Bases de la Programmation Orientée Objet / Java

DUT / Module M213

PPN Informatique 2013 - Page 30/67

Mission du module

➔ **Définie par le PPN Informatique 2013**

Savoir développer un programme dans un ***langage de P.O.O.****,* ***à partir d'une conception détaillée****.*

- approche industrielle du développement des logiciels (software factories)
- processus et cycle de vie des logiciels applicatifs
- position centrale dans le cycle
- contraintes de productivité et de maintenance

Cycle de vie des logiciels

➔ **Plusieurs modèles théoriques**

- Maîtrise d'ouvrage / Maîtrise d'oeuvre
- Nécessité impérative d'une expression de besoin formalisée (suivant modèles types / URD)
- mise en œuvre ensuite de méthodes d'analyse et de conception
- enchaînement d'étapes parfaitement identifiées et formalisées, avec documentation type associée (basée sur des diagrammes et des pictogrammes)

Travaux décrits dans le module

➔ **Dans la continuité des étapes amont**

- exploitation de la documentation et des diagrammes obtenus (diagrammes structurels, diagrammes comportementaux et diagrammes d'interaction)
- approche fragmentée du code opérationnel et des tests unitaires associés (**analogue TAD**)
- mise en œuvre ensuite d'un processus d'intégration des modules et de validation (Cf modules PPN complémentaires DUT/S3 et DUT/S4)

Points clés décrits dans le PPN

- Concepts fondamentaux de la P.O.O. (encapsulation, composition, polymorphisme, héritage, …)
- Lecture d'une conception détaillée orientée objet (UML notamment)
- Développement et mise en œuvre des tests unitaires
- Utilisation de briques logicielles, API, bibliothèques, … (composants logiciels communs)
- Sensibilisation aux bonnes pratiques de la prog. (gestion de versions notamment)

Choix complémentaires

- Choix d'une méthodologie d'analyse et de conception (Unified Modeling Language / **UML**)
- Choix d'un environnement de conception (**Visual Paradigm**)
- Choix d'un langage orienté objets (**Java**)
- Choix d'un environnement de développement (IDE **JCreator** pour S2 et **Eclipse** pour S3/S4)

Objectifs du cours

➔ Transmettre à l'auditoire la connaissance des concepts et techniques suivants :

- le **paradigme objet** et ses avantages
- la place de ce paradigme dans le cycle de vie
- liens avec le module M214 (notamment UML)
- **l'usage des objets en phase de programmation**
- **l'implémentation en Java du paradigme objet**
- les tests unitaires d'une application Java

Pré-requis indispensable

➔ **Modules M112 & M113 (S1)**

- élaboration d'algorithmes
- programmation en langage impératif
- structuration des données en mémoire
- maîtrise du langage C
- mécanismes d'allocation mémoire
- **concepts et mise en œuvre des T.A.D.**

Module connexe M214

Comprendre et modéliser une conception détaillée (démarche modulaire)

- modélisation objets pour l'analyse et la conception détaillée par exemple en UML
- gestion des versions dans le développement
- documentation du code
- sensibilisation aux bonnes pratiques de la conception et du développement

Module connexe M224

Gestion de projets informatique

- la démarche projet et les acteurs d'un projet
- l'équipe projet et la répartition des rôles
- le cahier des charges : analyse des besoins
- tâches, planification, enchaînement, cycle de vie
- outils d'ordonnancement (graphe de Pert, diagramme de Gantt)
- documentation

Modules complémentaires (1)

M313 / Algorithmique avancée (java)

Savoir mettre en œuvre des structures de données avancées (y compris récursives) et les algorithmes qui les manipulent

M315 / Conception et programmation avancées

Produire une conception détaillée en appliquant des patrons de conception, la mettre en œuvre en utilisant des bonnes pratiques de programmation orientée objet

Modules complémentaires (2)

M412 / Programmation répartie (java)

Savoir programmer une application répartie (multi processus / multi threads / distribuée sur un réseau)

Objectifs des T.D. & T.P.

➔ **Permettre aux étudiants de :**

- vérifier et approfondir les concepts introduits en cours
- appliquer le paradigme objet sur des cas simples
- exploiter tous les aspects du langage Java
- mettre en œuvre la gestion des exceptions
- développer des modules de tests unitaires
- analyser et modifier des classes écrites par d'autres
- mettre en œuvre les collections (java.util.*)
- mettre en œuvre les fichiers (java.io.*)

Contrôle des connaissances (1)

➔ **Contrôle continu en TD/TP**

- Contrôle individuel partiel lors de chaque séance
- Contrôle de la maîtrise du cours
- Rendu périodique des TP **en fin de séance**

➔ **1 I.E. au cours du semestre**

- Durée 1H30 / 4 exercices / En salle d'examen
- Documents limités à une feuille manuscrite recto/verso

Contrôle des connaissances (2)

➔ **1 D.S. global en fin de module**

- Durée 1H30 / 4 exercices / En salle d'examen
- Documents limités à une feuille manuscrite recto/verso

Intervenants pédagogiques

➔ **Cours** (10H)

- A. Thuaire – Responsable du module

➔ **TD/TP** (20H + 30H)

- L. Brenac (2 groupes)
- R. Lecat (1 groupe)
- P. Perry (1 groupe)
- F. Rallo (1 groupe)
- A. Thuaire (1 groupe)

Bibliographie (1)

➔ **Livre électronique en ligne**

- Penser en Java - Bruce ECKEL (*http://java.developpez.com/livres/penserenjava*)
- Apprenez à programmer en Java – C. HERBY (*http://fr.openclassrooms.com/uploads/fr/ftp/livre/java/*)

Site du zero

Bibliographie (2)

➔**Polycops en ligne**

- E. Lefrancois (Université de Genève) M213
- G. Valet (Lycée Diderot) M213
- P. Gérard (IUT Villetaneuse – Paris XIII) M214

Bibliographie (3)

➔**Autres sources**

- Privilégier les livres d'initiation (Java et UML)
- Tutoriaux des sites **oracle** et **developpez**.com

➔**Documentation en ligne**

http://docs.oracle.com/javase/6/docs/api/

(package java.lang)

Bibliographie (4)

➔ **Site de référence**

http://www.oracle.com/fr/technologies/java/overview/