L'approche Orientée Objet et UML

UML

Rémy Courdier

UML

Les diagrammes de modélisation

email : Remy.Courdier@univ-reunion.fr

Plan du cours

UML

- Introduction au Génie Logiciel
- L'approche Orientée Objet et Notation UML
- **Les diagrammes de modélisation**
- Relations entre les différents diagrammes
- De l'analyse à la conception
- Relation entre les notations OMT et UML

Chapitre 3 : Les diagrammes de modélisation

UML

- Les diagrammes d'objets
- Les diagrammes de collaboration
- Les diagrammes de classes
- Les diagrammes de cas d'utilisation
- Les diagrammes de séquence
- Les diagrammes d'états-transitions
- Les diagrammes d'activités
- Les diagrammes de composants
- Les diagrammes de déploiement

Moyens de visualiser et de manipuler des éléments de modélisation

3.1Diagrammes objets, de collaboration et de classes

UML

- Les diagrammes d'objets (ou d'instances)
  - ✓ représenter les objets et leurs relations sans représenter les envois de messages (représentation statique).
  - ✓ permettre la compréhension générale du système
  - ✓ faciliter la compréhension des structures complexes (récursives)
- Les diagrammes de collaboration
  - ✓ correspondent à une extension des diagrammes d'objets.
  - ✓ représentation de la structure spatiale statique qui permet la mise en relation d'un groupe d'objets
  - ✓ représentation des interactions entre objets.
- Les diagrammes de classes
  - ✓ représente la structure abstraite statique en terme de classe et de relations
  - ✓ description abstraite des liens potentiels d'un objet vers d'autres objets

Petits conseils pour un schéma UML efficace

UML

- ***Lorsque l'on passe à la conceptualisation d'une application de taille, on garde souvent les mauvaises habitudes datant de diagrammes plus simples. Voici quelques règles pour réaliser des diagrammes clairs et lisibles par tous.***

  - ✓ **Placer avant de relier**
    Les petits projets n'ont qu'un temps, et il faut rapidement perdre l'habitude de construire son diagramme UML au fur et à mesure, à associer deux classes par une ligne dès que celles-ci sont créées. Bien au contraire, l'utilité d'UML étant de vous permettre de réfléchir à votre application avant de vous lancer dans sa réalisation, il faut prendre (et non "perdre") son temps à faire la liste de tous les éléments à placer dans le diagramme, ensuite seulement les placer de manière logique dans le diagramme, et enfin, en tout dernier, associer les éléments entre eux.

Petits conseils pour un schéma UML efficace (2)

UML

- ✓ **Ne pas croiser les associations**
  L'association entre deux éléments est primordiale à la bonne compréhension d'un diagramme. Autant que faire se peut, il faut éviter de placer les éléments de telle sorte que leur association puisse se croiser : un ensemble d'associations clairement distinctes permet une lecture beaucoup plus rapide de l'ensemble du diagramme.
  Dans le cas où deux associations doivent se croiser, il faut bien marquer la distinction entre celles-ci : au point de croisement, l'une d'entre elles doit faire un "saut" par-dessus l'autre, indiquant ainsi qu'elle ne sont pas liées.

- ✓ **Mettre des codes couleurs**
  Même avec des éléments bien disposés et des associations distinctes, le diagramme d'une grosse application peut facilement devenir illisible à l'oeil humain (sans pour autant gêner sa lecture par un logiciel). Pour ne pas être le seul à savoir lire votre diagramme, prenez le temps de marquer selon différentes couleurs les éléments d'une même catégorie, ou à marquer d'une couleur vive les éléments principaux de votre diagramme. Il deviendra ainsi un bien meilleur outil de travail de groupe.

Conseils Méthodologiques (suite)

UML

- ✓ **Diviser pour mieux régner**
  Il faut savoir rester sobre : dès que vous vous apercevez que votre diagramme a des grandes chances de contenir beaucoup d'éléments dans tous les sens, découpez-le en plusieurs sous-diagramme, auxquels le diagramme principal fera référence. La plupart des outils de conception UML vous permettent de lier deux fichiers UML directement depuis l'interface : n'hésitez pas à y faire appel !
  Accessoirement, cela vous permet de simplifier votre diagramme lors de présentation à vos managers ou à d'autres groupes, probablement peut intéressés par les détails...

- ✓ **Mettre du sens**
  N'oubliez jamais d'utiliser des flèches là où elles sont requises, c'est-à-dire dans une association à sens unique, indiquant qu'un message ne peut être transmis que dans un seul sens. Cela implique de nombreuses choses au niveau de développement, et savoir qu'un élément n'a pas besoin d'émettre des messages, mais seulement d'en envoyer, peut simplifier de beaucoup la réalisation de l'application.

3.2 Les Diagrammes de cas d'utilisation

UML

■ Définition

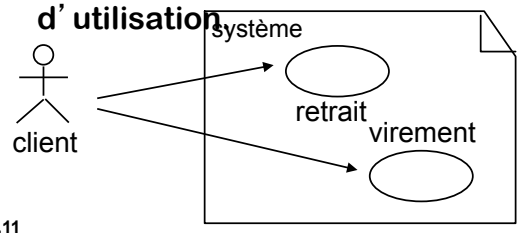
- ✓ **Description sous forme d'actions et de réactions du comportement d'un système du point de vue de l'utilisateur**

■ Objectif

- ✓ **Définir les limites du système à concevoir**
- ✓ **Définir les relations entre le système et l'environnement**
- ✓ **Partitionner l'ensemble des besoins d'un système**

■ Notation UML

- ✓ **Un utilisateur ou acteur est représenté par un petit personnage, un cas d'utilisation ou fonctionnalité du système par un ovale, les flèches issues d'un personnage pointent vers les cas d'utilisation**



Les acteurs

UML

- **Quatre catégories d'acteurs**
  - ✓ **acteurs principaux : utilisent les fonctions principales du système**
  - ✓ **acteurs secondaires : tâches administratives ou de maintenance**
  - ✓ **matériel externe : dispositifs matériels incontournables utilisés**
  - ✓ **autres systèmes : systèmes avec lesquels le système doit interagir**
- **Les 3 types de relations entre acteurs et cas d'utilisation**
  - ✓ **relation de communication : Déclenche**
    - **déclenchement d'un cas d'utilisation par un un acteur**
  - ✓ **relation d'utilisation : Utilise**
    - **un cas d'utilisation comprend le comportement d'un autre cas d'utilisation**
  - ✓ **relation d'extension : Etend**
    - **le cas d'utilisation source étend ou enrichi le comportement du**

Spécification d'un cas d'utilisation

UML

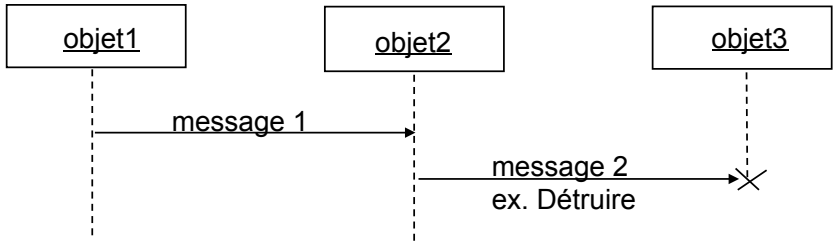
La spécification d'un cas d'utilisation comprend :

- **L'événement qui déclenche le cas d'utilisation**
- **L'événement qui cause l'arrêt du cas d'utilisation**
- **L'interaction entre le cas d'utilisation et les acteurs**
- **Les échanges d'informations**
- **La chronologie et l'origine des informations**
- **Les répétitions de comportements**
- **Les situations optionnelles (Choix a, Choix b, Choix c)**

3.3 Les diagrammes de séquence

UML

- Un diagramme de séquence montre des interactions entre objets selon un point de vue temporel
- La représentation se concentre sur l'expression des interactions
- Notation :



envoi synchrone ⟶ (expéditeur bloqué jusqu'à acceptation du destinataire)

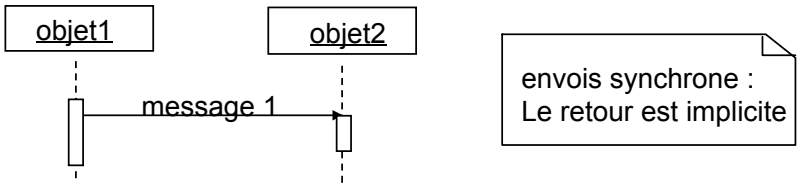
envoi réflexif ⟲ (envoi d'un objet à soi-même)

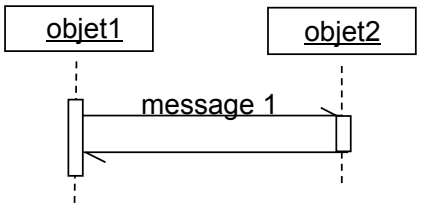
envoi asynchrone ⟶ (n'interrompt pas l'exécution de l'expéditeur)

Période d'activités

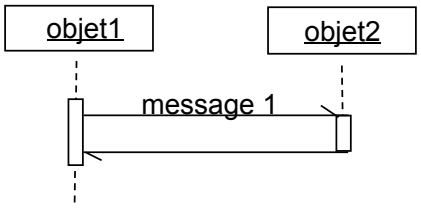
UML

- Une période d'activité correspond au temps pendant lequel un objet effectue une action





Retour explicite d'envoi asynchrone



Retour explicite avant suicide

3.4 Les diagrammes d'états-transitions

UML

Un diagramme d'états-transitions est un automate d'états fini déterministe associé à une classe

- **Abstraction des comportements possibles**
  - ✓chaque objet suit le comportement décrit dans l'automate associé à sa classe, son état caractérise ses conditions dynamiques
- **On associe un tel automate à toute classe qui présente un comportement réactif marqué**
- **Statecharts de D. Harel**
  - ✓D. Harel, "Statecharts : A Visual Formalisme for Complexe Systems", Science of Computer Programming, vol. 8, 1987.
  - ✓Automates hiérarchiques, possédant les concepts d'orthogonalités, d'agrégation et de généralisation

Sémantique du diagramme

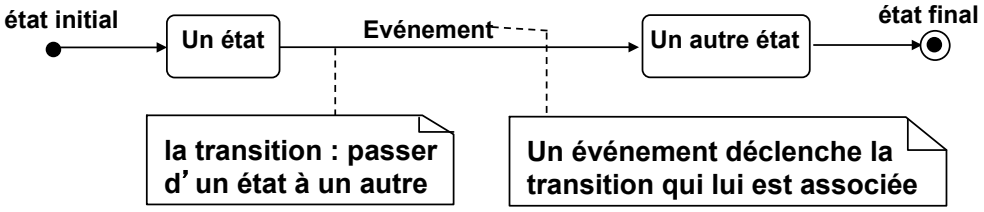
UML

- Ce diagramme sert à représenter des automates d'états finis, sous forme de graphes d'états, reliés par des arcs orientés qui décrivent les transitions.
- Les diagrammes d'états-transitions permettent de décrire les changements d'états d'un objet ou d'un composant, en réponse aux interactions avec d'autres objets/composants ou avec des acteurs.
- Un état se caractérise par sa durée et sa stabilité, il représente une conjonction instantanée des valeurs des attributs d'un objet.
- Une transition représente le passage instantané d'un état vers un autre.
- Une transition est déclenchée par un événement. En d'autres termes : c'est l'arrivée d'un événement qui conditionne la transition.
- Les transitions peuvent aussi être automatiques, lorsqu'on ne spécifie pas l'événement qui la déclenche.
- En plus de spécifier un événement précis, il est aussi possible de conditionner une transition, à l'aide de "gardes" : il s'agit d'expressions booléennes, exprimées en langage naturel (et encadrées de crochets).

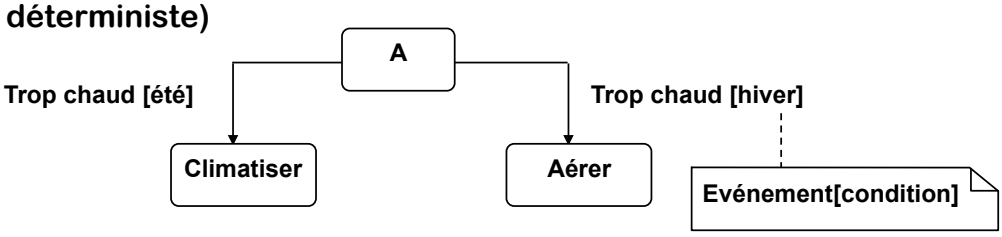
Notations

UML

- **Etat, Transition et Evénement**



- **Les gardes**
  - ✓ Une garde est une condition qui valide ou non le déclenchement d'une transition lors de l'occurence d'un événement
  - ✓ Elles doivent être mutuellement exclusives (aspect déterministe)



Evénements & Opérations, Actions

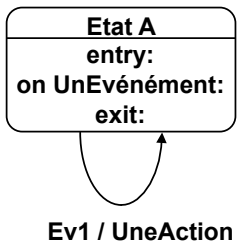
UML

- **Le lien entre Evénements et Opération du diagramme de classe apparaît dans l'automate**
  - ✓ Chaque transition peut être décorée par le nom d'une action à exécuter lorsque la transaction est déclenchée par un événement
  - ✓ L'action correspond à une des opérations déclarées dans la classe de l'objet destinataire de l'événement
- **Les états peuvent contenir des actions**
  - ✓ exécutées à l'entrée ou à la sortie
  - ✓ exécutées lors de l'occurrence d'un événement pendant que l'objet est dans l'état

Les mot-clés "*entry:*" et "*exit:*"
- indique les actions d'entrée et de sortie

Le mot-clé "*on*" *Un événement:*
- indique une action sur événement externe

Une transition réflexive entraîne les actions de sortie et d'entrée

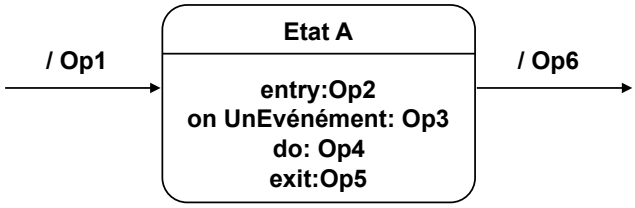


Actions et activités

UML

- **Définitions**
  - ✓ **Une action correspond à une opération dont le temps d'exécution est négligeable**
  - ✓ **Une activité est une opération qui prend du temps qui est exécutée pendant qu'un objet est dans un état donné.**
  - ✓ **Le mot-clé "*do:*" indique une activité**
  - ✓ **Activité cyclique : s'arrête lorsqu'une transition de sortie est déclenchée. activité séquentielle : l'état peut être quitté si lorsque l'activité parvient à son terme l'une des transitions est franchissable**

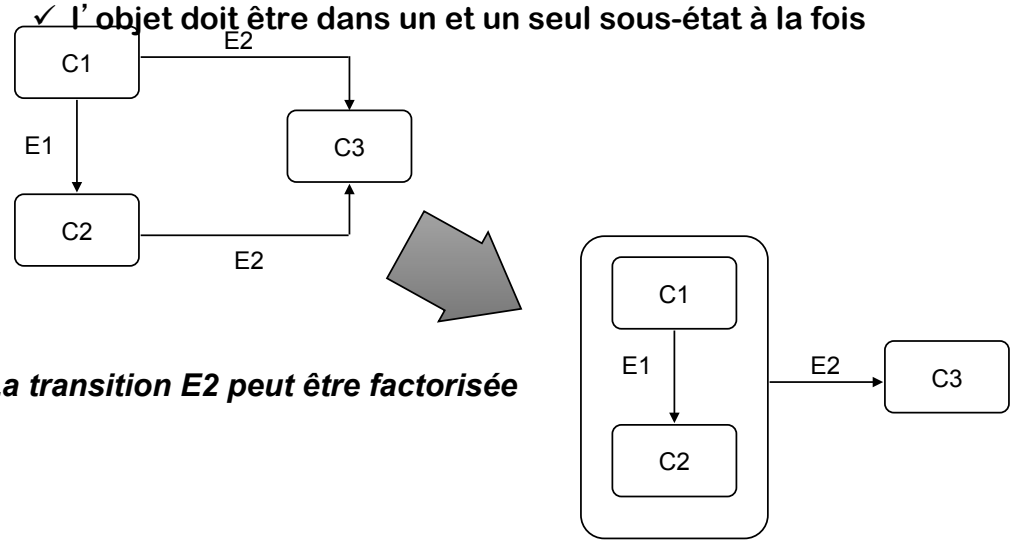
- **Les 6 types d'Actions/Activités**



Notion avancée : Généralisation d'état

UML

- **Un état peut être décomposé en sous-états disjoints**
  - ✓ **états généraux = super-états, états spécialisés = sous-états**
  - ✓ **l'objet doit être dans un et un seul sous-état à la fois**

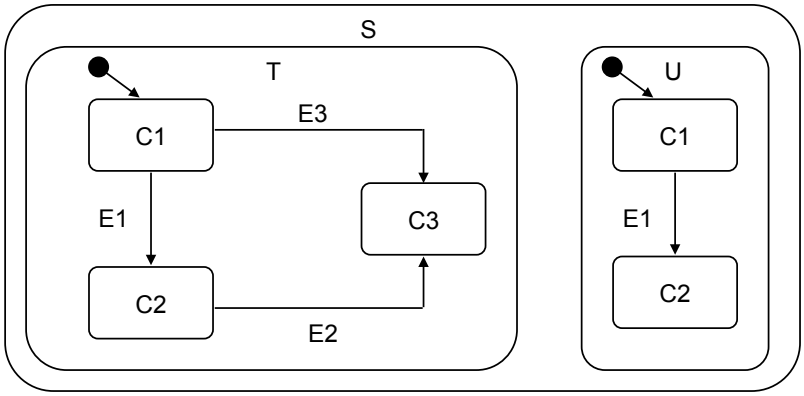


La transition E2 peut être factorisée

Notion avancée : Agrégation d'état

UML

- Composition d'un état à partir de plusieurs autres états indépendants
  - ✓ on parle d'agrégation d'états et d'état composant
  - ✓ l'objet doit être simultanément dans tous les états composant l'agrégation d'états



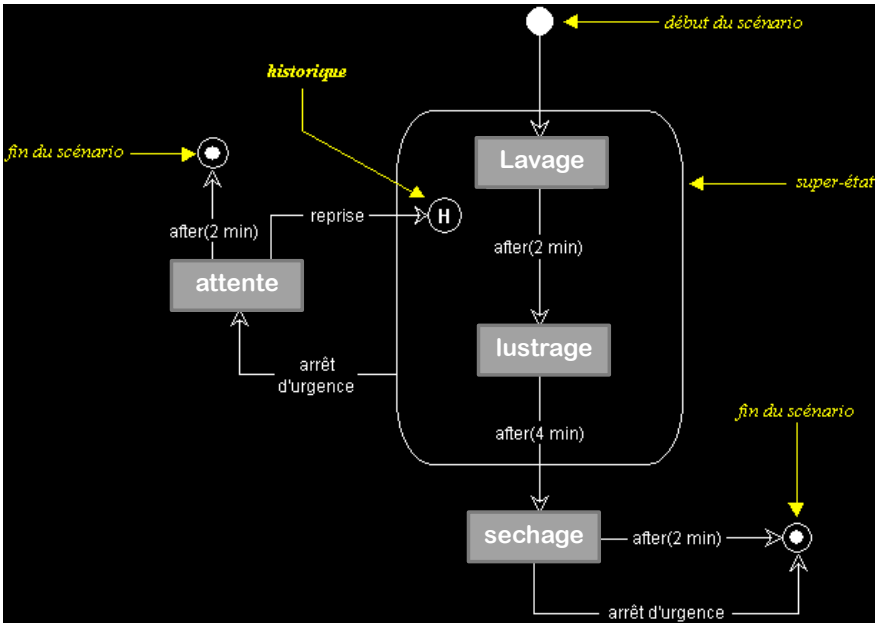
L'état S - produit cartésien des états T et U

Les diagrammes d'états-transitions : exemple

UML

Le diagramme d'états-transitions présenté, montre les différents états par lesquels passe une machine à laver les voitures.

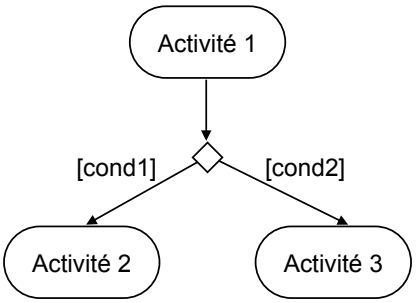
En phase de lustrage ou de lavage, le client peut appuyer sur le bouton d'arrêt d'urgence. S'il appuie sur ce bouton, la machine se met en attente. Il a alors deux minutes pour reprendre le lavage ou le lustrage (la machine continue en phase de lavage ou de lustrage, suivant l'état dans lequel elle a été interrompue), sans quoi la machine s'arrête. En phase de séchage, le client peut aussi interrompre la machine. Mais dans ce cas, la machine s'arrêtera définitivement (avant de reprendre un autre cycle entier).

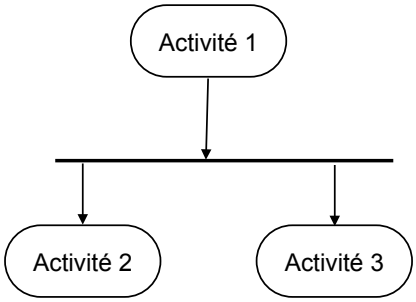


3.5 Les Diagrammes d'activités

UML

- Variante des diagrammes d'états-transitions destinée à représenter le comportement interne d'une méthode
- Ils représentent :
  - ✓ le déroulement d'étapes regroupées séquentiellement
  - ✓ les synchronisations entre flots de contrôles
- Notations :





3.6 Les diagrammes de composants

UML

Description des éléments physiques et leurs relations dans l'environnement de réalisation

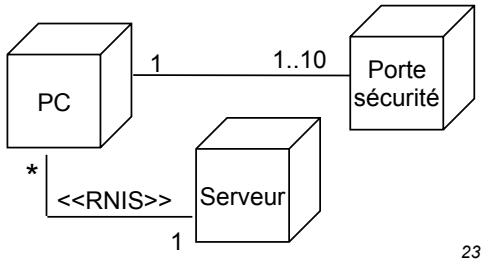
- Un composant
  - ✓ élément informatique qui entre dans la fabrication d'une application.
  - ✓ ex : fichiers, paquetage ADA, bibliothèque chargée dynamiquement,...
  - ✓ On distingue 3 types de composants :
    Spécification ou Interface, Corps , Spécification générique
- Les dépendances entre composants
- Les processus et tâches
- Les programmes principaux
- Les sous-programmes
- Les sous-systèmes

3.7 Les diagrammes de déploiement

UML

Ils montrent la disposition physique des matériels qui composent le système ainsi que la répartition des exécutables sur ces matériels

- **Les nœuds**
  - ✓ un nœud est une ressource matérielle physique
    - Dispositif (ex. Modem), Processeur (ex. PC), Mémoire (ex. Disque)
- **Liens entre nœuds**
  - ✓ lignes qui symbolisent un support de communication à priori bidirectionnel
    - connexion (ex. TCP-IP, RNIS,...)



UML

Fin du Chapitre 3

Les diagrammes de modélisation