Analyse, Conception des Systèmes Informatiques

Atelier de Génie Logiciel
Objecteering

O. Boissier, SMA/G2I/ENS Mines Saint-Etienne, Olivier.Boissier@emse.fr, **Septembre 2004**

Sommaire

- ✓ **Définition**
- Objecteering
- Principes
- Configuration et les outils
- Construction des modèles UML
- Générations
  - documentations
  - codes
- Design Patterns

Atelier de Génie Logiciel

- une ou plusieurs méthodes d 'analyse et de conception reconnues,
- éditeurs graphiques et syntaxiques,
- procédures de développement normalisées,
- générateurs de code, de documentation
- outils de génie logiciel tels que :
  - gestionnaire de projet, de configuration,
  - estimation de coûts,
  - qualimétrie, maintenance du logiciel, …

Objecteering

- AGL mettant en œuvre la notation UML
- développé par la société Softeam
- version 5.2.2.
- S'exécute sous Windows, Solaris
- analyse, conception UML
- générations de code (C++, Java, SQL, IDL, EJB…)
- rétro-ingéniérie (Java ->UML, C++->UML)
- outil ouvert :
  - intégration possible d 'autres outils (ex. SNIFF++)
  - développement de nouveaux modules
- www.objecteering.com

Objecteering (suite)

- Adaptation aux besoins de l 'utilisateur :
  - Création de nouveaux modules et utilisation, (Méta-projets, *Objecteering*/*UML Profile Builder*)
  - Création de nouveaux gabarits de documentation (*Objecteering*/*UML Profile Builder*)
  - Adaptation du processus de modélisation (Process Wizard) en y ajoutant des aspects spécifiques à des secteurs particuliers (critères de qualité, de productivité, …)

Objecteering (suite)

- Multi-utilisateurs : échange de données, des modèles entre les projets d 'utilisateurs différents
  - utilisant tous objecteering (*Objecteering*/*MultiUser*).
  - Utilisant des AGL différents utilisant UML (XMI, standard OMG pour les échanges de modèles)

Sommaire

- Définition
- Objecteering
- ✓ **Principes**
- Configuration et les outils
- Construction des modèles UML
- Générations
  - documentations
  - codes
- Design Patterns

Principes : Référentiel central

- bases : fichiers binaires contenant les informations liées aux objets (*.*ofp*), contenant :
  - projets : espaces de travail pour créer et utiliser des modèles.
  - méta-projets : environnements de développements pour les *profils* et *modules* UML.
- Administration par :
  - baseadm : administration et réparation des bases d 'un site

Principes : Modules

- Ensemble de fonctionnalités disponibles dans l 'outil, qu 'un utilisateur peut choisir de charger ou non.
- Choisi pour un projet (ex : choix des modules C++ generation, Oracle generation, documentation generation).
  - ➔*Menus* spécifiques pour chacun de ces modules, ensemble de *Marques* liées aux modules (méthode virtuelle en C++, index pour Oracle, etc.), ensemble de *Notes*, ensemble de *Fichiers* spécifiques manipulables dans l 'explorer, ensemble de *Gabarits* pour la documentation ou pour le code.

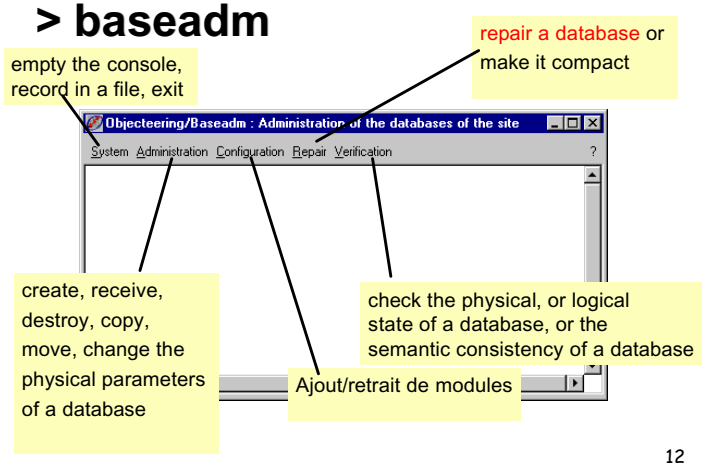
Principes : Modules existants

- Multi-user : gestion du travail collaboratif et d 'espaces de travail,
- XMI : génération et re-lecture de fichiers XMI.
- Documentation : Production de documentation spécifiques selon des plans prédéfinis ou adaptés à l 'organisation.
- C++ : génération de code C++, de Makefiles
- C++ Reverse Engineering :
- Java : génération de code Java, Makefiles, …
- Design Patterns pour Java/C++
- SQL : génération de code SQL
- Metrics : implémentation d 'un ensemble de métriques pour l 'évaluation de la qualité des modèles produits.
- CORBA : génération de code IDL.
- Continuus : gestion de la configuration.
- UML Profile Builder : définition de profiles UML, …

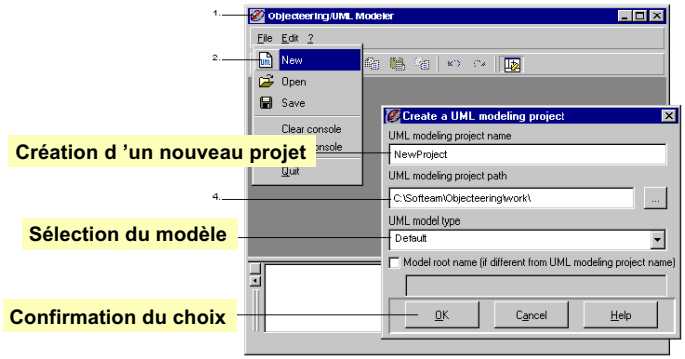
Configuration

- <installation_directory>/user/env.csh -> .cshrc ou <installation_directory>/user/env.sh -> .profile
- Variables d 'environnement à ne pas modifier :
  - OBJING_PATH : chemin d 'accès au répertoire d 'installation d 'Objecteering
  - OBJING_DATA : chemin d 'accès au répertoire qui assure la gestion du site.
  - OBJING_LANG : = us
  - OBJING_PRINTER : imprimante à utiliser
- Aide
  - setenv O_HELP_BROWSER netscape

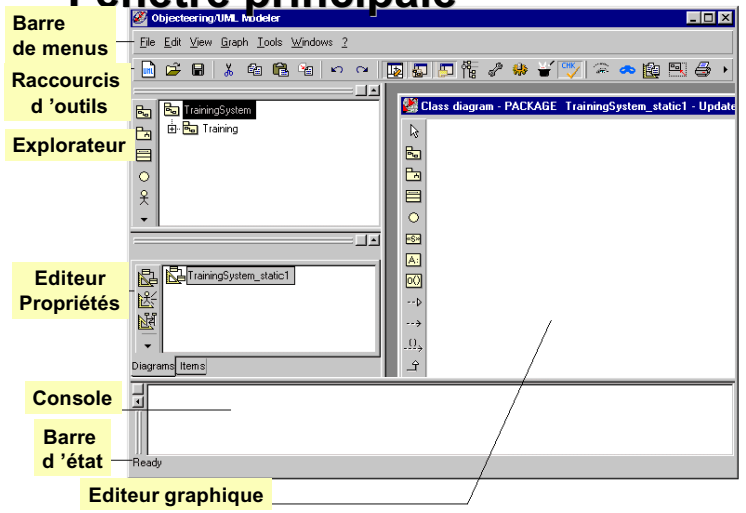
> baseadm



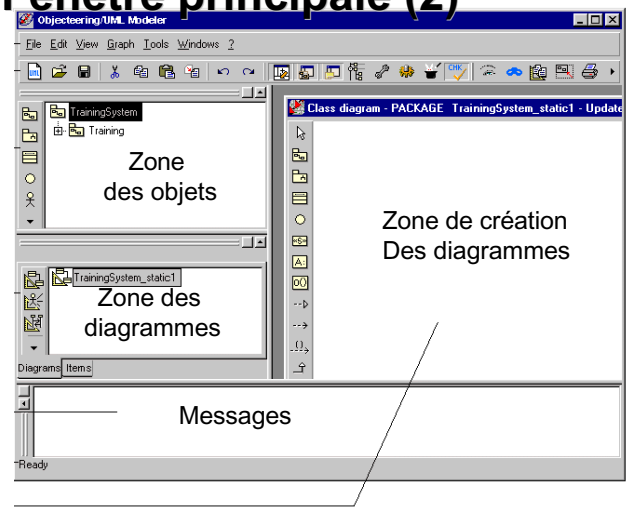
> objing



Fenêtre principale

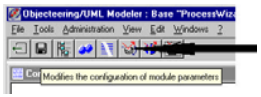


Fenêtre principale (2)

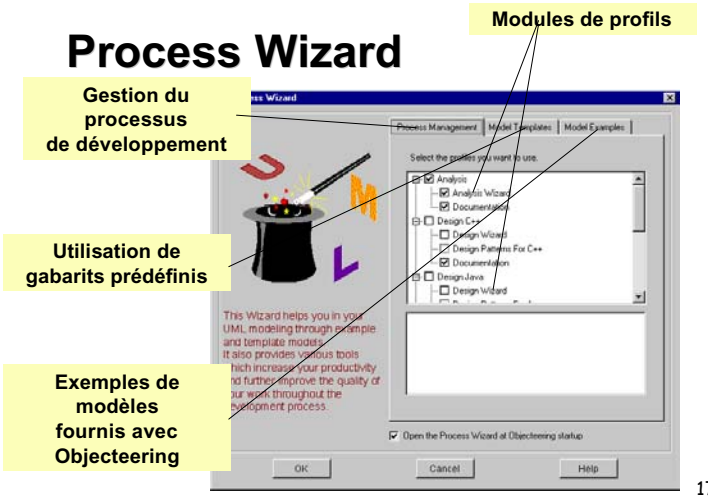


Process Wizards

- Aides disponibles pendant le processus de modélisation :
  - automatisation d 'un certain nombre d 'opérations (traçabilité, cohérence)
  - respect de critères de modélisation, vérification de la qualité des modèles
  - standardisation des résultats issus de différents utilisateurs,
  - adaptation du processus aux besoins de l 'organisation.
- Manipulation de *modules de profils* (extension d 'éléments de modélisation, règles de présentation des diagrammes, règles de validation, règles de transformation)
- Accessible au lancement, dans la barre de menus, dans le menu contextuel des items créés,
- changements de paramètres via



Process Wizard



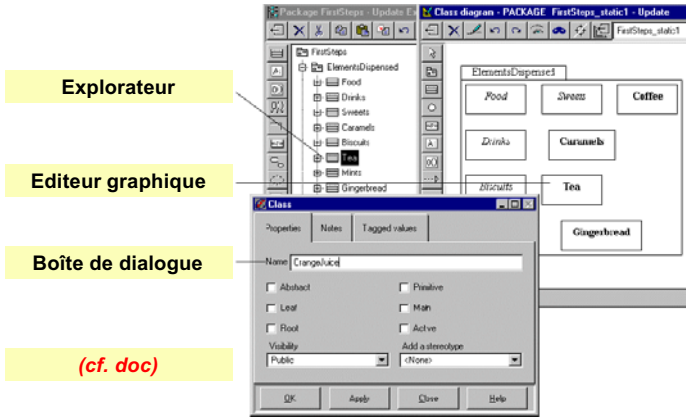
Sommaire

- Définition
- Objecteering
- Principes
- Configuration et les outils
- ✓ **Construction des modèles UML**
- Générations
  - documentations
  - codes
- Design Patterns

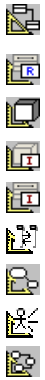
Construction des modèles UML

- Deux démarches possibles :
  - définition des modèles puis définition des instances, exemples de validation et d ’illustration des modèles ;
  - définition des instances puis construction des modèles correspondants ;

Construction des modèles UML



Diagrammes UML disponibles



- diagrammes de classes
- diagrammes de collaboration
- diagrammes de déploiement et composants
- diagramme d 'instance d 'un déploiement
- diagramme d 'objet
- diagramme de séquence
- diagramme d 'état
- diagramme de cas d 'utilisation
- diagramme d 'activité

Vérifications de cohérence

- Entrée de données : aide pendant la tâche de modélisation en fournissant la liste des éléments possibles selon l 'élément édité.
- Gestion de la cohérence : modifications dans une vue du modèle (éditeurs graphiques, explorateurs)
- vérification de la cohérence : vérification de la validité de l 'élément entré en fonction du modèle courant.
- ***(Cf. doc)***

Sommaire

- Définition
- Objecteering
- Principes
- Configuration et les outils
- Construction des modèles UML
- ✓ **Générations**
  - documentations
  - codes
- Design Patterns

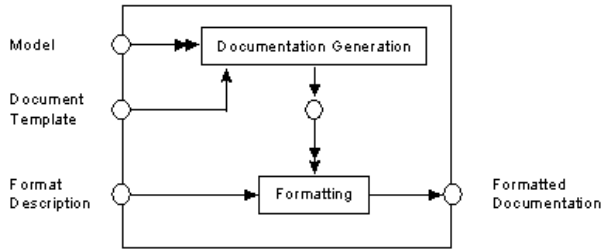
Génération de documentations

- Deux gabarits de documents disponibles:
  - document de spécification (analyse)
  - document de conception
- plusieurs formats de génération (Postscript, RTF, HTML, Ascii)
- Documentation paramétrée
- Documentation basée sur :
  - information des modèles,
  - valeurs d 'étiquettes (analysis, noanalysis, design, …),
  - notes (summary, description, …),
  - gabarit de document.

Génération de documentations (suite)

- Analyse (analysis.us, analysis.fr)
  - 1. Contexte général
  - 2. Glossaire, dictionnaire, description générale de l 'analyse (packages), utilisateurs et besoins (use case)
  - 3. Classes et packages
  - 4. Traçabilité entre les différentes notions des modèles.
- Conception (design.us, design.fr)
  - 1. Contexte général
  - 2. Architecture
  - 3. Conception générale
  - 4. Traçabilité
  - 5. Intégration

Génération de documentations (suite)



Génération de code JAVA

- Le module Objecteering/Java regroupe :
  - génération de code Java
  - compilation du code généré
  - génération de documentation java
  - rétro-ingéniérie de bibliothèques existantes,
  - Java Design Patterns
- Plusieurs possibilités de paramétrisation,
- Nécessité d'étiqueter des éléments du modèle, de définir des notes ***(cf. doc)***
- Compilation et correction des erreurs dans le modèle UML.

Design Patterns pour Java et C++

- Automatisation de la phase d 'implémentation d 'un Pattern.
- La phase de choix du Pattern à appliquer est laissée à l 'analyse et à la conception.
- Différents Patterns sont disponibles :
  - Singleton, State, Prototype, Memento, Visitor, Proxy, Adapter, Counted Pointer Idiom
  - Pour Java (Java Design Pattern) : RMI, EventSource, EventListerner, Finalize