Un aperçu de l'UML

- Notion de modélisation
- Techniques de modélisation orientée objets
- Histoire de l'UML
- Brève introduction à l'UML

  Les notions

  Les diagrammes
- Mécanismes généraux

Modélisation

- Un **modèle** est **une représentation simplifiée** d'une partie du système réel avec un but spécifique

  - Le système réel est complexe, alors il est nécessaire d'une simplification

- Un modèle est souvent **une représentation visuelle simplifiée** du système réel

- Un modèle représente le système:
  - à certain niveau d'abstraction,
  - selon un point de vue,
  - par des moyens de description (texte, image, ...)

- La **modélisation** est le processus de représenter un système par des modèles

Modélisation: pourquoi

- Mieux comprendre le système
- Facilité la communication

  Fournir des moyens de communication entre des développeurs
- Mieux compléter le système

  reconnaître la cohérence entre les modèles et les besoins pour améliorer et compléter le système
- Spécifier la structure et le comportement du système
- Documenter des décisions importantes

Modélisation

Méta-modèle

- est une représentation d'un modèle
- peut être utilisé pour
  - Décrire la syntaxe et la sémantique d'un modèle
  - Manipuler les modèles avec des outils
  - Transformer des modèles
  - Vérifier et maintenir la cohérence entre les modèle

Principes de modélisation

- Le choix du modèle approprié

  - Vue de données : modèle entité - association

  - Vue de structure : algorithme

  - Vue orientée objets : classes et relations entre elles

- Les modèle doivent représenter le système à différents niveaux d'abstraction (selon les besoins des utilisateurs)

Suite

- Les modèles doivent être connectés au monde réel

  - Les modèles construits sont proches des systèmes réels

  - Un système doit être modélisé par un ensemble de modèles

  - Un modèle n'est pas suffisant

  - Il faut décrire différentes vues du système: dynamique, statique, installation, utilisation, ...

Modélisation

Un bon modèle devrait

- Utiliser une notation standardisée
- Être compréhensible pour les clients et les utilisateurs
- Permettre aux ingénieurs logiciels de bien saisir le système
- Fournir une vue abstraite du système
- Être visuel

Avantages de modélisation

- Faciliter la révision et l'évolution du système
- Réduire des erreurs en permettant de détecter tôt des erreurs dans les phases de développement
- Réduire le coût de développement
- Réduire le temps-à-marché
- Réduire la complexité par le mécanisme d'abstraction

Techniques de modélisation orientée objets

- 1975 – 1990 : plusieurs techniques de modélisation orientée objets sont développées
- 1990 – 1994 : il existe plus 50 techniques de modélisation orientée objets
- Les techniques les plus connues
  - **OOD** (Object Oriented Design)
  - **OOSE** (Object Oriented Software Engineering)
  - **OMT** (Object Modeling Technique)

Technique OMT

Développée par Rumbaugh (1991)

Se compose de 3 vues

- Vue statique
  - Modèle d'entité-relation
- Vue dynamique
  - Diagramme d'états
- Vue fonctionnelle
  - Diagramme de flots d'information

Technique OOD

Développée par Booch (1991)

- Vue statique
  - Diagramme de classes
  - Diagramme d'objets
- Vue dynamique
  - Diagramme d'états
  - Diagramme de temps

Technique OOSE

Développée par Jacobson (1992)

Se compose de 5 modèles

- Modèle des besoins (scénario d'utilisation)
- Modèle d'analyse (niveau conceptuel)
- Modèle de conception (niveau logique)
- Modèle d'implémentation (niveau physique)
- Modèle de test

Histoire de l'UML

Trop de techniques de modélisation orientée objets

- Nécessité d'une standardisation
  - Unification des techniques de modélisation

En 1994

- Rumbaugh et Booch unifient leurs approches pour le projet UML chez Rational Software

En 1995

- La première version sort sous le nom « Unified Method »v0.8
- Jacobson se joint à l'équipe

Histoire de l'UML (Suite 1)

En 1996

- La naissance de l'UML v0.9 intégrant OOSE
- La première conférence de l'UML est organisée

En 1997

- L'UML v1.0 soumis à l'OMG (Object Management Group)
- L'UML v1.1 standardisé par l'OMG

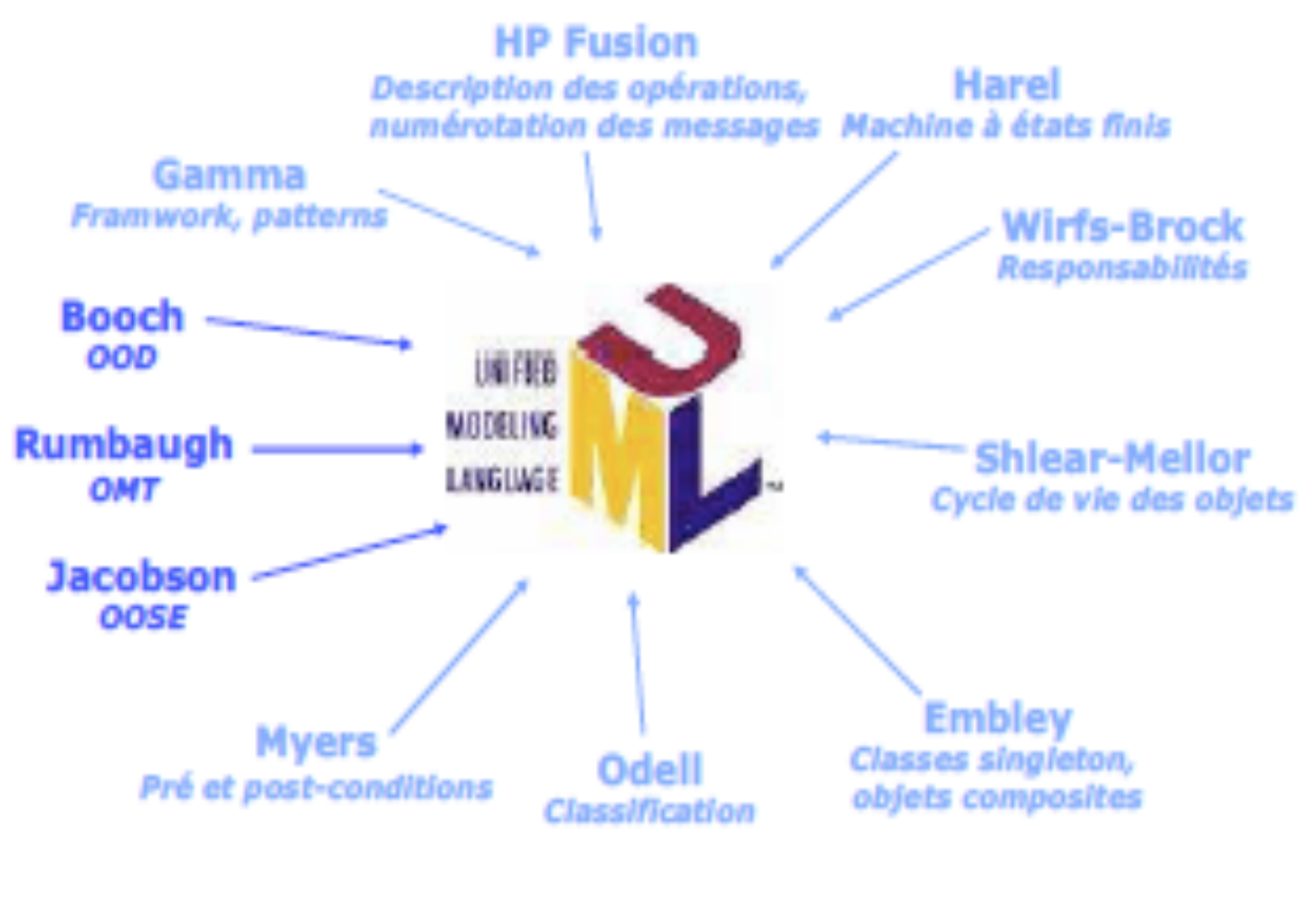
En 1998

- l'OMG distribue l'UML v1.2

Actuellement (2010): l'UML v2.0

Histoire de l'UML (Suite 2)

Contributions à l'UML



UML : un langage

- UML : Unified Modeling Language
- L'UML est **un langage**

  - Composer du vocabulaire, de la syntaxe et de la sémantique

- L'UML est **un langage de modélisation**

  - Permettre de représenter un système à différents niveaux: conceptuel, physique
  - Se composer du vocabulaire et des règles pour décrire des différents modèles représentant un système

Langage UML (Suite 1)

- L'UML n'est pas une méthodologie ou un processus
- L'UML laisse la liberté de conception
- L'UML peut être combiné avec plusieurs processus de développement
- L'UML est **un langage de visualisation**

  - Utiliser des représentations graphiques

  - Apporter une meilleure vue du système (en utilisant des représentations graphiques)

Langage UML (Suite 2)

- L'UML est **un langage de spécification**

  - Permettre de spécifier un système sans ambiguïté
  - Permettre de spécifier un système à différents : analyse, conception, installation

- L'UML est **un langage de construction**

  - Permettre de simuler le système
  - Les modèles UML sont facilement transformée en code source

- L'UML est **un langage de documentation**

  - Permettre de décrire toutes les étapes de développement du système
  - Les modèles construits sont des documents complets du système

UML : les modèles

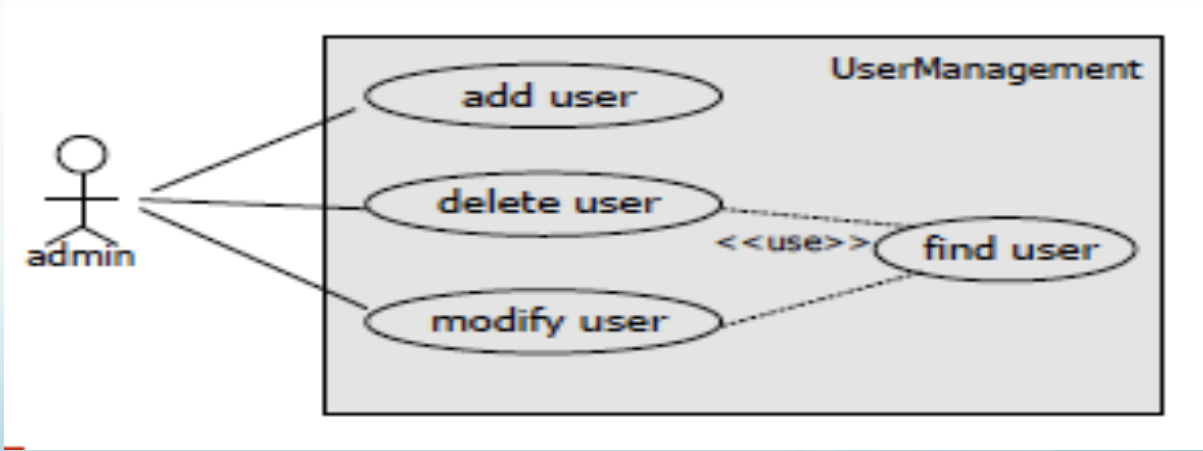
Se compose de 9 (13) diagrammes

- Modélisation **des besoins**
  - Diagrammes de cas d'utilisation
- Modélisation de **la structure statique**
  - Diagrammes de classes
  - Diagrammes d'objets
- Modélisation **du comportement dynamique**
  - Diagrammes d'interaction (Diagrammes de séquence , Diagrammes de collaborations)
  - Diagrammes d'activité
  - Diagrammes d'états
- Modélisation **de l'architecture**
  - Diagrammes de composants
  - Diagrammes de déploiement

Diagramme de cas d'utilisation

- Montrer les utilisations possibles d'un système
- Décrire la vue statique du système selon le **point de vue des utilisateurs**
- Très importants pour saisir les fonctions du système

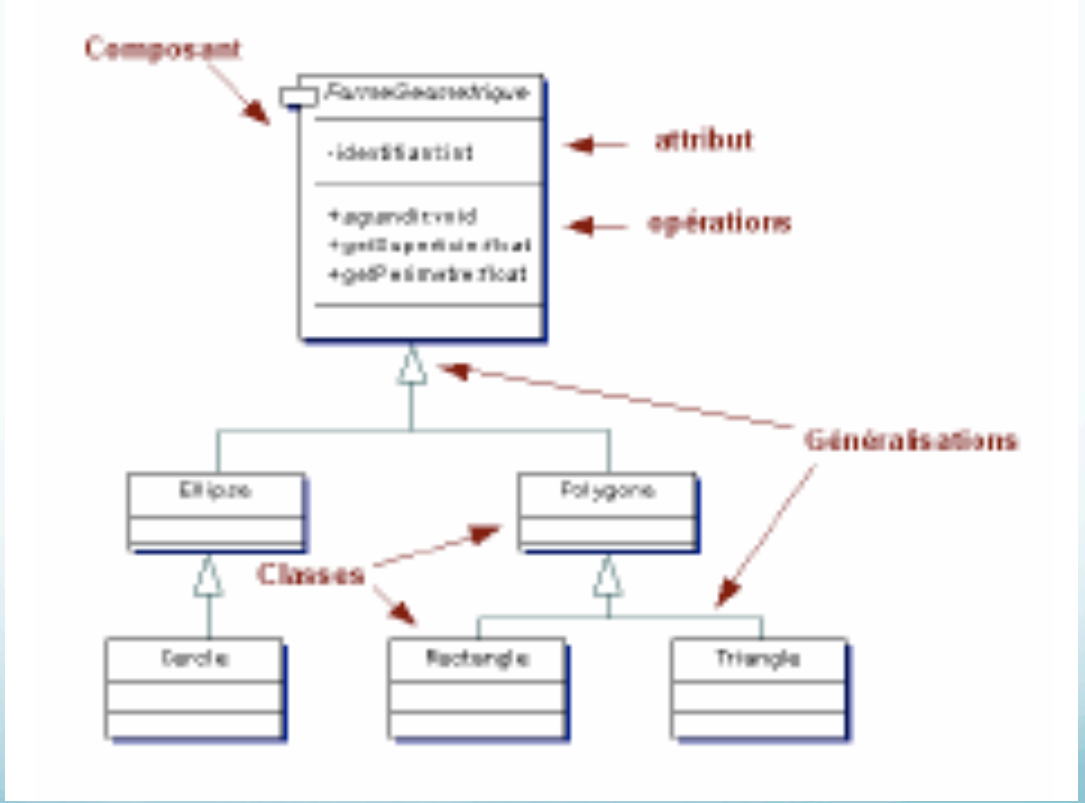
Exemple:



Diagrammes de classes

- Décrire des classes et leurs interactions
- Décrire la vue statique du système

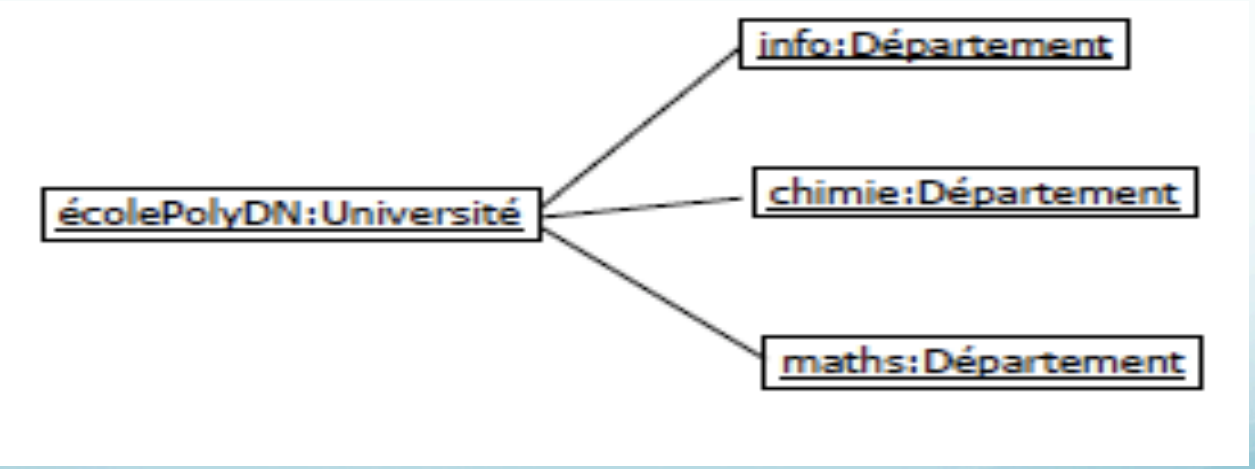
Exemple



Diagrammes d'objets

- Décrire un ensemble des objets et leurs interactions
- Un diagramme d'objets représentent les mêmes informations qu'un diagramme de classes mais en vue des instances des classes

Exemple



Diagrammes d'interactions

- Décrire le comportement du système par les interactions entre des objets qui le composent.
- Montrer **la vue dynamique** du système
- Les diagrammes d'interactions sont une extension des diagrammes d'objets en précisant les interactions entre les objets
- Composer de deux types de diagramme

  - Diagrammes de séquence

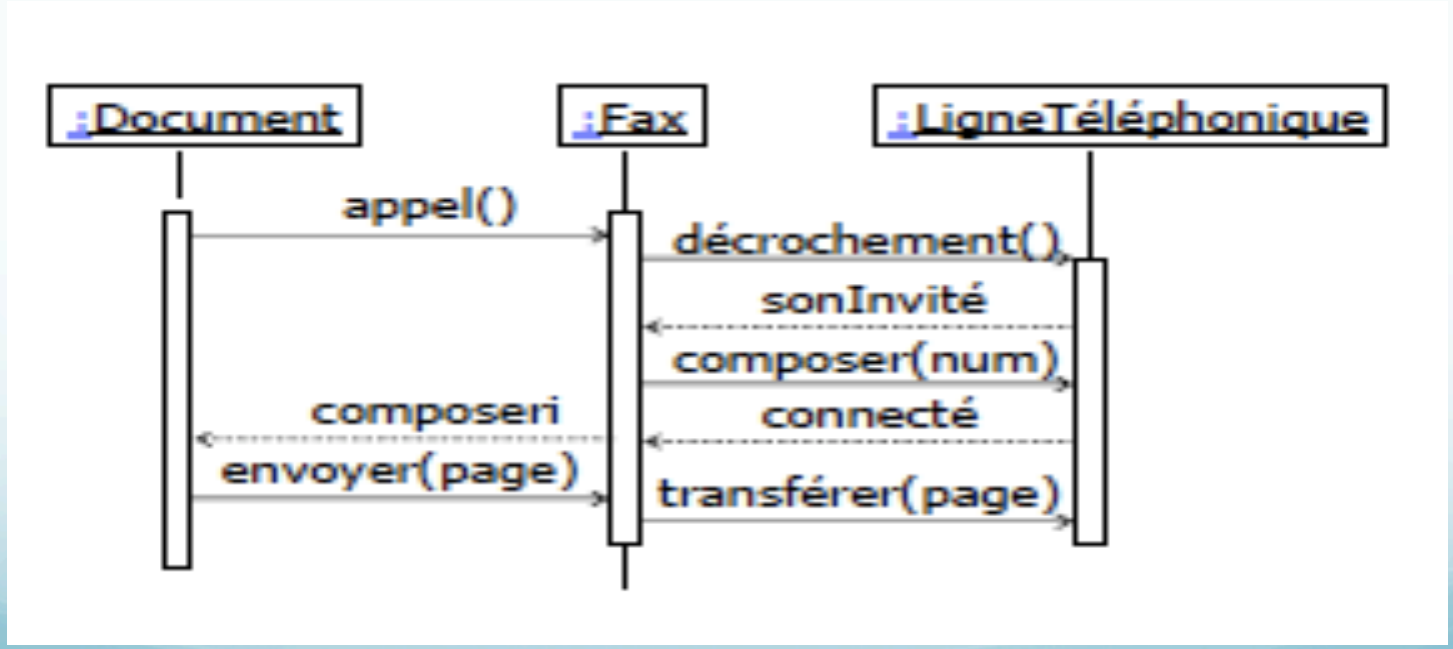
  - Décrire les interactions entre les objets en mettant l'accent sur le séquencement des messages

Diagrammes d'interactions (1)

- Diagrammes de collaboration

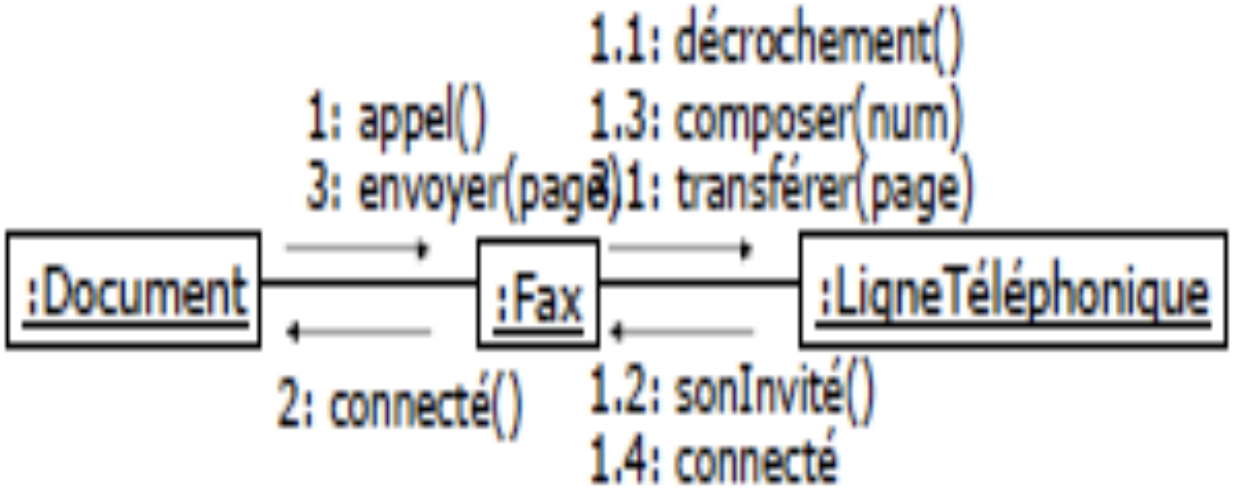
  - Décrire les interactions entre les objets en mettant l'accent sur la structure des objets

Diagrammes de séquence



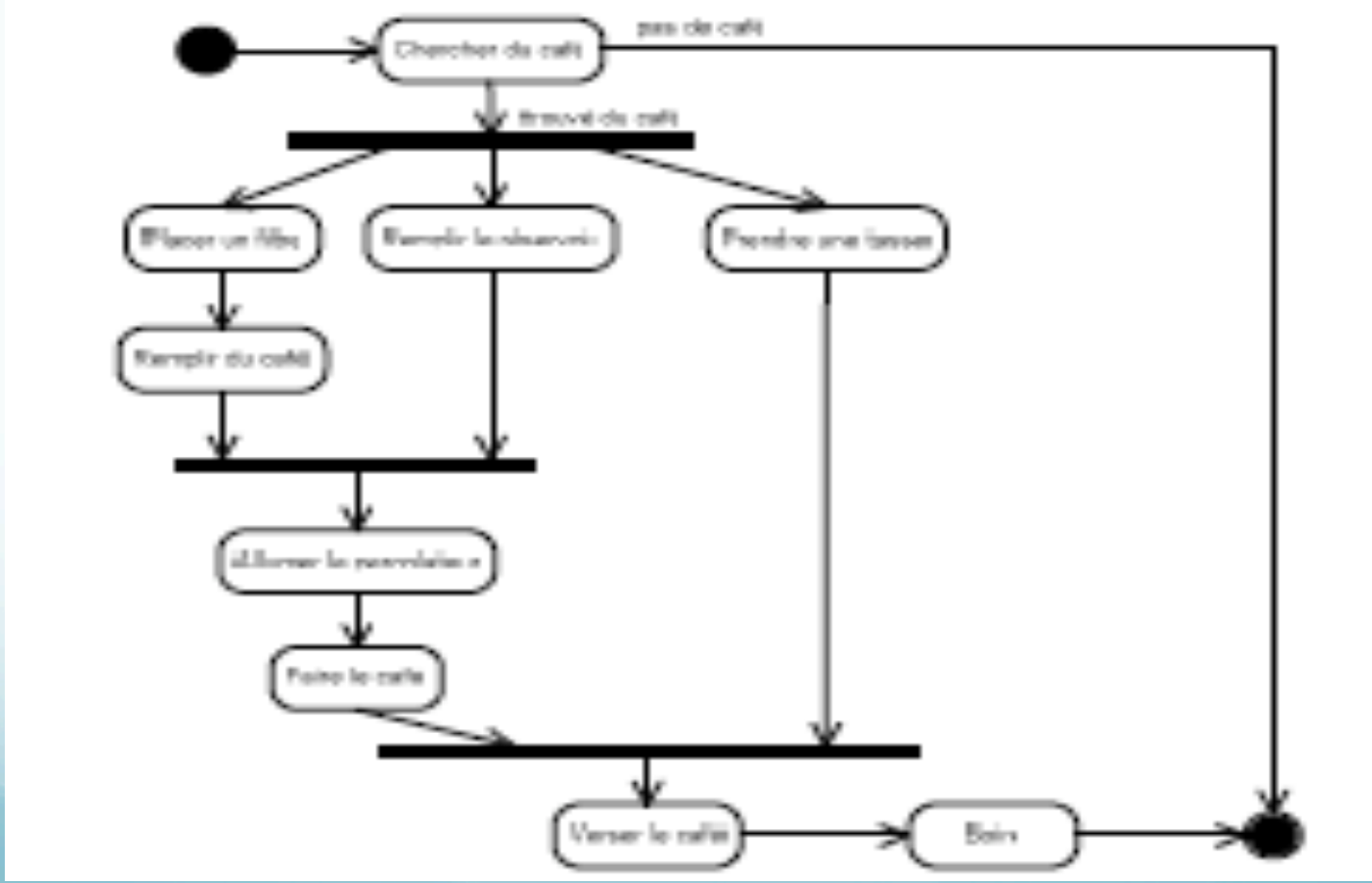
Diagrammes d'interactions(2)

Diagrammes de collaboration



Diagrammes d'activités

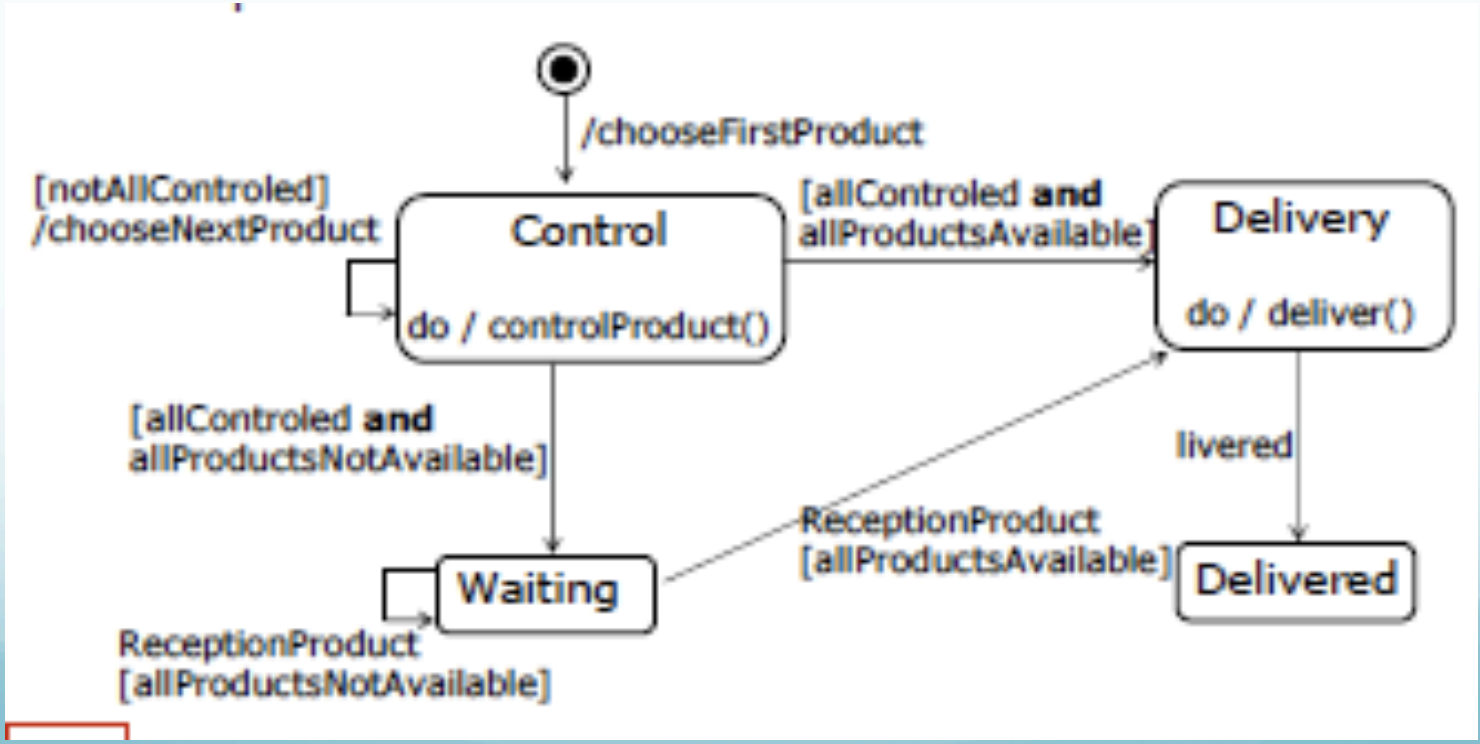
- Décrire les flots de l'information dans le système
- La vue dynamique du système



Diagrammes d'états

- Décrire comment le système se comporte de façon Interne
- La vue statique du système

Exemple:



Diagrammes de déploiement

Décrire l'organisation physique de différents composants (machines) du système (matériel)

Les mécanismes généraux

- Mécanisme de structuration des diagrammes de classes

  -Les paquetages

- Mécanisme intégré d'extension

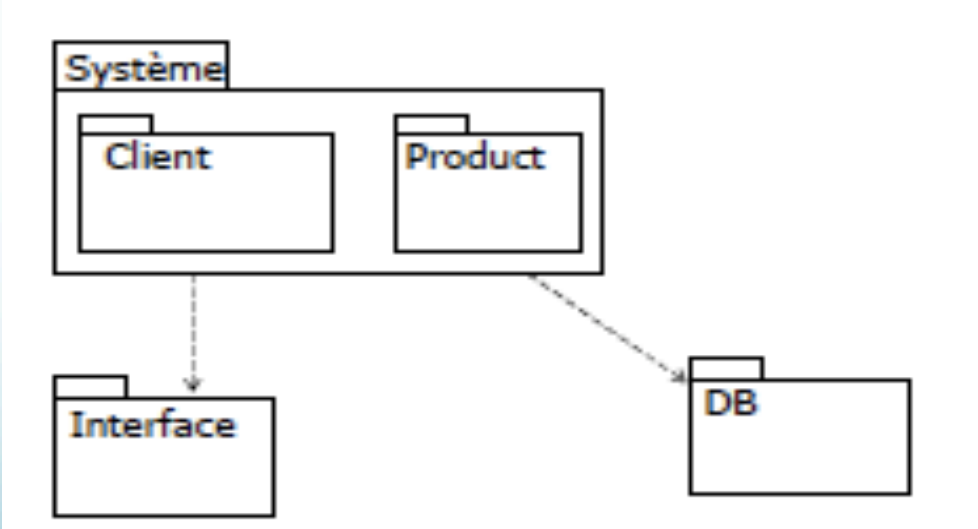
  -Les stéréotypes

  -Les valeurs marquées (tagged values)

- Les notes
- Les contraintes

Les paquetages

- Permettre de structurer des diagrammes de classes
- Construire une structure de dépendance entre des paquetage

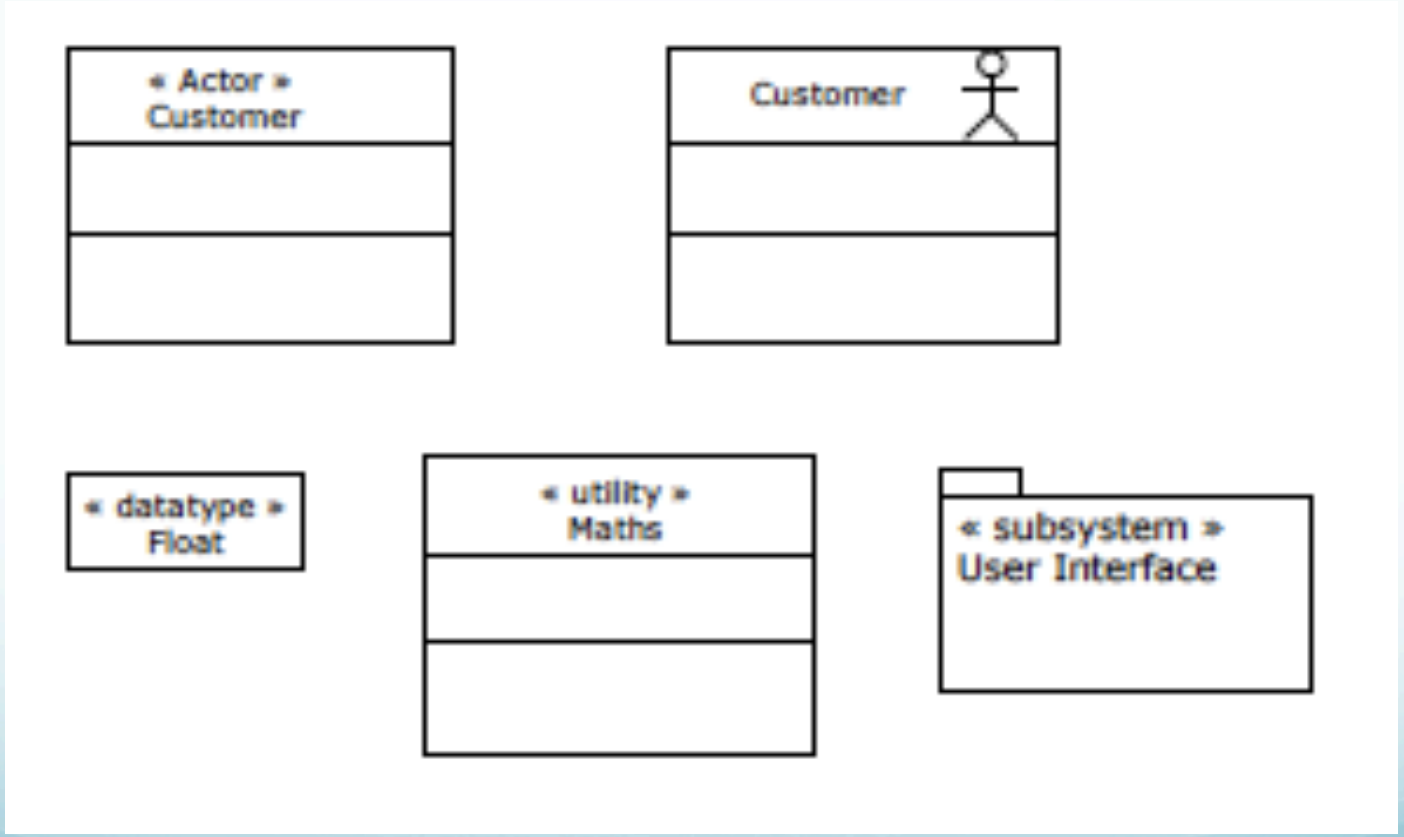


Les stéréotypes

- est un Un mécanisme intégré d'extension permet élargir le vocabulaire d'UML
- Sont employés pour créer de nouveaux types d'élément en UML qui dérivent des sortes existantes mais qui sont adaptées à un problème donné
- Il existe des stéréotypes prédéfinis dans l'UML
- Notation
  « nom du stéréotype »
  Possibilité d'ajouter un icône

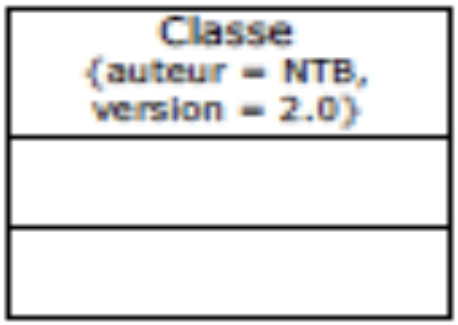
Les stéréotypes

Exemple



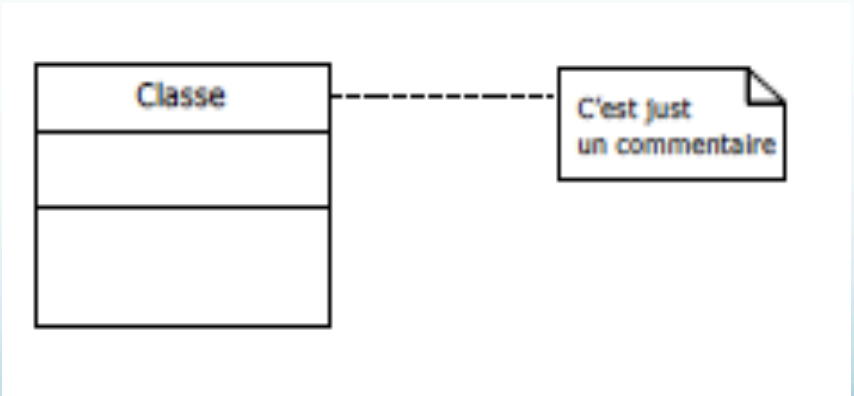
Les valeurs marquées

- Un autre mécanisme d'extension
- Fournir des informations supplémentaires sur les éléments d'UML
- Paires de type {nom = valeur}



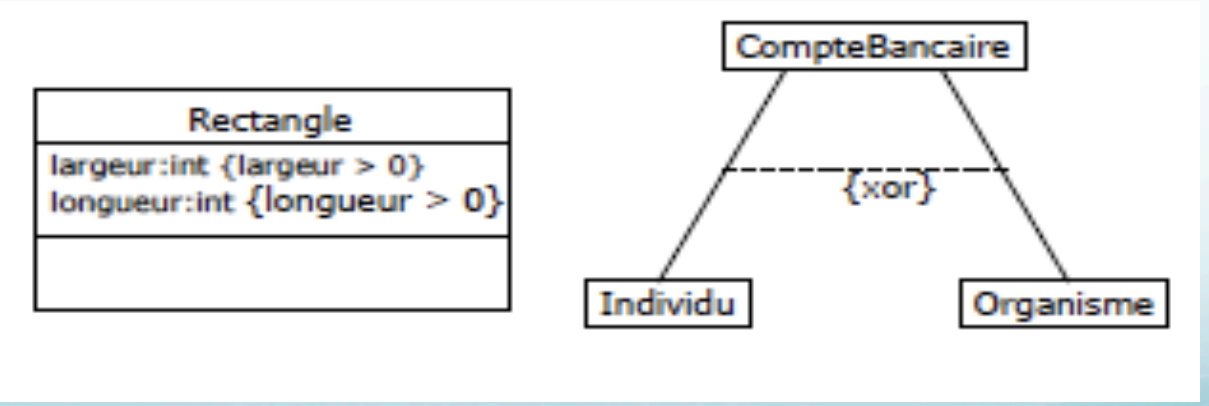
Les notes

- Commentaires attachés à un ou plusieurs éléments de modélisation
- Fournir l'information supplémentaire sur les éléments de
- Modélisation Appartiennent à la vue, pas aux modèles



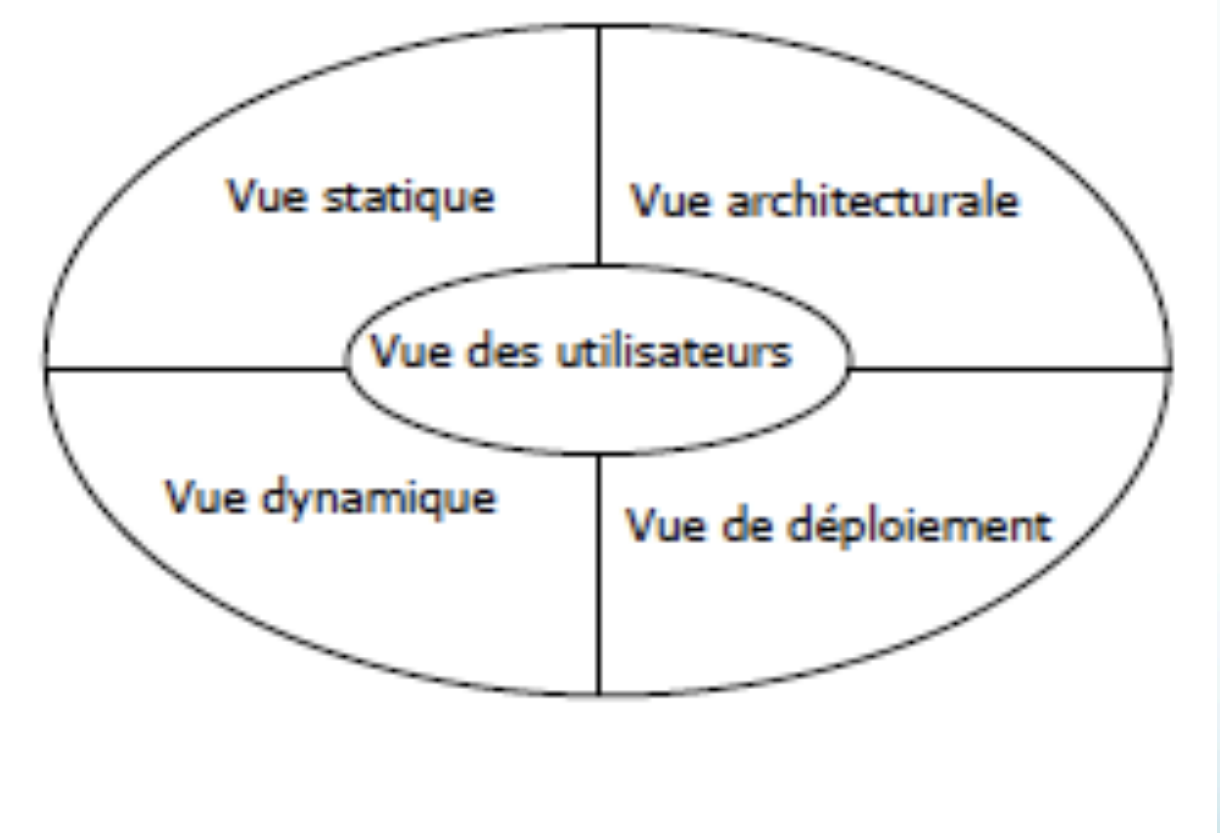
Les contraintes

- Restrictions qui limitent l'utilisation d'un élément ou la sémantique d'élément
- Exprimé en langage naturel
- Exprimé en OCL (Object Constraint Language)
- Exemple



Les vues

- Un système se modélise en 5 vues différentes dans



Les vues

Les diagrammes et les vues