## Modélisation Objet avec UML

- La genèse d'UML
- Un survol d'UML
- La notation UML

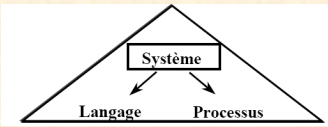
D'après le cours de Pierre-Alain Muller

## Genèse : Complexité des logiciels

- Les tendances
  - Programmation sans programmer
  - Micro-architectures (*patterns*)
  - Importance de l'architecture
  - Informatique distribuée
  - Multimédia

## De quoi a-t-on besoin ?

- Un langage de modélisation
  - Notation claire
  - Sémantique précise
- Un processus de génie logiciel




## Langage de modélisation

- Générique
- Expressif
- Flexible (configurable, extensible)
- Syntaxe et sémantique
- Unification par convergence aujourd'hui

## Processus

- Générique
- Impossible à standardiser
  - Personnes, applications, cultures...
- Cadre configurable
- Unification par convergence dans le futur

## Comment modéliser ?

- La manière de modéliser influence fortement
  - La compréhension du problème
  - La solution
- Il n'existe pas de modèle universel
  - Jeux de modèles faiblement couplés
  - Multiplicité des niveaux d'abstraction
- Les meilleurs modèles sont en prise sur le monde réel

## Evolution des méthodes

- D'abord des méthodes structurées
- A partir des années 80 émergence des méthodes orientées-objets
- Les principales méthodes objets convergent
  - Différences superficielles
  - Notation, terminologie
- L'expérience permet de séparer le bon grain de l'ivraie

## La prolifération des méthodes objet

- Une cinquantaine de méthodes objet dans les cinq dernières années
  - Confusion, attentisme
- Consensus autour d'idées communes
  - Objets, classes, associations, sous-systèmes, cas d'utilisation

## Rapprochement de Booch et OMT

- Booch'93 et OMT-2 sont plus ressemblantes que différentes
  - Booch'93 adopte les associations, les diagrammes d'Harel, les traces d'événements
  - OMT-2 introduit les flots de messages et retire les diagrammes de flot de données
- Booch-93 construction
- OMT-2 analyse et abstraction

## L'unification des méthodes

- La guerre des méthodes ne fait plus avancer la technologie des objets
- Recherche d'un langage commun unique
  - Utilisable par toutes les méthodes
  - Adapté à toutes les phases du développement
  - Compatible avec toutes les techniques de réalisation

## Différentes sortes de systèmes

- Logiciels
  - Ingénierie des logiciels
- Logiciels et matériels
  - Ingénierie des systèmes
- Personnes
  - Ingénierie des affaires
- **Unification sur plusieurs domaines d'applications**

## La notation unifiée

- Basée sur les méthodes de BOOCH, OMT et OOSE
- Influencée par les bonnes idées des autres méthodes
- Mûrie par le travail en commun

## Principales influences

- Souvent une histoire imbriquée

**Booch** Catégories et sous-systèmes
**Embley** Classes singletons et objets composites
**Fusion** Description des opérations, numérotation des messages
**Gamma, et al.***Frameworks*, *patterns*, et notes
**Harel** Automates (*Statecharts*)
**Jacobson** Cas d'utilisation (*use cases*)
**Meyer** Pré- et post-conditions
**Odell** Classification dynamique, éclairage sur les événements
**OMT** Associations
**Shlaer-Mellor**Cycle de vie des objets
**Wirfs-Brock** Responsabilités (CRC)

## Portée de la notation unifiée

- Standardiser les artefacts du développement
  - Modèles, notation et diagrammes
- Ne pas standardiser le processus
  - Dirigé par les cas d'utilisation
  - Centré sur l'architecture
  - Itératif et incrémental

## Les objectifs

- Représenter des systèmes entiers
- Etablir un couplage explicite entre les concepts et les artefacts exécutables
- Prendre en compte les facteurs d'échelle
- Créer un langage de modélisation utilisable à la fois par les humains et les machines

## Approche retenue

- Identifier la sémantique des concepts de base
- Classer les concepts
- Construire un métamodèle
- Choisir une notation graphique
- Regrouper par niveau d'abstraction, complexité et domaine

## Métamodèle

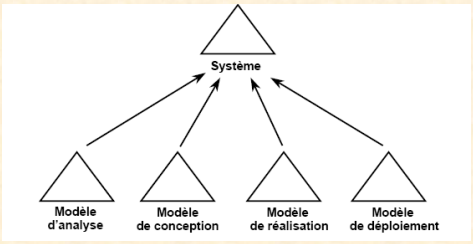
- Identification des concepts fondamentaux
  - Définition de la sémantique de ces concepts
  - Choix d'une représentation graphique
- Métamodélisation d'UML avec UML
  - Description formelle des éléments de modélisation
- Austère, pas pédagogique
  - Méthodologistes
  - Constructeurs d'outils

## Les modèles et les vues

- Un modèle est un quanta de développement
  - Cohérence interne forte
  - Couplage faible avec les autres modèles
  - Relié à une phase de développement
- Une vue est une projection au travers des éléments de modélisation
  - Graphique
  - Peut englober plusieurs modèles

## Exemples de modèles

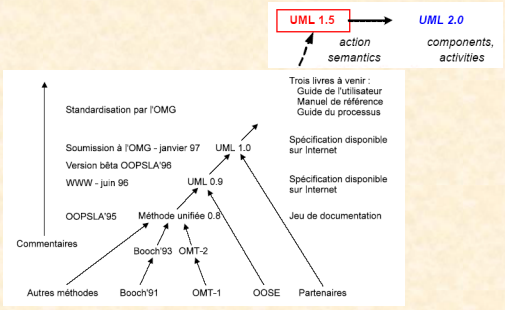
- Un système possède plusieurs modèles




## Les étapes

- Octobre 95
  - *Unified Method* V0.8
- Octobre 96
  - UML V0.91 (*The Unified Modeling Language for Object-Oriented Development*)
- Janvier 97
  - UML 1.0 est soumise à l'OMG
- Septembre 97
  - Approbation par le comité technique de l'OMG

## Evolution de UML




## Acceptation de UML

- UML est dans le domaine public
- Soutenue par le marché
  - Microsoft, HP, IBM, Oracle...
- Successeur naturel des méthodes de Booch, OMT et OOSE
- UML est le fruit de l'expérience et des besoins de la communauté des utilisateurs

## Les partenaires

- Courant 96 UML devient un enjeu stratégique
- Consortium de partenaires
  - DEC, HP, i-Logix, Intellicorp, IBM, ICON Computing, MCI Systemhouse, Microsoft, Oracle, Rational Software, TI et Unisys
  - IBM, Platinum, Data Access Technologies, Reich Technologies, Softeam, Taskon A/S

## Domaines d'application

- Applicable à tout développement logiciel (à objets)
  - Systèmes d'information, SIG...
  - Systèmes temps réels, embarqués...
  - Interfaces, simulateurs, calcul
  - Applications diverses
- Couverture complète du cycle de développement.
  - Analyse des besoins
  - ...
  - Intégration et tests

## En résumé

- UML est une notation, pas une méthode
- UML est un langage de modélisation objet
- UML convient pour toutes les méthodes objet
- UML est dans le domaine public

## Survol : Eléments de modélisation

- Briques pour capturer la sémantique des applications
- Pas accessibles directement aux utilisateurs
- Représentation interne (outils)
- Représentation externe (échange entre outils)

## Eléments de modélisation

- Les objets 
  - Une entité d'un monde réel ou virtuel
- Les classes 
  - La description d'un ensemble d'objets
- Les états 
  - Une étape de la vie d'un objet
- Les tâches 
  - Un flot de contrôle indépendant

## Eléments de modélisation

- Les cas d'utilisation 
  - Une manière dont un acteur utilise le système
- Les collaborations 
  - La réalisation d'un cas d'utilisation par une société d'objets collaborants
- Les micro-architectures (*patterns*) 
  - Un générateur pour la structure et l'interaction d'une société d'objets

## Eléments de modélisation

- Les composants 
  - Un module contenant des entités d'implémentation
- Les noeuds 
  - Un dispositif matériel capable d'exécuter du logiciel
- Les paquetages 
  - Une partition du modèle
- Les notes 
  - Un commentaire, une explication ou une annotation

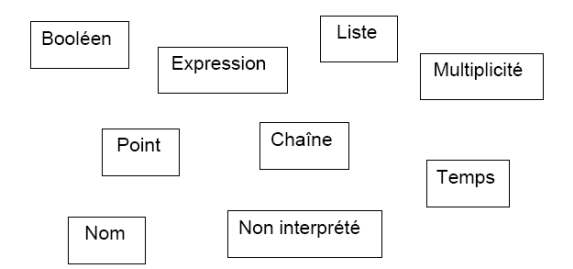
## Relations

- L'association 
  - Une connexion sémantique entre instances
- La généralisation 
  - Une relation de classification
- La dépendance 
  - L'utilisation d'un élément par un autre
- La trace 
  - Dépendance inter-modèles

## Mécanismes communs

- Les stéréotypes <<stéréotype>>
  - Extension des classes du métamodèle
- Les étiquettes
  - Paire (nom, valeur)
- Les notes
  - Commentaire textuel
- Les contraintes {contrainte}
  - Relation sémantique entre éléments

## Types primitifs

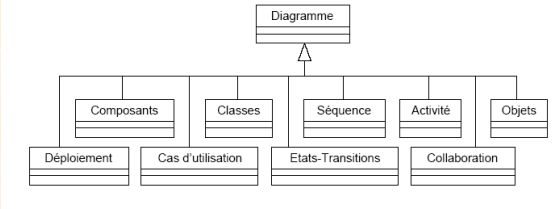



## Notation UML

- Manipulée par les utilisateurs
- Simple, intuitive, expressive, cohérente
- Vues graphiques (multiples) des éléments de modélisation

## Les diagrammes d'UML

- 9 types de diagrammes




## Diagrammes

- Les diagrammes de classes
  - Les classes et les relations statiques
- Les diagrammes d'objets
  - Les objets et les liens
- Les diagrammes de séquence
  - Vision temporelle des interactions
- Les diagrammes de collaboration
  - Vision spatiale des interactions

## Diagrammes (suite)

- Les diagrammes de cas d'utilisation
  - Les acteurs et l'utilisation du système
- Les diagrammes d'états-transitions
  - Le comportement des objets
- Les diagrammes d'activités
  - Le flot de contrôle interne aux opérations

## Diagrammes (suite)

- Les diagrammes de composants
  - Les composants d'implémentation et leurs relations
- Les diagrammes de déploiement
  - La structure matérielle et la distribution des objets et des composants