UML

Diagramme de communication

Introduction

L'illustration d'une interaction :

le diagramme de séquence;

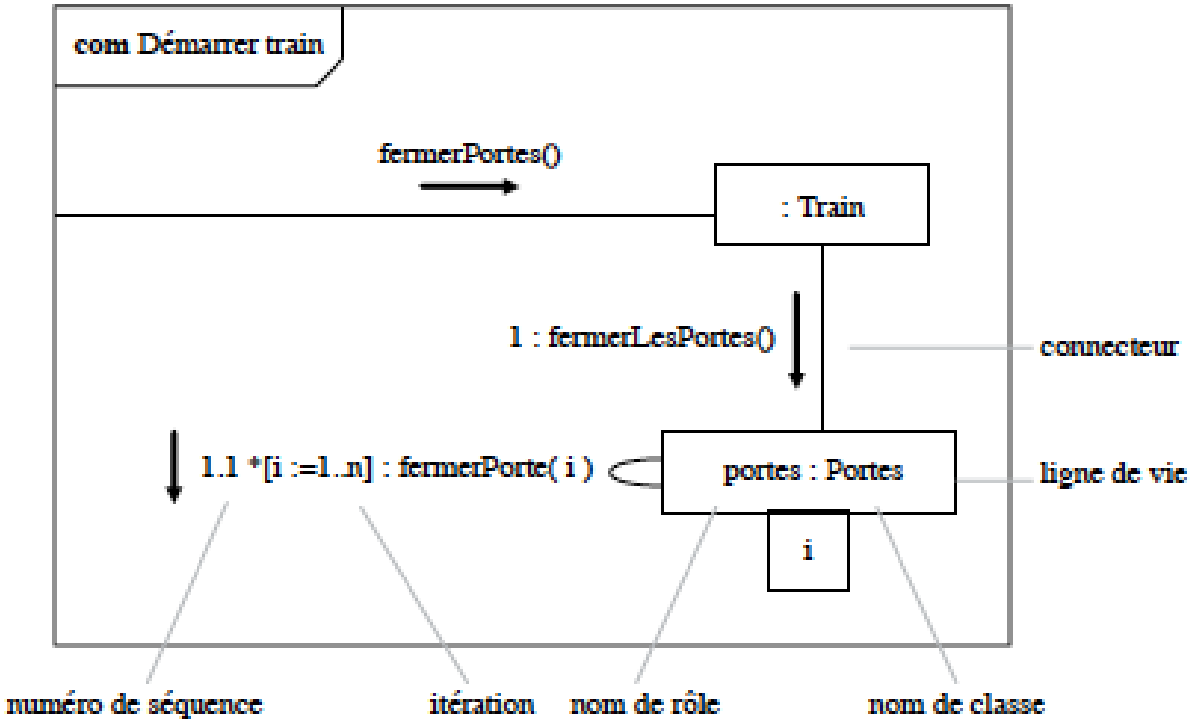
le diagramme de communication;

le diagramme de timing;

- Les diagrammes de communication et de séquence représentent des interactions entre des lignes de vie.
- Un diagramme de séquence montre des interactions sous un angle temporel, et plus particulièrement le séquencement temporel de messages échangés entre des lignes de vie, tandis qu'un diagramme de communication montre une représentation spatiale des lignes de vie.

Notations

Les lignes de vie sont représentées par des rectangles contenant une étiquette dont la syntaxe
est :
<nomDuRôle> : <NomDuType>
Au moins un des deux noms doit être mentionné dans l'étiquette (si le nom du rôle est omis, le caractère : est obligatoire).



Description

- Le rôle permet de définir le contexte d'utilisation de l'interaction. Si la ligne de vie est un objet, celui-ci peut avoir au cours de sa vie plusieurs rôles : une instance d'une classe Personne peut jouer le rôle d'un étudiant dans un contexte donné, et devenir le conducteur d'une voiture à un autre moment.

Les connecteurs peuvent montrer la multiplicité des lignes de vie qui participent à une interaction (exemple : les portes d'un train …)

UML

Diagramme d’états-transitions

Introduction

- Les diagrammes d'états-transitions d'UML décrivent le comportement interne d'un objet à l'aide d'un automate à états finis.

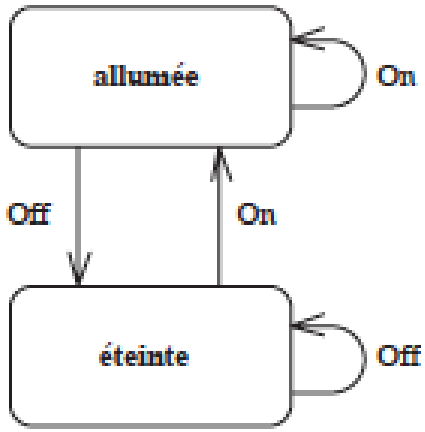
Ils présentent les séquences possibles d'états et d'actions qu'une instance de classe peut traiter au cours de son cycle de vie en réaction à des événements discrets (de type signaux, invocations de méthode).

- 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 : cet historique est caractérisé par un état.

Exemple

- …une lampe munie de deux boutons-poussoirs : une pression sur « On » allume la lampe et une pression sur « Off » l'éteint.

Une pression sur « On » ne produit pas d'effet si la lampe est déjà allumée ; la réaction d'une instance de Lampe à cet événement dépend de son état interne.



Exemple2 : le comportement simplifié d'une fenêtre d'application, qui répond aux stimuli de trois boutons placés dans l'angle.

Notions

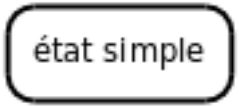
- Le diagramme d'états-transitions est le seul diagramme à offrir une vision complète et non ambiguë de l'ensemble des comportements de l'élément auquel il est attaché. En effet, un diagramme d'interaction n'offre qu'une vue partielle correspondant à un scénario sans spécifier comment les différents scénarii interagissent entre eux.

- Il est souhaitable de construire un diagramme d'états-transitions pour chaque classeur (qui, le plus souvent, est une classe) possédant un comportement dynamique important.

- Les transitions d'un diagramme d'états-transitions sont donc déclenchées par des événements, appelés « déclencheurs » ou triggers.

Etats

Etats simples: par un rectangles aux coins arrondis :



États composites : contient des sous-états;

- Le nom de l'état peut être spécifié dans le rectangles et doit être unique dans le diagrammes d'états-transitions, ou dans l'état enveloppant. On peut l'omettre, ce qui produit un état anonyme.

- 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. La configuration de l'état global de l'objet est le jeu des états (élémentaires) qui sont actifs à un instant donné.

Les événements

- Un appel de méthode sur l'objet courant génère un événement de type *call*.
- Le passage de faux à vrai de la valeur de vérité d'une condition booléenne génère implicitement un événement de type *change*.
- La réception d'un signal asynchrone, explicitement émis par un autre objet, génère un événement de type *signal*.
- L'écoulement d'une durée déterminée après un événement donné génère un événement de type *after*.
- La fin d'une activité de type *do/,* interne à un état génère implicitement un événement appelé *completion event*.

- Un événement de type call ou signal est déclaré ainsi :

  nom-événement '(' liste-paramètres ')'

  où chaque paramètre a la forme :

  nom-paramètre ':' type-paramètre

Les événements de type call sont donc des méthodes déclarées au niveau du diagramme de classes.

Les états

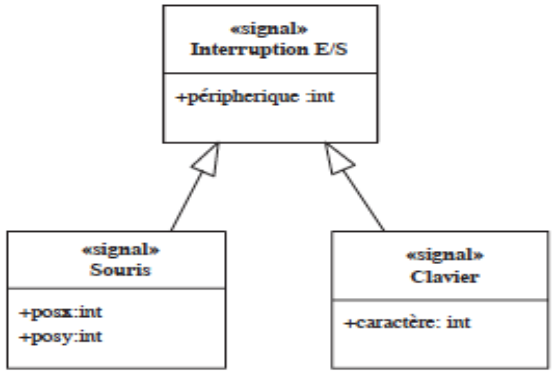
- Le modèle dynamique comprend plusieurs diagrammes d'états.
- Chaque diagramme d'états ne concerne qu'une seule classe.
- Chaque automate à états finis s'exécute concurremment et peut changer d'état de façon indépendante des autres.

Lorsqu'un objet est créé, il entre dans l'état initial.

L'état final est un pseudo-état qui indique que l'exécution de l'automate ou du sous-état est terminée.

Les événements - type signal (1)

- Un signal est un message émis de façon asynchrone, c'est-à-dire que l'appelant peut poursuivre son exécution sans attendre la bonne réception du signal.



Trois signaux sont déclarés comme des classes du paquetage et peuvent être liés entre eux par des relations d'héritage. Si le signal A dérive du signal B, il déclenche par transitivité toutes les activités liées à B en plus des siennes propres. Les attributs de classe sont interprétés comme les arguments de l'événement de type signal.

Les événements - type signal (2)

- Une transition d'un diagramme d'états-transitions est représentée par un arc plein, liant un état dit « source » à un état « cible ». Elle est dotée d'une étiquette contenant une expression de la forme :

nom-événement '(' liste-param-événement ')' '[' garde ']' '/' activité

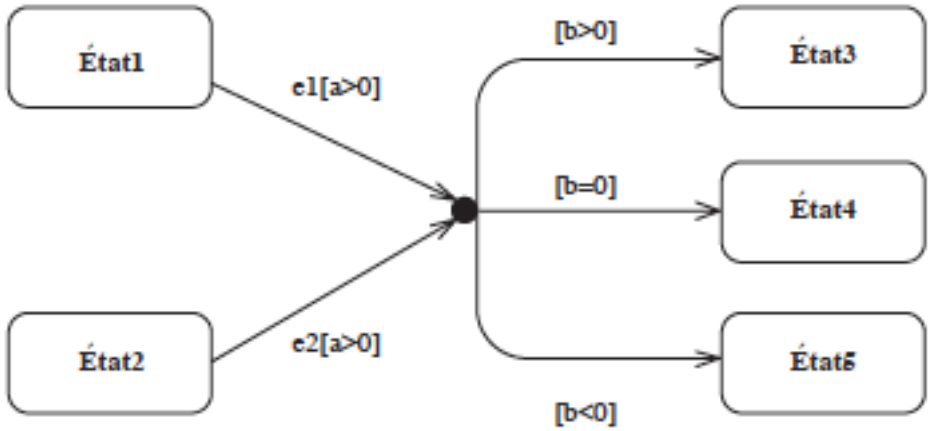
où les paramètres éventuels de l'événement sont séparés par des virgules, la garde (par défaut vraie) désigne une condition qui doit être remplie pour pouvoir déclencher la transition, et l'activité exprimée dans une syntaxe libre désigne des instructions à effectuer au moment du tir.

- Le déclencheur de la transition est un événement de type call, signal, change ou after, ou n'est pas spécifié pour les transitions automatiques

Point de décision

- représenter des alternatives pour le franchissement d'une transition : des pseudo-états particuliers :

  les points de jonction (représentés par un petit cercle plein) et les points de choix (représentés par un losange)



Possibilité de faire une équivalence avec des transitions gardées.

Point de choix dynamique

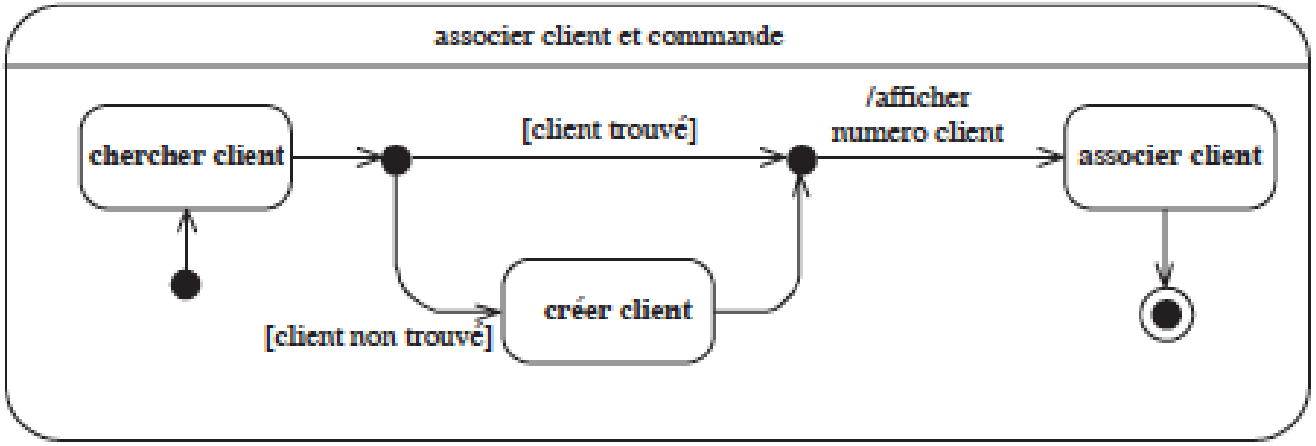
- représenté par un losange;
- Au contraire d'un point de jonction, les gardes après ce point de choix sont évaluées au moment où il est atteint.

- Exemple:
- Un formulaire en ligne est rempli par un utilisateur. Quand il valide son formulaire en appuyant sur le bouton go, une vérification de la cohérence des données fournies est réalisée par validerEntrée(). Si les informations paraissent correctes, on lui demande de confirmer, sinon on affiche les erreurs détectées et il doit remplir de nouveau le formulaire.

- La validation se fait sur le segment avant le point de choix. Ce fonctionnement ne peut être décrit par un simple point de jonction.

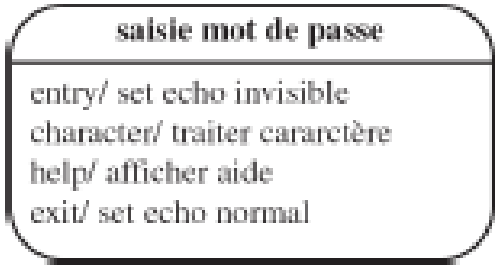
Représentation d'alternatives avec les points de jonction

Le point de jonction est bien adapté à la représentation de clauses conditionnelles de type if/endif.



Transitions et états composites

- Un objet reste dans un état durant une certaine durée et des transitions internes peuvent intervenir.
- Une transition interne ne modifie pas l'état courant, mais suit globalement les règles d'une transition simple entre deux états.
- Trois déclencheurs particuliers sont introduits permettant le tir de transitions internes : **entry/, do/,** et **exit/.**



Déclencheurs de transitions internes

Les transitions prédéfini :

- « **entry** » définit une activité à effectuer à chaque fois que l'on rentre dans l'état considéré.
- « **exit** » définit une activité à effectuer quand on quitte l'état.
- « **do** » définit une activité continue qui est réalisée tant que l'on se trouve dans l'état, ou jusqu'à ce que le calcul associé soit terminé.

Les autres transitions :

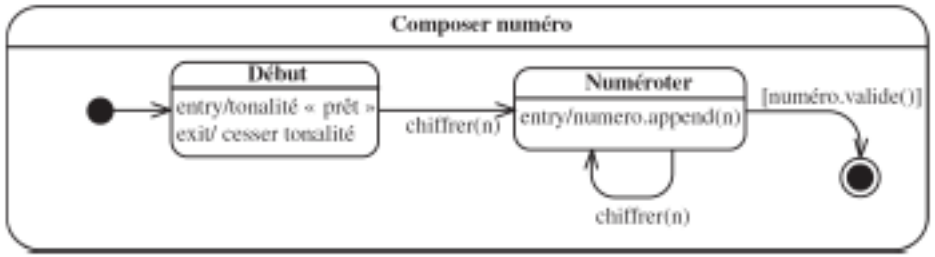
- Les transitions internes sont spécifiées dans le compartiment inférieur de l'état, sous le compartiment du nom.

- Notation : **nomEvenement (params) [garde] / activiteARealiser**

Etats composites (1)

- décomposé en deux ou plusieurs sous-états.

- Tout état ou sous-état peut ainsi être décomposé en sous-états imbriqués sans limite a priori de profondeur.
- Un état composite est représenté par les deux compartiments de nom et d'actions internes habituelles, et par un compartiment contenant le sous-diagramme.



Etats composites (2)

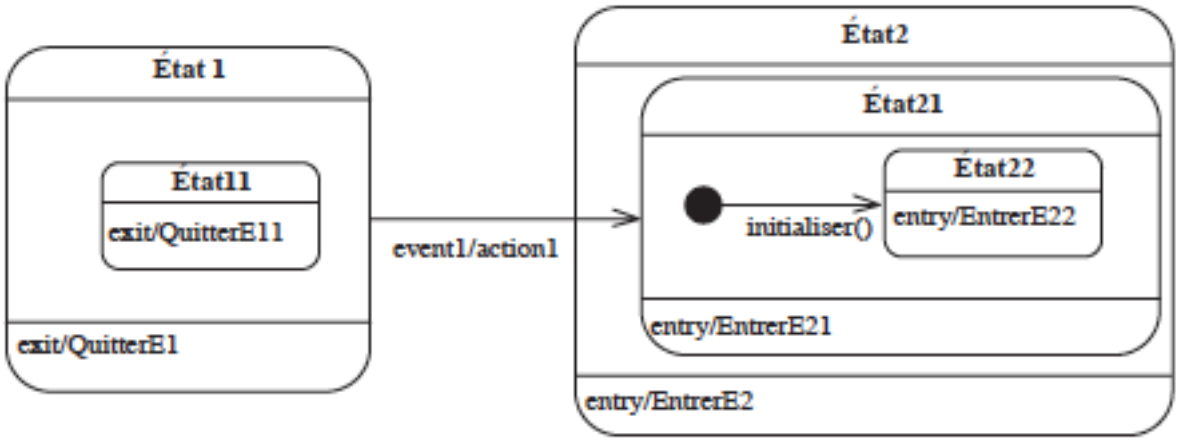
- Les transitions peuvent avoir pour cible la frontière d'un état composite. Elle sont alors équivalentes à une transition ayant pour cible l'état initial de l'état composite.

- Une transition ayant pour source la frontière d'un état composite est équivalente à une transition qui s'applique à tout sous-état de l'état composite source.

Etats composites (3)

- Ordre d'appel :

Depuis l'état État11, la réception de l'événement event1 provoque la séquence d'activités QuitterE11, QuitterE1, action1, EntrerE2, EntrerE21, initialiser, EntrerE22, et place le système dans l'état État22.



Etats composites et Historique (1)

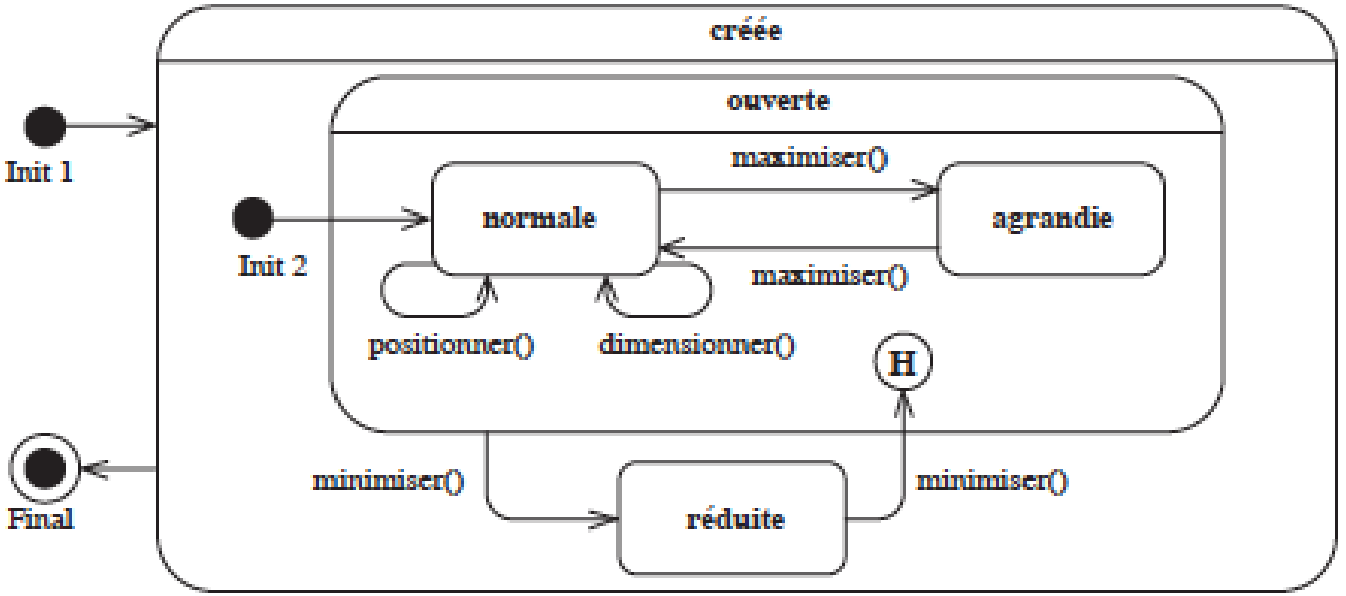
- Chaque région ou sous-diagramme d'états-transitions peut avoir un pseudo-état historique, noté par un **cercle contenant un H**.

Une transition ayant pour cible le pseudo-état historique est équivalente à une transition qui a pour cible le dernier état visité dans la région contenant le H.

H* désigne un historique profond : un historique valable pour tous les niveaux d'imbrication.

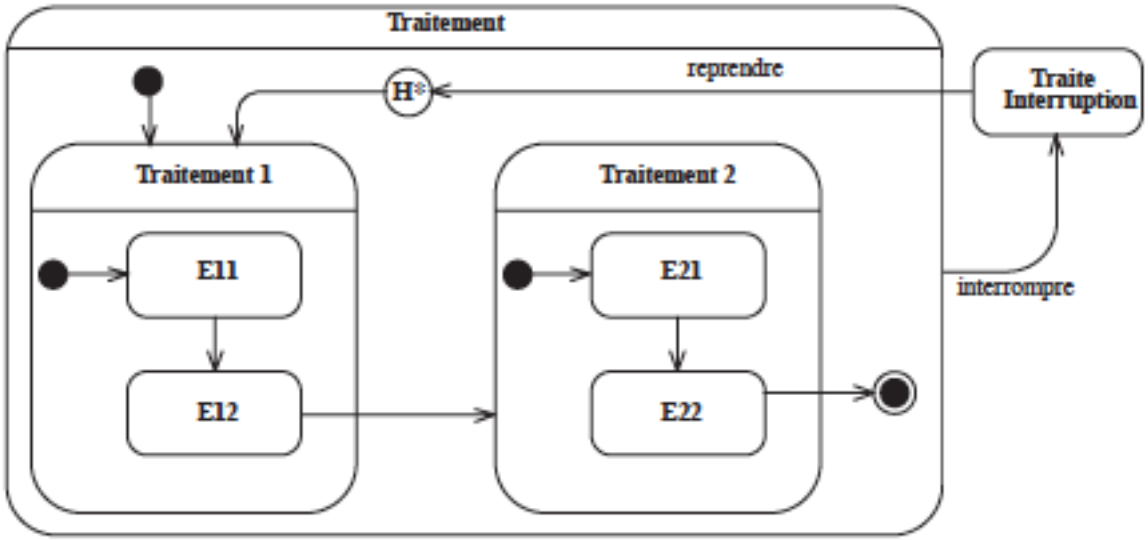
Exemple : Fenêtre d'application - le pseudo-état historique permet de retrouver l'état précédent (normale ou agrandie) quand on sort de l'état réduite.

Rappel : Fenêtre d'application



Etats composites et Historique (2)

- l'utilisation d'un historique de surface H, au lieu de H*, permettrait de retrouver l'état Traitement1 ou Traitement2 dans leur sous-état initial, mais pas les sous-états imbriqués E11, E12, E21, E22, qui étaient occupés avant l'interruption.



Interfaces des états composites

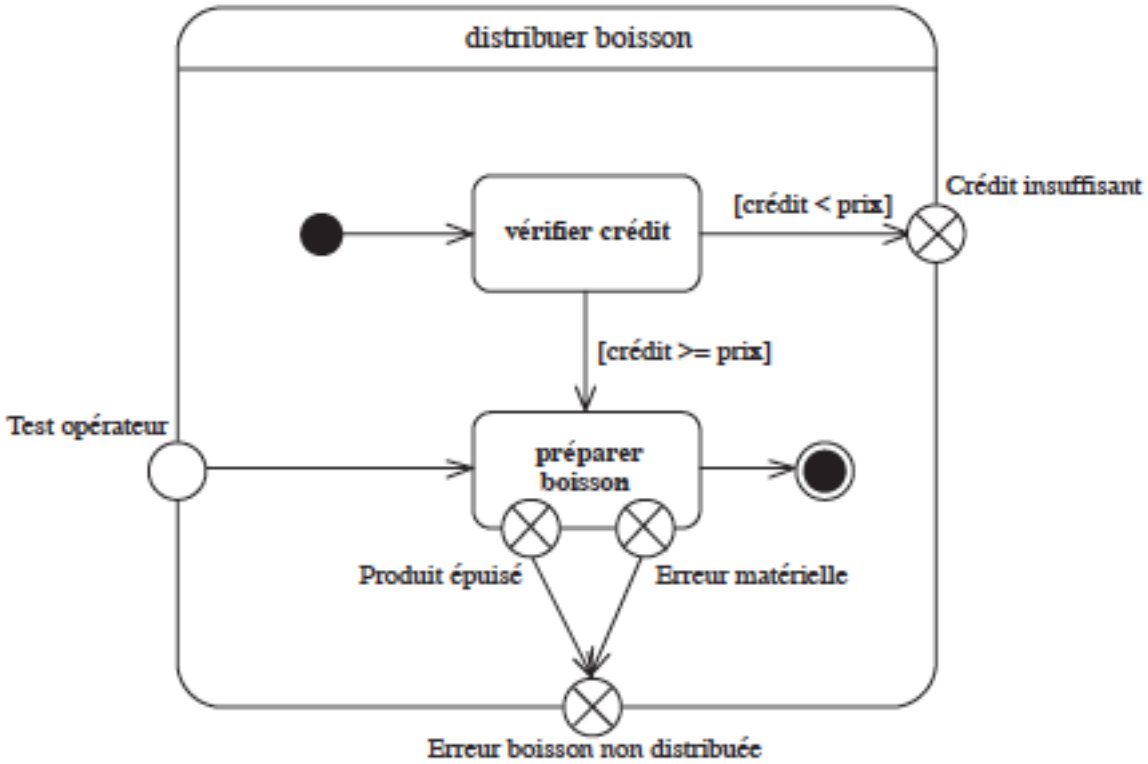
-les sous-états d'un état composite et de les définir dans un autre diagramme.

Pour exprimer la connexion des diagrammes entre eux - utiliser des points de connexion.

Les points d'entrée et de sortie sont respectivement représentés par un cercle vide et un cercle barré à la frontière de l'état.

Ces interfaces permettent d'abstraire les sous-états des macro-états (réutilisabilité).

Exemple : Distribuer boisson


distribuer boisson
vérifier crédit
[crédit < prix]
Crédit insuffisant
[crédit >= prix]
Test opérateur
préparer boisson
Produit épuisé
Erreur matérielle
Erreur boisson non distribuée