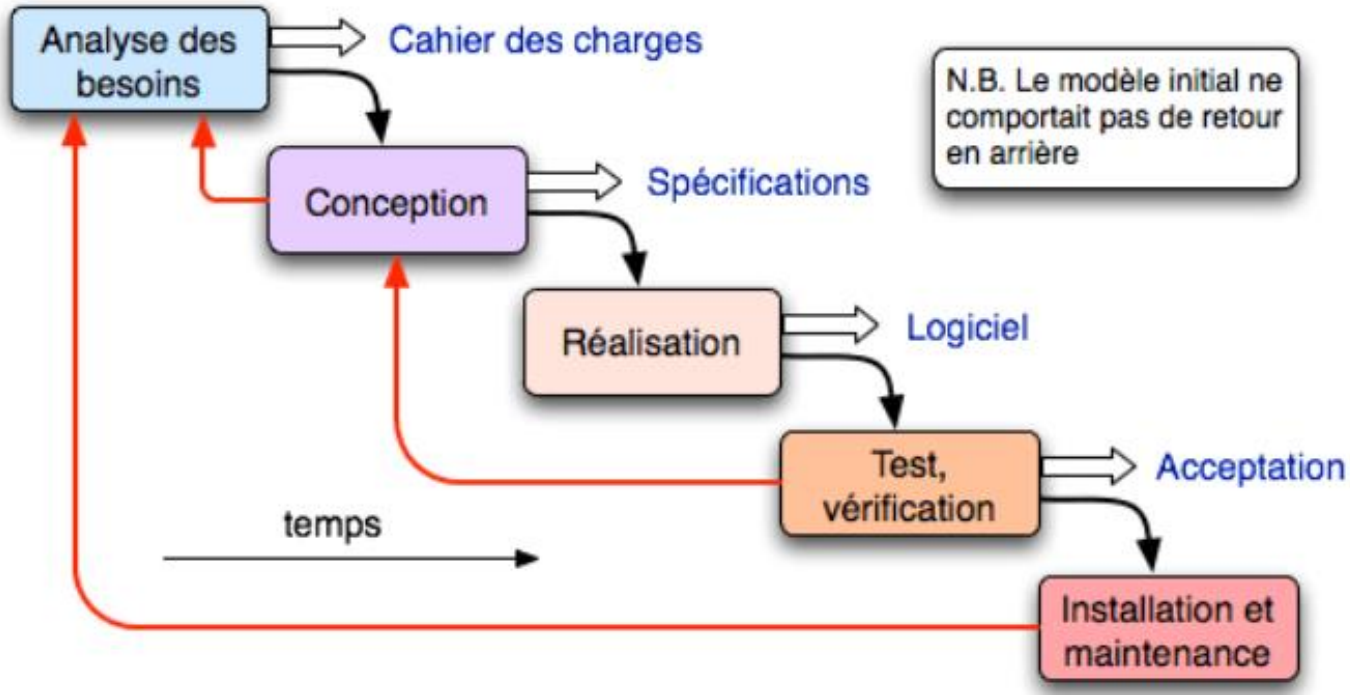
Génie Logiciel

- Dispensé par Dr. Msc. Ir. **MWAMBA KASONGO Dahouda**
  Docteur en génie logiciel et systèmes d'information
  Machine and Deep Learning Engineer

- Assisté Master Student, Ir. Jason MUSA

Heure : 08H00 – 12H00



Fiche matière

Universite Protestante de Lubumbashi
Faculté de Sciences Informatiques

Enseignant responsible de la matière : Génie Logiciel

Contact: dahouda37@gmail.com/ dahouda37@upl-univ.ac

Matière : Génie Logiciel

Filière : BAC4 Génie Logiciel (GL)

Crédit: 4 [60H]

Volume horaire d'enseignement hebdomadaire: Cours (Nombre d'heures par semaine): 16H

Objectif du cours

❖ Prérequis

- Notions sur les systèmes d'informations
- Notion sur la Programmation oriente objet

❖ Objectifs généraux

À la fin de ce cours, les étudiants seront capables de :

- - Comprendre les phases clés du cycle de vie du développement logiciel (SDLC) et l'importance de chaque phase dans la création de produits logiciels réussis.
- - Comprendre la vision globale du processus de développement logiciel.

Objectif du cours

Objectifs spécifiques

- Initier les étudiants aux outils essentiels pour la conception, la construction et la maintenance de logiciels de qualité.

- Introduire aux étudiants les bonnes pratiques, les méthodologies de développement logiciel et les normes de l'industrie.
- Apprenez à recueillir, analyser et documenter les exigences logicielles des parties prenantes.
- Acquérir des connaissances sur les principes de conception de logiciels et les modèles d'architecture.
- Capacité à analyser des problèmes complexes, à les décomposer et à mettre en œuvre des solutions efficaces.
- Expérience dans l'application de la pensée logique et critique pour résoudre des problèmes de codage et de conception.

Fiche matière

Contenu de la matière

Chapitre 1. Introduction au génie logiciel

Chapitre 2. Modélisation avec UML

Chapitre 3. Diagramme de cas d'utilisation: vue fonctionnelle

Chapitre 4. Diagrammes de classe et d'objet : vue statique

Chapitre 5. Diagrammes de séquence, activité et état/ transition: vue dynamique

Chapitre 6. Autres notions et diagrammes UML

Chapitre 7. Introduction aux méthodes de développement : (RUP, XP)

Chapitre 8. Patrons de conception et leur place au sein du processus de développement

Contenu de matière

Chapitre 1. Introduction au génie logiciel

- Ce chapitre traite de l'évolution du génie logiciel, expliquant pourquoi cette discipline a émergé.
- Il couvre des concepts de base comme les définitions du génie logiciel, les raisons pour lesquelles il est important et ses objectifs principaux, tels que la création de logiciels de qualité, respectant les délais et les coûts.

Chapitre 2. Modélisation avec UML

UML est un langage de modélisation standardisé utilisé pour représenter les aspects conceptuels et techniques des systèmes logiciels. Ce chapitre explique comment utiliser UML pour concevoir et documenter des systèmes complex es de manière graphique et structurée.

Contenu de matière

Chapitre 3. Diagramme de cas d'utilisation: vue fonctionnelle

- Ce chapitre introduit les **diagrammes de cas d'utilisation**, qui décrivent les interactions entre les utilisateurs (acteurs) et le système.

- Il s'agit d'une vue fonctionnelle du système, illustrant ce que le système est censé faire du point de vue de l'utilisateur.

Chapitre 4. Diagrammes de classe et d'objet : vue statique

- Les **diagrammes de classe** montrent la structure statique d'un système, c'est-à-dire les classes et les relations ent re elles (héritage, association, etc.).

- Les **diagrammes d'objet** représentent des instances spécifiques de ces classes. Ce chapitre traite de la modélisati on statique du système

Contenu de matière

Chapitre 5. Diagrammes de séquence, activité et état/ transition: vue dynamique

Ce chapitre couvre la **modélisation dynamique** du comportement du système à travers :

- ✓ **Diagrammes de séquence** : ils montrent les interactions entre les objets dans un ordre temporel spécifique.
- ✓ **Diagrammes d'activité** : ils représentent le flux de contrôle ou de données dans le système.
- ✓ **Diagrammes d'état/transition** : ils montrent les états possibles d'un objet et les transitions d'un état à un autre en fonction des événements

Chapitre 6. Autres notions et diagrammes UML

Ce chapitre explore d'autres types de diagrammes UML (comme les diagrammes de composants, de déploiement) et de s concepts complémentaires pour une modélisation plus détaillée ou spécialisée.

Contenu de matière

Chapitre 7. Introduction aux méthodes de développement : (XP, UP, RUP)

Ce chapitre introduit différentes **méthodologies de développement logiciel**, notamment :

- ✓ **XP (Extreme Programming)** : une méthodologie agile qui met l'accent sur la flexibilité et la réactivité aux changements.
- ✓ **UP (Unified Process)** et **2TUP** : des variantes ou extensions du processus unifié
- ✓ **RUP (Rational Unified Process)** : un cadre itératif et incrémental pour le développement logiciel.

Chapitre 8. Design patterns MVC et leur place au sein du processus de développement

Le Chapitre 8 traite des patrons de conception (design patterns), et plus spécifiquement du pattern MVC (Modèle-Vue-Contrôleur), qui est l'un des plus populaires dans le domaine du développement logiciel, en particulier pour les applications web.

Fiche matière

Mode d'evaluation : Moyenne anuelle

Moyenne	Points
Presence au cours	10 pts
TD	10 Pts
TP	10 pts
Interrogation	20 pts
Total Moyenne annuelle	/50 pts

Références bibliographiques

1. Introduction to Software Engineering, Ronald J. Leach, CRC Press, Taylor & Francis Group, 2016

2. Uml 2 pratique de la modélisation, Benoît Charroux, Yann Thierry-Mieg, Aomar Osmani

3. M. Blaha et J. Rumbaugh. Modélisation et conception orientées objet avec UML 2. 2ème édition Pearson Education, 2005

4. Sommerville, I. (2010). *Software Engineering* (9th ed.). Pearson Education.

5. Pressman, R. S. (2014). *Software Engineering: A Practitioner's Approach* (8th ed.). McGraw-Hill.

6. Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The Unified Modeling Language User Guide* (2nd ed.). Addison-Wesley.

7. Larman, C. (2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Devel opment*. Prentice Hall.

8. Fowler, M. (2003). *UML Distilled: A Brief Guide to the Standard Object Modeling Language* (3rd ed.). Addison-Wesley.

9. Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*.

10. Schmidt, D. C., Stal, M., Rohnert, H., & Buschmann, F. (2000). *Pattern-Oriented Software Architecture: Patterns for Concu rrent and Networked Objects* (Vol. 2).

Références bibliographiques

11. **Hitz, M., Kappel, G., & Schrefl, M. (2021).** *Object-Oriented and Entity-Relationship Modelling*. Springer.
12. **Sommerville, I. (2020).** *Software Engineering (10th Edition)*. Pearson.
13. **Booch, G., Rumbaugh, J., & Jacobson, I. (2020).** *The Unified Modeling Language User Guide*. Addison-Wesley.
14. **Ambler, S. W. (2020).** *Agile Modeling: Effective Practices for eXtreme Programming and the Unified Process*.
15. **Fowler, M. (2019).** *UML Distilled: A Brief Guide to the Standard Object Modeling Language*. Addison-Wesley.
16. **Pressman, R. S., & Maxim, B. (2020).** *Software Engineering: A Practitioner's Approach (9th Edition)*. McGraw-Hill.
17. **Rubin, K. S. (2019).** *Essential Scrum: A Practical Guide to the Most Popular Agile Process*. Addison-Wesley.
18. **Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (2019).** *Design Patterns: Elements of Reusable Object-Oriented Softw are*.
19. **Larman, C. (2021).** *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative D evelopment*. Prentice Hall.
20. **Patterson, D. A., & Hennessy, J. L. (2020).** *Computer Organization and Design: The Hardware/Software Interface*.