Génie Logiciel
et
Approche Orientée-Objets

Pierre-Alain Muller

Plan du cours

- Introduction
- Complexité des logiciels
- Les objets et les classes
- Les liens et les relations
- La classification
- Le processus
- Les patterns
- Des exemples et des exercices

Des bouquins

- Il y en a BEAUCOUP ...
- Object-Oriented Analysis and Design
  - Grady Booch, Benjamin Cummings
- Object-Oriented Modeling and Design
  - James Rumbaugh, Prentice Hall
- Object-Oriented Software Engineering
  - Ivar Jacobson, Addison-Wesley
- Managing the Software Process
  - Watts S. Humphrey

Génie Logiciel

- L’art et la manière de construire des logiciels
- Faire passer l’élaboration du logiciel du stade artisanal vers le stade industriel
- Un besoin plus sensible pour les grands projets

La crise du logiciel

- Identifiée dès 1968
- L'image de notre incapacité à maîtriser les coûts du développement de logiciel
- Le besoin de méthodes, de techniques et d'outils pour fabriquer les logiciels
- "Build the right system, build the system right"

Les différents fronts

- L’analyse et la conception (SA, RT, SADT, HOOD, Merise, ...)
- La programmation (Smalltalk, Ada, C++, L4G, SQL, ...)
- La documentation (2167, 2167a, ...)
- Les tests (Boîte blanche, boîte noire, ...)
- Les environnements (Concerto, Entreprise2, ...)
- Le cycle de vie (Cascade, V, W, ISO 9000,...)

L’approche orientée-objets

- Une vision unificatrice initialement basée sur une représentation simplifiée du monde réel
- Un état d’esprit applicable sur tous les fronts du génie logiciel
- La technologie incontournable de la prochaine décennie

L’âge de raison

- L’approche orientée-objets à 30 ans
- 1966 Une idée à Oslo
- 1969 La recherche au PARC (Xerox)
- 1980 Version industrielle de Smalltalk
- 1986 1200 personnes pour OOPSLA’86
- 1996 Omniprésence des solutions objets
- 2006 Stabilisation de la transition vers l’objet