microprocesseur, automate programmable, ...). Les variables d'entrée de la fonction "Traiter" sont alors les informations fournies par la fonction "Acquérir" (capteurs, ...) et les variables de sortie sont les ordres pour la fonction "Distribuer" de la chaîne d'énergie, éventuellement via la fonction "Communiquer".

2.1 Les états

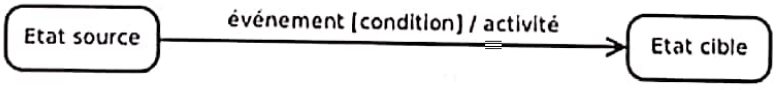
Définition :

Il existe différentes sortes d'états :

- l'état initial : pseudo-état qui indique un point d'entrée dans un graphe ;
- l'état final : non obligatoire, il indique que le système décrit n'a plus d'état actif ;
- un état : un état représente une période de la vie du système. Pendant cette période, le système accomplit une ou plusieurs actions, ou attend un évènement. Il peut être actif ou non ; plusieurs états peuvent être actifs en même temps dans le même système. Un état est dessiné sous la forme d'un rectangle aux coins arrondis, contenant son nom.



2.2 Les transitions



Définition :

Une transition représente le passage instantané d'un état vers un autre. Une transition ne peut donc pas avoir de durée. On appelle état source l'état de départ d'une transition et état destination l'état d'arrivée.

Important : une transition n'est évaluée que si l'état source est actif.

2.2.1 Évènements

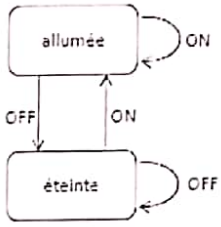
Une transition *peut* être déclenchée par un **évènement**. Dans ce cas, c'est l'arrivée d'un évènement qui conditionne le franchissement de la transition.

Il existe 2 sortes d'évènements.

- Evènement signal : un signal est émis à destination d'un objet ; cette émission est asynchrone, c'est-à-dire que le destinataire ne l'attend pas, et qu'elle peut survenir n'importe quand. Par exemple : l'appui sur un bouton-poussoir ;
- Evènement temporisé : un évènement de ce type fait intervenir le temps. Il nécessite l'utilisation des mots réservés **when(date)** pour spécifier un temps absolu, ou **after(durée)** pour spécifier une durée à partir de l'instant d'activation de l'état précédent.

Exemple simple :

Considérons une lampe munie de deux boutons poussoirs ON et OFF. Les états de la lampe sont : « allumée » et « éteinte ». Les évènements possibles sont : « ON » et « OFF ». Ces évènements déclenchent une transition entre états.



Cependant, il n'est pas indispensable d'avoir un événement déclencheur. **En absence d'évènement**, la transition est franchie lorsque l'activité associée à l'état source est terminée.

Par exemple, dans un état d'initialisation, le système sait à quel moment l'initialisation est terminée et il n'est donc pas nécessaire de spécifier un événement. Ce type de transition est appelée transition « automatique ». Attention, dans le cas où l'activité n'a pas de fin (allumage d'un voyant), la transition ne sera jamais franchie !

2.2.2 Condition de garde

En plus de spécifier un évènement précis, il est aussi possible de conditionner une transition, à l'aide d'une **"condition de garde"** : il s'agit d'une expression booléenne encadrée de crochets, évaluée lorsque l'état précédant la transition est vrai et que l'évènement déclencheur se produit. Si la condition de garde est vraie, la transition est alors franchie, sinon elle ne l'est pas et l'événement est perdu.

Attention à la différence entre évènement et condition de garde :

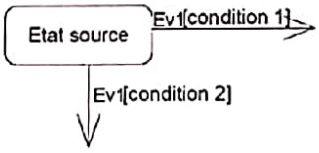
- un **événement est parfaitement daté dans le temps**, il correspond par exemple à un passage d'une variable de 0 à 1 à un instant précis (front montant) ;
- **une condition de garde n'est pas datée**, elle doit être vraie à l'instant où l'événement survient pour que la transition soit franchie.

Exemples :

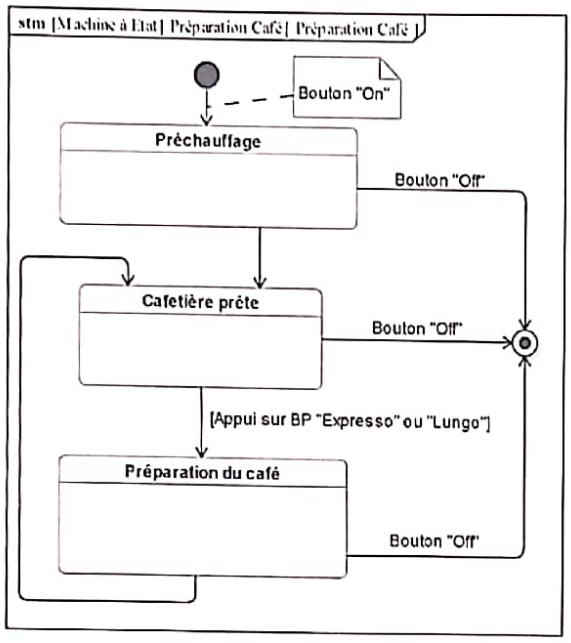
- *exemple d'évènement* : appui sur un bouton-poussoir, capteur fin de course atteint, etc.
- *exemple de condition de garde* : vitesse du véhicule non nulle, température > 20°C, etc.

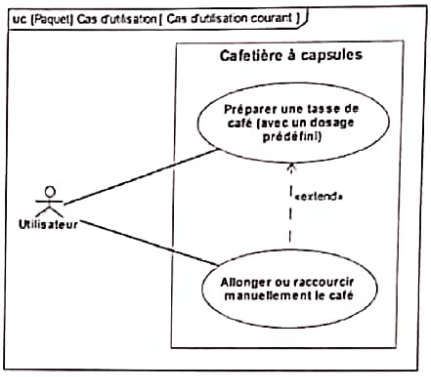
2.3 Transition conditionnelle

Plusieurs transitions peuvent quitter un même état. Une seule d'entre elles doit être déclenchée ; les événements et / ou les conditions de garde doivent donc être exclusives.



Exemple : diagramme d'états de la machine à café





2.4 Pseudo-états de « choice » et de « junction »

D'autres représentations sont possibles, en utilisant des pseudo-états.

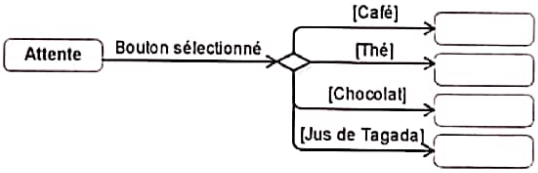
Définition :

Les pseudo-états sont des éléments graphiques permettant de préciser certaines règles de comportement d'une machine d'états. Il ne s'agit pas d'états dans la mesure où ces symboles ne correspondent pas à un état possible du composant étudié.

1. **«choice»**

Ce pseudo-état :

- possède au moins une entrée et deux sorties ;
- n'est atteint que lorsque l'évènement en amont apparaît.

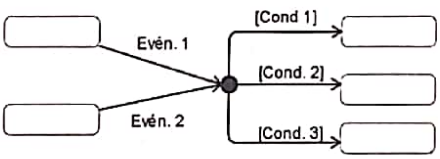


Les conditions de garde sont toutes évaluées simultanément. L'évolution du système se poursuit si une condition située en aval est vraie. Les conditions de gardes doivent être exclusives. Le mot clé « else » peut être utilisé pour englober tout ce qui n'est pas décrit dans les autres expressions booléennes.

2. **«junction»**

Ce pseudo-état est généralement utilisé pour relier plusieurs états en amont à plusieurs états en aval.

Le comportement est semblable au pseudo-état « choice », à la différence que pour qu'un chemin soit emprunté, l'une des conditions situées en aval de la jonction doit être vraie pendant que l'un des évènements situés en amont se produit.



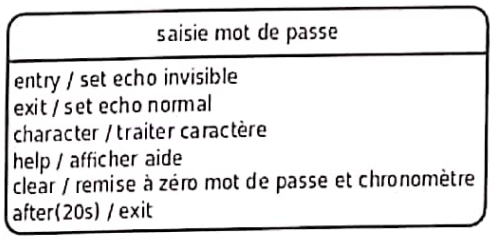
L'évaluation des conditions de garde situées en aval du point de jonction est réalisée avant que le pseudo-état (point de jonction) ne soit atteint.

2.5 Les actions

Une action peut être lancée au moment du passage d'une transition (Voir /activité sur la figure présentant les transitions).

Principe : le lancement des actions à l'intérieur de l'état actif est organisé selon des mots réservés :

- **entry/** est suivi des actions exécutées lorsque l'état devient actif ;
- **do/** est suivi d'une ou plusieurs actions exécutées dans l'ordre de leur écriture, à partir de l'instant où l'activité entry/ est terminée ;
- **exit/** est suivi des actions qui se déroulent lorsque l'état se désactive.

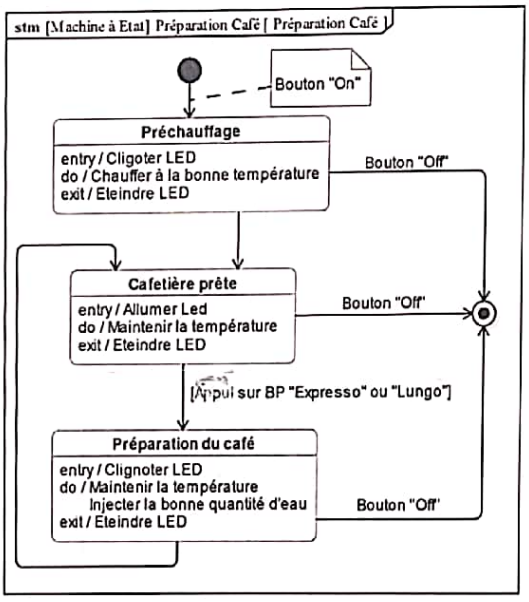


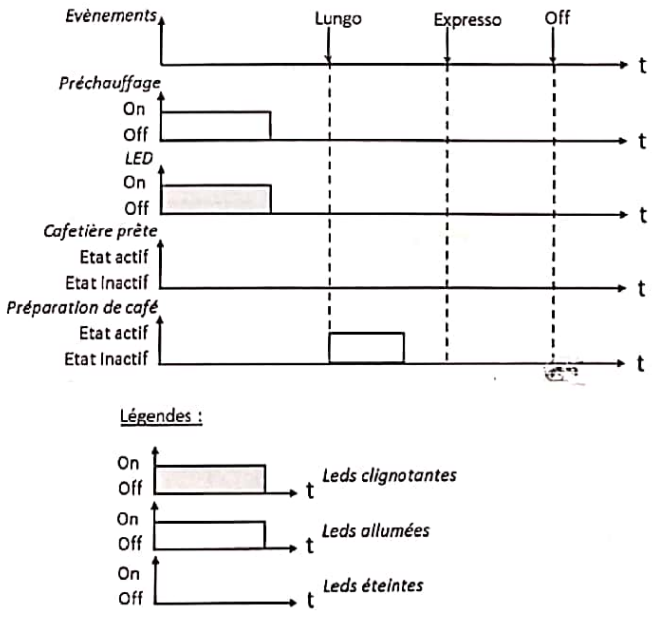
Remarques :

- on peut aussi ne pas utiliser de mot réservé, auquel cas cela correspond à un do/ ;
- un état peut ne pas contenir d'action. Il sert alors à attendre le déclenchement de la transition suivante ;
- pendant que l'état est actif, un évènement peut lancer une action avec la syntaxe : **évènement/** suivi de l'action. Cette action est lancée chaque fois que l'évènement survient, tant que l'état est actif.

Exercice : diagramme d'états de la machine à café (cf. page 2)

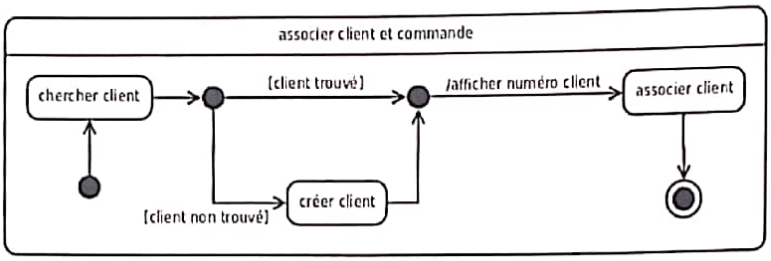
A partir du diagramme d'état ci-dessous, compléter le chronogramme décrivant le fonctionnement du système.



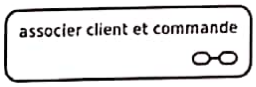


2.6 État composite

Définition : un état composite est un état décomposé en régions contenant chacune un ou plusieurs sous-états.

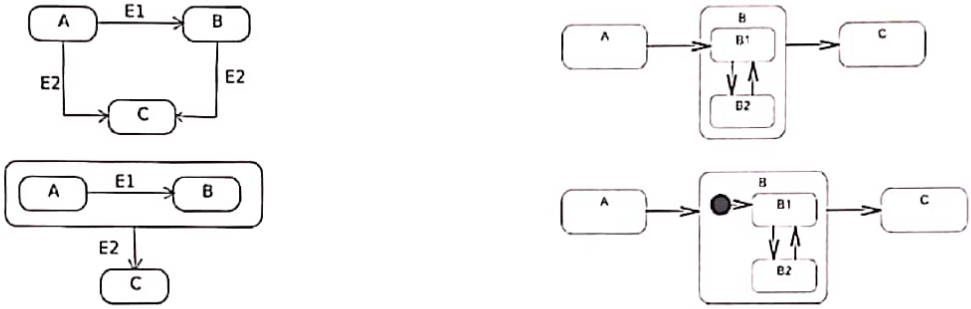


Remarque : une notation abrégée permet d'indiquer qu'un état est composite et que sa définition est donnée sur un autre diagramme.

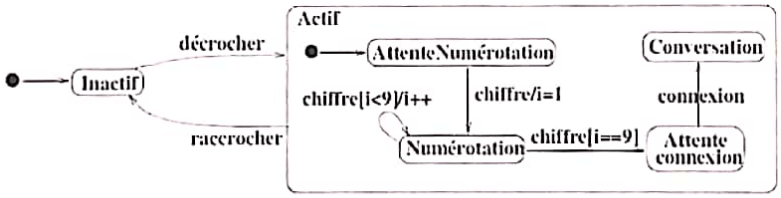


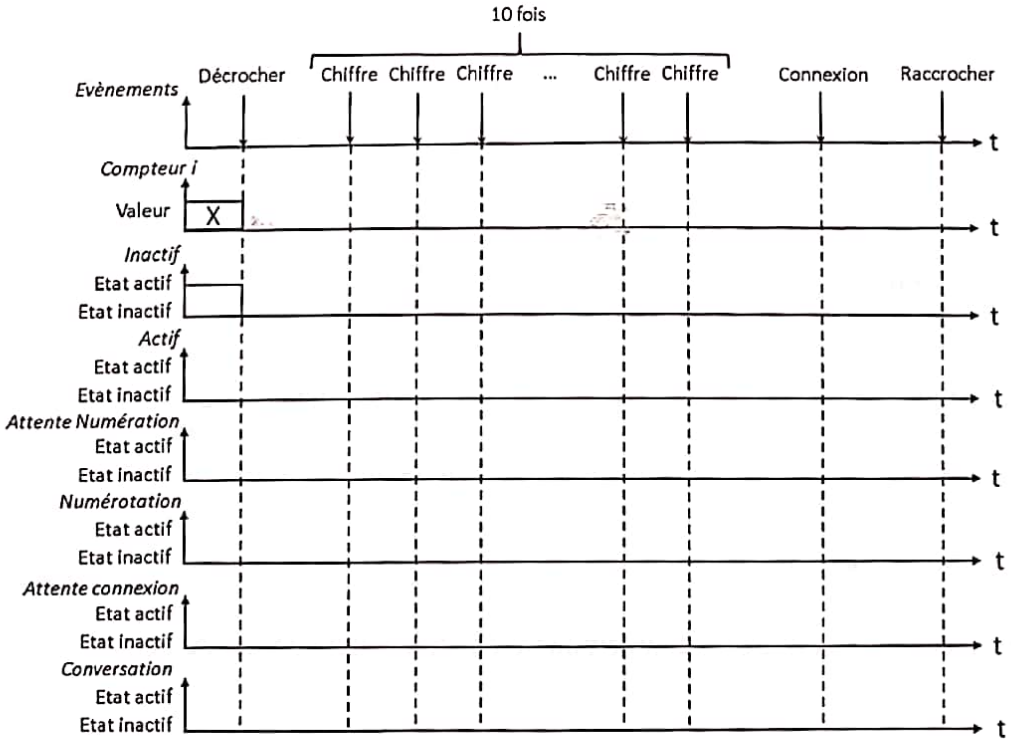
Exemple 1 :

- à gauche, deux diagrammes équivalents ;
- à droite, deux diagrammes équivalents avec la transition d'entrée reportée dans l'état composite (diagramme du haut) ou avec un état initial emboité (diagramme du bas). L'état B est appelé état englobant.

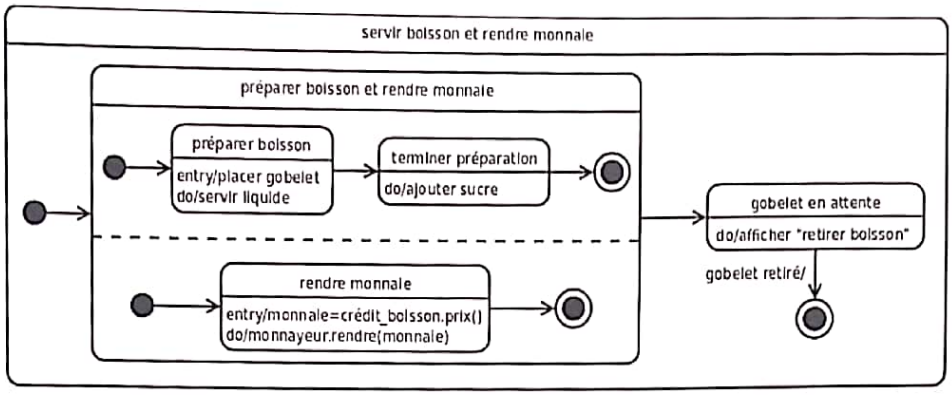


Exemple 2 : compléter le chronogramme ci-dessous à partir du diagramme d'état proposé.





2.7 États concurrents



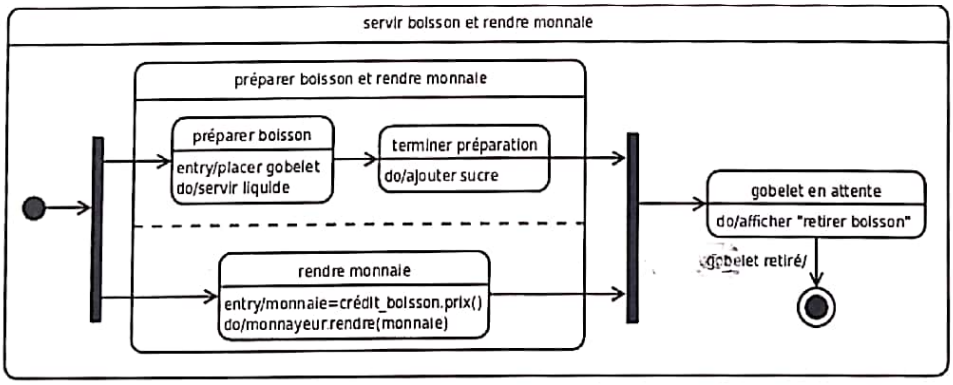
Les diagrammes d'états-transitions permettent de décrire efficacement les mécanismes concurrents grâce à l'utilisation d'**états orthogonaux**. Un état orthogonal est un état composite comportant plus d'une région, chaque région représentant un flot d'exécution. Graphiquement, dans un état orthogonal, les différentes régions sont séparées par un trait horizontal en pointillé allant du bord gauche au bord droit de l'état composite.

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 concurrentes.

Toutes les régions concurrentes d'un état composite orthogonal doivent atteindre leur état final pour que l'état composite soit considéré comme terminé.

La figure illustre l'utilisation d'un état composite orthogonal pour modéliser le fait que la préparation de la boisson d'un distributeur de boissons se fait en parallèle au rendu de la monnaie.

Il est également possible de représenter ce type de comportement au moyen de transitions concurrentes. De telles transitions sont qualifiées de **complexes**. Les transitions complexes sont représentées par une barre épaisse et peuvent, éventuellement, être nommées :



3. Description du fonctionnement séquentiel par diagramme de séquence

Le diagramme de séquence est un diagramme comportemental, il détaille, pour un cas d'utilisation spécifique, les interactions entre plusieurs entités (acteur, système, sous-systèmes) au moyen de messages. Il ne décrit que l'enchainement séquentiel de ces interactions :

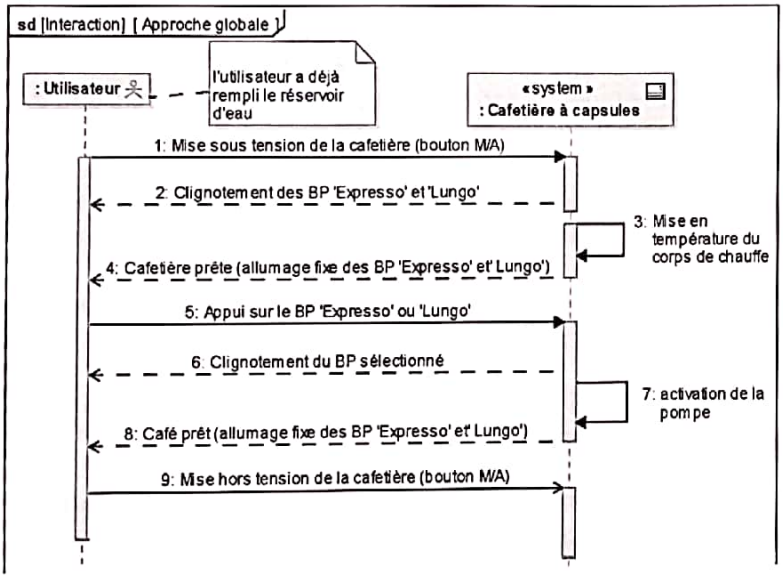
- dans l'ordre chronologique (le temps s'écoule du haut vers le bas) ;
- par des messages échangés entre les différentes entités (acteur, système, sous-systèmes), qui peuvent déclencher des activités chez le receveur du message ;

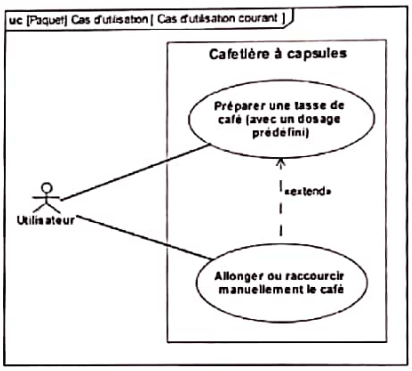
- sans détailler les comportements individuels de composants du système (d'autres diagrammes doivent être utilisés pour détailler les comportements des composants).

Remarques :

- il n'y a pas d'échelle de temps sur un diagramme de séquence, une activité de 10 s peut être représentée de la même façon qu'une activité de 30 min ;
- plusieurs diagrammes de séquence peuvent être définis pour un même système, ils sont à rattacher à un cas d'utilisation (qu'il décrit en entier ou en partie) ;
- il n'est pas obligatoire de détailler toutes les interactions, cela dépend du niveau de détail et de l'objectif recherché.

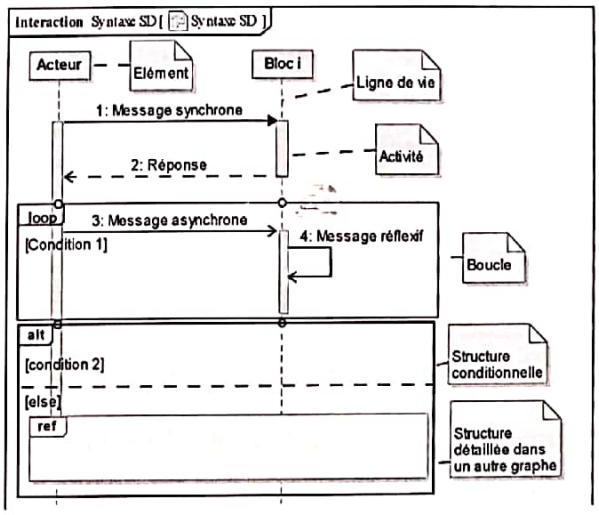
Exemple 1 : Diagramme de séquence de la machine à café à capsule





Définition : les principaux éléments graphiques utilisés dans les diagrammes de séquences

- **les lignes de vie** : lignes verticales pointillées représentant l'existence d'un élément (bloc ou acteur) du diagramme de cas d'utilisation. Sur une ligne de vie, le temps s'écoule du haut vers le bas (sans échelle spécifique) ;
- **les activités** : rectangles verticaux (optionnels) placés sur les lignes de vie des éléments qui décrivent une période d'activité de l'élément concerné. S'il n'y a pas d'activité sur une ligne de vie d'un élément, celui-ci peut être en attente d'un message ;
- **les messages** : éléments unidirectionnels de communication (lien horizontal orienté) entre deux lignes de vie.



La réception d'un message déclenche une activité chez le receveur (la cible). Plusieurs types de messages existent :

- message synchrone : l'expéditeur du message synchrone attend une réponse. Pendant ce temps d'attente de la réception, l'émetteur du message ne peut être occupé à une autre tâche : il attend la réponse. Un message synchrone est dessiné par un trait continu avec une flèche pleine ;
- la réponse (à un message synchrone) est dessinée en trait pointillé avec une flèche vide ;
- message asynchrone : l'expéditeur n'attend pas de réponse de la part du receveur. Ce message est représenté par un trait continu avec une flèche vide ;
- message réflexif (interactions internes).

Finalement, **les fragments** sont des rectangles horizontaux qui englobent une partie de la séquence (boucle (loop), structure parallèle (par), structure conditionnelle (alt), référence à un autre diagramme de séquences (réf)....).

Exemple 2 : fonctionnement d'une climatisation réversible (qui produit du froid ou du chaud).