

Chapitre 3

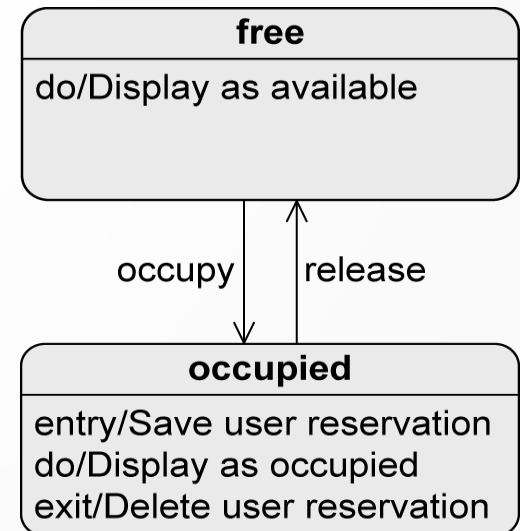
Diagramme d'états-transitions



Plan

2

- Introduction
- Etats simples
- Transitions
- Evénements
- Etats composites
- Points d'entrée et de sortie
- Exemples



Rappel

3

- Les diagrammes de cas d'utilisation modélisent à QUOI sert le système, en organisant les interactions possibles avec les acteurs.
- Les diagrammes de séquence permettent de décrire COMMENT les éléments du système interagissent entre eux et avec les acteurs pour réaliser l'objectif du système.
- Diagramme de séquence
 - Diagramme d'interaction
 - fait partie des diagrammes comportementaux (dynamiques)

Introduction

4

- Appelé également **diagramme d'états** ou **diagramme de machine d'états** (state machine diagram)
- **Objectif** : Décrire le comportement dynamique d'une entité (système, objet...)
- Comportement décrit par des états et transitions entre les états
- Intérêt :
 - Vue synthétique de la dynamique de l'entité
 - Regroupe un ensemble de scénarios

Automates

5

- Un diagramme d'états-transitions est un graphe qui représente un automate à états finis
- Un automate à états finis est la spécification de la séquence d'états que subira un objet au cours de son cycle de vie.
 - Description de tous les états possibles **d'un unique objet** à travers l'ensemble des cas d'utilisation dans lequel il est impliqué
 - Utile pour les objets qui ont un comportement complexe.
 - Un diagramme d'états est alors réalisé pour la classe qui décrit ces objets au comportement complexe.
 - A ne pas faire pour tous les objets
 - Généralement, à faire uniquement pour les objets ayant 3 états ou plus
 - Sauf cas particulier, à chaque instant, chaque objet est dans un et un seul état.

Éléments fondamentaux d'un diagramme d'états

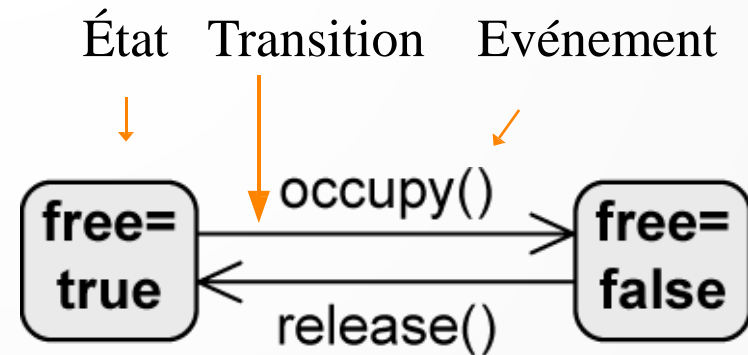
6

- Éléments fondamentaux du diagramme d'états :
 - **Les états** : un état représente une situation dans la vie d'un objet pendant lequel il satisfait une certaine condition, exécute certaines activités, attend certains évènements.
 - **Les transitions entre états** : pour marquer le changement d'état d'un objet.
 - **Les évènements** (ou déclencheurs) qui provoquent des changements d'état.

Éléments fondamentaux d'un diagramme d'états

7

- Les états sont représentés par des rectangles aux coins arrondis
- Les transitions sont représentées par des arcs orientés liant les états entre eux.
- Plusieurs types d'événements pouvant déclencher une transition
 - Appel d'une méthode, signal, ...



Modèle dynamique

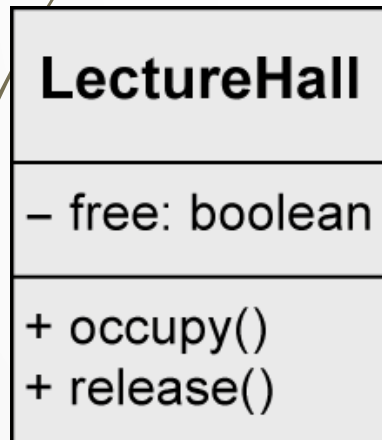
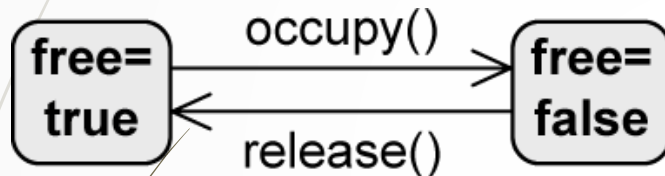
8

- Le modèle dynamique comprend plusieurs diagrammes d'états.
- Chaque diagramme s'exécute concurremment et peut changer d'état de façon indépendante des autres
- Attention
 - Chaque diagramme d'états ne concerne qu'une seule classe.
 - Uniquement pour les classes ayant des comportements complexes
 - Généralement, cela ne dépasse pas le 10% des classes du système

Exemple : Amphithéâtre

9

- Exemple : description du comportement d'un amphithéâtre (Lecture Hall)



```
classe LectureHall {  
    private boolean free;  
  
    public void occupy() {  
        free = false ;  
    }  
    public void release() {  
        free = true ;  
    }  
}
```

Etats

10

- États = nœuds du diagramme d'états-transitions
- Plusieurs types d'états simples:
 - Etat initial (un seul par diagramme)
 - Etat final (aucun ou plusieurs possibles)
 - Etat de terminaison (aucun ou plusieurs possibles)
 - Etats intermédiaires, étapes de la vie du système, de l'objet (plusieurs possibles)
- Etats composites:
 - peuvent contenir des sous-diagrammes d'états-transitions.

Actions et Activités

11

➤ Action

- Opération instantanée (durée négligeable) toujours intégralement réalisée.
- Non interruptible
- Exemple d'action : affecter une valeur à un attribut, créer ou détruire un objet, invoquer une méthode d'un autre objet ou de l'objet lui-même

➤ Activité

- Opération qui nécessite un certain temps d'exécution.
- peut consister en plusieurs actions
- Peut être interrompue à chaque instant.
- UML n'impose pas la façon de décrire les activités

Activités liées à un état ou à une transition

12

- ▶ Pendant le cycle de vie d'un objet, il peut exécuter plusieurs activités.
 - ▶ Des activités au sein des états
 - ▶ Des activités qui sont exécutées lors du franchissement d'une transition
- ▶ Plus précisément, on peut spécifier les différentes activités
 - ▶ Pendant l'état (attachée à l'état)
 - ▶ A l'entrée et la sortie d'un état
 - ▶ Au sein d'un état, lors de la réception d'un événement
 - ▶ Lors du franchissement d'une transition

- « Début » d'un diagramme d'états-transitions
- Lorsqu'un objet est créé, il entre dans l'état initial
- Pseudo-état
 - Transitoire, c'est-à-dire que le système ne peut pas rester dans cet état
 - Plutôt une structure de contrôle qu'un véritable état
- Aucune transition entrante
- Si plus d'une transition sortante
 - Les gardes doivent être mutuellement exclusives et couvrir tous les cas possibles pour garantir qu'exactly un état cible est atteint
- Si l'état initial devient actif, l'objet passe immédiatement à l'état suivant
 - Aucun événement autorisé sur les transitions sortantes (exception : `new()`)

Etat final et Etat de Terminaison

14

➤ Etat final



- Etat réel
- Marque la fin du diagramme ou des sous-états
- L'objet peut rester dans un état final pour toujours

➤ Etat de Terminaison

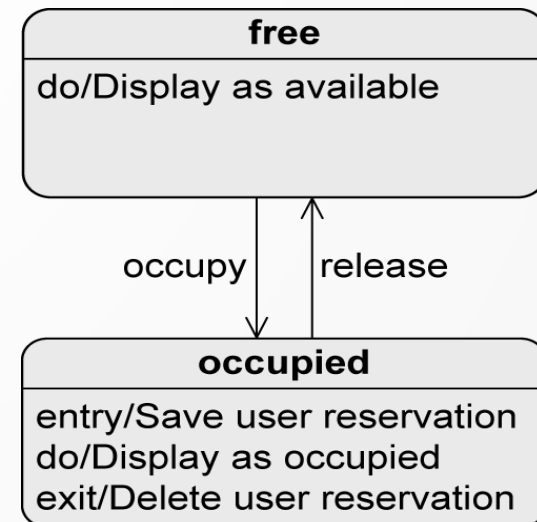
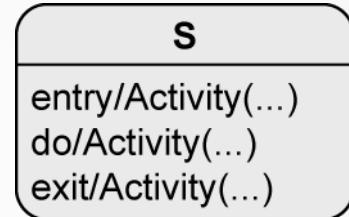


- Pseudo-état
- Termine le diagramme d'états-transitions
- L'objet modélisé cesse d'exister (= est supprimé). La fin de cycle de vie de l'objet.

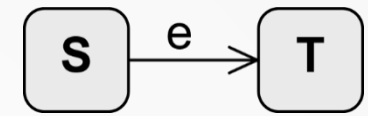
Etats simples

15

- Contenu d'un état simple :
 - le nom
 - l'activité attachée à cet état
 - les activités réalisées pendant cet état
- Lorsqu'un état est actif
 - L'objet est dans cet état
 - Toutes les activités internes spécifiées dans cet état peuvent être exécutées à des moments précis.
- Les principaux types d'activités liées à un état
 - entry / **Activité(...)**
 - Exécutée lorsque l'objet entre dans l'état
 - exit / **Activité(...)**
 - Exécutée lorsque l'objet quitte l'état
 - do / **Activité(...)**
 - Exécutée tant que l'objet reste dans cet état



Transition

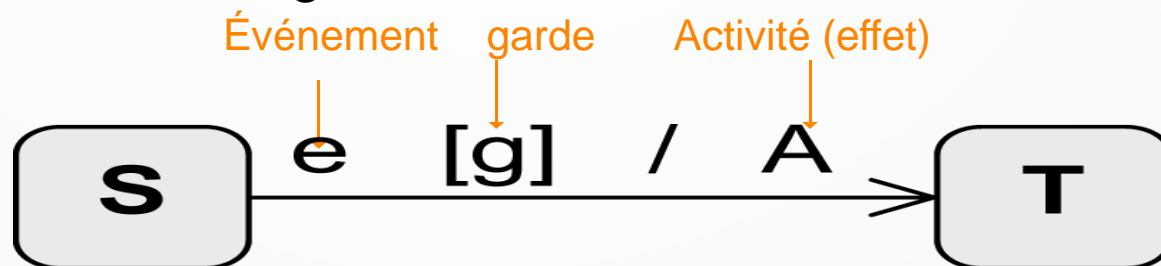


16

- Une transition entre deux états est représentée par un arc qui les lie l'un à l'autre.
 - Le passage est unidirectionnel
 - Elle indique qu'un objet peut changer d'état et **exécuter certaines activités, si un événement déclencheur se produit** et que les conditions de **garde sont vérifiées**.
- Sa syntaxe est la suivante :

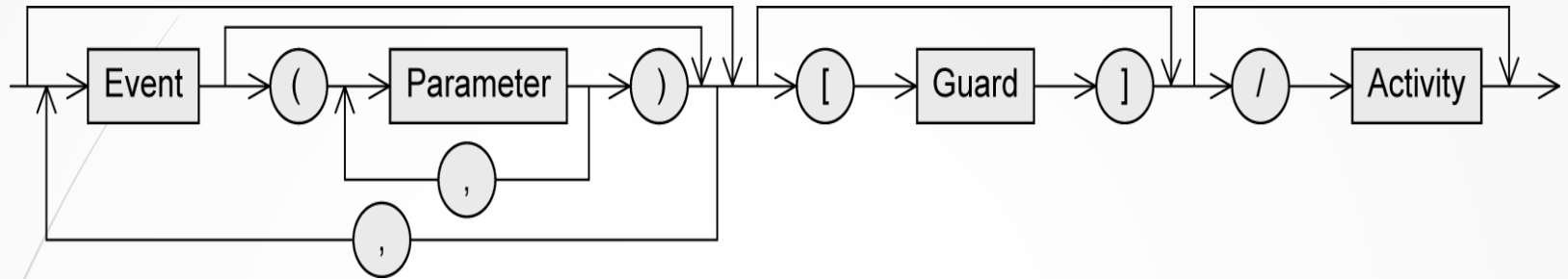
nomEvenement (params) [garde] / activiteARealiser

- Garde désigne une condition qui doit être remplie pour pouvoir déclencher la transition.
- ActivitéARealiser désigne des instructions à effectuer au moment du passage.



Transition - syntaxe

17



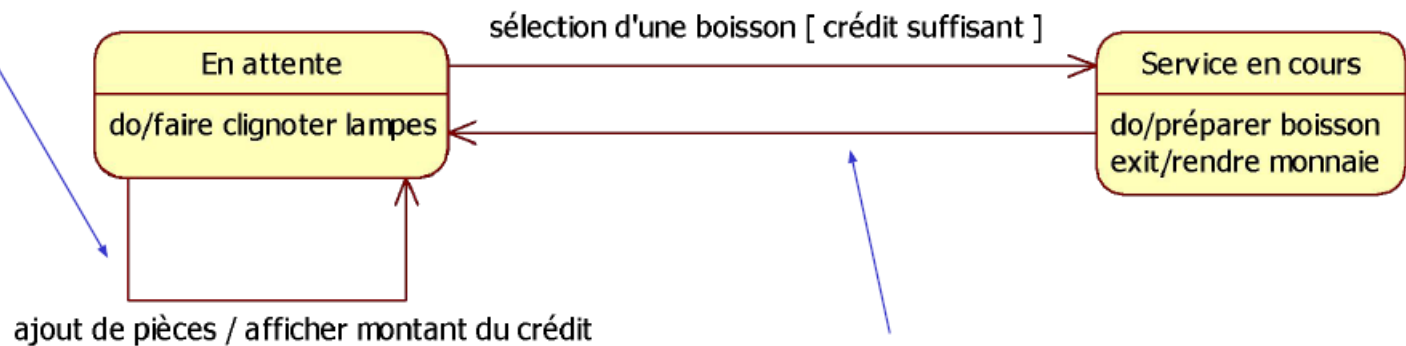
- Si l'événement **Event** se produit, la garde **Guard** est testée
 - Si la garde est vraie
 - Toutes les activités dans l'état actuel sont terminées
 - Toute activité de sortie est exécutée
 - La transition s'effectue
 - L'activité **Activity** est exécutée pendant la transition d'état
 - Si la garde est fausse
 - Aucune transition d'état n'a lieu, l'événement est ignoré

Transition – cas particuliers

18

- Transition automatique :
 - lorsque qu'il n'y a pas de nom d'événement sur une transition, il est sous-entendu que la transition aura lieu dès la fin de l'activité dans l'état courant (événement **Completion**).
- Auto-transition :
 - transition d'un état vers lui-même
 - appelé également transition externe
- Exemple : diagramme d'états du distributeur de boissons

Auto-transition

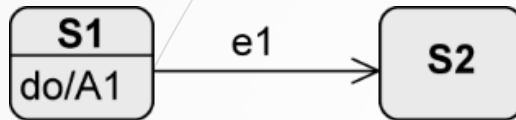


Transition automatique dès la fin de l'activité « préparer boisson »

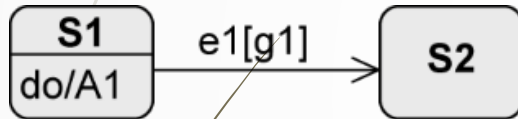
Transition - exemples

19

➤ Quand ont lieu les transitions suivantes ?



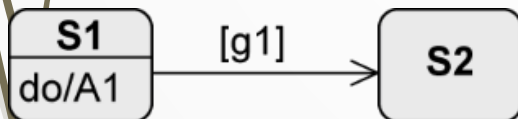
- Si e1 se produit, A1 est abandonnée et l'objet passe à l'état S2



- Si e1 se produit et que g1 est évaluée à vrai, A1 est abandonnée et l'objet passe à l'état S2



- Dès que l'exécution de A1 est terminée, un événement **Completion** est généré et initie le passage au S2



- Dès que l'exécution de A1 est terminée, un événement **Completion** est généré ; si g1 est évaluée à vrai la transition a lieu ; Sinon, la transition ne pourra jamais avoir lieu

Événements déclencheurs

20

- Les transitions d'un diagramme d'états-transitions sont déclenchées par des événements déclencheurs dont les principaux types sont :
 - **Call** : Un appel de méthode sur l'objet courant génère un événement de type **call**.
 - **Change** : Le passage de faux à vrai de la valeur de vérité d'une condition booléenne génère implicitement un événement de type **change**.
 - **Signal** : La réception d'un signal asynchrone, explicitement émis par un autre objet, génère un événement de type **signal**.
 - **after** : L'écoulement d'une durée déterminée après un événement donné génère un événement de type **after**. Par défaut, le temps commence à s'écouler dès l'entrée dans l'état courant.

Événements call et signal

21

- Un événement de type **call** ou **signal** est déclaré ainsi :
nomEvenement (params)
- Les événements de type **call** sont des méthodes déclarées au niveau du diagramme de classes.
 - Par exemple : `occupy(user, lectureHall)`, `register(exam)`
- Les **signaux** sont déclarés par la définition d'une **classe portant le stéréotype « signal »**, ne fournissant pas d'opérations, et dont les attributs sont interprétés comme des arguments.
 - Recevoir un signal : `rightmousedown`, `sendSMS(message)`, ...

Événements change et after

22

- Un événement de type **change** est introduit de la façon suivante :

when (conditionBooleenne)

- Il prend la forme d'un **test continu** et se déclenche potentiellement à chaque changement de valeurs des variables intervenant dans la condition.
- Cas particulier : transition temporelle absolue
 - Exemple : when (time == 16), when(date==20150101)

- Un événement **time** de type **after** est spécifié par :

after (duree)

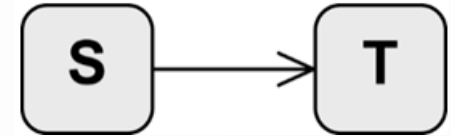
- Le paramètre s'évalue comme une durée, par défaut écoulée depuis l'entrée dans l'état courant.
- Transition temporelle relative
- Par exemple : after(10 secondes)

Événements Completion et Any Receive

23

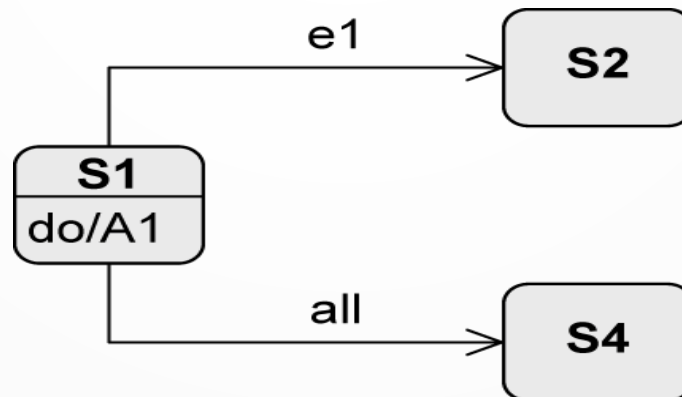
➤ Événement **Completion**

- Généré automatiquement lorsque tout ce qu'on a à faire dans l'état actuel est terminé



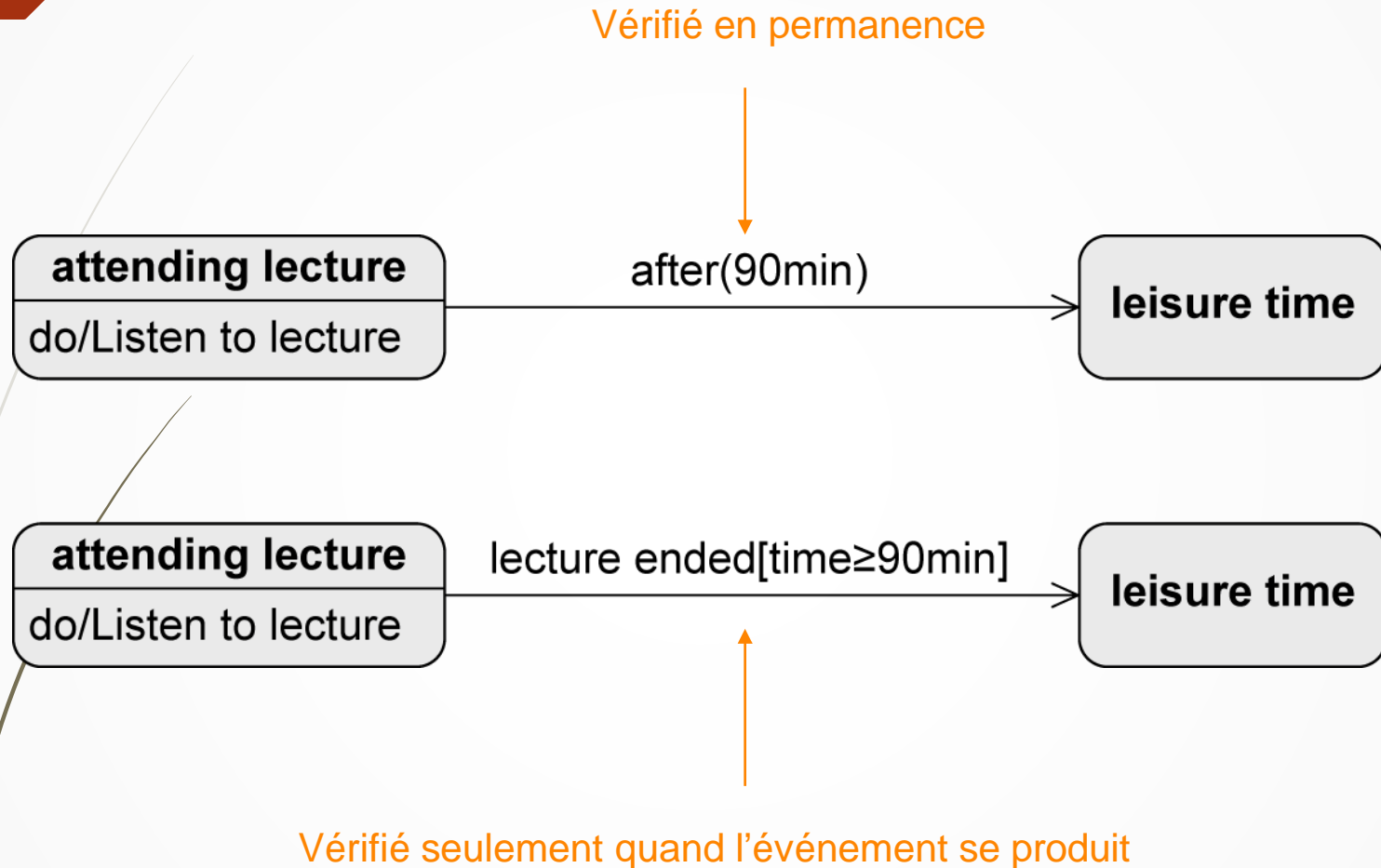
➤ Événement **Any Receive**

- Se produit lorsqu'un événement se produit et qui ne déclenche aucune autre transition de l'état actif
 - Mot-clé **all**



Événement Change vs garde

24



Question: Que se passe-t-il si la conférence dure moins de 90 minutes ?

Transition interne

25

- 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 prédéfinis** sont introduits permettant le tir de transitions internes : **entry/**, **do/**, et **exit/**.
- D'autres événements peuvent survenir au sein de l'état et qui vont déclencher l'exécution de certaines activités

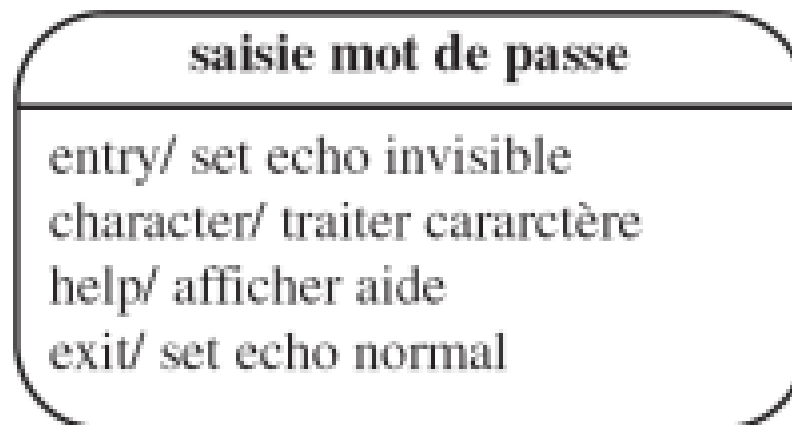
saisie mot de passe

entry/ set echo invisible
character/ traiter caractère
help/ afficher aide
exit/ set echo normal

Transition interne

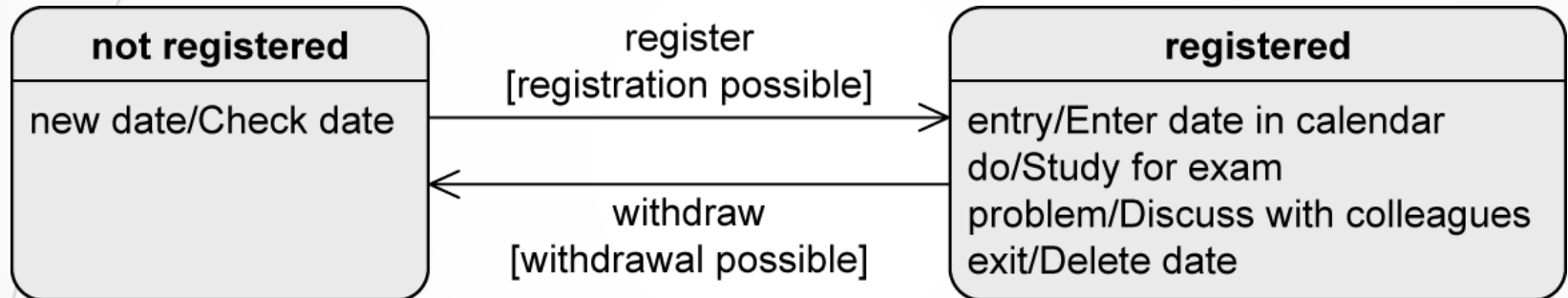
26

- Dans une transition interne, le contexte de l'activité est préservé
 - On ne sort pas de l'état
- Les transitions internes sont spécifiées dans le compartiment inférieur de l'état, sous le compartiment du nom.
- Chaque transition interne est décrite selon la syntaxe suivante :
nomEvenement (params) [garde] / activiteARealiser
- **Exemple** : les transitions déclenchées par les événements **character** et **help**



Exemple – état d'inscription à un examen

27



Transition interne vs Auto-transition

28

- Auto-transition :
 - Est une **transition externe** d'un état vers lui-même
 - Le contexte est réinitialisé (on sort et on re-rentre dans l'état)
- La transition **event1/Activity3**
 - Interne ou externe : quelle différence ?

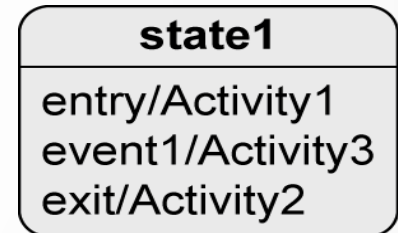
■ Si event1 se produit

- L'objet reste dans **state1**
- **Activity3** est exécutée

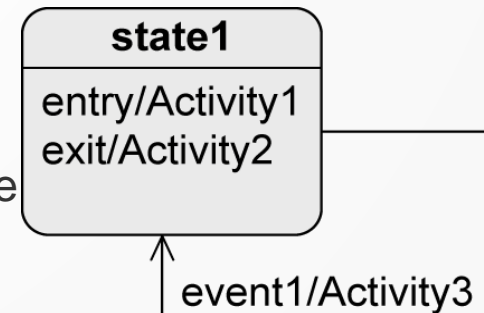
■ Si event1 se produit

- L'objet quitte **state1** et **Activity2** est exécutée
- **Activity3** est exécutée
- L'objet entre dans **state1** et **Activity1** est exécutée

Transition interne



Auto-Transition



Ordre d'exécution des activités

29

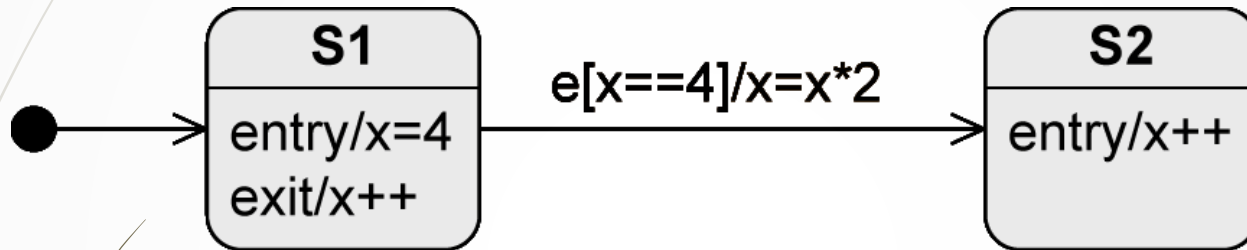
Si un événement se produit sur une transition :

- En entrée
 - Activité sur la transition d'entrée
 - Activité d'entrée
 - Activité associée à l'état
- En interne
 - Interruption de l'activité en cours (contexte sauvé)
 - Activité interne
 - Reprise de l'activité
- En sortie
 - Interruption de l'activité en cours (contexte perdu)
 - Activité de sortie
 - Activité sur la transition de sortie
- Auto-transition
 - Interruption de l'activité en cours (contexte perdu)
 - Activité de sortie
 - Activité sur l'auto-transition
 - Activité d'entrée
 - Activité associée à l'état

Exemple – Ordre d'exécution des activités

30

- Supposons que S1 soit actif... quelle est la valeur de **x** après l'apparition de l'événement **e** ?



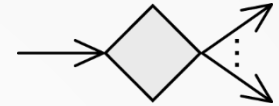
- S1 devient actif (l'objet est dans l'état S1), x est mis à la valeur 4
- e se produit, la garde est vérifiée et évaluée à vrai
- L'objet quitte S1, x est mis à 5
- La transition a lieu, x est mis à 10
- L'objet entre dans l'état S2, x est mis à 11

Transitions alternatives et concurrentes

31

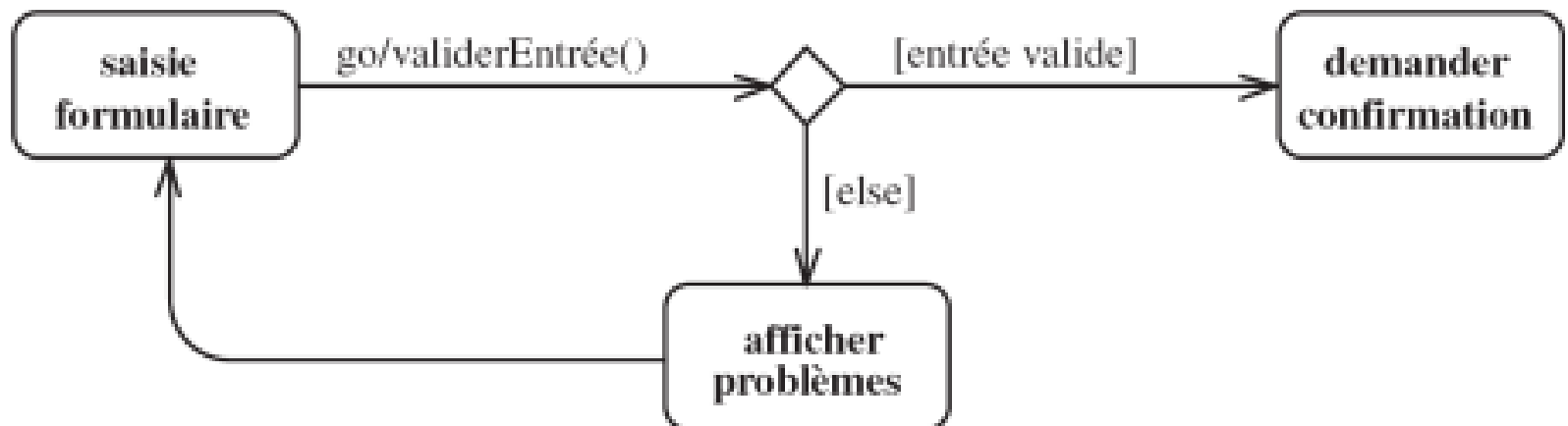
- Il est possible de représenter des alternatives pour le franchissement d'une transition.
 - On utilise pour cela des pseudo-états particuliers :
 - le **nœud de décision**, et
 - le **nœud de jonction**
- Les diagrammes d'états permettent de décrire efficacement les mécanismes ou des sous-états concurrents
 - On utilise pour cela
 - le nœud de parallélisation ou transition **fork**
 - Le nœud de synchronisation ou transition **join**

Nœud de décision



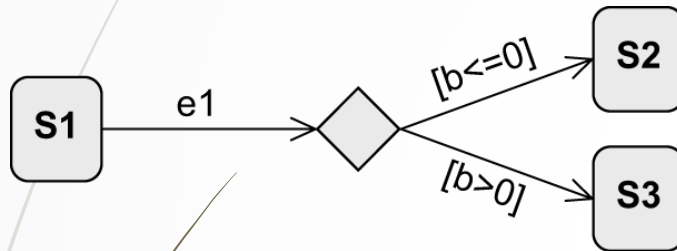
32

- Utilisé pour modéliser des transitions alternatives
- Le nœud de décision relie un état d'origine à plusieurs états de destination
- Les transitions sortantes sont dotées de gardes. Les gardes après le nœud 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 nœud de décision.
 - Les **gardes** sont **mutuellement exclusives**.



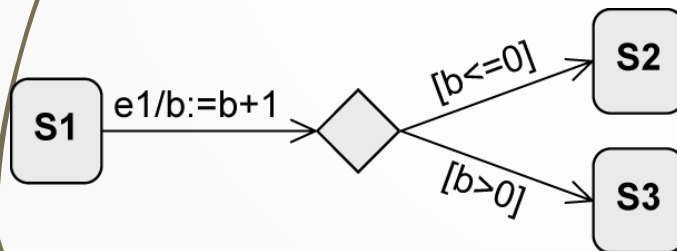
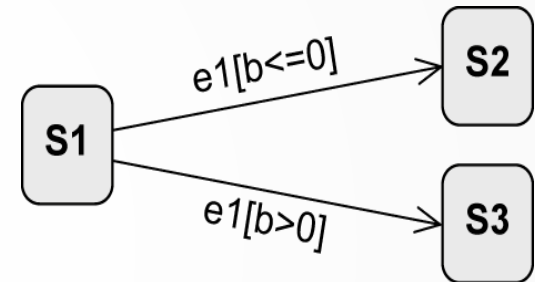
Nœud de décision

33



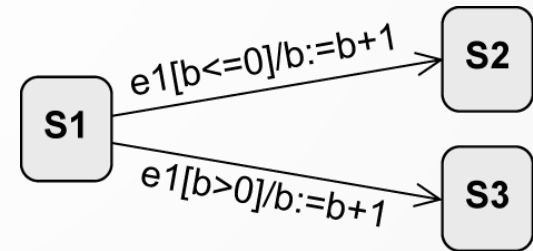
équivalent?

=



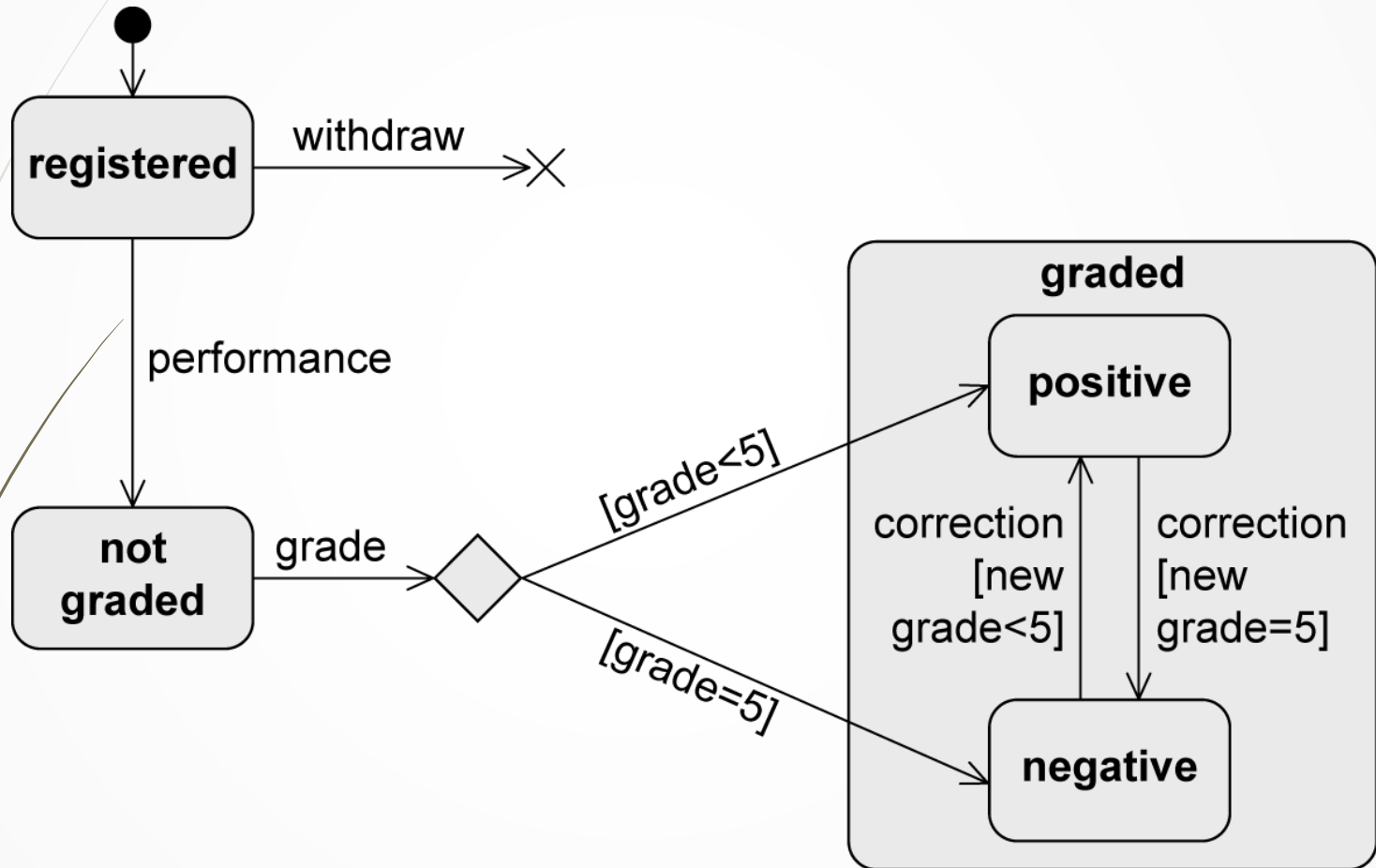
équivalent?

≠

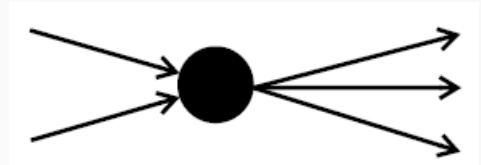


Nœud de décision : exemple

34



Nœud de jonction



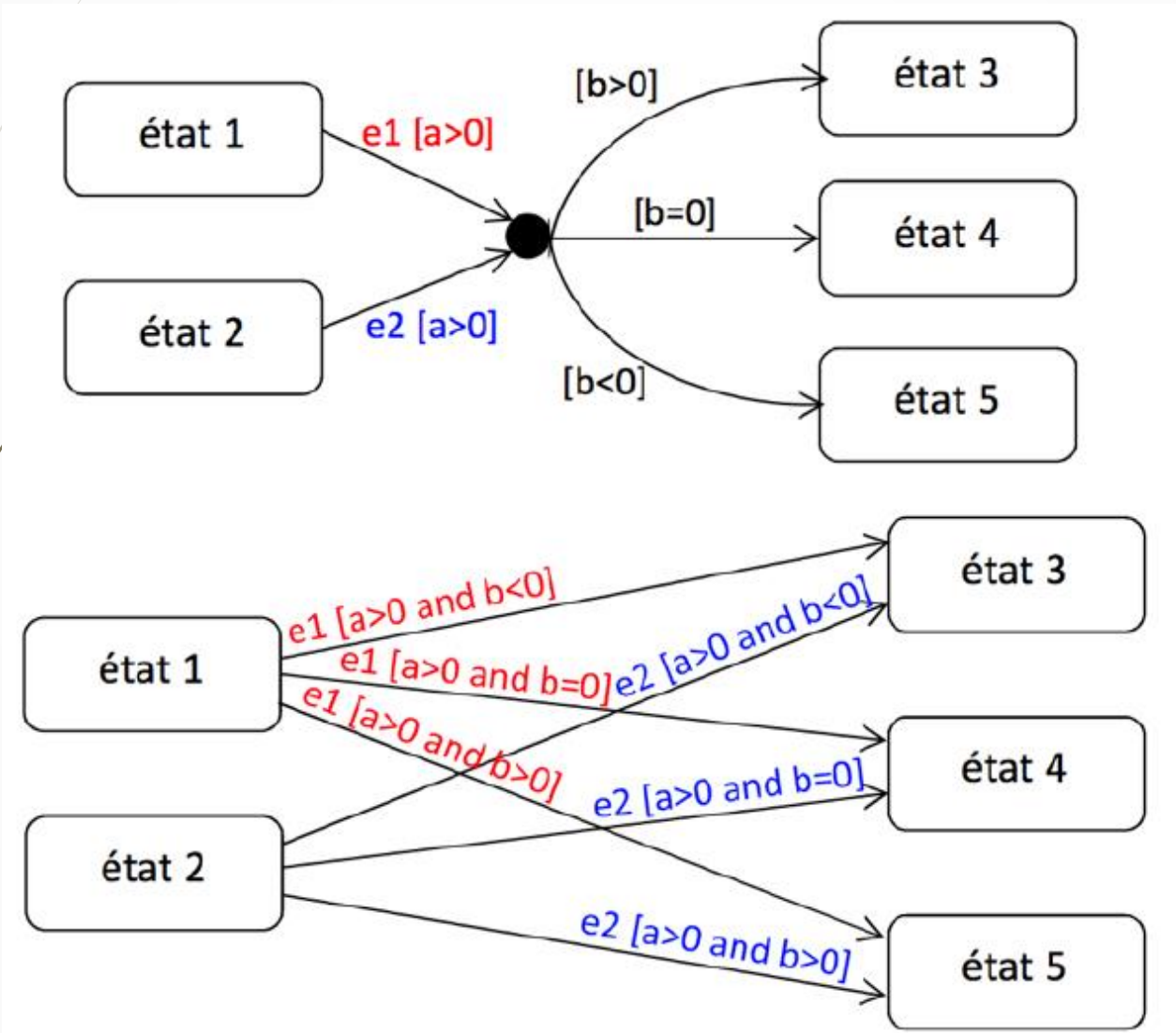
35

- Permet de partager des segments de transition (notation plus compacte, amélioration de la visibilité des chemins alternatifs)
- Plusieurs transitions peuvent viser et/ou quitter un point de jonction
- Tous les chemins (suites de segments) à travers le point de jonction sont potentiellement valides (on peut donc représenter un comportement équivalent en créant une transition pour chaque paire de segment avant et après le point de jonction)
- Pour pouvoir emprunter un chemin, toutes les gardes le long de ce chemin doivent s'évaluer à VRAI dès le franchissement du premier segment

Nœud de jonction

36

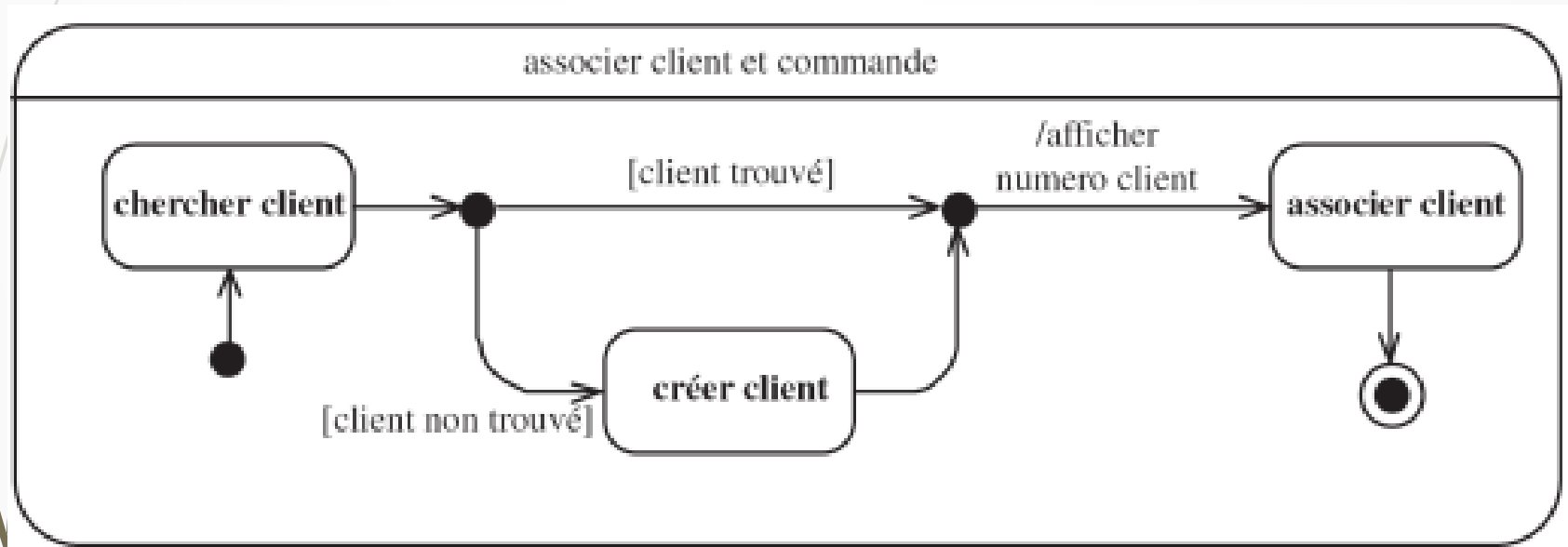
- Deux représentations équivalentes



Nœud de jonction

37

- Bien adapté pour représenter des chemins optionnels
if (...) {...}
- Utilisation de deux points de jonction

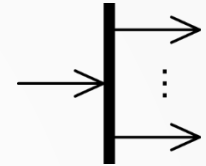


Nœuds de parallélisation et de synchronisation

38

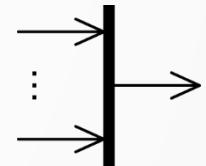
➤ Nœud de parallélisation

- Une transition **fork**
- Pseudo-état
- Divise le flot de contrôle en plusieurs flots simultanés
- 1 transition entrante
- Plus d'une transition sortante (>1)



➤ Nœud de synchronisation

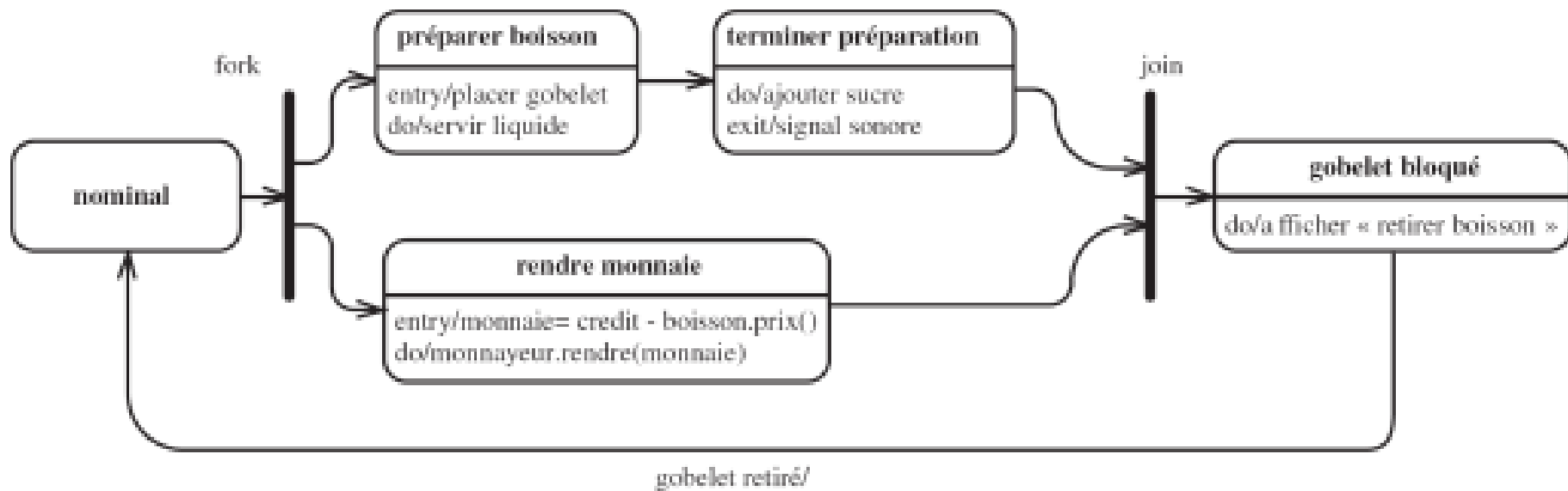
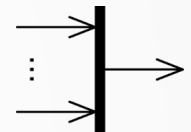
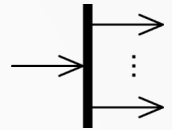
- Une transition **join**
- Pseudo-état
- Fusionne plusieurs flots simultanés
- Plus d'une transition entrante (>1)
- 1 transition sortante



Transitions concurrentes

39

- Une transition **fork** correspond à la création de deux ou plusieurs états concurrents.
- Une transition **join** correspond à une barrière de synchronisation qui supprime la concurrence.
- Pour pouvoir continuer leur exécution, toutes les tâches concurrentes doivent préalablement être prêtes à franchir la transition.



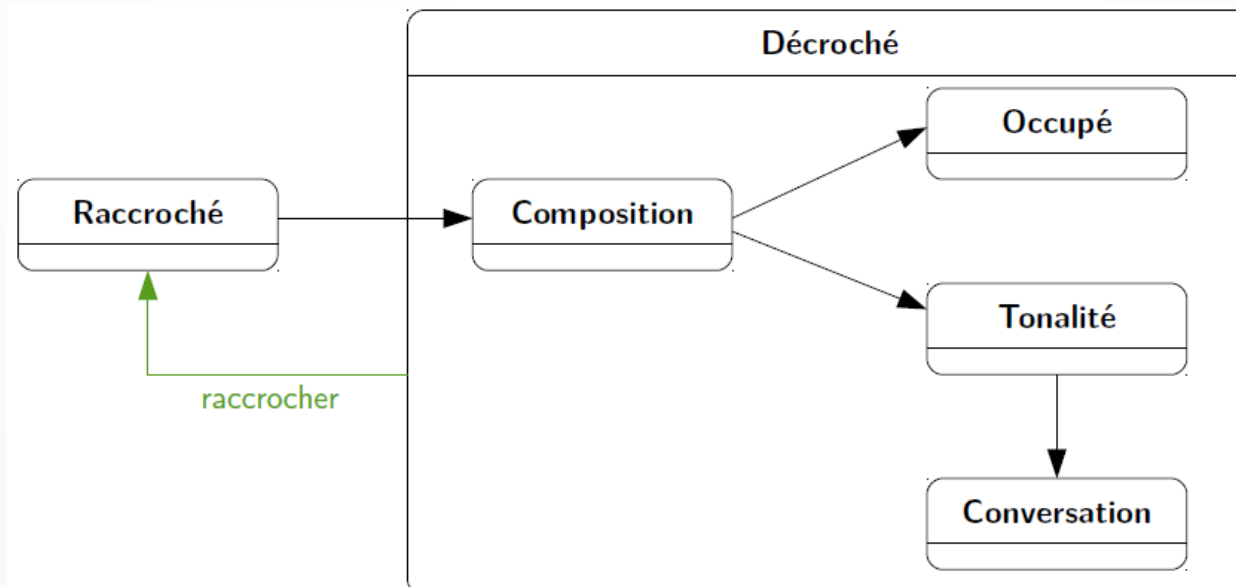
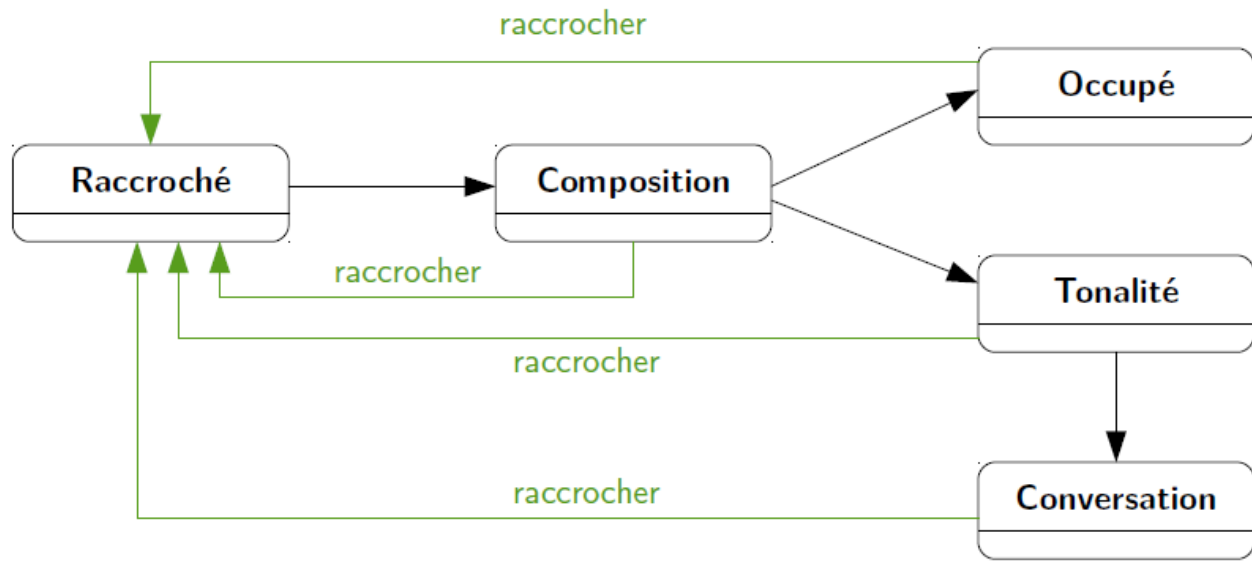
Etat composite (1)

40

- C'est un état regroupant un ensemble d'états
- Objectifs :
 - Hiérarchiser les états
 - Structurer les comportements complexes
 - Factoriser les actions
- Un état composite, par opposition à un état dit « simple », est 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.

Etat composite (2) : téléphone

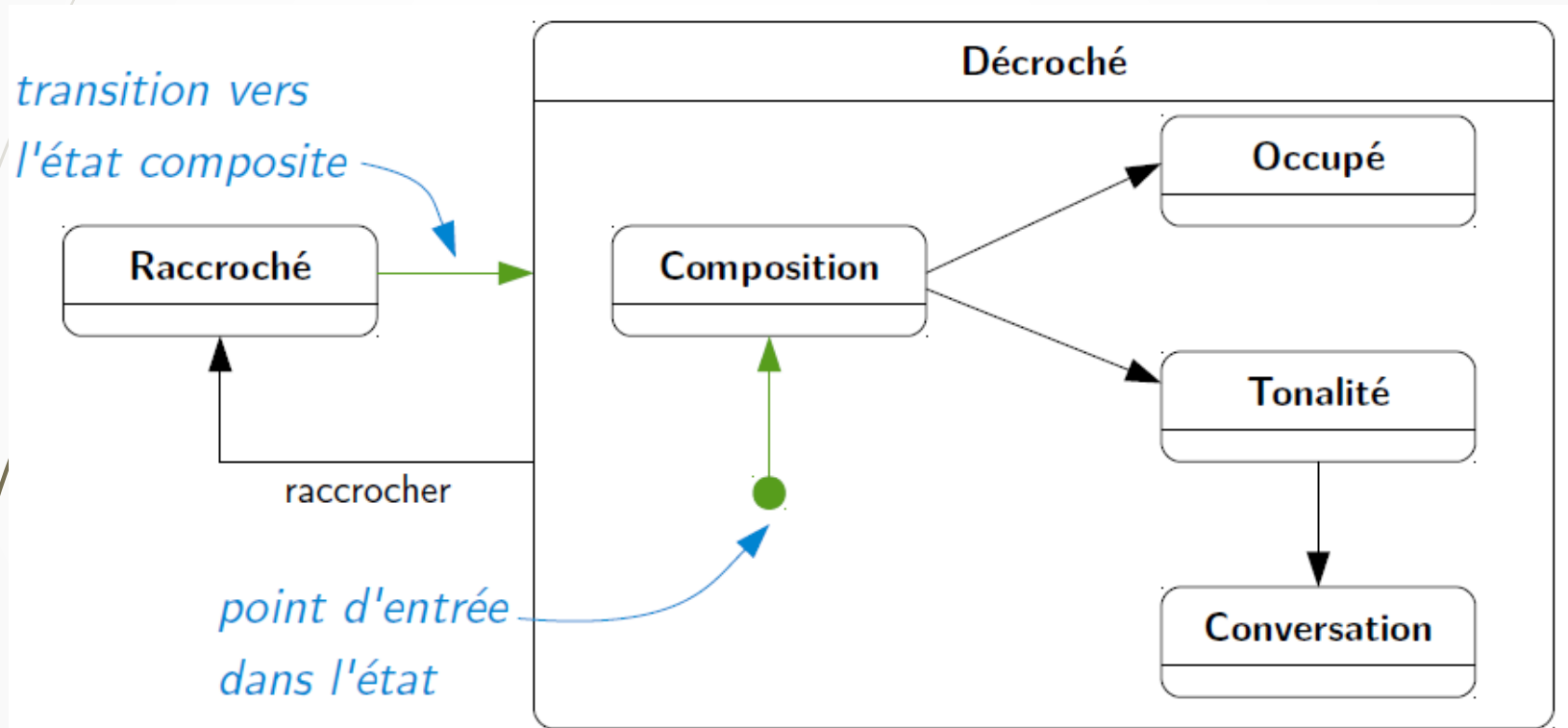
41



Etat composite (3)

42

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

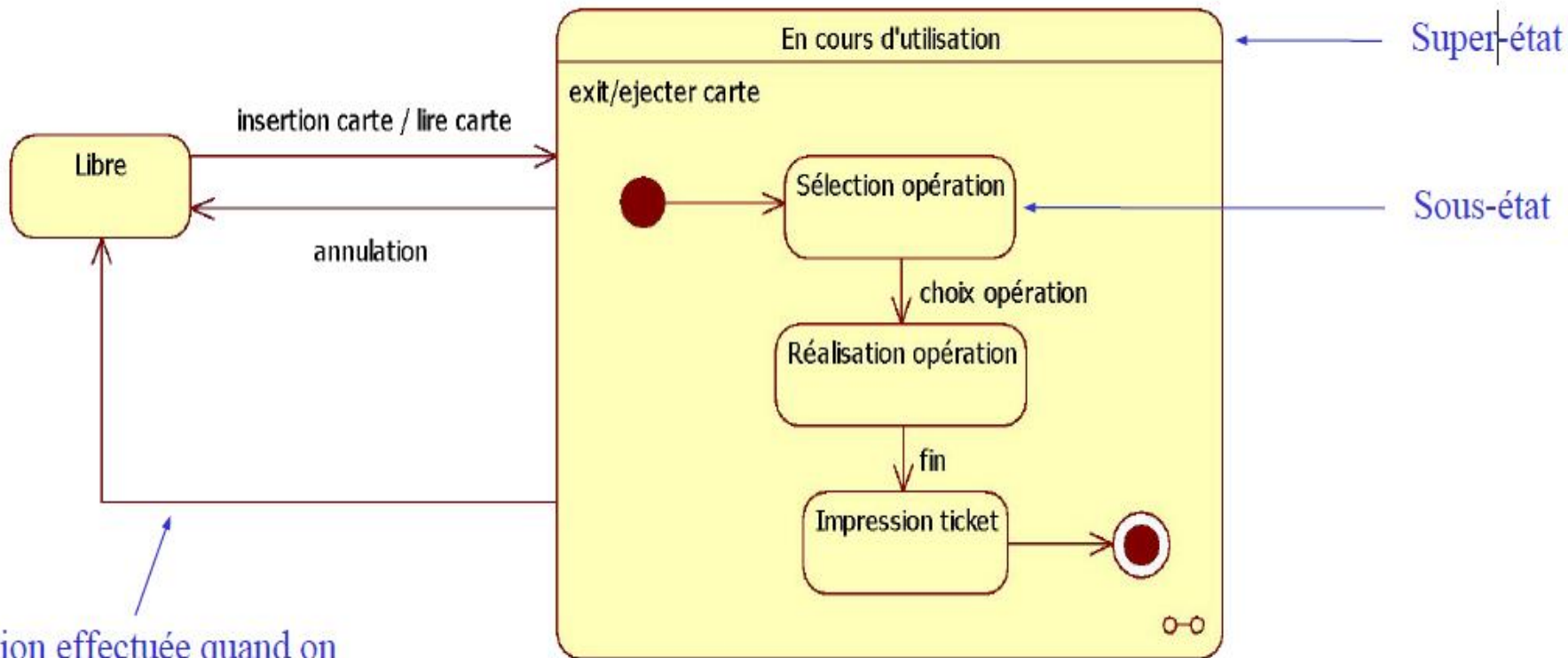


- Modélisation équivalente

Etat composite (4)

43

- 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.

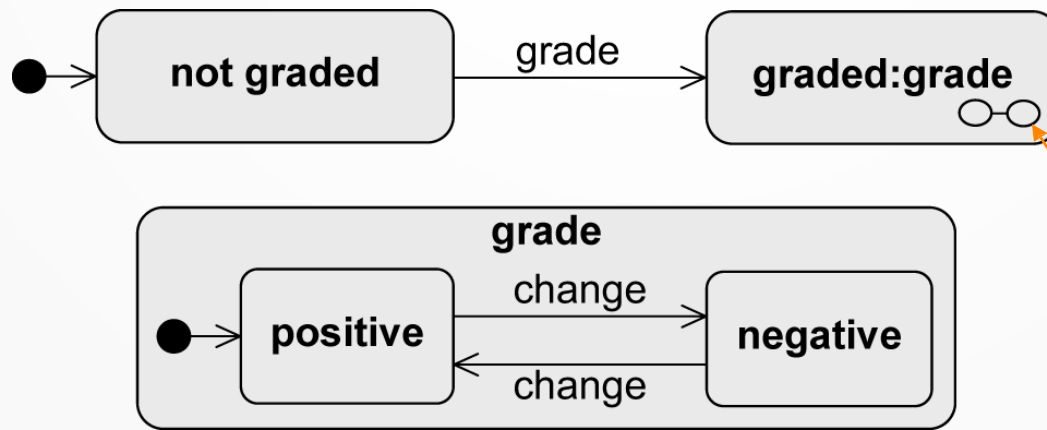


Transition effectuée quand on arrive à l'état final du super-état

Une sous-machine à états

44

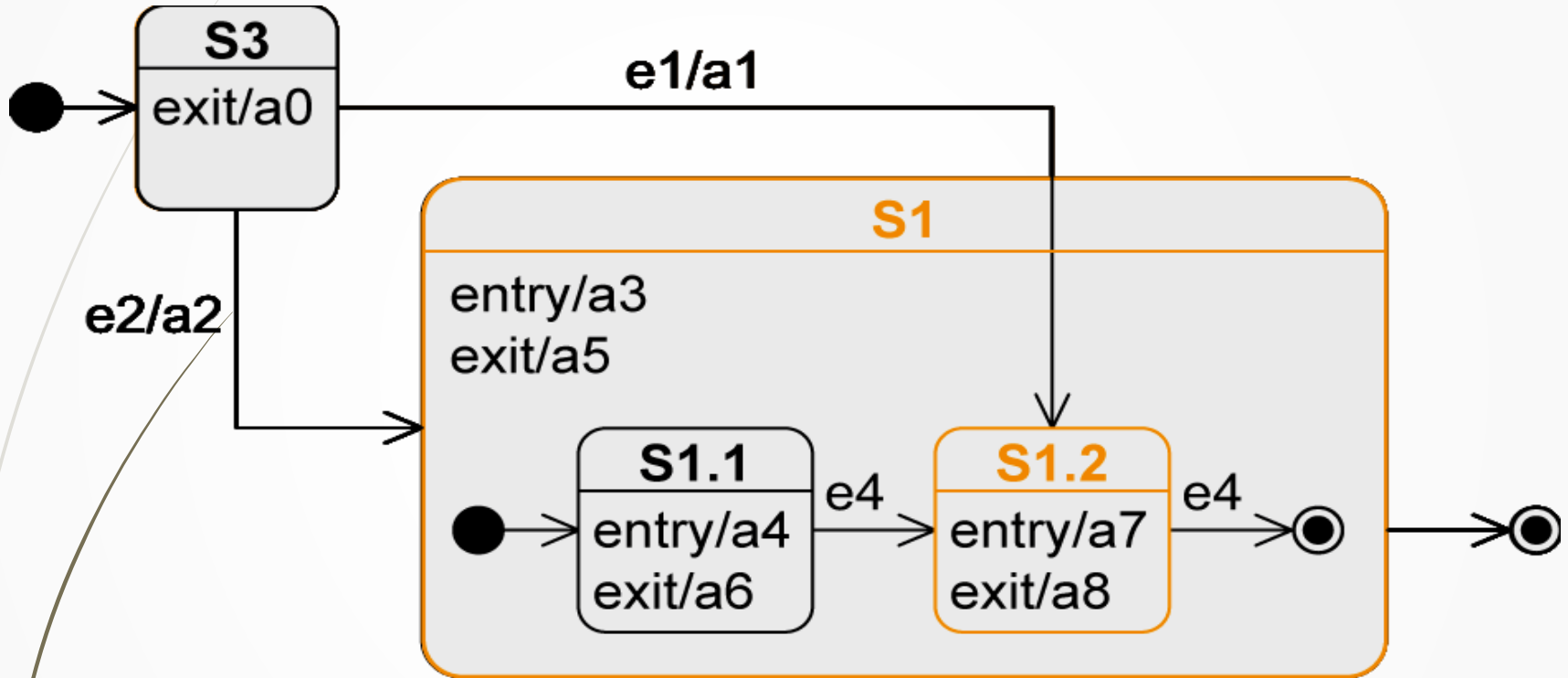
- Une sous-machine à états (submachine state, SMS) représente l'invocation d'un diagramme d'états définie ailleurs.
- Pour réutiliser des parties de diagrammes d'états-transitions dans d'autres diagrammes d'états-transitions
- Notation : **état** : **SubmachineState**
- Dès que la sous-machine à états est activée, le comportement de la sous-machine est exécuté
 - Correspond à l'appel d'un sous-programme dans les langages de programmation



symbole de raffinement
(facultatif)

Transition entrant dans un sous-état d'un état composite

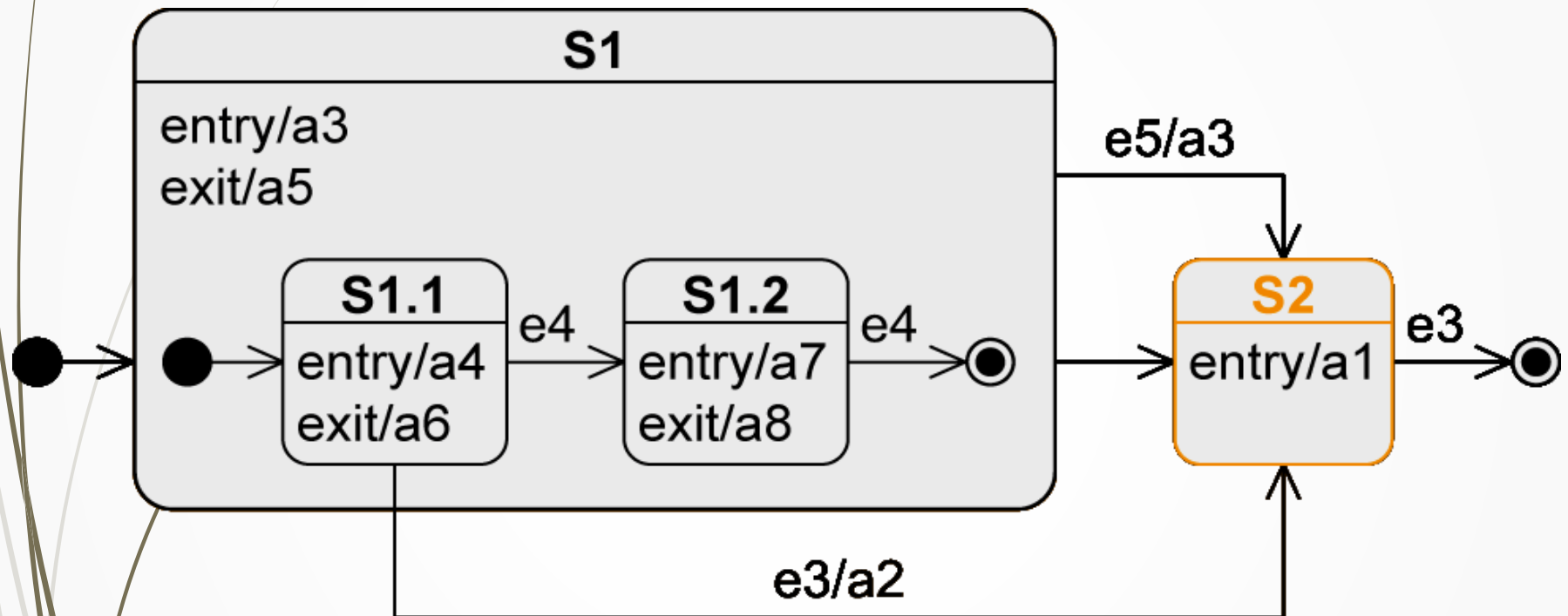
45



Événement	État	Activités exécutées
"Début"	S3	
e1	S1/S1.2	a0-a1-a3-a7

Transition sortant d'un sous-état d'un état composite

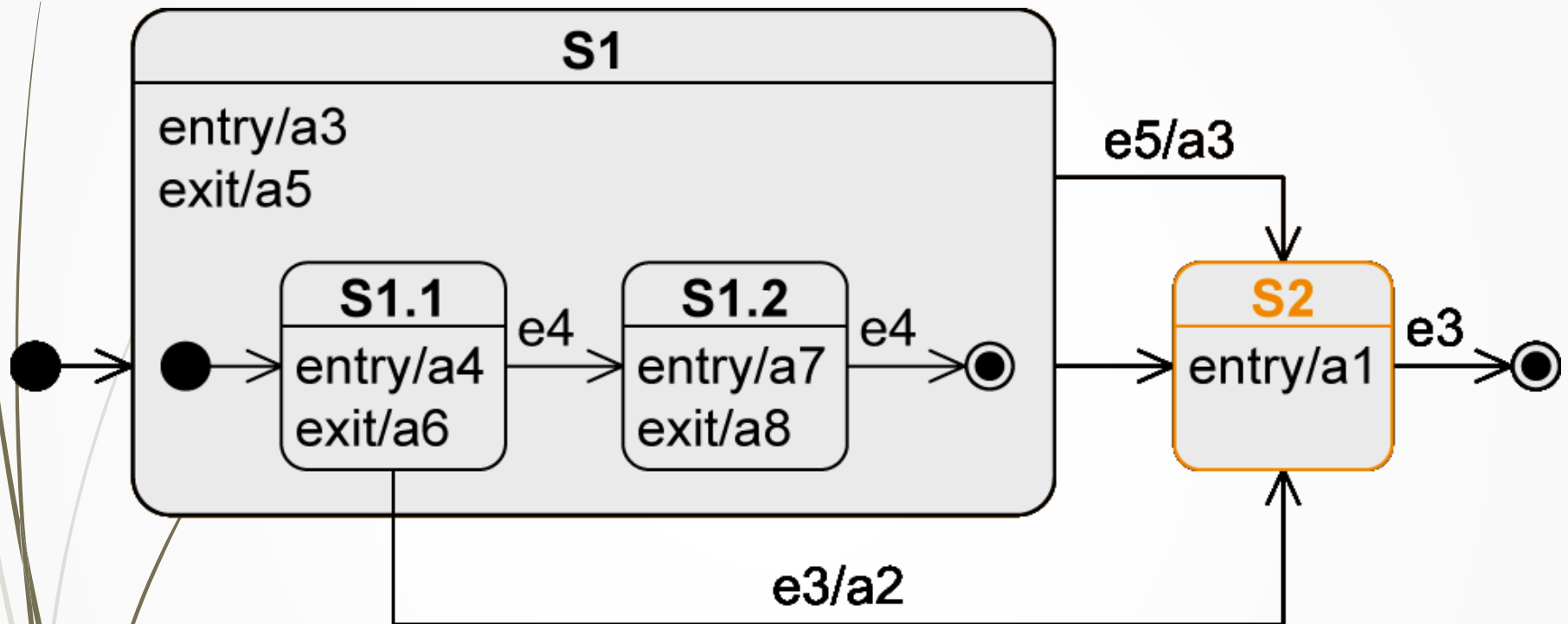
46



Événement	État	Activités exécutées
"Début"	S1/S1.1	a3-a4
e3	S2	a6-a5-a2-a1

Transition sortant d'un état composite

47

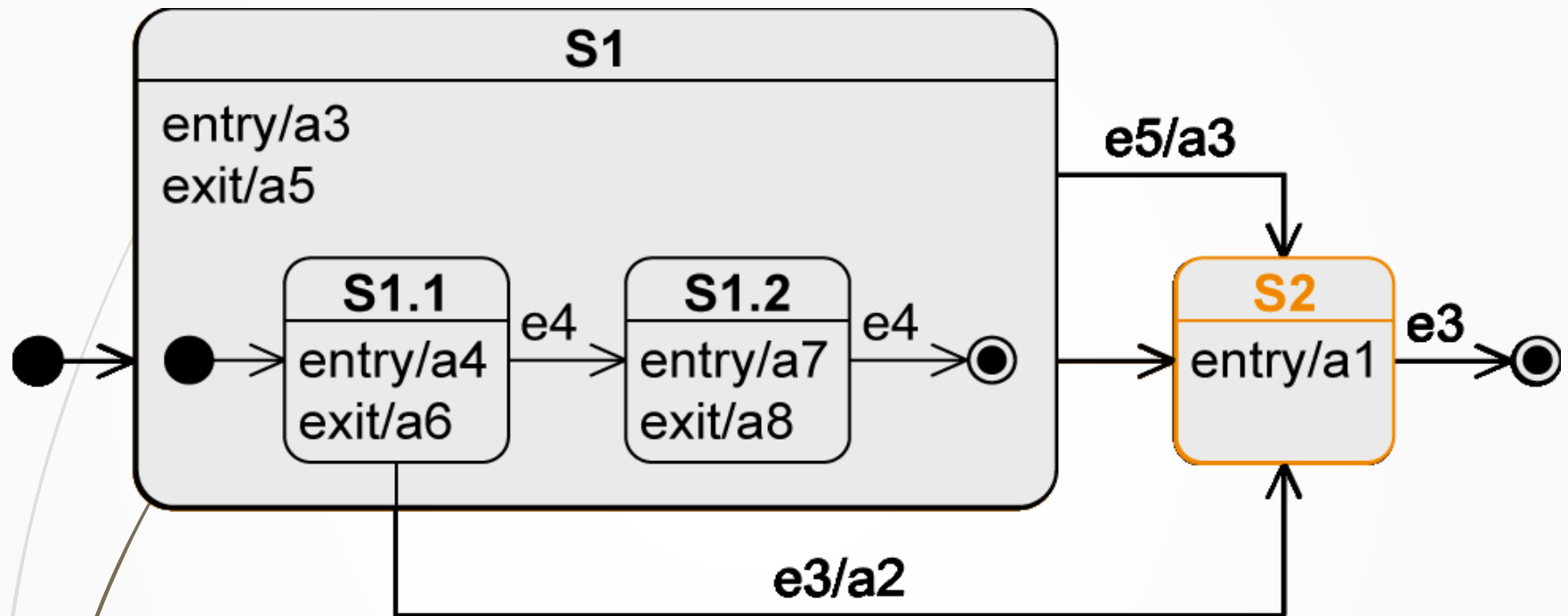


e	Événement	État	Activités exécutées
	"Début"	S1/S1.1	a3-a4
	e5	S2	a6-a5-a3-a1

Quel que soit le sous-état de S1 qui est actif, dès que e5 se produit, le système passe à S2

Transition automatique depuis l'état composite

48

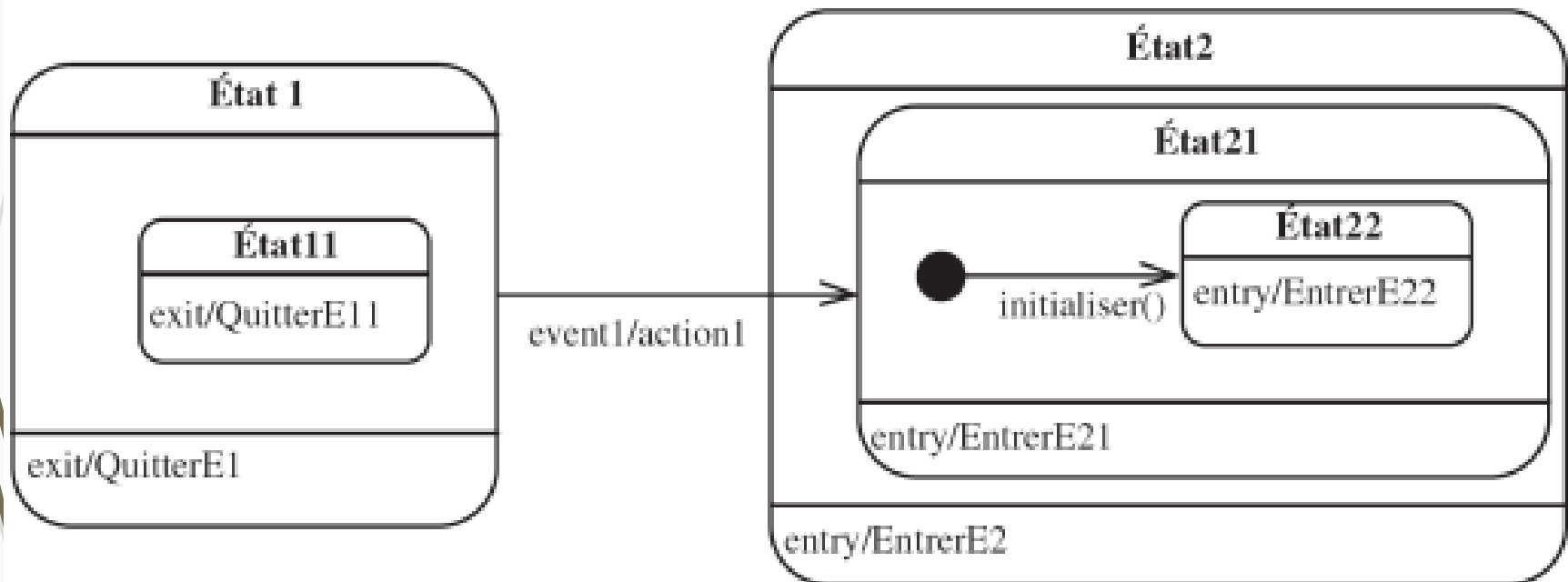


Événement	État	Activités exécutées
Départ	S1/S1.1	a3-a4
e4	S1/S1.2	a6-a7
e4	S2	a8-a5-a1

Transition sortant d'un état composite

49

- Depuis **État11**, quand l'événement **event1** survient
 - On produit la séquence d'activités : QuitterE11, QuitterE1, action1, EntrerE2, Entrer21, initialiser, EntrerE22
 - L'objet se trouve alors dans Etat22.

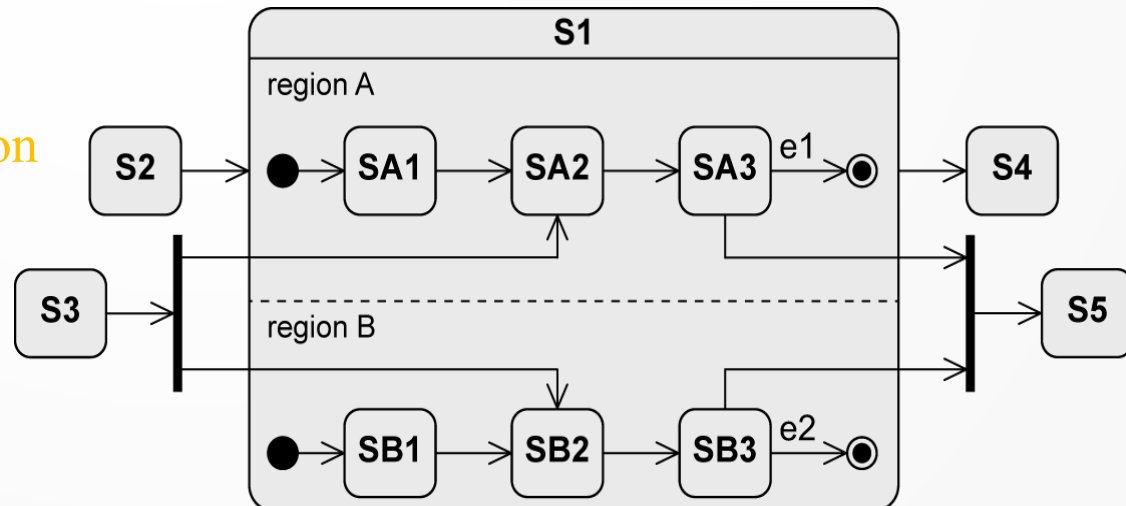


État concurrent ou orthogonal

50

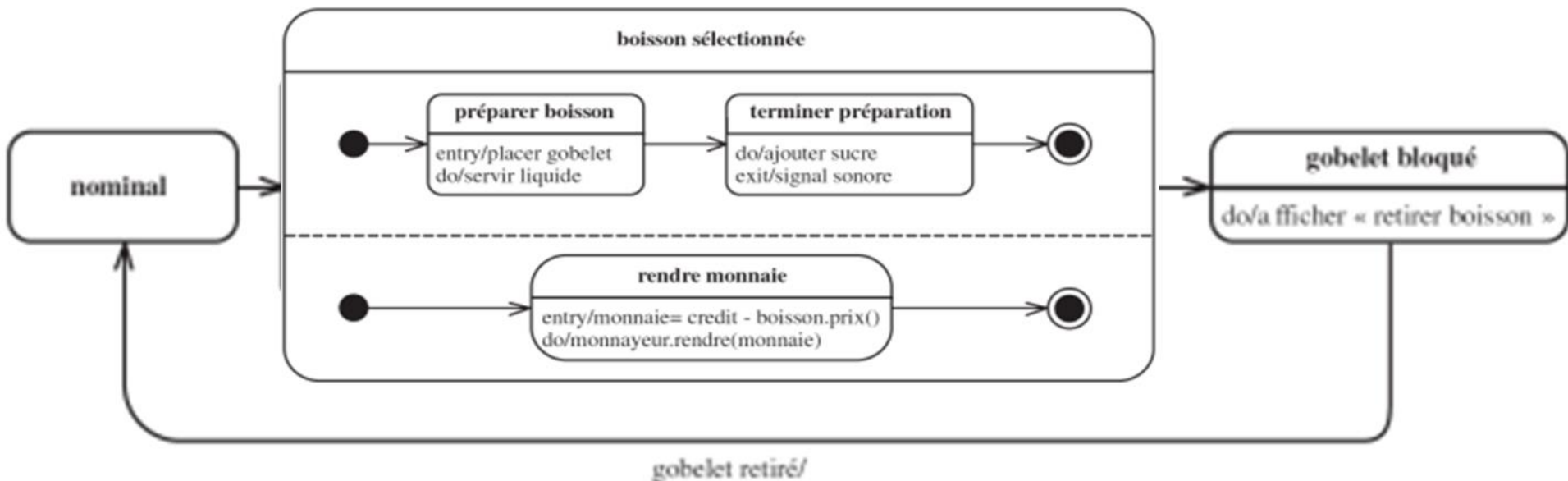
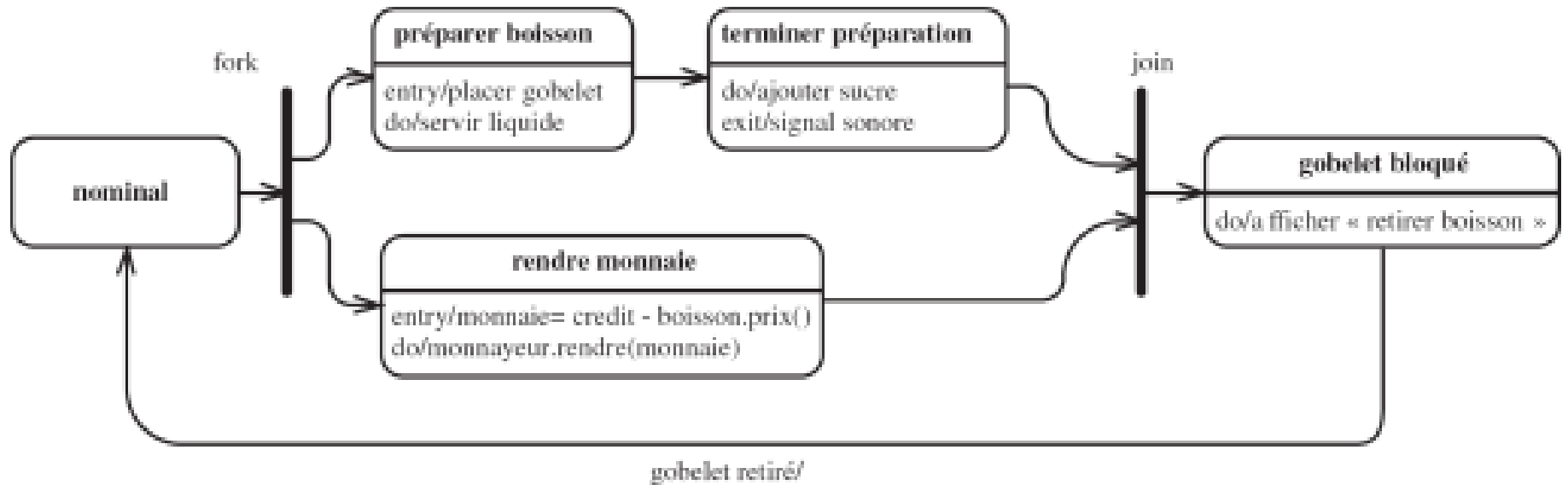
- L'état composite est divisé en deux (ou plusieurs) régions séparées par une ligne pointillée
- Chaque région possède son propre diagramme d'états.
- **Un état de chaque région est toujours actif à tout moment**
- **Entrée** : le passage à la frontière de l'état concurrent active les états initiaux de toutes les régions
- **Sortie** : l'état final doit être atteint dans toutes les régions pour déclencher l'événement **Completion**

Utiliser les nœuds parallélisation et de synchronisation pour atteindre différents sous-états



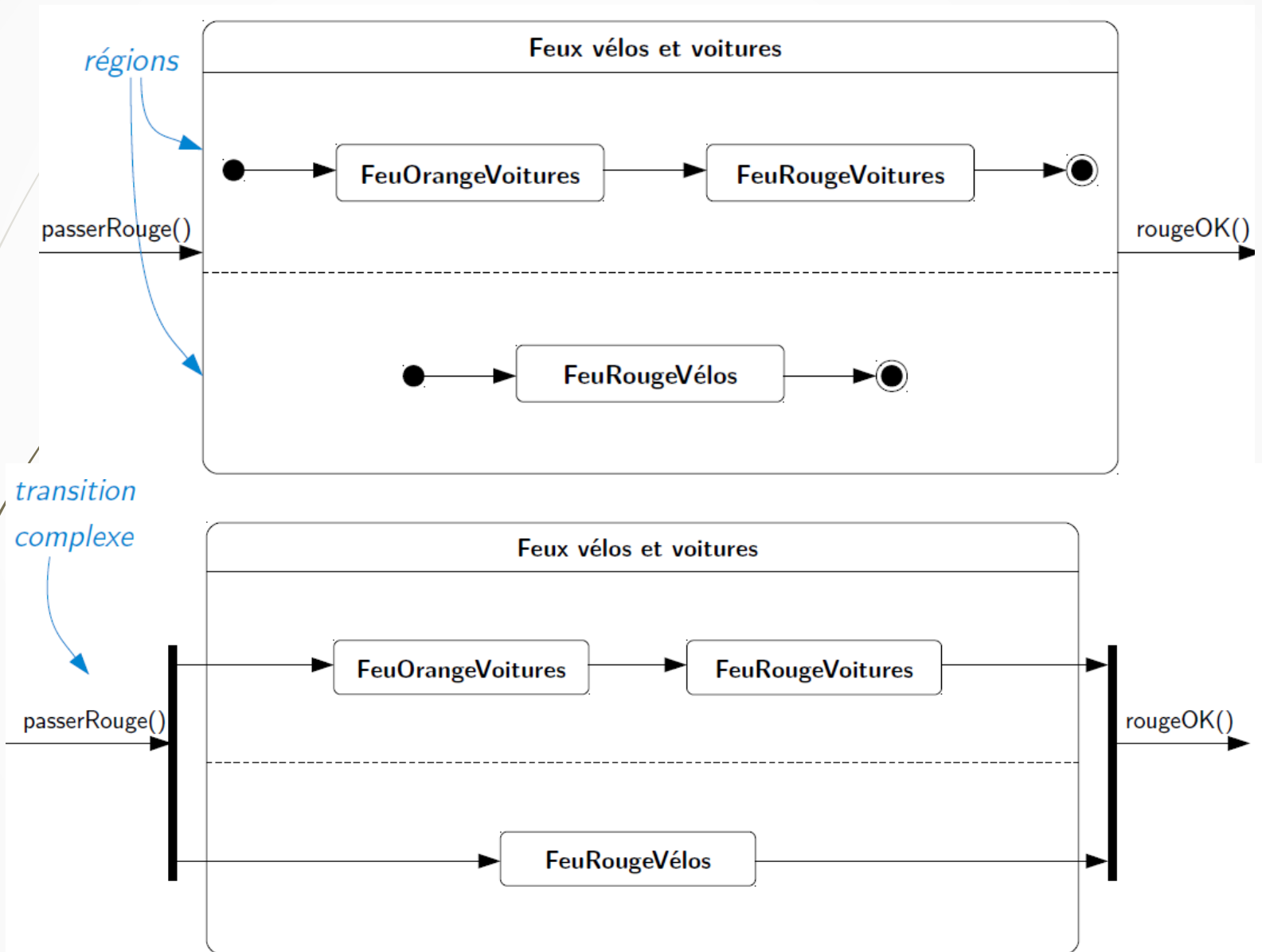
Distributeur de boisson

51



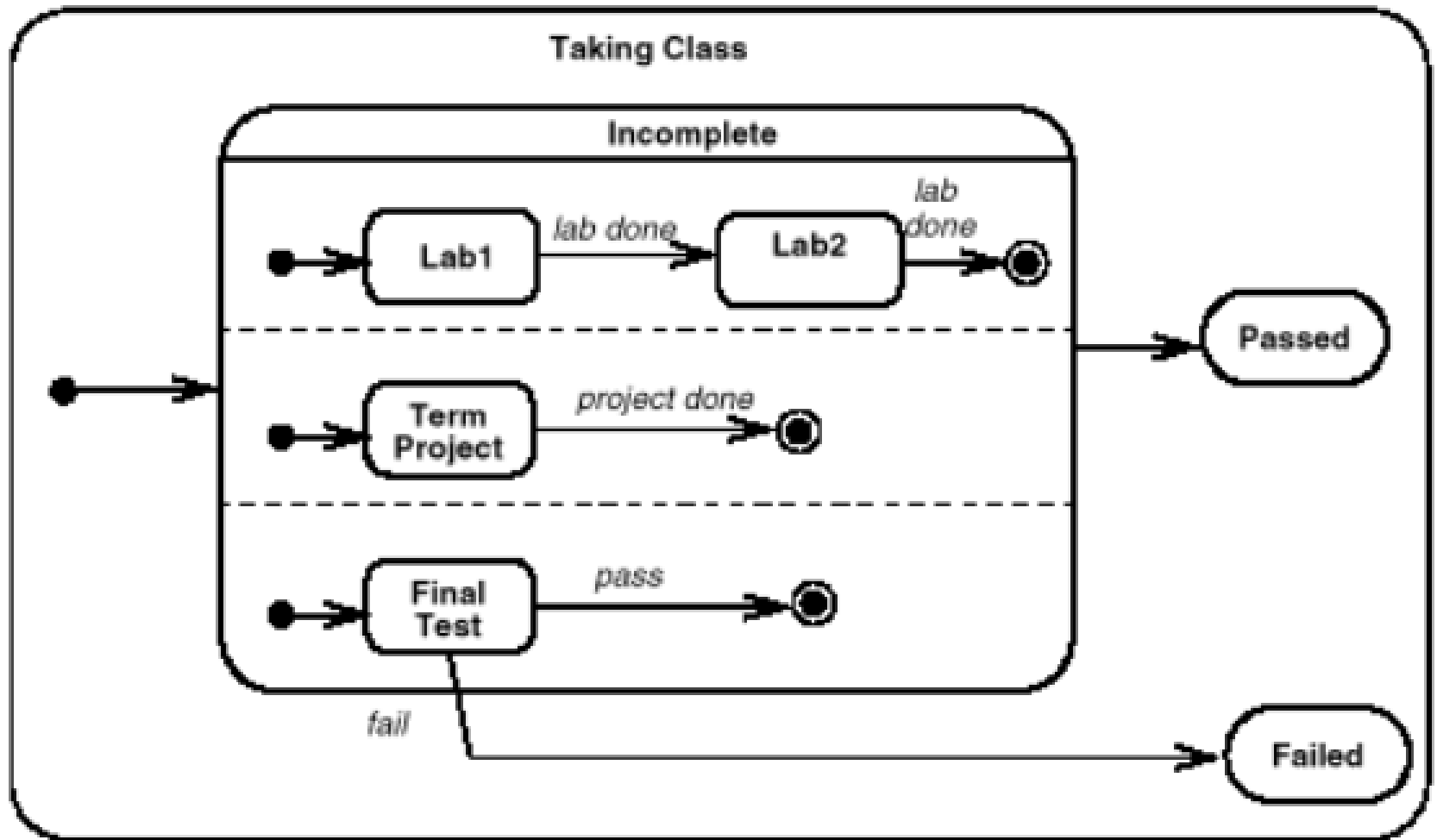
Représentations équivalentes

52



État orthogonal : exemple

53



État orthogonal : exemple - suite

54

- L'exemple précédent montre un état composite **Incomplete** qui se décompose en trois régions. L'entrée dans l'état composite
- **Taking Class** fait passer les régions dans des états **Lab1**, **Term Project** et **Final Test**. Supposons tout d'abord que les événements suivants se produisent :
 - **lab done** : la région du haut passe dans l'état **Lab2**.
 - **project done** : la région de milieu est donc finie.
 - **fail** : alors les trois régions sont terminées et l'état est **Failed** ;
 - sinon (c'est l'événement **pass** qui est survenu) la région du bas est donc finie et dès que l'événement **lab done** surviendra, la région du haut sera finie et l'état sera finalement **Passed**.

Pseudo-état historique

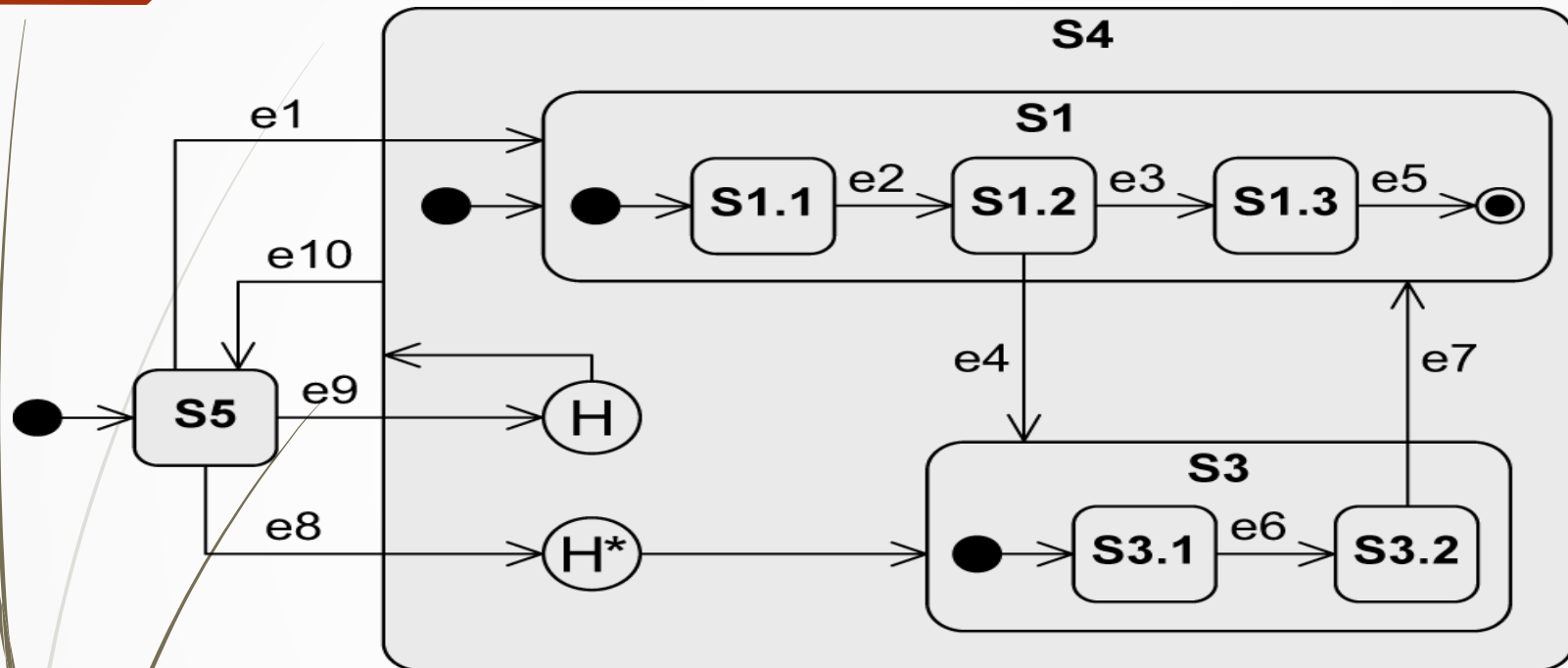
55

- Quand une transition sort d'un état composite :
 - Le dernier sous-état atteint n'est plus connu,
 - Une nouvelle entrée dans l'état redémarre au premier sous-état.
- Un **pseudo-état historique** permet de mémoriser le dernier sous-état atteint.
- Ce **pseudo-état historique** est noté par un H cerclé. (H)
- Une transition ayant pour cible le pseudo-état historique est équivalente à une transition qui a pour cible le dernier sous-état visité dans la région contenant le H.
 - Active le dernier sous-état et toutes les activités d'entrée sont menées séquentiellement de l'extérieur vers l'intérieur de l'état composite
- Un autre pseudo-état, H^* , désigne un **historique profond**, c-à-d un historique valable pour tous les niveaux d'imbrication.

(H^*)

Pseudo-état historique : exemple

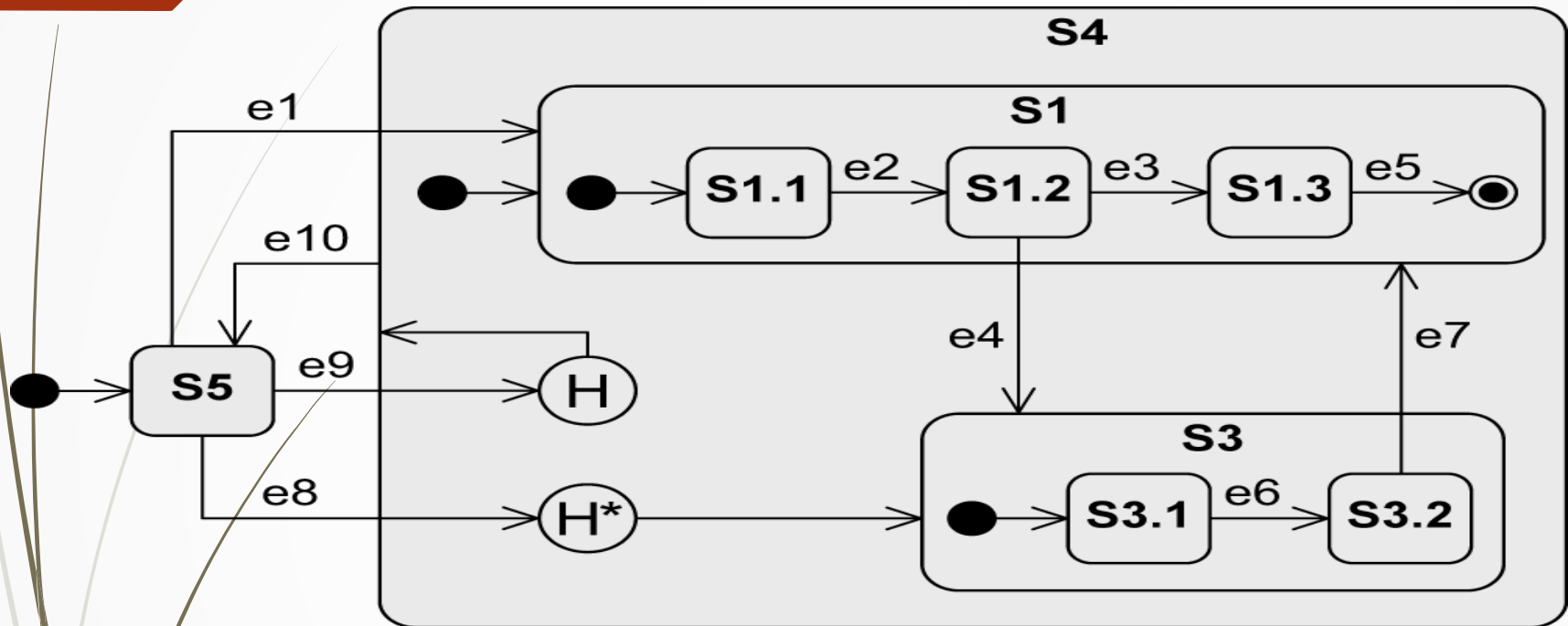
56



Événement	État
Départ	S5
e1	S4/S1/S1.1
e2	S1.2
e10	S5
e9	(H→) S1/S1.1

Pseudo-état historique : exemple

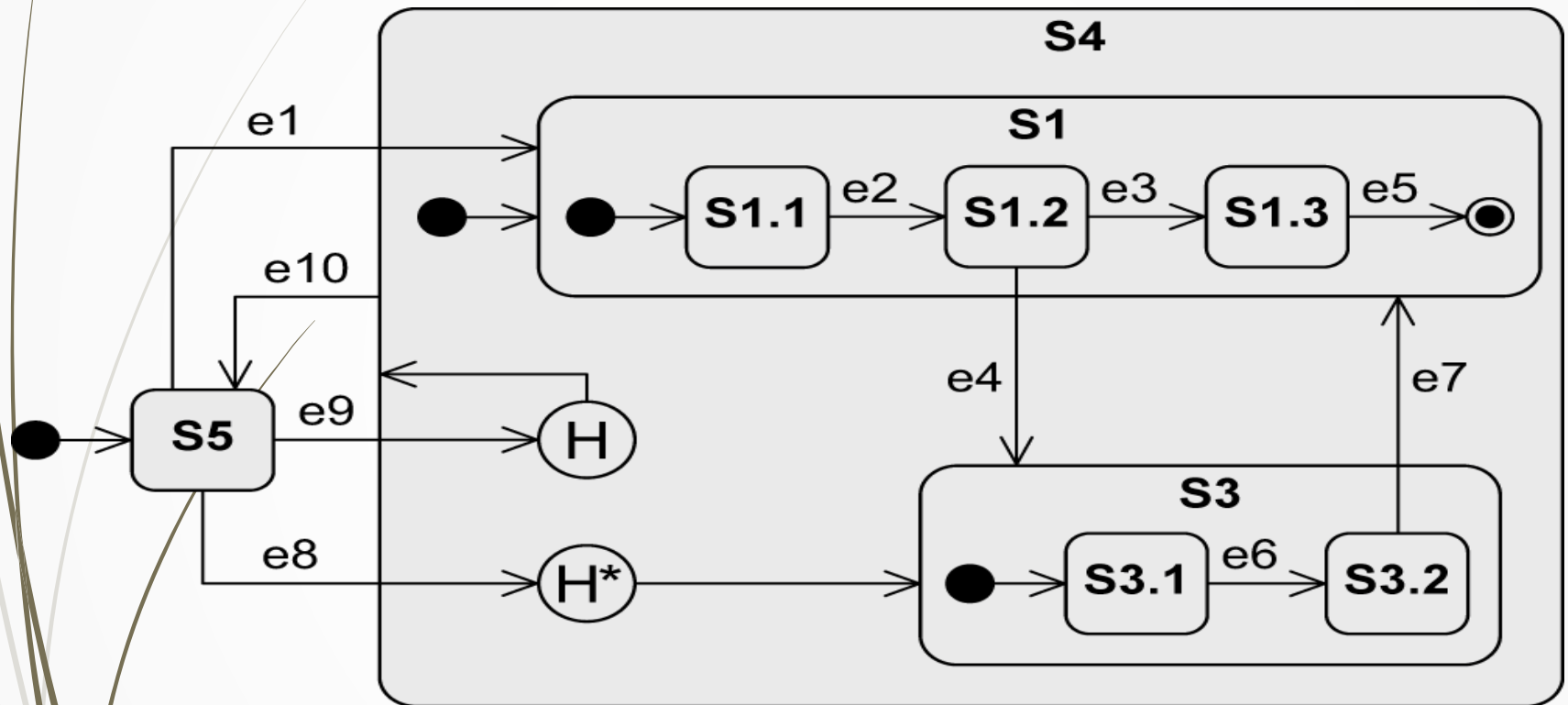
57



Événement	État
Départ	S5
e1	S4/S1/S1.1
e2	S1.2
e10	S5
e8	(H*→)S1.2

Pseudo-état historique : exemple

58



Événement

État

Départ

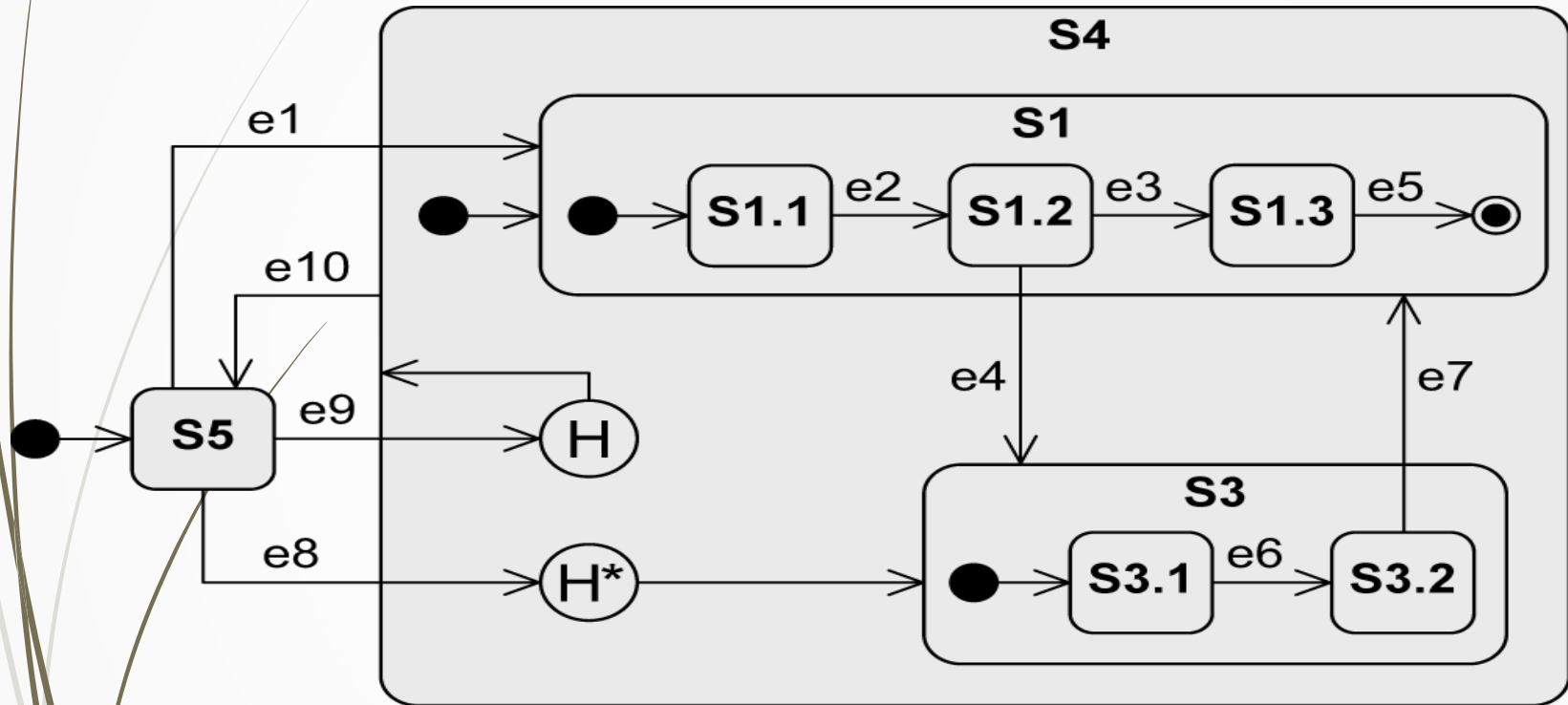
S5

e9

(H→) S1/S1.1

Pseudo-état historique : exemple

59

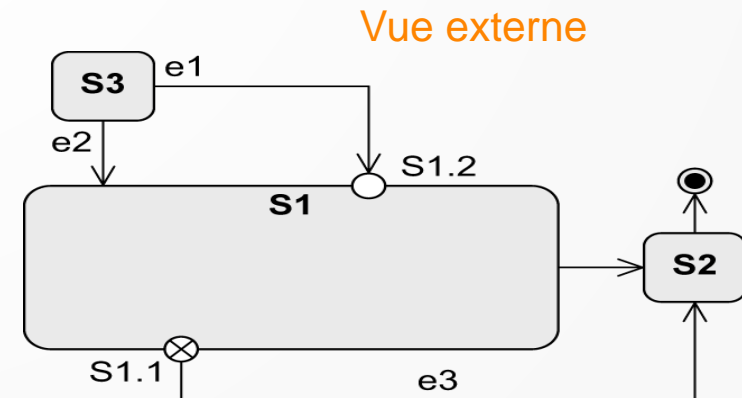
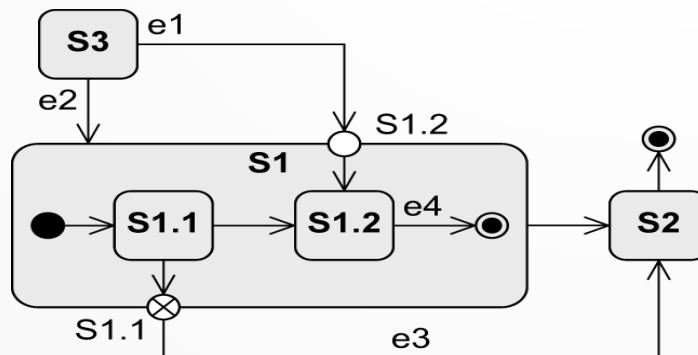


Événement	État
Départ	S5
e8	(H*→) S3/S3.1

Points d'entrée et de sortie

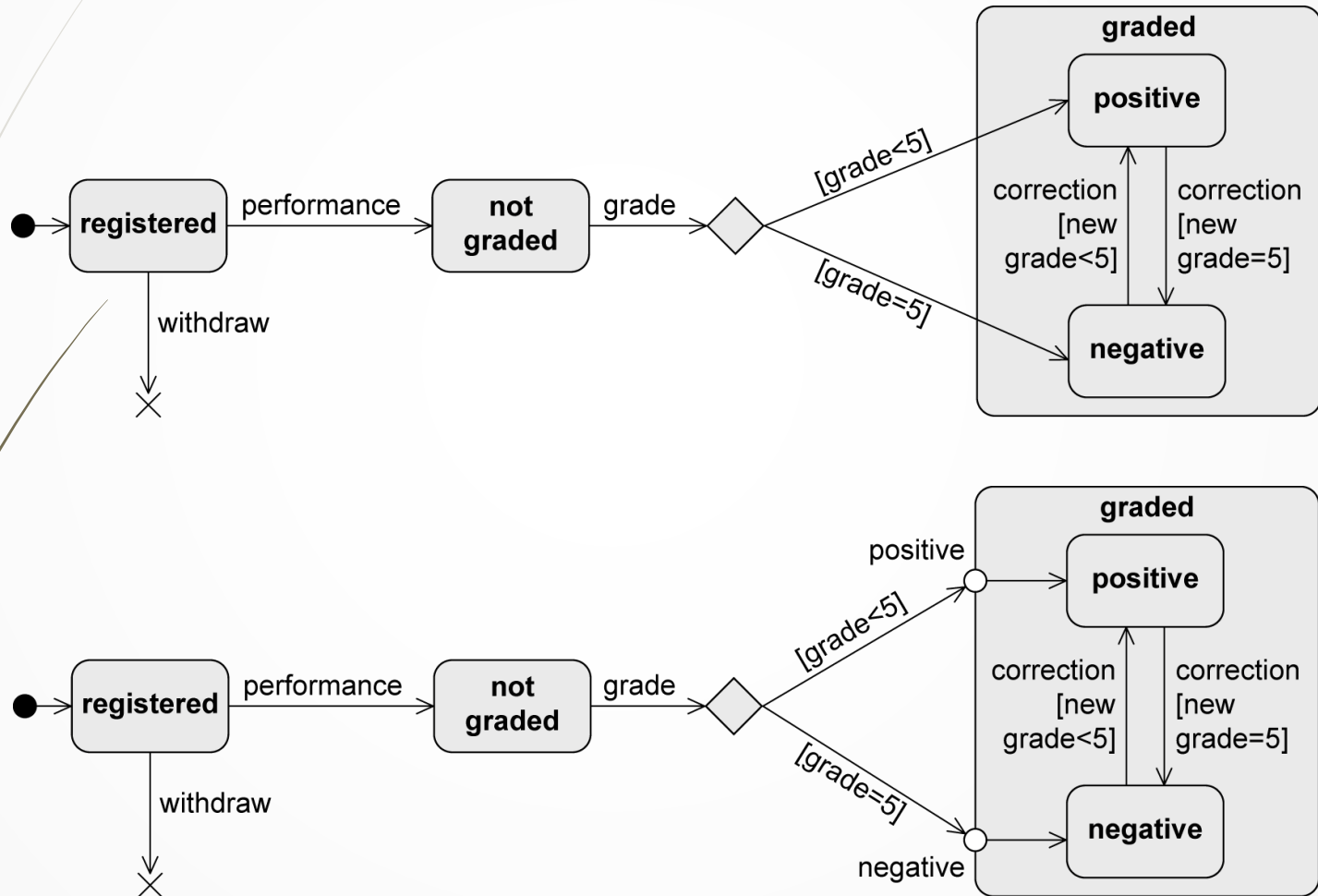
60

- Lorsqu'un état composite peut être entré ou sorti via un état autre que les deux états, initial et final
- Et que la transition externe n'a pas besoin de connaître la structure de l'état composite
- On utilise les points d'entrée et de sortie
- Ce sont des sous-états de l'état composite et sont représentés par un cercle
 - Avec un X pour les points de sortie
 - Vide pour les points d'entrée
- Ces interfaces permettent d'abstraire les sous-états des états composites (réutilisabilité).



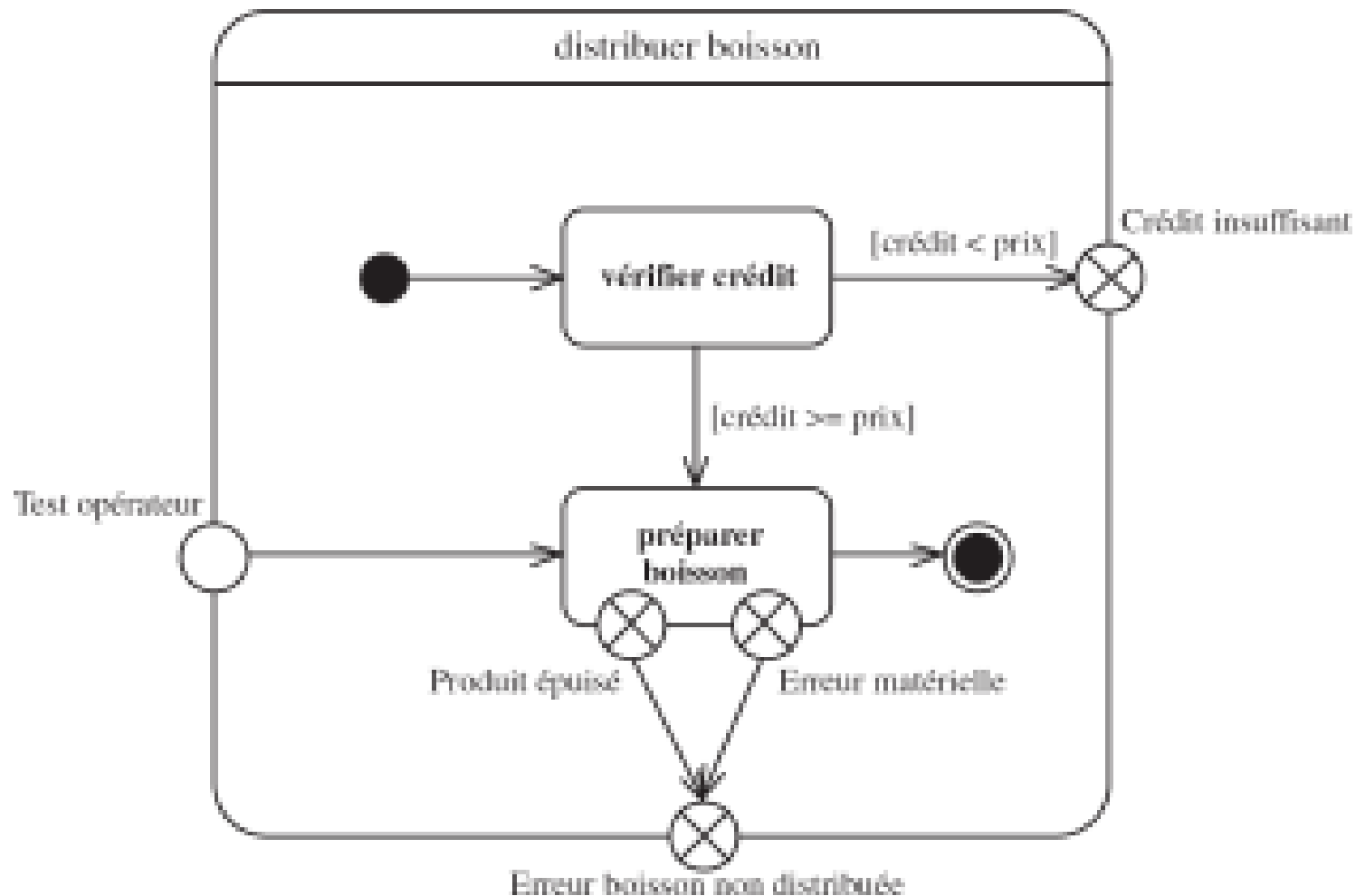
Points d'entrée et de sortie : exemple

61



Points d'entrée et de sortie : exemple

62



Utilisation des diagrammes états-transitions

63

- En phase d'analyse :
 - Description de la **dynamique du système** vu de l'extérieur
 - Synthèse des scénarios liés aux cas d'utilisation
 - Événements = action des acteurs
- En phase de conception :
 - Description de la **dynamique d'un objet particulier**
 - Événements = appels d'opérations

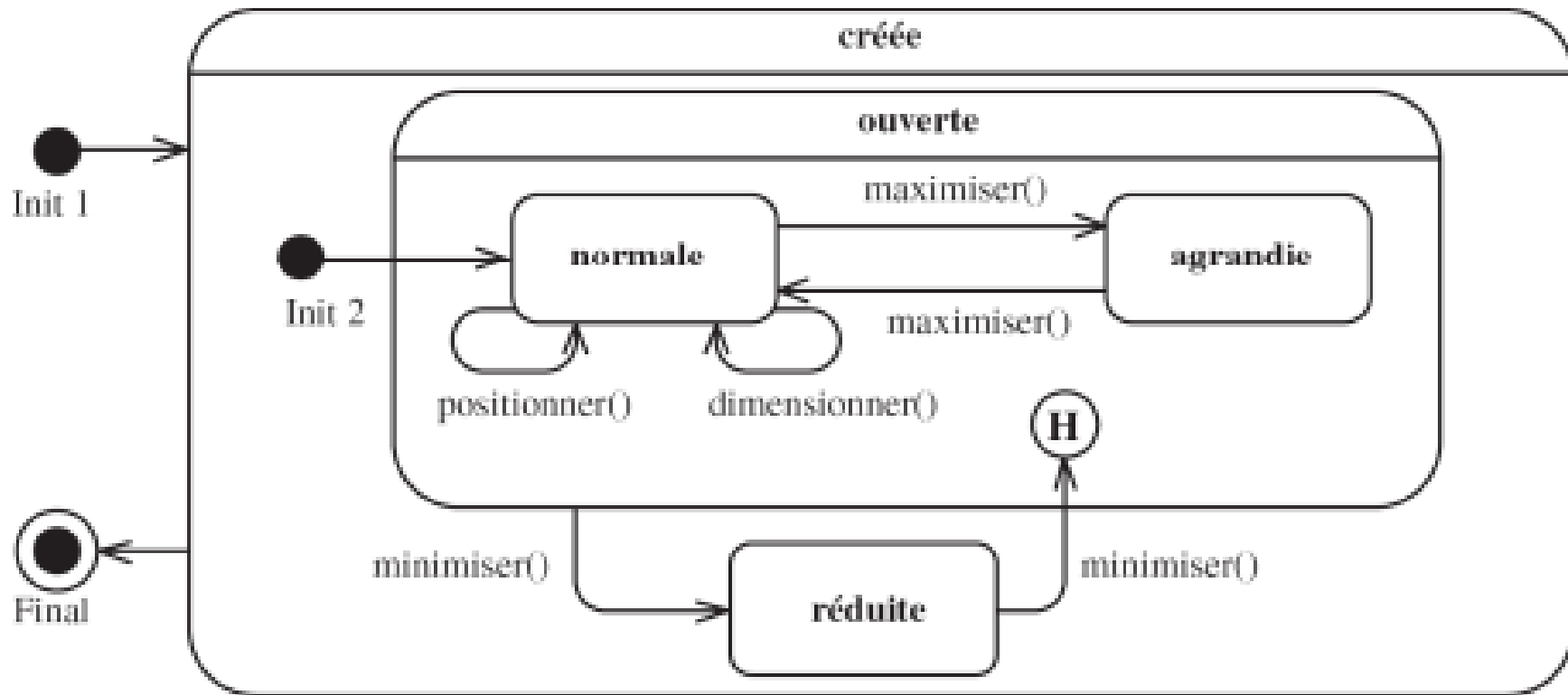
Exemple : fenêtre d'ordinateur

64

- Une fenêtre d'application peut être dans trois états :
 - Normale, Agrandie ou Réduite
 - A l'ouverture, elle est créée dans son état normal.
 - Elle peut alors être déplacée et re-dimensionnée.
 - Lorsqu'elle est agrandie, elle occupe toute la surface de l'écran.
 - Lorsqu'elle est réduite, elle est représentée par une icône dans la barre des tâches.
 - A l'état agrandie ou réduite, elle ne peut être ni déplacée ni re-dimensionnée

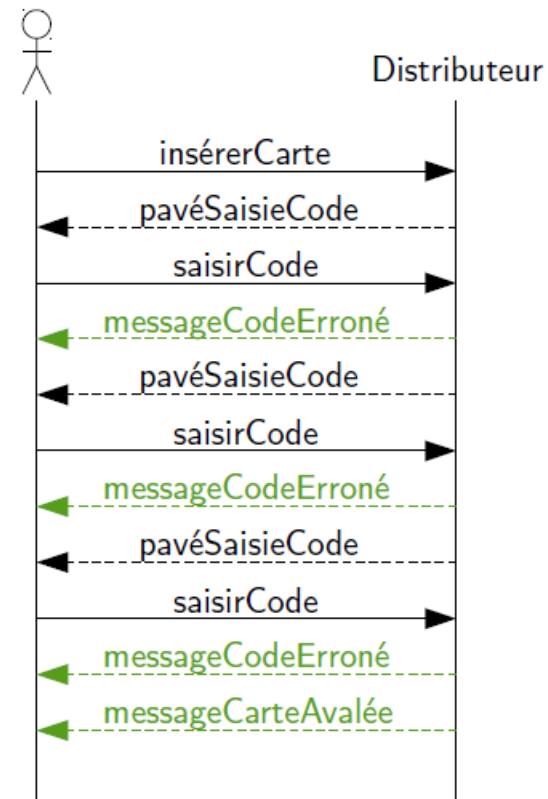
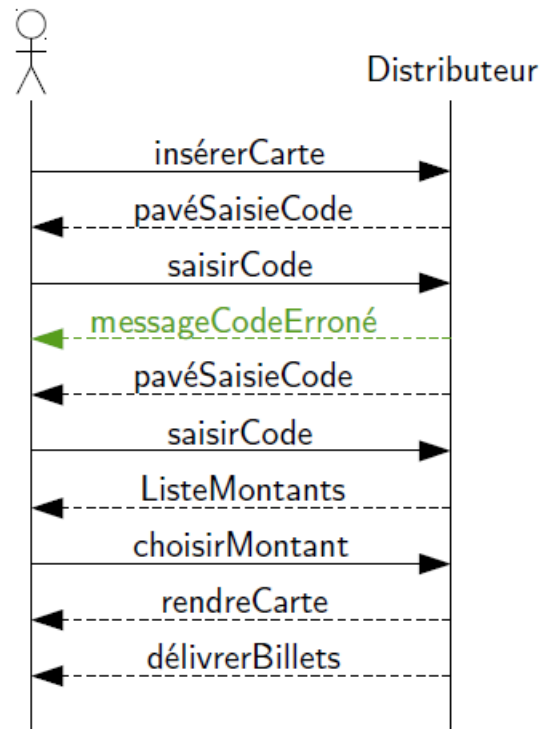
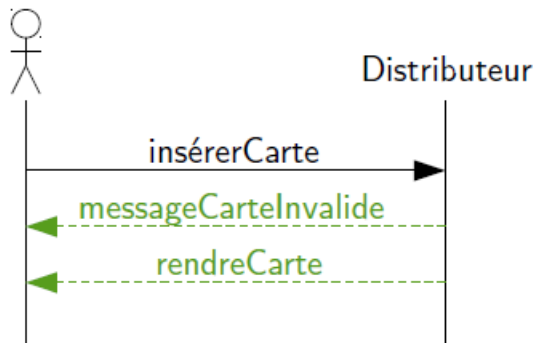
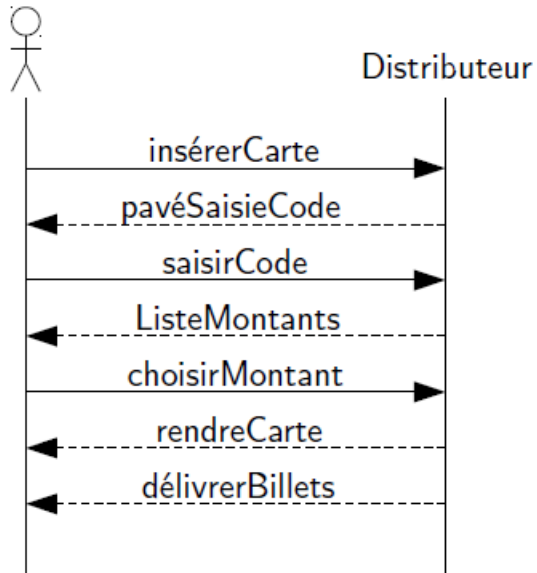
Exemple : fenêtre d'ordinateur

65



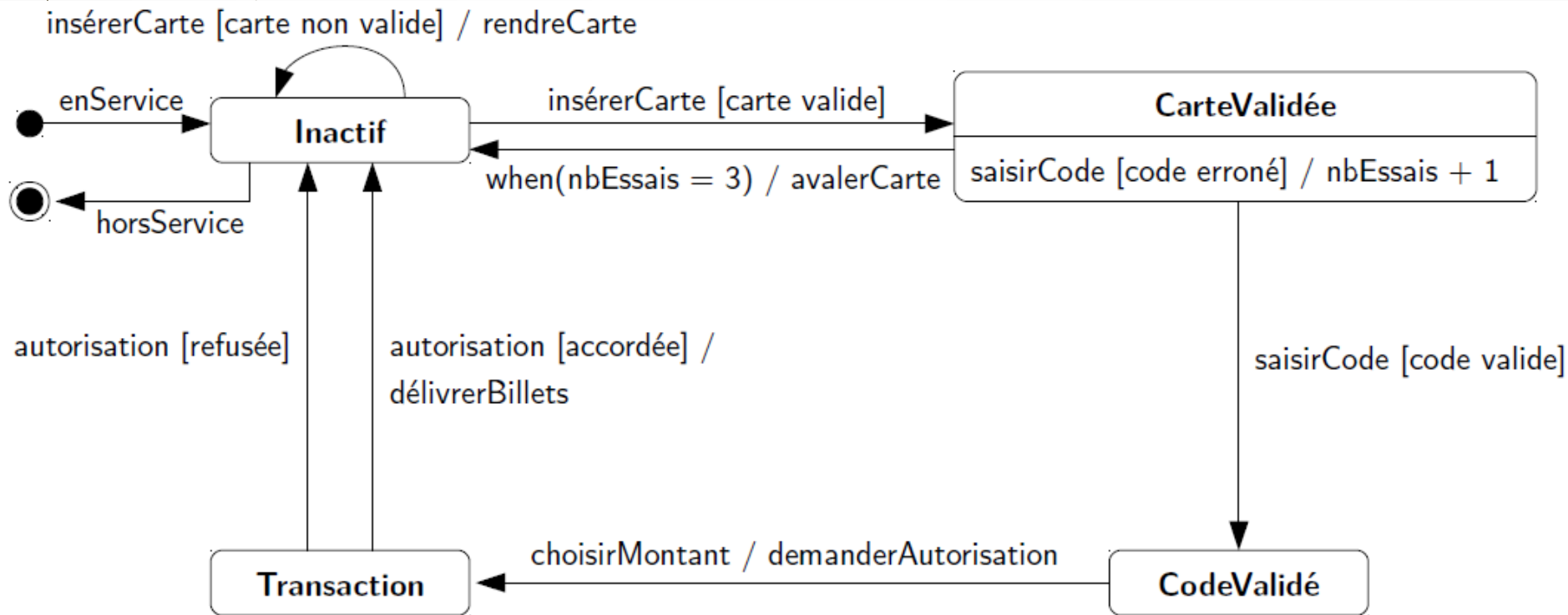
Exemple : DAB

66



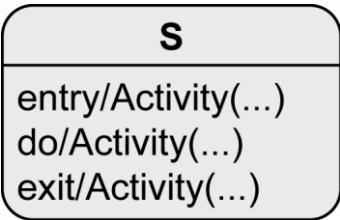
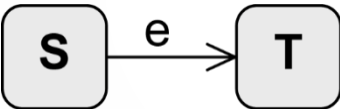



Exemple : Diagramme d'états correspondant

67



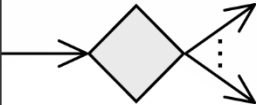
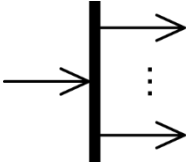
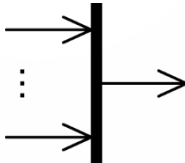
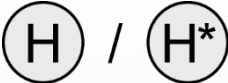
Notation

68

Nom	Notation	Description
Etat		Description d'un « intervalle de temps » spécifique dans lequel se trouve un objet au cours de son "cycle de vie". Au sein d'un État, les activités peuvent être exécutées par l'objet.
Transition		transition depuis un état source S à un état cible T lorsqu'un événement e se produit
Etat Initial		Début du diagramme d'états-transitions
Etat Final		Fin du diagramme d'états-transitions
Etat de terminaison		Fin de l'état d'un objet dans un diagramme d'états-transition

Notation

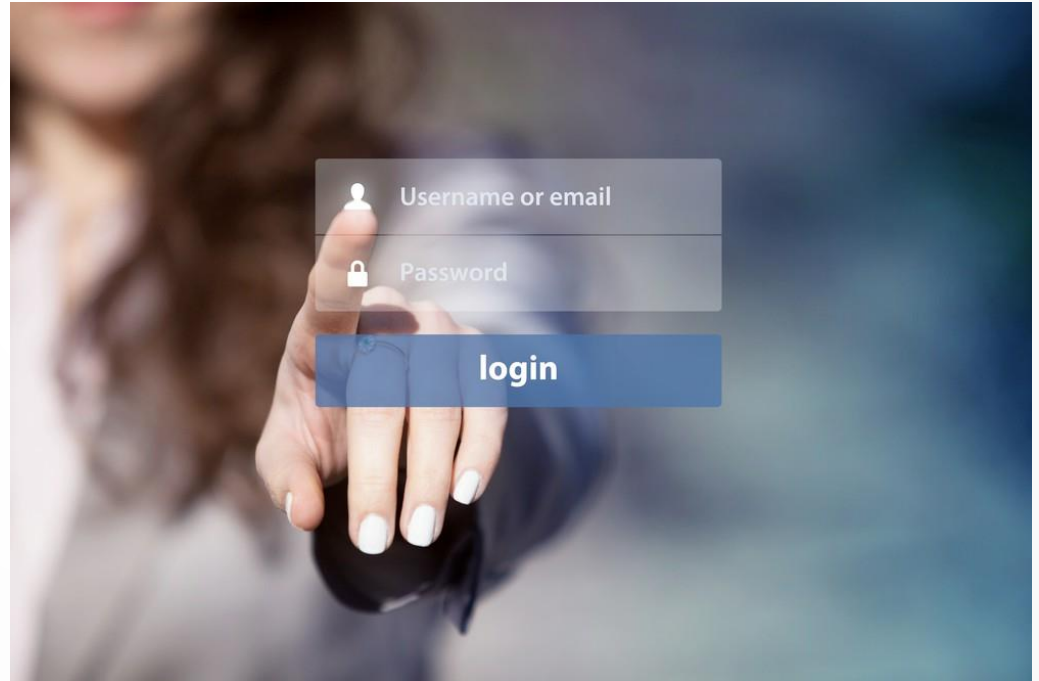
69

Nom	Syntaxe	Description
Nœud de décision		Nœud depuis lequel plusieurs transitions alternatives peuvent partir
Nœud de Parallélisation(fork)		Fractionnement d'une transition en plusieurs transitions parallèles
Nœud de synchronization (join)		Fusion de plusieurs transitions parallèles en une seule transition
Etat historique / Historique profond		« Adresse de retour » vers un sous-état ou un sous-état imbriqué d'un état composite

Etude de cas « Contrôle d'accès »

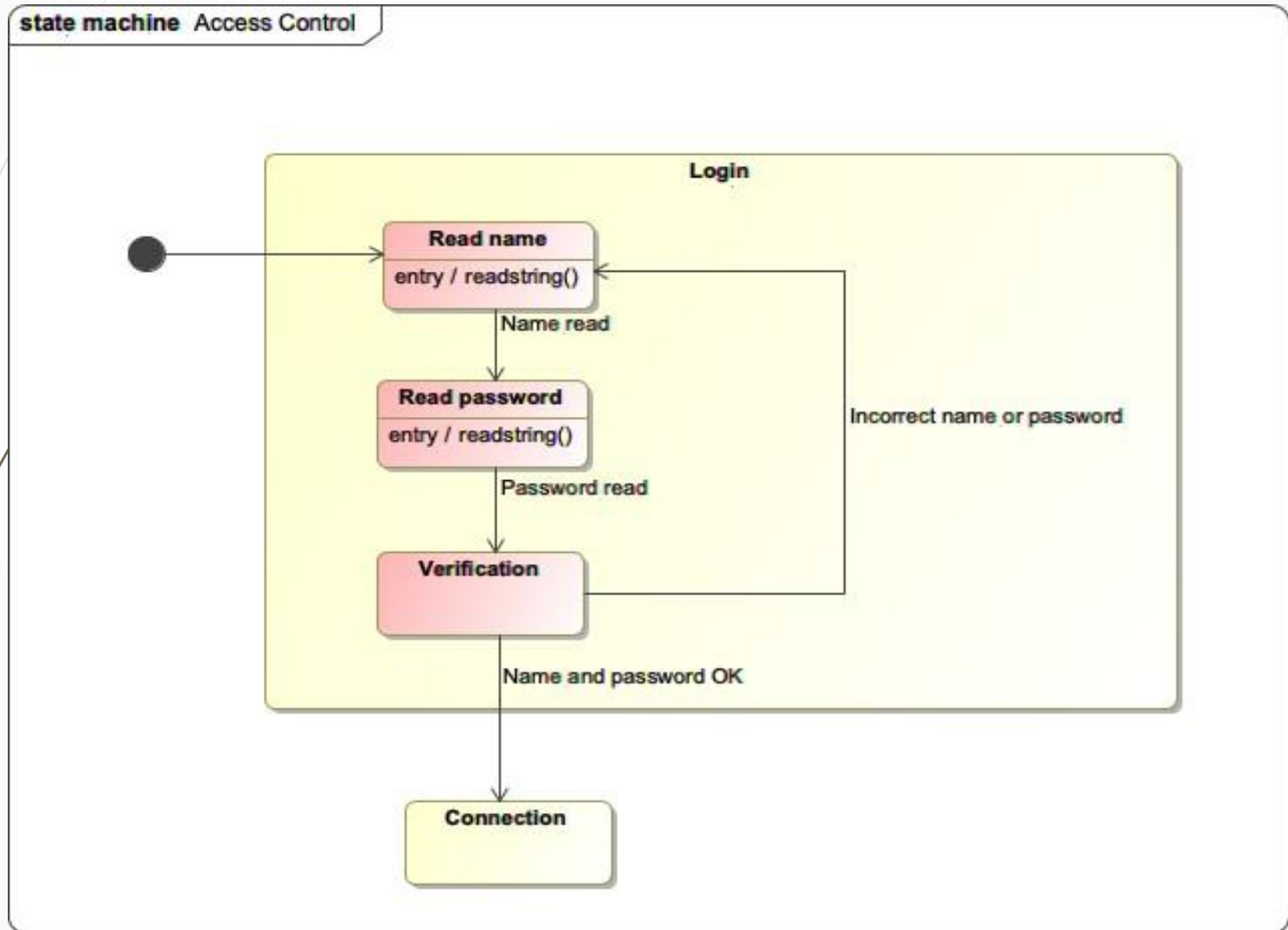
70

- Élaborons le diagramme d'états-transitions pour un contrôle d'accès par log in.



Etude de cas « Contrôle d'accès »

71



Etude de cas « Machine à laver »

72

- Élaborons le diagramme d'états-transitions pour une machine à laver, qui lave, rince et essore les vêtements. Notez que, en cas de coupure de courant pendant son fonctionnement, lorsque le courant est rétabli, il continue à fonctionner dans le même état dans lequel elle était avant la panne de courant.



Etude de cas « Machine à laver »

73

