UML 1ère partie

Référence:
http://uml.developpez.com/lp/cours/uml_free_fr_cours.html

PLAN

- Définition et historique
- Vue générale
- A quoi ça sert?
- Modéliser avec UML : qu'est-ce qu'un modèle?
- Modéliser avec UML : comment?
- Les vues de UML
  - Vue logique
  - Vue des processus
  - Vue des composants
  - Vue de déploiement
  - Vue des cas d'utilisation

Définition et historique

- UML : Unified Modeling Language est un langage unifié de modélisation objets.
- Unification : né (1997) de la fusion de 3 langages
  - OMT : par Rumbaugh (Rational Software)
  - OOD : par Booch (General Electric)
  - OOSE : par Jacobson (Ericsson)
- Fruit du terrain et de 3 industriels renommés

Définition et historique

- OMT : Object Modeling Techniques a pour vues :
  - Statiques : ne modifie pas l'objet
  - Dynamiques : peut modifier l'objet
  - Fonctionnelles : séquences d'appels de fonctions d'une utilisation
- OOD : Oriented Object Design a pour vues :
  - Logiques : décomposition logicielle
  - Physiques : décomposition matérielle
- OOSE : Oriented Object Software Engineering a des vues basées sur le cycle vie logiciel
  - Analyse
  - Conception
  - Réalisation/Implémentation
  - Test/Maintenance

Vue générale

- Cycle de vie logiciel
  - UML couvre toutes les phases du cycle de vie
- Indépendance
  - UML est indépendant du domaine d'application et des langages d'implémentation
- Standard
  - intégré dans de nombreux outils de modélisation (GDPro, Rational Rose, TogetherSoft, Visio, …)
- Format d'échange
  - Utilise XML pour échanger les modèles UML

A quoi ça sert?

- UML langage semi formel
  - UML est structuré sur un métamodèle qui définit :
    - ☑ les éléments de modélisation (les concepts manipulés par le langage(classe, cas d'utilisation, package, etc.),
    - ☑ la sémantique de ces éléments (leur définition et le sens de leur utilisation).
  - Un métamodèle = description très formelle des concepts d'un langage. Il limite les ambiguïtés.
  - Le métamodèle d'UML permet de classer les concepts du langage selon leur niveau d'abstraction ou domaine d'application.
  - Le langage formel utilisé pour le métamodèle UML est OMG-MOF
  - Les éléments du métamodèle ont une représentation graphique associée

A quoi ça sert?

- UML est un cadre d'analyse des objets
  - offre plusieurs vues complémentaires qui permettent de décrire le cycle de vie d'une application
- Ces vues utilisent les notions fondamentales de la conception objets
- UML est un support de communication
  - Sa notation graphique permet de visualiser une solution objet
  - L'aspect formel de sa notation limite les ambiguïtés
  - L'aspect visuel facilite la comparaison
  - Son indépendance par rapport au langage de programmation en fait un langage universel objet

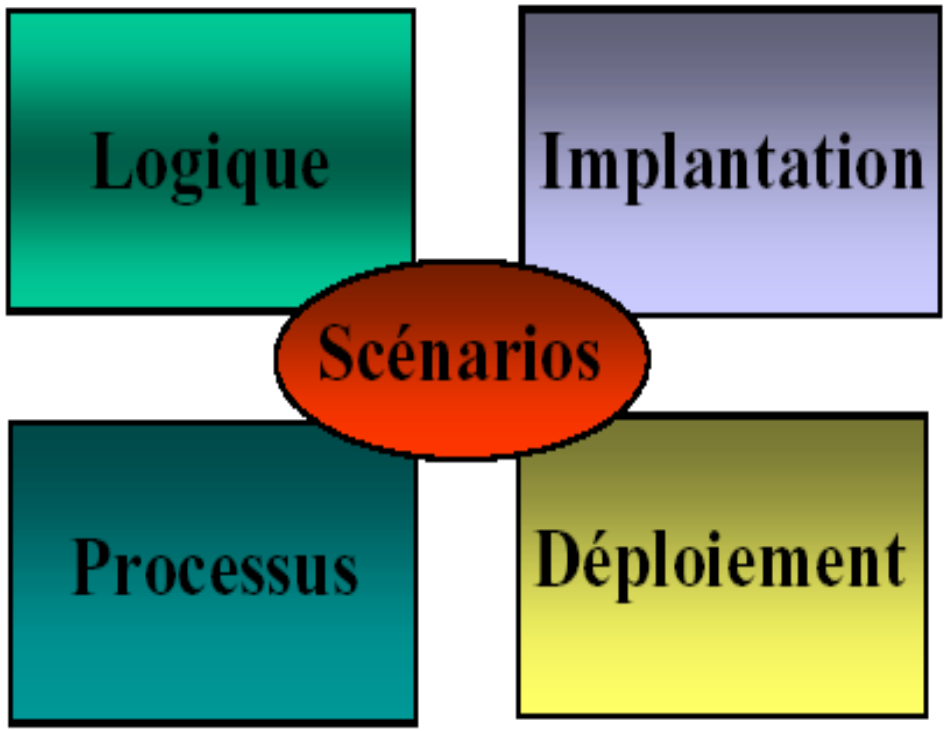
Qu'est-ce qu'un modèle?

- Un modèle est une représentation semi formelle, formelle ou abstraite d'un problème posé en langage naturel
- Un modèle se réalise à travers un langage de modélisation
- Selon le découpage des informations un modèle peut être orienté objet ou fonction
- Un modèle objets fait ressortir les objets du langage naturel
- Dans cet exercice les objets sont généralement les sujets ou les compléments d'objets du problème posé en lange naturel.

Comment?

- UML ne définit pas le processus d'élaboration des modèles
- Mais suggère 3 démarches
  - Itérative et incrémentale : L'idée est de quitter d'un prototype et de l'améliorer
  - Besoins utilisateurs : Dans ce cas ce sont les utilisateurs qui guident la réalisation du modèle.
    - ☑ Validation des différents livrables du modèle par les utilisateurs.
  - Centré sur l'architecture : en utilisant les différentes vues de UML proposé par Ph. Kruchten (cf. figure suivante)

Comment?



- **Figure :** 4+1 vues centré sur l'architecture [Élements génie logiciel, Ecole Polytechnique Montéal, 2001]

Les vues de UML

- 5 différentes vues
  - La vue **logique** se concentre sur l'abstraction et l'encapsulation (classes, , stéréotypes, diagramme de classes)
  - La vue d'**implantation** est une vue de bas niveau composée de modules logiciels
  - La vue des **processus** utilisé dans un environnement multitâches
  - La vue de **déploiement** est utilisé dans un environnement distribué pour représenter l'architecture et la topologie
  - La vue **des cas d'utilisation ou de scénarios** est une vue qui sert de guide à tous les autres

La vue logique

- Décomposition orientée objets
  - Les objets, les classes
  - Relations entre objets et comportement des objets
    - ☑Diagramme activités
    - ☑Diagramme d'états,
    - ☑Diagramme de collaboration,
    - ☑Diagramme de classe
    - ☑et Diagramme de dépendance de dossiers logiques
  - Structuration sous forme de dossier/répertoire (ou package)

La vue des processus

- Décomposition en tâches/processus
- Organisation en groupe de processus
- Les interactions entre processus
- Synchronisation et communication parallèle
- Différents types de communication
  - Asynchrone,
  - Synchrone,
  - Implicite, etc.
- Les paramètres clés
  - Disponibilité, fiabilité
  - Intégrité, performance
  - Contrôle, synchronisation

La vue d'implantation ou de composants

- Décomposition statique des modules et sous systèmes
  - L'allocation des éléments de modélisation dans des modules (fichiers sources, exécutables, etc...).
  - L'organisation des composants : la distribution du code, les dépendances entre les composants.
  - Les contraintes de développement (bibliothèques externes...).
  - Organisation complexe
    - ☑ les modules en « ***sous-systèmes*** » ,
    - ☑ dépendance entre les interfaces des sous-systèmes et d'autres sous-systèmes ou modules.

La vue de déploiement

- Description des ressources matérielles/logicielles en environnement distribué
  - La disposition et nature physique des matériels, ainsi que leurs performances
  - L'implantation des modules principaux sur les nœuds du réseau
  - Les exigences en terme de performances (temps de réponse, tolérance aux fautes et pannes….)
  - Les paramètres clés
    - ☑ Disponibilité, fiabilité, performance
    - ☑ Contrôle, Intégrité, synchronisation
    - ☑ Installation, maintenance

La vue des scénarios ou cas d'utilisation

- Sert de cadre ou guide aux autres vues
- Définit les besoins des clients du système
- Contient les cas d'utilisation et les acteurs
- A l'aide de scénarios et de cas d'utilisation
  - Contribue à définir un modèle d'architecture pertinent et cohérent
- Unifie les quatre autres vues de l'architecture
- Permet d'identifier les interfaces critiques
- Elle permet un regroupement logique des cas d'utilisation