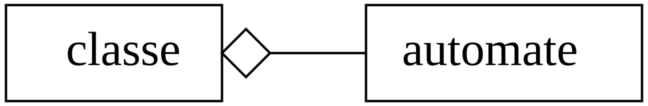
Les diagrammes d'états-transitions

➜ Visualisent des automates déterministes[1].

➜ On relie l'automate à la classe considérée. On ne représente pas les automates des objets qui ne changent pas (ou peu) d'état.



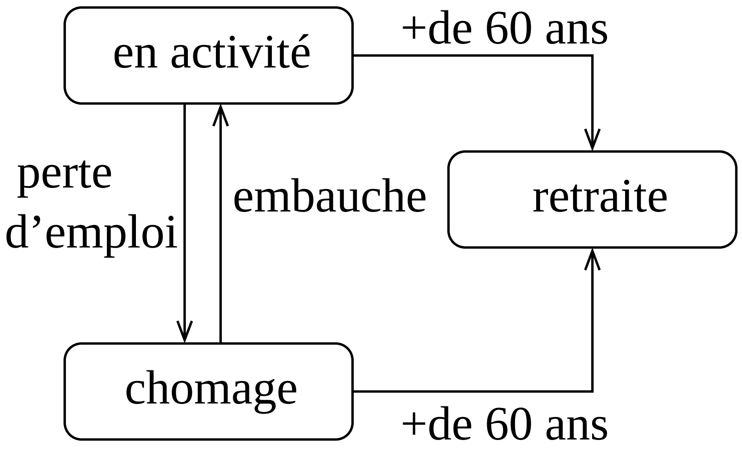
[1] formalisme de Harel, D. 1987. *Statecharts : a Visual Formalism for Complex Systems*. Science of Computer Programming vol 8.

Les diagrammes d'états-transitions - 2

- Un objet est à tout moment dans un état donné.
- L'état d'un objet est constitué des valeurs instantanées de ses attributs.

Les diagrammes d'états-transitions - 3

- L'objet passe d'un état à un autre par les transitions.
- Déclenché par un événement, les transitions permettent le passage d'un état à un autre instantanément.



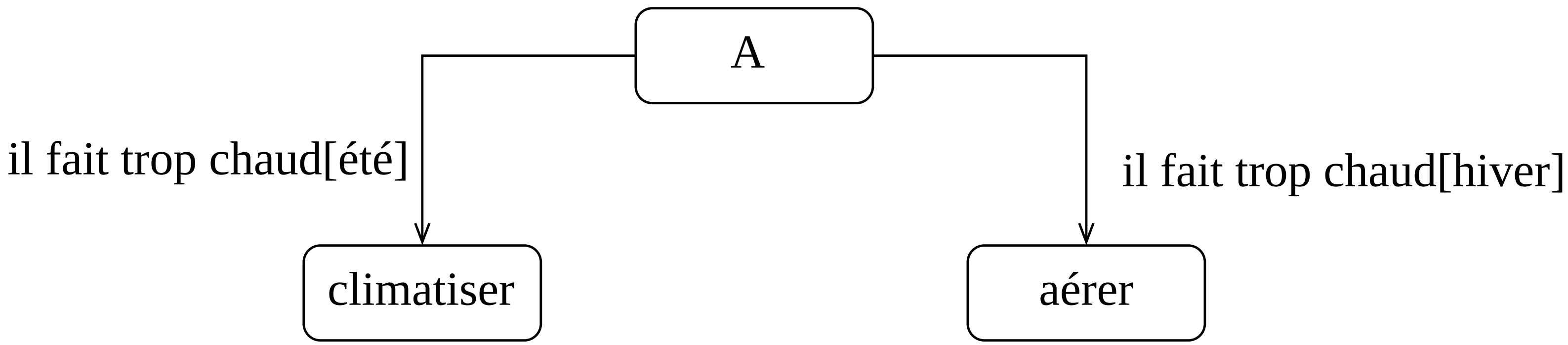
Les événements

- La syntaxe d'un événement dans un diagramme est la suivante :
  Nom_événement (Nom_paramètre : type, ...)[condition]

  condition est la garde qui valide ou non le déclenchement d'une transition quand l'événement s'est produit.

- On peut associer à chaque transition une **action** à exécuter lors du franchissement dû à un événement. Les spécifications de l'action sont contenues dans l'objet destinataire.

- **Exemple** : l'événement `il fait trop chaud` entraîne la climatisation ou l'ouverture des fenêtres selon la saison.

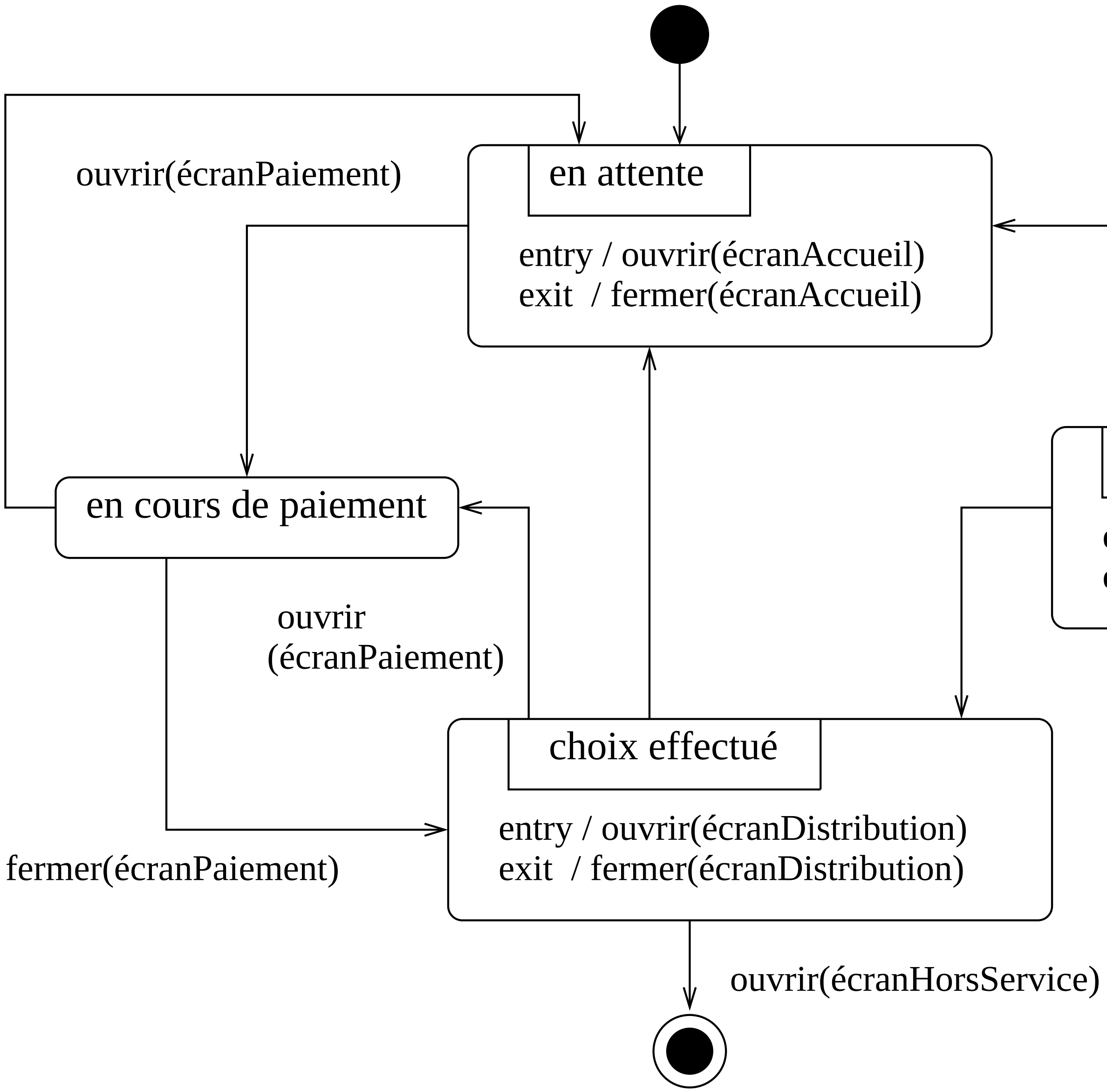


Les événements

➔ Il est possible de préciser les actions à exécuter lorsque l'on est dans un état donné, en entrant ou en sortant.
Pour cela UML donne plusieurs mots-clés :

- **entry** : action à exécuter dès l'entrée dans l'état.
- **exit** : action à exécuter lors de la sortie de l'état.
- **on** : action interne provoquée par un événement qui ne provoque pas le passage à un nouvel état.
- **do** : activité à exécuter (une activité est une action dont le temps d'exécution est non négligeable).

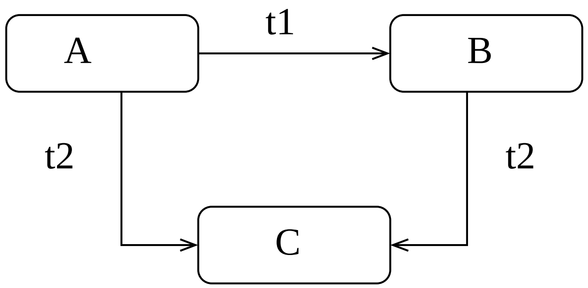
Exemple d'un diagramme d´états

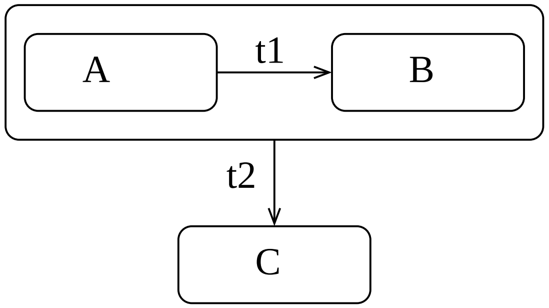


Généralisation d'états

- Pour remédier au problème de l'explosion du nombre des états et de leur connexions, il est possible de définir des **super-classes** d'états et des **sous-classes** qui en héritent. La démarche d'abstraction est identique à la **généralisation/spécification** des classes.

- Un état peut-être décomposé en plusieurs **sous-états disjoints** (ou-exclusif), un objet ne peut-être que dans un seul état à la fois.





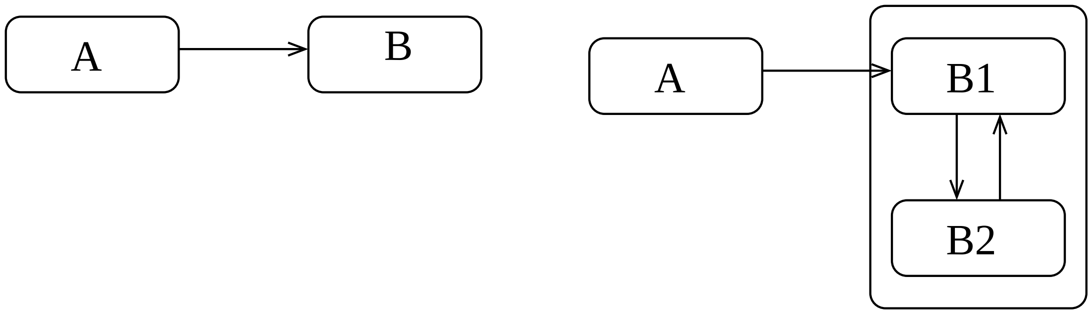
Généralisation d'états - 2

- Un sous-état hérite des variables d'états et de transitions externes de sa super-classe.

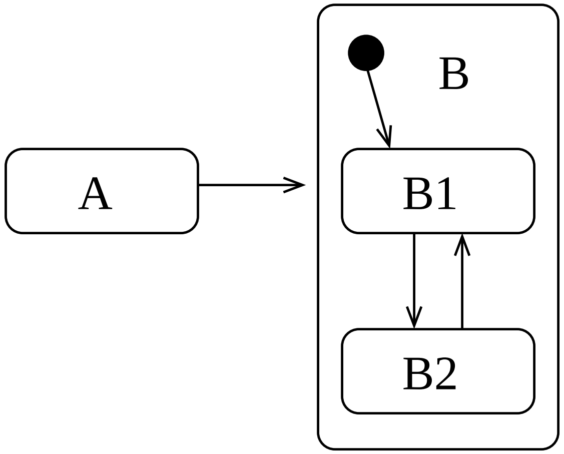
- Un seul état (le super-état ou un des sous-états) hérite des transitions d'entrée car un seul état peut-être la cible d'une transition.

- Si la décomposition a pour objectif de définir un état particulier pour le traitement d'une transition interne, cette transition ne fera pas l'objet d'un héritage.

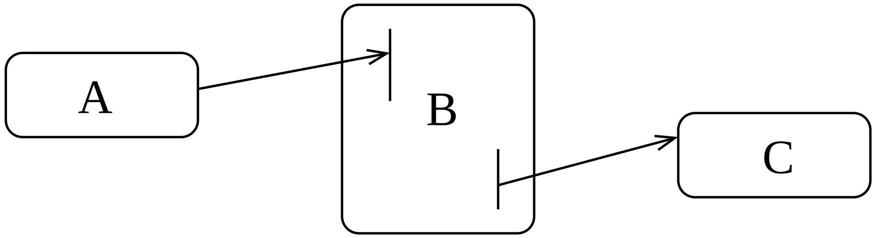
Cas d'une décomposition qui rompt la barrière d'asbtraction :



Solution

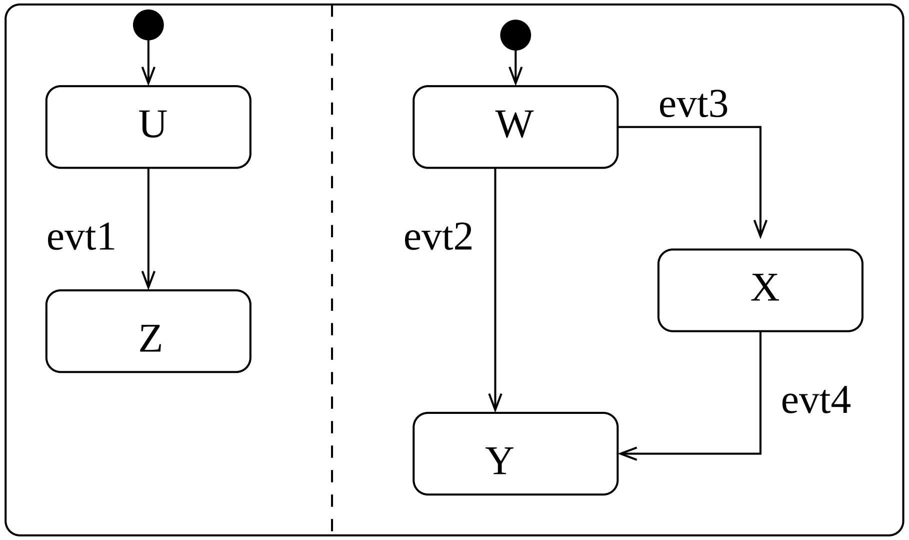


Représentation simplifiée

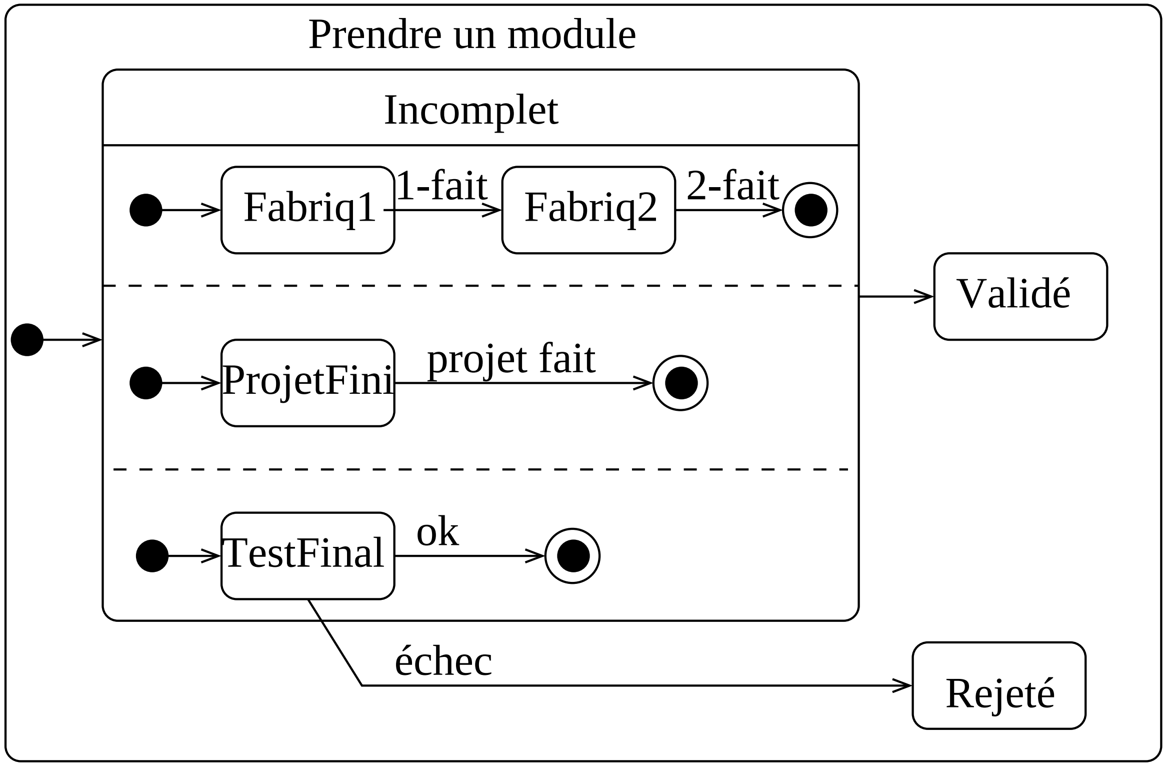


Agrégation d'états

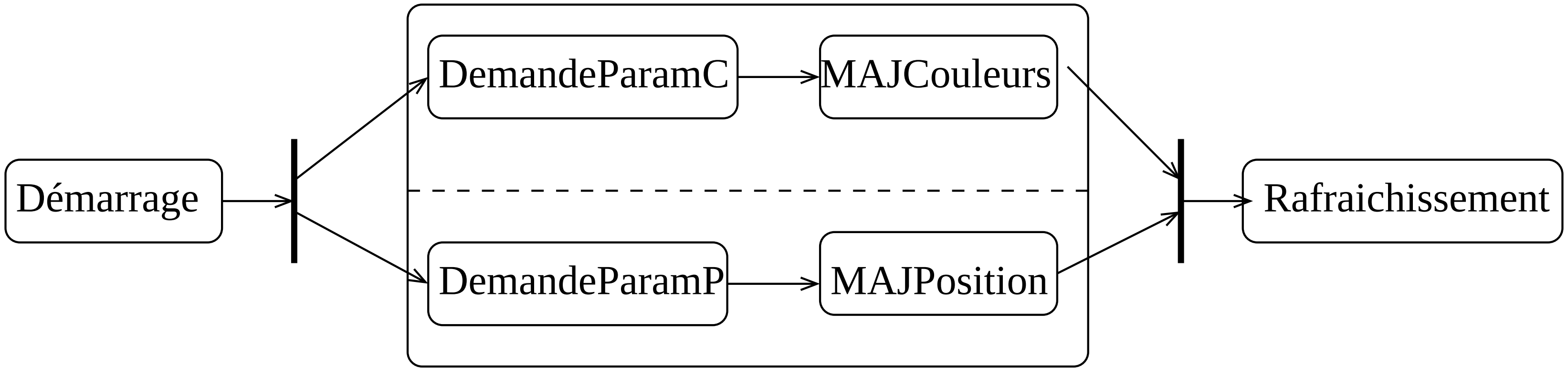
➔ L'agrégation d'états est un état composé de plusieurs automates qui évoluent simultanément et indépendamment.



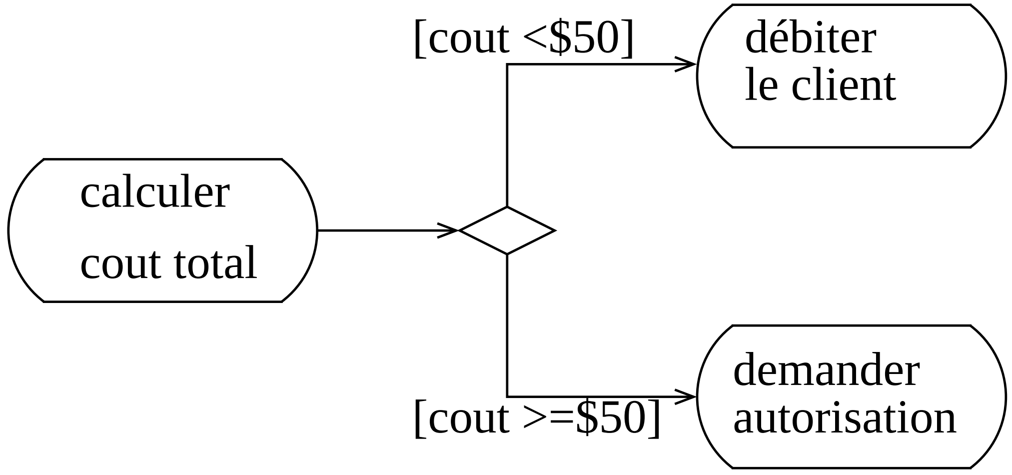
Agrégation d'états - 2



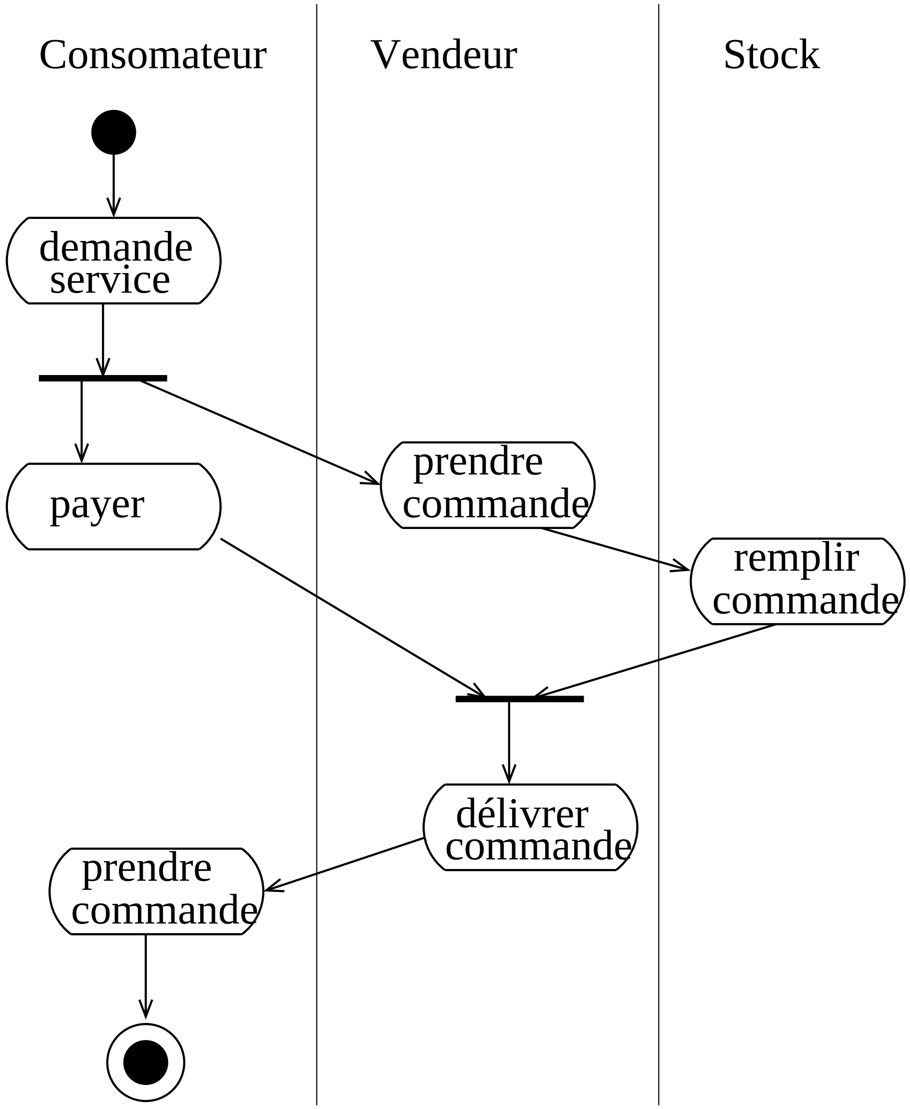
Transitions complexes



Décisions


[cout <$50]
débiter
le client
calculer
cout total
demander
autorisation
[cout >=$50]

Diagrammes d'activités



Action et Objets