Plan de cours

IFT719 – Processus du génie logiciels (hiver 2010**)**

Enseignant

Luc LAVOIE
Courriel :Luc.Lavoie@USherbrooke.ca
Bureau : D4-1010-12
Téléphone : (819) 821-8000 poste 62015
Site : http://pages.usherbrooke.ca/llavoie/
Disponibilité : sur rendez-vous.

Horaire

Lundi	08:30 à 10:20	D4-2022
Mercredi	08:30 à 10:20	D4-2022

Version et statut

1.1.0 - en vigueur (comprend les changements aux dates de remise des rapports)
1.0.0 – version initiale en date du 2010-01-11

---

1 Introduction

1.1 Objet et portée du document

Le document décrit l'activité IFT719 « **Processus du génie logiciels** » offerte au trimestre d'hiver 2010 par le Département d'informatique de la Faculté des sciences. On y présente les objectifs, le contenu, l'organisation et les modalités d'évaluation du cours.

1.2 Définitions

IEEE	*The Institute of Electrical and Electronics Engineers, inc.*
PGC	plan de gestion de configuration (IEEE SCMP *software configuration management plan*).
PGP	plan de gestion de projet (IEEE SPMP *software project management plan*).
SAS	spécification d'architecture du système.
SCL	spécification de conception du logiciel (IEEE SDD *software design document*).
SES	spécification des exigences du système (IEEE SRS *software requirement specification*).
UML	*Unified Modeling Language*.

1.3 Références

1.3.1 Références essentielles

[IGL301]
COLLECTIF GL ;
IGL301 – Spécification et validation des exigences (notes complémentaires et synthétiques).
http://pages.usherbrooke.ca/llavoie/enseignement/IGL301
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Sherbrooke, Canada, janvier 2009.

[GLOGUS]
LAVOIE, Luc ;
GLOGUS – recueil de modèles de documents pour le développement logiciel.
http://pages.usherbrooke.ca/llavoie/glogus.php
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Sherbrooke, Canada, janvier 2009.

1.3.2 Références importantes

[Bray2002]
K. BRAY;
An Introduction to requirements engineering;
Addison-Wesley, 2002.
ISBN 0-201-76792-9 ; UdeS QA 76.758 B744 2002.

[Elmasri2007]
ELMASRI, Ramez ; NAVATHE, Shamkant B.;
Fundamentals of database systems
Fifth Edition, Pearson Addison Wesley, 2007.
ISBN 0-321-36957-2.

[Leffingwell2003]
D. LEFFINGWELL, D. WIDRIG;
Managing software requirements – A use case approach ;
2nd edition, Addison-Wesley, 2003.
ISBN 0-321-12247-X; UdeS QA 76.76 D47L44 2003.
[WOL2004]
Wall-On-Line : l'e-gouvernement wallon,
La boîte à outils : 15 méthodes d'implication des utilisateurs,
http://egov.wallonie.be/boite_outils_methodes/index.htm
(version en date du 17 décembre 2004 consultée le 11 mai 2007)
également disponible sous
http://pages.usherbrooke.ca/llavoie/projets/GLOGUS/wall-on-line.pdf

1.3.3 Références utiles

[Braude2001]
Eric J. BRAUDE;
Software engineering: an object-oriented perspective;
John Wiley & sons, 2001;
ISBN 0-471-32208-3 [QA 76.758 B74 2000]
[Davis2007]
M. Davis ;
Requirements Bibliography,
http://web.uccs.edu/adavis/UCCS/reqbib.htm
(consulté le 2007-03-15)
[GDT]
Grand dictionnaire terminologique.
Office québécois de la langue française.
http://www.grandictionnaire.com
(consulté le 2008-12-15).
[Hull2004]
E. HULL, K. JACKSON, J. DICK;
Requirements engineering;
2/E, Springer, 2004;
[TA 168 H85 2005]
[IEEE1233]
IEEE Guide for Developing System Requirements Specifications;
IEEE Std 1233-1998, IEEE, New York, 1998;
[QA 76.76 S73I438 1998 – disponible au comptoir de la bibliothèque de Sciences et Génie]
[IEEE830]
IEEE Recommended Practice for Software Requirements Specifications;
IEEE Std 830-1998, IEEE, New York, 1998;
[QA 76.76 S73I44 1998 – disponible au comptoir de la bibliothèque de Sciences et Génie]
[IEEE12207]
Industry Implementation of International Standard ISO/IEC 12207-1995;
IEEE 12207, IEEE, New York, 1995.
[QA 76.76 S73I44 1998 – disponible au comptoir de la bibliothèque de Sciences et Génie]
[ISO12207]
ISO/IEC 12207 - Information Technology—Software Life-Cycle Processes;
1995.

[Jackson1995]
Michael JACKSON;
Software Requirements & Specifications;
Addison Wesley, 1995; ISBN 0-201-87712-0.
[Jackson2001]
Michael JACKSON;
Problem frames;
ACM Press Book, Addison Wesley, 2001; ISBN 0-201-59627-X.
[Jacobson1999] (traduit en français, voir [Jacobson2000])
Ivar JACOBSON, Grady BOOCH, James RUMBAUGH;
The unified software development process;
Addison-Wesley, 1999; ISBN 0-201-57169-2.
[Jacobson2000] (traduction de [Jacobson1999])
Ivar JACOBSON, Grady BOOCH, James RUMBAUGH;
Le processus unifié de développement logiciel;
Eyrolles, 2000 ; ISBN 2-212-09142-7 ; [UdeS 76.76 D47J3514 2000].
[Larman2005]
Craig LARMAN;
Applying UML and patterns - an introduction to object-oriented analysis and design and iterative development;
3/E, Prentice-Hall, Upper Sadel River (NJ), 2005; ISBN 0-13-148906-2.
[Lauesen2002]
S. Lauesen;
Software Requirements: Styles and Techniques;
Addison Wesley Professional, 2002; [QA 76.754 L38 2002].
[Pressman2005]
PRESSMAN, Roger S.
Software Engineering - A practioner's Approach.
Sixth Edition, McGraw-Hill, 2005; ISBN 0-07-301933-X.
[Sommerville2007]
SOMMERVILLE, Ian
Software Engineering.
Height Edition, Addison-Wesley, 2007; ISBN 978-0-321-31379-9.
[VanVliet2008]
VAN VLIET, Hans
Software Engineering - Principles and Pratice.
Third Edition, Wiley, 2008; ISBN 978-0-470-03146-9.

2 Présentation

2.1 Mise en contexte

Le génie logiciel traite de la configuration d'une machine universelle (ordinateur) dans le but d'atteindre un objectif spécifique. Le logiciel de configuration peut lui aussi être vu comme une machine, mais il diffère des autres machines en ce sens qu'il est intangible. Le génie logiciel doit son nom et sa constitution comme un domaine de connaissance propre à la tenue d'un séminaire organisé par l'OTAN à Garmisch-Partenkirchen en Autriche en 1968. Böhm et Bauer en sont probablement les parrains.

Le logiciel de configuration d'une machine universelle est désigné sous plusieurs appellations différentes, selon la caractéristique mise de l'avant : logiciel (intangibilité), programme (déterminisme), système (complexité).

Puisqu'on construit généralement un système pour atteindre un but donné, il est préférable de déterminer et de détailler d'abord quel est ce but. Ce qui nous amène à l'ingénierie des exigences, la partie du génie logiciel qui permet de déterminer quel système sera développé.

L'élaboration d'une architecture logicielle découle de cette première étape. Elle a pour but de délimiter les unités organiques qui permettront de réaliser les fonctions du logiciel de façon efficace, traçable et modifiable (pour ne nommer que quelques-unes des propriétés recherchées).

Même si dans certains cas il est possible d'induire l'architecture de la spécification formelle, la vérifiabilité et la validité du produit résultant nécessitent un examen indépendant. Voilà une troisième étape.

Les composants organiques doivent communiquer entre eux et les humains interagir avec certains d'entre eux. L'ingénierie des interfaces machine-machine et des interfaces personne-machine intervient alors dans une quatrième étape.

La conception globale et la conception détaillée, souvent regroupées sous la désignation de programmation, visent à construire effectivement chacun des composants organiques et des composants d'interface. La conception globale s'intéresse aux algorithmes, aux structures de données et à la complexité algorithmique des solutions proposées pour incarner les composants logiciels. La conception détaillée s'intéresse à l'expression des solutions élaborées lors de la conception globale. Parfois considérées comme deux étapes distinctes, parfois comme une seule étape, la conception globale et la conception détaillée permettent la concrétisation du logiciel effectif qui devient alors produit.

Les frontières entre chacune de ces étapes ne sont pas arbitraires, mais ne sont pas toujours tranchées. L'ordre dans lequel elles réalisées peut varier considérablement selon qu'il s'agit d'un projet prédictif ou expérimental. L'agencement des activités et des tâches en phases, en vagues, en cascades, en itérations ou en spirales nécessite de nombreuses adaptations. Plusieurs procédés ont donc vu le jour qui, lorsqu'ils sont contextualisés, déterminent les processus de développement effectivement utilisés.

Ces processus de développement sont le plus souvent réalisés dans le contexte d'un projet dont la gestion nécessite elle-même plusieurs processus (gestion de portée, de durée, de coût, des ressources humaines, des risques, etc.). L'interaction entre tous ces processus est alors déterminante pour le succès du projet.

Le cours s'intéressera cependant principalement aux processus techniques, liés au développement proprement dit du logiciel.

2.2 Fiche signalétique

Objectif

Effectuer l'analyse du processus même de développement des logiciels; utiliser et appliquer les techniques de réingénierie et de réutilisation.

Contenu

Bref aperçu des approches et des normes du développement de logiciels. Étude de quelques cycles de base de développement de logiciels par le paradigme de décision/justification. Illustration sur des exemples. Approches de réingénierie et de rétro-ingénierie des logiciels: limites et

perspectives. Techniques de réutilisation des logiciels. Environnements et ateliers de développement assisté des logiciels. Études de cas.

Crédits

3

Organisation

Cours : 3 heures par semaine
Travail personnel : 6 heures par semaine

Référence

http://www.usherbrooke.ca/fiches-cours/ift719.htm

2.3 Objectifs spécifiques

Au terme de cette activité pédagogique, la personne l'ayant réussie sera capable de :

- ◊ planifier et diriger le processus de développement et, plus particulièrement celui d'ingénierie des exigences ;
- ◊ optimiser les relations entre les différents processus logiciels ;
- ◊ contrôler, superviser et maîtriser les processus de développement logiciels.

3 Contenu

1. **Revue des principaux procédés de développement logiciel**
2. **Processus d'ingénierie des exigences**
3. **Revue des techniques d'exploration**
4. **Documentation des processus de développement logiciel**
5. **Méthodes d'analyse, de spécification et de suivi des exigences**
6. **Techniques d'analyse, de spécification et de suivi des exigences**
7. **Des spécifications aux essais**
8. **Des spécifications à l'architecture**

4 Organisation

4.1 Modalités d'enseignement

Les participants sont tenus d'assister aux cours de l'activité IGL301 et de produire un rapport synthèse de chacun des modules, complété par un approfondissement fondé sur une recherche personnelle à partir de travaux scientifiques récents. Les dates de remise des rapports sont indiquées au calendrier (voir 4.3).

4.2 Modalités d'évaluation

En plus des deux examens individuels, l'évaluation porte sur :

◊ huit rapports de synthèse (un par thème du contenu) ;
◊ un séminaire sur un sujet de recherche déterminé conjointement avec l'enseignant.

La durée de l'examen de mi-trimestre est de 110 minutes et celle de l'examen final est de trois heures – aucune documentation n'est permise et l'usage d'appareils informatiques, électroniques ou de communication (ordinateur, calculatrice, téléphone, etc.) est interdit.

Tableau 1 – Sommaire des évaluations

Évaluation	Valeur	Commentaire
Examen de mi-trimestre	20 %	Individuel
Examen final	40 %	Individuel
Rapports de synthèse	24 %	Individuel (8 x 3 %)
Séminaire	16 %	Individuel
Total	100 %	

Tout étudiant, toute étudiante, qui omet de remettre un travail au moment prescrit par l'échéancier doit rencontrer l'enseignant afin de déterminer une nouvelle date de remise. Dans tous les cas, une pénalité de 10 % par jour de retard est imposée.

L'évaluation est faite en tenant compte de la clarté des documents et du respect de la méthodologie de gestion de projet. Conformément aux articles 36, 37 et 38 du règlement facultaire d'évaluation des apprentissages[1], l'enseignant peut retourner à l'étudiante ou à l'étudiant tout travail non conforme aux exigences quant à la qualité de la langue et aux normes de présentation.

Toute situation de plagiat sera traitée en conformité, entre autres, avec l'article 8.1.2 du Règlement des études[2] de l'Université de Sherbrooke.

En cas de circonstances extraordinaires au-delà du contrôle de l'université de Sherbrooke et sur décision de celle-ci, l'évaluation des apprentissages de cette activité est sujette à changement.

1 http://www.usherbrooke.ca/accueil/documents/politiques/pol 2500-008/pol evaluation/sciences.html
2 http://www.usherbrooke.ca/programmes/etude

4.3 Calendrier

Tableau 2 – Calendrier des activités

N°	date	activité	contenu	évaluation (remise)
1	2009-01-04	cours	Introduction	
2	2009-01-11	cours + TD	1	
3	2009-01-18	cours + TD	2	Rapport 1
4	2009-01-25	cours + TD	3	
5	2009-02-01	cours + TD	4	Rapport 2
6	2009-02-08	cours + TD	5-6	
7	2009-02-15	étude	5-6	
8	2009-02-22	examen	(1 – 6)	examen de mi-trimestre
9	2009-03-02	relâche		Rapports 3 et 4
10	2009-03-08	cours + TD	5-6	
11	2009-03-15	cours + TD	5-6	Rapport 5
12	2009-03-22	cours + TD	5-6	
13	2009-03-29	cours + TD	7	Rapport 6
14	2009-04-05	cours + TD	7-8	
15	2009-04-12	cours + TD	8	Rapport 7
16	à déterminer	examen	(1 – 8)	examen final
17	à déterminer	séminaires		Rapport 8

Notes :

- Les numéros des rapports renvoient à ceux des thèmes du contenu (section 3).
- Les jours de remise sont les mercredis.
- Il y a relâche le 4 janvier (Nouvel An), le 20 janvier (Carnaval) et le 5 avril (lundi de Pâques).
- La date de l'examen de mi-trimestre est fixée par la Faculté (le 22 ou le 24 février).
- La date de l'examen final est fixée par la Faculté (entre le 13 et le 23 avril).
- La date des séminaires sera fixée après publication de la date d'examen final.