SLR202 – Modélisation UML : vue structurelle et simulation comportementale Introduction

Etienne Borde@telecom-paristech.fr
Sylvie.Vignes@telecom-paristech.fr

Institut Mines-Télécom
Télécom ParisTech
Département Informatique et Réseaux
Groupe Systems, Software,Services

Éléments de Génie Logiciel

Un premier contact avec le Génie Logiciel

- *En référence aux standards ISO/IEEE*:
  « Ensemble des activités de conception et de mise en œuvre des produits et des procédures tendant à rationaliser la production du logiciel et son suivi »
- *But de l'UE*:
  Apprendre à pratiquer les activités de modélisation en utilisant un langage graphique UML, consensus industriel, selon de «bonnes pratiques», applicables dans vos projets logiciels

Le Génie Logiciel dans le contexte industriel

- Logiciels essentiels au fonctionnement d'une entreprise:
  Système d'Information du « cœur de métier »
  - Ex : administration du réseau ou facturation d'un opérateur télécom …
- Logiciels critiques :
  exécution vitale dont l'erreur peut coûter des vies humaines ou coûter très cher
  - Ex: transport, médecine, finances …

Un modèle

- est une représentation abstraite de la réalité qui masque certains détails du monde réel,
- permet de réduire la complexité du problème en éliminant les détails qui n'influencent pas son comportement de manière significative,
- décrit ce que le concepteur croit important pour la compréhension et la prédiction du problème à modéliser.

- Les objectifs du problème permettent de cibler le modèle et de déterminer les limites.
- Un modèle représente une famille de systèmes.

Modéliser un système permet

- de comprendre son fonctionnement
  - en le représentant pour
    - connaître les besoins
    - le spécifier,
    - le construire
    - le documenter
- Gérer la complexité
- Assurer sa cohérence
- Se focaliser sur les aspects essentiels
- Communiquer avec le client et dans l'équipe

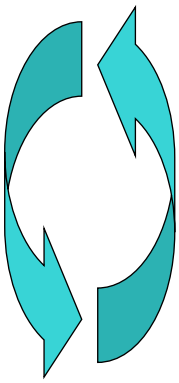
Le paradigme "Orienté Objet"

- Cette approche
  - Permet d'identifier les éléments du système
    - Pour en faire les objets
    - qui collaborent pour accomplir les services attendus.
  - Se base sur les concepts de base de l'objet
- La Conception Orientée objet est
  - Dans un premier temps ascendante
  - Mais pas uniquement
- Il faut penser
  - éléments essentiels du problème = cœur de la solution du système
  - dissocier l'activité de conception de l'activité de codage en LOO (Java ou C++) qui ont une implantation spécifique des concepts
- Alors, comment s'y prendre?

Pourquoi un langage de modélisation ?

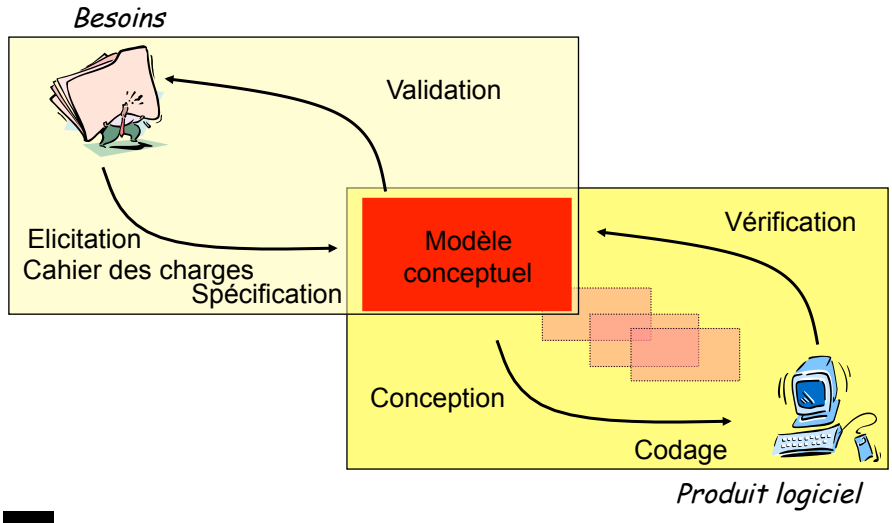
- Fournir un moyen d'abstraction
- Simplifier ⇒ Notation graphique
- Exprimer les artefacts produits par les activités de Génie Logiciel pendant les phases de développement
- Comparer des solutions pour structurer le système
- Ré-utiliser des modèles correspondant à des problèmes connus
- Assurer l'indépendance par rapport aux langages de programmation, aux plates-formes

Les activités d'ingénierie du logiciel



- Décomposer le modèle
  - en sous-modèles relativement indépendants
  - Un sous-modèle =
    - Phase analyse = une « vue »
    - Phase conception = un sous-système
  - Ébauche de l'architecture (= structure)
- Raffiner
  - Introduire des détails
- Transformer
  - Faire des choix de conception
  - Avoir des "équivalences" entre modélisation et codage
- Introduire la généricité
  - Solution d'un problème plus général qui peut se particulariser

Conception d'une solution informatique



SLR202 - 10

Des activités techniques fondamentales associées à la Modélisation

- La Validation
  - a pour objectif de s'assurer que le système ou le sous-système développé a la capacité à fournir les services attendus par le client et les utilisateurs finaux.
- La Vérification
  - ensemble des activités permettant d'assurer la conformité technique de réalisation du système ou du sous-système logiciel en développement

* Référence ISO 8402
Quality Management & Quality Assurance

SLR202 - 11

UML : le standard, les diagrammes.

Avant UML

- D'abord les langages de programmation
  - 1973: SMALLTALK
  - 1989 : Eiffel (B. Meyer)
  - 1993 : C++
  - 1995 : Java Ada95
- Puis apparition de **multiples méthodes OO**
  - HOOD (87);OOA(91);OMT(91);OOD(91); …
  - chacune avec sa notation
- Création de l'OMG : Object Management Group (en 89)
  - définir et promouvoir des standards (CORBA, UML…)
  - Sept. 97 *Unified Modeling Language* V1.1

SLR202 - 13

Les trois principaux auteurs initiaux (années 90-97)

- Grady Booch,
  - Methode Booch
- Jim Rumbaugh
  - OMT
- Ivar Jacobson
  - Objectory-OOSE

SLR202 - 14

Historique Unified Modeling Language

UML V2.5 beta (in process version)

UML V2.4.1 (2013)

UML V2.3 (2010)

UML V2.0 (2005)

UML 1.3 (06/99), 1.4 (02/01), 1.5 (03)

UML 1.0 Standard OMG (15/11/97)

UML V0.9

Unified Method V0.8

BOOCH93 OMT2

BOOCH91 OMT OOSE

Unified Modeling Language

SLR202 - 15

Object Management Group

- site http://www.omg.org/
- site http://www.uml.org/
  - Les spécifications du standard (2.5)
    - UML Superstructure
    - ...
  - Les contributeurs à la standardisation
  - Des liens vers
    - des tutoriaux
    - des outils industriels ou gratuits

consensus

SLR202 - 16

The OMG specification states:

- *"The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system.*
- *The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components."*

SLR202 - 17

L'index du standard UML …

- Diagrammes structurels (UML Structure)
  - Class diagram
  - Object diagram
  - Component diagram
  - Deployment diagram
  - Package diagram
  - Composite structure diagram
- Diagrammes comportementaux (UML Behavior)
  - Use case diagram
  - Activity diagram
  - Statechart diagram
  - Diagrammes d'interaction (UML Interaction)
    - Sequence diagram
    - Collaboration diagram
    - Communication diagram
    - Interaction overview diagram
    - Timing diagram
- + mécanisme d'extension
  - Stéréotype

SLR202 - 18

Le sous-ensemble de diagrammes UML

- Diagrammes structurels
  - Class diagram
  - Object diagram
  - Package diagram
- Diagrammes comportementaux
  - Use case diagram
  - Activity diagram
  - Statechart diagram
  - Diagrammes d'interaction
    - Sequence diagram
    - Collaboration diagram
- + mécanisme d'extension
  - Stéréotype

SLR202 - 19

UML n'est pas une méthode

- UML doit être intégré à un processus
  - peut être lui-même modélisé (workflow)
  - comprenant un cycle de développement du modèle
- Un même type de diagramme peut servir à modéliser différentes phases de développement
- Une méthode fixe les diagrammes appropriés à chaque phase

SLR202 - 20

Recommandation du standard : l'utilisation d'UML doit être

- Pilotée par les cas d'utilisation
  - Adéquation aux besoins des utilisateurs
  - Dans tout le cycle : de la spécification à la maintenance!
- Centrée sur l'architecture
  - Conçue pour satisfaire les besoins
  - Prendre en compte
    - les évolutions futures
    - Les contraintes de réalisation
- Suivre un processus itératif et incrémental
  - Décomposer en petites itérations
  - À partir des cas d'utilisation et de l'analyse des risques
  - Livraisons incrémentales du système

Donc il faut de la méthode

- Les modèles ne se suffisent pas
- Pour organiser les activités, il faut suivre une méthode basée sur
  - des règles d'utilisation de la notation,
  - un workflow entre les "parties prenantes" :
    - Utilisateurs, maîtrise d'ouvrage, maîtrise d'œuvre, informaticiens développeurs
    - un guide des « artefacts »
    - une démarche cohérente
  - Liée à un outil informatique

Le projet SLR202: phases initiales du cycle de développement

- L'expression des besoins
  - Élicitation des exigences fonctionnelles et non fonctionnelles
  - Cahier des charges
  - *Spécification* des exigences fonctionnelles *du logiciel*
    - Décrire ces exigences par des modèles (UML)
    - cf gabarit du document de spécification au standard IEEE 830-98 (adapté à la modélisation en UML)
- L'analyse
  - étude de ce qu'il "faut" modéliser (priorité, éléments essentiels)
- La conception
  - Modèles (UML) permettant de décrire les aspects
    - structurels
    - et comportementaux du système
- L'implémentation
  - conception détaillée, choix techniques, Codage et tests

SLR202 : 2 parties complémentaires

- Modélisation de la structure du logiciel
  - Validation par rapport aux besoins
  - Vérification de cohérence du modèle éventuellement transformation vers une implémentation (conception détaillée vers code)
    - Démarche de modélisation
    - Modélisation d'une étude de cas Système d'Information
- Modélisation comportementale
  - Modélisation et validation de la dynamique du système
  - Vérification de changement d'états en réponse à des évènements
  - Code => simulation d'autom

Mêmes diagrammes UML dont **Statechart**, c'est l'objectif de vérification qui prend un autre point de vue!