Bases de la conception « orientée objet »

DUT Informatique
Semestre 2

Mourad Ouziri
mourad.ouziri@parisdescartes.fr

IUT de Paris Descartes

PARIS DESCARTES IUT

PARIS DESCARTES

Conception objet :

« Diagramme d'états-transitions UML »

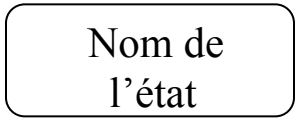
Objectifs :

- Schématiser le comportement attendu des objets de l'application
- Etablir un modèle de tests unitaires du comportement des objets

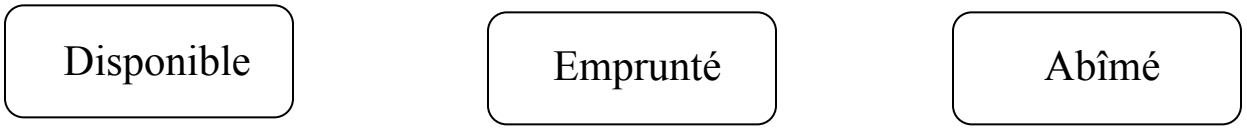
UML

DE – Diagramme d'états-transitions

- Objectif
  - Décrire le comportement des objets d'une classe
- Définition
  - Automate déterministe à états finis représentant les états pertinents d'un objet et les transitions faisant évoluer cet objet d'un état à un autre
- État
  - Situation durable d'un objet
  - Conjonction instantanée des valeurs des attributs et associations de l'objet
- Syntaxe graphique



- Exemple : quelques états d'un exemplaire de livre



UML

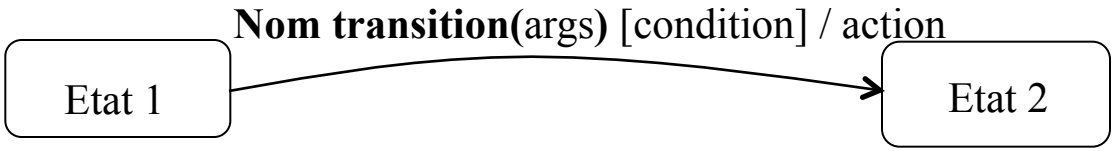
DE – Diagramme d'états-transitions

- Transition
  - Passage instantané d'un état à un autre (ou dans le même état)
  - Elle est déclenchée par un événement
  - Deux types d'événements :

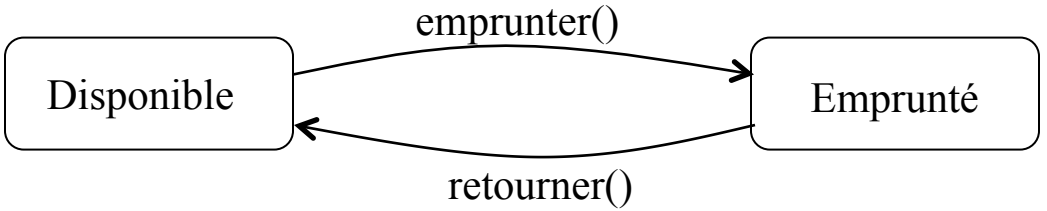
    Message : réception d'un message par l'objet concerné (méthode dans la classe correspondante)

    Temporel : correspond à un instant donné : *when(attr=10h)*, après un certain délai : *after(3jours)*
  - Elle peut être conditionnée : la transition n'est déclenchée que si la condition est vérifiée

- Syntaxe graphique



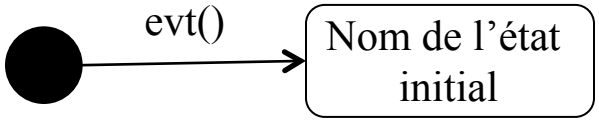
- Exemple



UML

DE – Diagramme d'états-transitions

- États particuliers
  - État initial

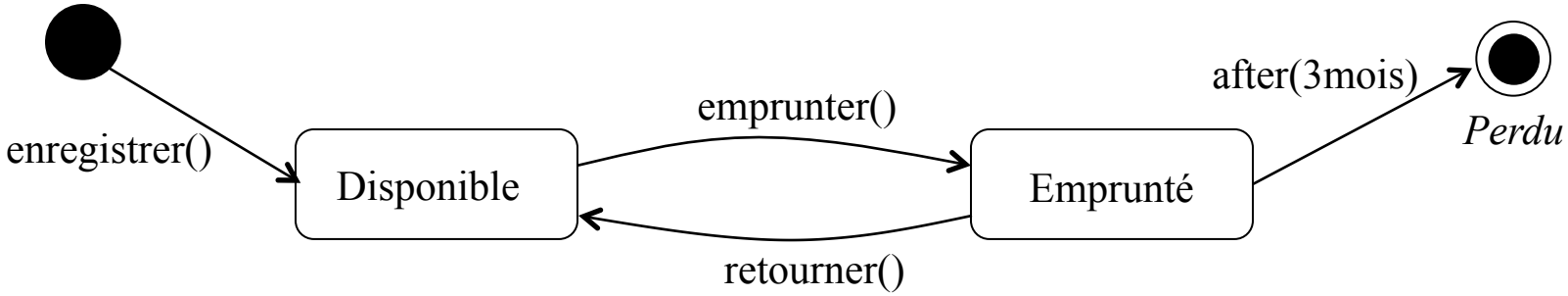


Premier état dès création d'un objet. Il est obligatoire et unique.

  - État final

État représentant la destruction de l'objet. Il est facultatif et peut être multiple.

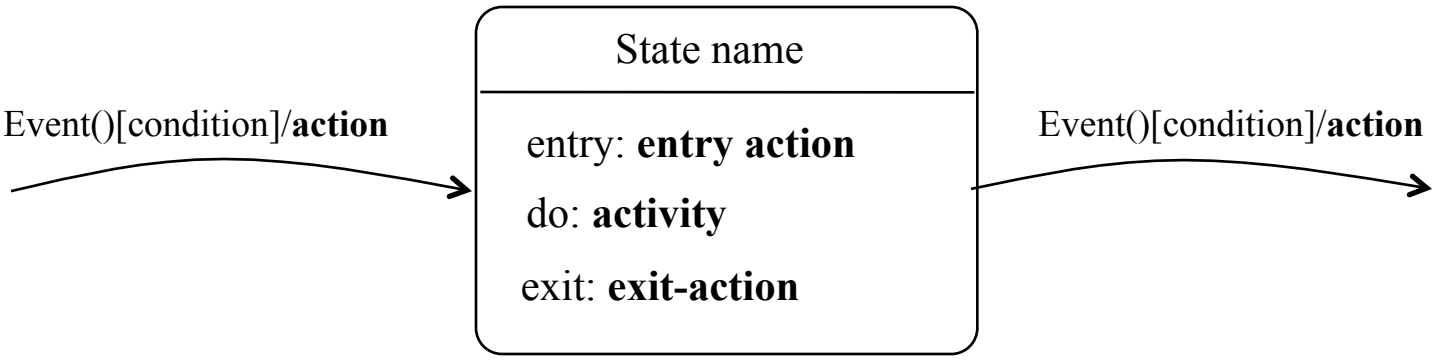
- Exemple



UML

DE – Diagramme d'états-transitions

- Action
  - Traitement instantané accompagnant une transition
  - Il peut être exécuté avec la transition, à l'entrée ou à la sortie d'un état
- Activité
  - Traitement nécessitant un certain temps d'exécution
  - Il est exécuté lorsque l'objet rentre dans l'état et interrompu à la sortie de l'état
- Syntaxe graphique



UML

DE – Diagramme d'états-transitions

- État composite ou généralisation d'états
  - Élément de structuration des diagrammes d'états-transitions
  - C'est un pseudo-état qui regroupe plusieurs états
  - Il permet de simplifier les diagrammes d'états
- Un état composite est créé à partir des :
  - États partageant les mêmes propriétés
  - États partageant les mêmes transitions d'entrée et de sortie
- Historique dans un état composite
  - Lorsqu'un objet rentre de nouveau dans un état composite, il peut retrouver son dernier (sous-)état de sortie
  - Le dernier état de sortie peut donc être mémorisé (symbole H dans l'état composite)
- Exemple

DE

Tests unitaires

- DE : modèle de comportement des objets d'une classe
- Tests unitaires : tests à l'échelle d'une classe
- Etat d'un objet
  - Situation durable de l'objet, obtenue à partir des valeurs de ses attributs
  - Valeur subjective et dépend du contexte de l'objet
- Indépendance de la classe testée du diagramme d'état
  - Possibilité de tester la classe avec différents diagrammes d'états
- Un diagramme d'états : une classe de tests
  - Les états du DE sont définis par la classe de tests associée
  - Les transitions sont testées dans cette même classe de tests associée

DE

Tests unitaires

- Classe de tests : tester deux types de transitions
  - Transitions possibles : définies dans le DE
    - Transition initiale : instanciation de l'objet
    - Transitions non initiales : toute autre transition définie dans le DE
  - Transition non possibles : celles non définies dans le DE
- Structure d'une classe de test DE

```
class TestClassseAvecDE1 {
    private Etat getEtat (Classe o) {
    }
    @Test
    public void testTransition1 () {
    }
}
```

```
enum Etat {
        Etat1,
        Etat2,
        …
        Etatn
}
```

DE

Tests unitaires

☞ Structure de test d'une transition définie

```
class TestClassseAvecDE1 {
    @Test
    public void testTransitionDéfinie1 () {
            //  GIVEN : état de départ
            //  WHEN : transition
            //  THEN : état final
            //  CLOSE : remise à zéro {optionnel}
    }
}
```

DE

Tests unitaires

☞ Structure de test d'une transition non définie

- L'objet se trouve forcément dans un état (de départ)
- La transition non définie passera l'objet dans un état indéfini

```
class TestClassseAvecDE1 {
    @Test (expected=UneException.class)
    public void testTransitionNonDéfinie1 () throws UneException {
                        //  GIVEN : état de départ
                        //  WHEN : transition
    }
}
```