

UNIVERSITÉ DE MONTRÉAL

L'ÉTUDE DES FORCES ET FAIBLESSES DES LOGICIELS LIBRES
DANS LA GESTION DE PROJET DISTRIBUÉE

MICHAËL DAUDIGNON
DÉPARTEMENT DE MATHÉMATIQUES
ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INDUSTRIEL)
DÉCEMBRE 2008

©Michaël Daudignon, 2008.



Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-47657-4
Our file Notre référence
ISBN: 978-0-494-47657-4

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.


Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

L'ÉTUDE DES FORCES ET FAIBLESSES DES LOGICIELS LIBRES
DANS LA GESTION DE PROJET DISTRIBUÉE

présenté par : DAUDIGNON Michaël

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. TRÉPANIÉ Martin, ing., Ph.D., président

M. BAPTISTE Pierre, ing., Doct., membre et directeur de recherche

M. BOURGAULT Mario, ing., Ph.D., membre et codirecteur de recherche

M. PELLERIN Robert, ing., Ph.D., membre

Dédicace

La dédicace du présent mémoire s'adresse à toutes les personnes qui sont convaincues que le savoir doit être partagé entre tous et qui participent activement à son partage.

Remerciements

Je tiens tout d'abord à remercier Monsieur Pierre Baptiste, mon directeur de recherche, qui a su m'accompagner, me guider et m'encourager de manière idéale selon moi, tout au long de mon travail. Mes remerciements s'adressent également à celles et ceux qui m'ont aidé à l'accomplissement du module *File Manager* pour la plate-forme *Moodle*, à savoir, Marie-Josée Leblanc, Dominic Villard, Sylvain Lefebvre, Valéry Frémaux et Romain Lombardo. Un grand merci à messieurs Trépanier et Bourgault pour leurs critiques constructives tant sur le plan de la recherche que sur celui de l'amélioration du module. Je souhaite remercier chaleureusement Michel Cambron, avec qui j'ai eu la chance de partager des conversations très enrichissantes sur la gestion de projet et ses implications en termes de fonctionnalités informatiques. De plus, je tiens à souligner que le soutien de mon entourage m'a été indispensable pour accomplir ce travail. Je remercie enfin le département de mathématiques et génie industriel pour le financement ainsi que tout le personnel de l'École Polytechnique Montréal pour avoir contribué aux excellentes conditions dans lesquelles j'ai travaillé.

Résumé

La gestion de projet moderne et l'informatique sont liés en ce sens que le gestionnaire de projet utilise des outils informatiques, tant la complexité des projets qu'il gère est grande. Or il existe une communauté du logiciel libre qui grandit et gagne en prospérité. Cette communauté distribue des produits de très bonne qualité dont le code source est accessible et qui sont couverts par des licences publiques qui garantissent les droits suivants à leurs utilisateurs : exécuter, modifier et redistribuer le logiciel ainsi que les modifications. À partir d'une revue bibliographique sur les thèmes de la gestion de projet et des logiciels libres, il a été développé d'une part, un modèle représentant les besoins du gestionnaire de projet en termes de fonctionnalités informatiques, et d'autre part, les enjeux liés au logiciel libre ont été identifiés. Ce modèle appelé "pyramide des besoins du gestionnaire de projet" ainsi que les enjeux identifiés ont permis par la suite d'évaluer l'offre des logiciels libres en gestion de projet.

La pyramide montre ainsi que le gestionnaire a besoin tout d'abord de définir son projet à l'aide d'un traitement texte et d'un tableur ; puis il effectue grâce à un planificateur, des activités de planification, suivi et contrôle en termes de délais, coûts, performances et ressources humaines et matérielles. Une fois ces premières activités maîtrisées, il pourra dans un deuxième temps, faire de la gestion de projet intégrée et quantitative, tout en tenant compte de l'incertitude. En d'autres termes, il sera capable de mettre en place des processus standardisés, de mesurer la performance et la qualité du projet, ainsi que de gérer les risques et donc, de mettre en place ce que l'on appelle, l'amélioration continue. Pour cela, il aura besoin d'une base de données partagées, de fonctionnalités avancées de communication comme le courrier électronique, les forums et de fonctionnalités permettant la gestion

des risques.

Suite à l'évaluation des produits à code ouvert, un projet international de redéveloppement d'un module pour un logiciel libre a été mis en place. La gestion de ce projet, nécessairement distribuée, a été réalisée à l'aide d'outils libres. Il a été montré que l'offre des logiciels libres était mature et qu'elle répondait positivement à la plupart des besoins du gestionnaire de projet. Ces produits *open source* sont même dans certains cas plus puissants en termes de fonctionnalités, de sécurité et de fiabilité, que leurs rivaux à code fermé. De plus, ils sont très flexibles puisque le processus de développement de tels produits nécessite qu'ils soient conçus de manière modulaire. Dès lors, si l'on se place au niveau des ERP *Enterprise Resource Planning* libres, on montre que tous les besoins actuels du gestionnaire de projet sont couverts de manière adéquate et ce, pour un coût d'acquisition nettement inférieur à celui d'une solution privée, tout en ayant une réelle flexibilité et un contrôle sur le devenir du produit. Ainsi, l'amélioration continue des outils de gestion de projet serait grandement facilitée par un choix "libre".

Abstract

Modern project management uses computer software to help managing the complexity of actual projects. It has to be noticed that a community of free and open source software (FOSS) exists and grows rapidly gaining in prosperity. This community offers a lot of very good products whose source code is available and protected by public licenses that guarantee the following rights to their users : execute, modify and redistribute the software and changes. A literature review on project management and FOSS was performed. On one hand, a model representing the the project manager's needs in computer functionalities was developped, and on the other hand, issues related to Open Source Software have been identified. The model, represented as a pyramid, shows a hierarchy of the project manager's needs. The identified issues and this model allowed thereafter to assess the offer of FOSS in the project management domain.

The manager's needs are : first, to define the project with a text processing software and a spreadsheet. Secondly, thanks to a planner, to perform some planning, monitoring and control tasks in terms of time, cost, performance and human and material resources. Once these early tasks controlled, he can thirdly perform integrated and quantitative project management. Finally he can take into account the uncertainty. In other words, he will be able to standardize processes, measure the performance and quality of the project, as well as manage risks and therefore, to establish what is called, continuous improvement. To perform this, a shared database, some advanced communication features (e-mail, Forums) as well as risk management features will be required.

Following the evaluation of open source software in project management,

an international redevelopment project of a module for a free software has been established. The distributed management of this project was performed using FOSS. It was shown that FOSS were mature and responded positively to most of the project manager's needs. Those open source products are in some cases even more powerful in functionalities, security and reliability than their private rivals. Moreover, they are very flexible because the developing process, that requires products to be designed in a modular fashion. Therefore, free and opens source ERP Software (Enterprise Resource Planning) cover all the current project manager's needs. Their acquisition cost is significantly lower than a private solution. Finally, there is a real gain in flexibility and control over the fate of the product. Thus, the continuous improvement of project management tools would be greatly facilitated by a "free" choice.

Table des matières

DÉDICACE	iv
REMERCIEMENTS	v
RÉSUMÉ	vi
ABSTRACT	viii
TABLE DES MATIÈRES	ix
LISTE DES TABLEAUX	xvi
LISTE DES FIGURES	xviii
LISTE DES SIGLES ET ABRÉVIATIONS	xix
INTRODUCTION	1
Partie I La gestion de projet et les logiciels libres	5
CHAPITRE 1 LA GESTION DE PROJET	7
1.1 Cadre de gestion du <i>Guide PMBOK®</i>	7
1.1.1 Management de l'intégration	8

1.1.2	Management du contenu	10
1.1.3	Management des délais	10
1.1.4	Management des coûts	11
1.1.5	Management de la qualité	12
1.1.6	Management des ressources humaines	14
1.1.7	Management des communications	16
1.1.8	Management des risques	17
1.1.9	Management des approvisionnements	19
1.1.10	Collectes de données et méthodes d'estimations	19
1.2	Modèle de développement du CMMI	21
1.2.1	Modèle continu	21
1.2.2	Modèle étagé	23
1.3	Besoins en logiciels	23
1.3.1	La pyramide des besoins	24
1.3.2	Logiciels et fonctionnalités principales	26
CHAPITRE 2	LES LOGICIELS LIBRES	28
2.1	Définitions	28
2.1.1	Code source	29
2.1.2	Deux philosophies : libre et <i>open source</i>	29

2.2	Licences publiques et gratuité	30
2.2.1	Licences publiques	31
2.2.2	Gratuité	32
2.3	Sécurité des logiciels libres	33
2.3.1	Définition complexe de la sécurité	34
2.3.2	La sécurité des logiciels libres	34
2.4	Nouveaux modèles d'affaires	35
2.4.1	Les distributeurs	36
2.4.2	Les éditeurs <i>open source</i>	37
2.4.3	Les fondations	37
2.4.4	Les firmes de conseil	38
2.4.5	Éditeurs privés offrant une version libre	38
Partie II Les logiciels libres en gestion de projet		39
CHAPITRE 3 RECHERCHE ET ÉVALUATION		41
3.1	Méthode de recherche	41
3.1.1	Répertoires de logiciels libres	42
3.1.2	Littérature	42
3.2	Méthode d'évaluation	42

CHAPITRE 4	ÉVALUATION DES OUTILS LIBRES . . .	47
4.1	Suite <i>Open Office</i>	47
4.2	Logiciels en gestion de projets	49
4.2.1	DotProject	49
4.2.2	Ganttproject	50
4.2.3	GNOME Planer	50
4.2.4	Open Workbench	51
4.2.5	PhpCollab	51
4.2.6	TaskJuggler	52
4.2.7	Résultats	52
4.3	Les ERP libres	56
4.3.1	Open ERP	56
4.3.2	Neogia	57
4.3.3	Project-Open	57
4.3.4	Résultats	58
CHAPITRE 5	CONFRONTATION.	60
5.1	MS Project 2007	60
5.2	Primavera 6	61
5.3	PS Next	61

5.4 Résultats	62
CHAPITRE 6 HYPOTHÈSES DE RECHERCHE	64
Partie III Test des hypothèses sur un cas réel	66
CHAPITRE 7 PROJET INTERNATIONAL	68
7.1 L'implantation de <i>Moodle</i> à Polytechnique crée un besoin . . .	68
7.1.1 <i>Moodle</i>	69
7.1.2 Module gestionnaire de fichiers	73
7.2 Objectifs du projet	75
7.2.1 Réactivation du partage de fichiers entre utilisateurs . .	75
7.2.2 Création d'espace de stockage par groupe	76
7.3 Équipe de projet	77
CHAPITRE 8 OBSERVATION DU PROJET	79
8.1 Méthode d'observation	79
8.2 La gestion du projet	80
8.2.1 <i>Moodle</i> plate-forme de gestion de projet distribuée . . .	80
8.2.2 Suite <i>Open Office</i> 2.3 pour la génération de documents	82
8.3 L'exécution du projet	82
8.3.1 <i>Bug tracking</i>	83
8.3.2 Environnement de développement Web	83

8.4	Impact du projet	84
8.4.1	Leçons apprises et perspectives	85
8.4.2	Statistiques d'utilisation du module à Polytechnique	86
8.4.3	Module utilisé dans le cadre du projet <i>Teach Advanced</i> <i>Online</i>	87
Partie IV Résultats, recommandations, et perspectives		90
CHAPITRE 9 RÉSULTATS		92
9.1	Les forces des logiciels libres en gestion de projets	92
9.1.1	Une offre mature et durable	92
9.1.2	Une réponse adéquate aux besoins en gestion de projet distribuée	93
9.1.3	Flexibilité, contrôle et évolution rapide	93
9.2	Les faiblesses des logiciels libres en gestion de projets	94
9.2.1	Trouver ce qui nous convient	94
9.2.2	Temps d'apprentissage et documentation	95
CHAPITRE 10 RECOMMANDATIONS.		96
10.1	Recommandations pour la plate-forme <i>Moodle</i>	96
10.2	Perspectives	97
CONCLUSION		99
BIBLIOGRAPHIE		99

Liste des tableaux

Tableau 1.1	Niveaux d'aptitudes du modèle étagé CMMI . .	24
Tableau 7.1	Équipe de projet	77
Tableau 8.1	Statistiques d'utilisation du module	86

Liste des figures

Figure 1.1	Modèle standard du <i>Guide PMBOK</i> ®	8
Figure 1.2	Cadre de gestion de projet CMMI	25
Figure 1.3	Pyramide des besoins du gestionnaire de projet	26
Figure 1.4	Pyramide des besoins en logiciels	27
Figure 3.1	Grille d'évaluation des besoins primaires	45
Figure 3.2	Grille d'évaluation des besoins avancés.	45
Figure 4.1	Évaluation logiciels libres (1)	54
Figure 4.2	Évaluation logiciels libres (2)	55
Figure 4.3	Évaluation ERP libres (1)	59
Figure 4.4	Évaluation ERP libres (2)	59
Figure 5.1	Évaluation logiciels propriétaires (1)	63

Figure 5.2	Évaluation logiciels propriétaires (2)	63
Figure 7.1	Structure de <i>Moodle</i>	72
Figure 7.2	Cas d'utilisation.	78
Figure 8.1	Site plate-forme (gestion du projet)	81
Figure 8.2	Capture d'écran du module <i>Technical project</i> .	82
Figure 8.3	Module traqueur de défauts	83

Liste des sigles et abréviations

BSD *Berkley Software Distribution*

CIA *Confidentiality Integrity Availability*

CLUF *Contrat de Licence Utilisateur Final*

CMMI *Capability Maturity Model[®] Integration*

CVS *Concurrent Versions System*

EISTI *École Internationale des Sciences du Traitement de l'Information*

ENT *Espaces Numériques de Travail*

ERP *Enterprise Resource Planning*

FOSS *Free and Open Source Software*

FSF *Free Software Foundation*

FTP *File Transfer Protocol*

GNOME *GNU Network Object Model Environment*

GNU *Acronyme récusé pour GNU's Not Unix*

GPL *General Public License*

HTML *Hypertext Markup Language*

IDE *Integrated Development Environment*

IP *Internet Protocol*

LGPL *Lesser General Public License*

MIT *Massachusetts Institute of Technology*

MPL *Mozilla Public License*

OSI *Open Source Initiative*

PDF *Portable Document Format*

PHP Acronyme récursif pour *Hypertext Preprocessor*

PMI *Project Management Institute*

SPD Structure de Découpage du Projet

SEI *Software Engineering Institute*

SQL *Structured Query Language*

TCP *Transmission Control Protocol*

NTIC Nouvelles Technologies de l'Information et de la Communication

TICE Technologies de l'Information et de la Communication pour l'Éducation

URL *Uniform Resource Locator*

ZIP *Zone Information Protocol*

Introduction

Dans un contexte extrêmement concurrentiel dû à la mondialisation, les entreprises ont été poussées à mettre en place ce que l'on appelle la gestion de projet. Un projet étant défini comme “une combinaison de ressources organisationnelles, agencées entre elles, pour créer quelque chose qui n’existait pas auparavant” [12]. La gestion de projet aide, quant à elle, à respecter “les exigences concurrentes de qualité, de contenu, de délai et de coût” [1] imposées par l’économie de marché globalisée. En d’autres termes, on cherche à répondre aux questions suivantes :

- combien est-ce que cela va coûter ?
- combien de temps faut-il pour réaliser le projet ?
- quelles sont les ressources à déployer pour réussir ?

La gestion de projet est désormais reconnue, depuis plusieurs années et dans tous les secteurs, comme un enjeu critique pour la croissance et la survie de toute entreprise. C’est la raison pour laquelle la communauté scientifique et professionnelle a cherché à codifier et organiser la gestion de projet autour de processus. Le *Project Management Institute* (PMI) a contribué de manière notoire à cette codification et le *Guide PMBOK®* regroupe les bonnes pratiques en gestion de projet. De plus, le département de la défense américaine a sollicité le *Software Engineering Institute* (SEI) et l’Université Carnegie Mellon afin de développer le modèle de maturité CMMI (*Capability Maturity Model® Integration*). Ce modèle est maintenant utilisé dans le monde entier et permet l’amélioration continue dans la maîtrise des exigences concurrentes dont il a été question plus avant.

Par ailleurs, il est indiscutable que les nouvelles technologies de l’information et de la communication (NTIC) ont fortement contribué à la mondia-

lisation. Il est également à noter que les projets sont devenus très complexes, tant dans leur contenu que dans leur gestion. Ainsi, les gestionnaires de projet ont eu besoin d'outils pour mettre en place les processus recommandés et ils ont fait appel à l'informatique. C'est alors qu'un grand marché est né et les éditeurs de logiciels l'ont bien compris. *Primavera Systems*, *Microsoft* et *Sciforma* figurent actuellement parmi les meilleurs sur ce marché, en témoignent par exemple les quelques 20 millions d'utilisateurs de *MS Project* en 2007. [28] Ces trois grandes compagnies proposent chacune leur solution propriétaire à la gestion de projet, c'est-à-dire qu'elle vendent à leurs utilisateurs le droit d'utiliser le logiciel, selon les termes d'un contrat de licence utilisateur final (CLUF).

Or, il existe une communauté du logiciel libre, grandissante et de plus en plus puissante. Tellement puissante que, selon l'auteur Gartner, il est impossible de l'éviter dorénavant. Il argumente en précisant que "les analystes prédisent qu'en 2011, au moins 80% des logiciels commerciaux contiendront du code ouvert". [10] De plus, Roberto Di Cosmo affirme que "sans logiciels libres, ni standards ouverts, il n'y aurait tout simplement plus d'Internet" [16]. Il illustre son propos avec quelques exemples et explique que le développement du réseau des réseaux repose précisément sur des protocoles ouverts comme TCP/IP pour les transmissions, sur des langages ouverts comme le HTML pour les sites Web et sur des logiciels libres comme le *Berkeley Internet Name Daemon* pour la résolution des noms de domaines. Enfin, toute la littérature indique que le mouvement du libre est né avec l'informatique, même s'il n'a pas été reconnu en tant que tel, car il était normal à l'époque de partager ses logiciels. Ainsi, on peut penser que "la montée en puissance de l'écosystème du logiciel libre va avoir une incidence sur le poste de travail" [39]. Dès lors, il semble tout à fait pertinent de s'intéresser aux outils libres dans le domaine de la gestion de projet distribuée. Est-ce

qu'il en existe ? Si oui, quels sont ceux qui pourraient répondre aux besoins du gestionnaire de projet ? Coûtent-ils vraiment moins cher ? Peut-on avoir confiance en ces logiciels ? Enfin, quel avenir leur est réservé ?

Afin de répondre à ces interrogations, il s'agit de se doter d'une méthode de recherche.

Dans un premier temps, il faut effectuer un travail de définition. En effet, qu'est-ce qu'exactly la gestion de projet et que sont les besoins en logiciels ? Puis il faut comprendre ce que sont les logiciels libres ou *open source*. La littérature est riche en ce qui concerne ces deux thèmes pris séparément. Elle peut donc aider à comprendre d'une part, les exigences d'un gestionnaire de projet et d'autre part, les valeurs du mouvement libre, sa raison d'être et son impact sur le monde. Une revue bibliographique s'impose donc d'elle-même dans la partie I.

En revanche, les publications se font plus rares en ce qui concerne les logiciels libres en gestion de projet distribuée. La plupart, comme celles du Journal du Net, se contentent de proposer une liste des offres arrivées à maturité, en indiquant les fonctionnalités principales disponibles. Malgré tout, il est intéressant, dans une partie II, de faire un tour d'horizon de cette offre et de la comparer avec les logiciels les plus connus qui ont été mentionnés en début de cette introduction.

Mais il doit être envisagé de dépasser cette simple évaluation. En effet, l'on se doit de tester les hypothèses qui auront été formulées dans le chapitre 6, suite à ce premier travail bibliographique. Il est donc approprié d'appliquer la gestion de projet distribuée en situation réelle. Pour cela, un projet international, totalement géré grâce à des logiciels libres, a été mis sur pieds. Ce projet, né d'un réel besoin commun à l'École Polytechnique de Montréal et à l'Université de Franche-Comté, ainsi que son observation, font l'objet de la

partie III.

Dans la dernière partie de cette recherche, il sera tout d'abord possible de tirer des conclusions sur les forces et faiblesses des logiciels libres en gestion de projet distribuée. Il s'agira donc de répondre aux hypothèses de recherche formulées. Puis, il sera utile d'émettre des recommandations quant à l'utilisation des logiciels libres, tout particulièrement en ce qui concerne la nouvelle plate-forme d'apprentissage en ligne dont se sont équipées l'École Polytechnique de Montréal et l'Université de Franche-Comté. Enfin, fort de toutes les informations accumulées au cours de ce cheminement, quelques perspectives pourront être abordées et ouvrir la voie à des recherches ultérieures.

Première partie

La gestion de projet et les logiciels libres

En se basant sur la littérature, cette première partie est destinée à définir la gestion de projet et les logiciels libres. Les deux objectifs visés sont : déterminer les besoins en logiciels des gestionnaires de projet et comprendre en quoi les logiciels libres ou *open source* pourraient être une solution valide en réponse aux besoins identifiés. Pour se faire, le *Guide PMBOK®* et le modèle de maturité CMMI permettront dans un premier chapitre d'identifier les processus clés de la gestion de projet et d'en apprécier les différentes composantes.

Le deuxième chapitre développera, à travers plusieurs questions fondamentales, les philosophies qui sont à l'origine de l'*open source* ou du mouvement du libre et en explicitera leurs nuances et leurs valeurs. Il expliquera également la nécessaire protection des droits d'auteurs de cette communauté. Enfin, la sécurité des logiciels libres sera vérifiée pour déboucher sur les nouveaux modèles d'affaires qui se sont créés grâce à "la montée en puissance de l'écosystème du logiciel libre" [39].

Chapitre 1

La gestion de projet

Dans ce chapitre, il s'agit de définir la gestion de projet afin d'en dégager les besoins en logiciels pour le gestionnaire de projet. Les deux modèles retenus sont ceux du *Project Management Institute* (PMI) d'une part, et du *Software Engineering Institute* (SEI) et de l'Université Carnegie Mellon d'autre part. Ils sont sans conteste les plus reconnus dans le monde professionnel. Leurs approches sont complémentaires en ce sens que le *Guide PMBOK®* regroupe les domaines de connaissance et les processus en management de projet – de l'initialisation à la clôture du projet – ; tandis que le modèle de maturité CMMI (*Capability Maturity Model® Integration*) présente une vue selon le degré de maîtrise des processus par l'entreprise (d'un niveau basique à un niveau avancé).

1.1 Cadre de gestion du *Guide PMBOK®*

La figure 1.1 présente le cadre de gestion de projet défini par le *Guide PMBOK®*. Il s'organise autour de neuf groupes de processus (ou domaines de connaissance) qu'il faut mettre en place pour gérer adéquatement tout projet. Chaque groupe n'a pas le même degré d'importance en fonction du domaine dans lequel évolue l'entreprise et du type de projet. Il appartient donc au gestionnaire de projet d'évaluer et de choisir les processus qu'il sou-

haite mettre en place ou privilégier. Pour des raisons évidentes d'analyse, il est proposé de passer au travers des neuf domaines de connaissances en gardant l'objectif en vue : dégager les besoins en logiciels pour le gestionnaire de projet.

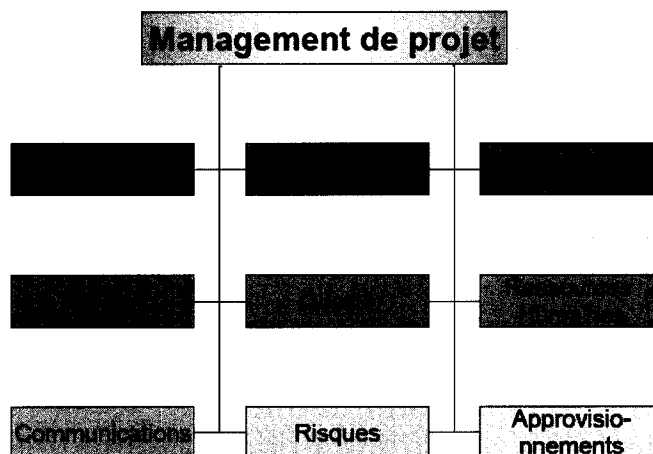


Figure 1.1 – Modèle standard du *Guide PMBOK®*

1.1.1 Management de l'intégration

Le guide mentionne dès le troisième chapitre les documents principaux du projet, à savoir :

- la charte de projet, qui autorise formellement le projet ;
- l'énoncé du contenu du projet qui décrit le travail à accomplir et les livrables à produire ;
- et le plan de management du projet qui définit comment le travail sera réalisé.

Ces documents regroupent toute l'information capitale à la réalisation du projet et leur origine se trouve dans le domaine de l'intégration du projet. Il

est en effet stipulé qu'au moins une version préliminaire de ces documents doit y être élaborée. Cependant, les autres groupes de processus apportent leur contribution. Le plan de management du projet comporte par exemple plusieurs plans subsidiaires, chacun provenant d'un domaine de connaissance différent. Ainsi, il peut en posséder huit :

- le plan de management du contenu ;
- le plan de management de l'échéancier ;
- le plan de management des coûts ;
- le plan de management de la qualité ;
- le plan de management des ressources humaines ;
- le plan de management des communications ;
- le plan de management des risques ;
- et le plan de management des approvisionnements ;

La charte de projet et l'énoncé préliminaire du contenu contiennent respectivement l'énoncé des travaux et les exigences fondamentales du projet. La définition de l'énoncé des travaux est sans équivoque : “une description narrative des produits et services que le projet doit fournir” [1]. Ces deux premiers documents sont donc des textes. Le premier besoin d'un gestionnaire de projet se trouve ainsi être un logiciel de traitement de texte. Le plan de management du projet étant un document complexe, il est nécessaire d'avancer vers les autres domaines de connaissances pour comprendre ce qu'il contient précisément. De plus, suite à ce parcours, il sera aisé de comprendre les autres processus de l'intégration du projet qui sont : diriger et piloter l'exécution du projet, surveiller et maîtriser le travail, la maîtrise intégrée des modifications et la clôture du projet. Ces quatre derniers processus sont soutenus par ceux des autres domaines de connaissances. Les besoins en logiciels seront donc identifiés au travers des prochaines sections.

1.1.2 Management du contenu

En ce qui concerne le management du contenu, il y a tout d'abord un plan à définir qui "documente la façon dont le contenu du projet sera défini, vérifié et maîtrisé" [1] ainsi que le contenu du projet lui-même (objectifs, exigences, limites, livrables). Il s'agit là encore de documents texte. Puis vient la structure de découpage du projet (SDP). On souhaite grâce à ce processus subdiviser le projet en composants plus simples et plus faciles à maîtriser et il s'agira par la suite d'en maîtriser les mises à jour et leurs impacts sur les autres composants du projet. Ici, on peut penser qu'un simple schéma ou tableau pourrait suffire dans la plupart des cas, mais lorsque le projet est très complexe, cela peut devenir très fastidieux. Ainsi, un tableur, dans lequel il est possible d'établir des liens entre pages ou entre différents documents, trouve ses limites. Il faut donc envisager des outils de gestion des exigences.

1.1.3 Management des délais

Lorsqu'on parle de management des délais, on pense immédiatement à l'échéancier. Pour établir un échéancier, il faut dans un premier temps identifier les activités, trouver leur séquence logique, c'est-à-dire leur ordre et leurs dépendances ; puis dans un deuxième temps, estimer les ressources nécessaires aux activités ainsi que la durée de ces activités. On peut utiliser un tableur lorsque le nombre d'activités et de ressources est restreint, mais le gestionnaire a besoin d'un outil spécialisé afin de générer des chartes de Gantt ou obtenir quasi-instantanément une optimisation du chemin critique. Il peut également vouloir appliquer certaines techniques comme l'analyse d'éventualités ou le nivellement des ressources. Avec un tel logiciel de gestion de projet, il lui est possible de "passer rapidement en revue de nombreuses

options de l'échéancier" [1].

1.1.4 Management des coûts

Le management des coûts consiste en l'estimation des coûts, l'établissement du budget et en la maîtrise des coûts.

Le gestionnaire estime ou fait estimer tout d'abord les coûts pour chaque activité ou lot de travail. Les estimations peuvent être réalisées par différentes méthodes dont quelques unes seront exposées dans la sous-section 1.1.10.

On obtient ensuite le budget en agrégeant les coûts des activités individuelles ou des lots de travail. En général, le gestionnaire utilise un tableur ou bien un outil de gestion de projet qui lui permet d'attribuer des coûts à chaque activité et établit le budget automatiquement.

Le dernier processus est la maîtrise des coûts qui se déroule en quatre temps. Il s'agit tout d'abord de documenter, dans le plan de management des coûts, les procédures à suivre pour modifier la base des coûts ainsi que les mécanismes et les niveaux d'approbation nécessaires à l'autorisation des modifications. Ensuite, il faut analyser la mesure de la performance. Pour cela, il existe plusieurs techniques qui aident à estimer l'ampleur des écarts qui surviendront inévitablement (valeur planifiée, valeur acquise, coût réel, etc.) Ces techniques nécessitent la collecte de données puis quelques calculs mathématiques. Il apparaît immédiatement qu'un tableur semble répondre à ce besoin. Le troisième temps de la maîtrise des coûts est la prévision. Elle repose sur des prédictions à partir d'informations et de connaissances disponibles à un moment donné. Le gestionnaire peut faire encore une fois appel à des techniques mathématiques pour évaluer le coût ou la quantité de travail restants. Dans la dernière étape, on effectue des revues de performance

du projet, c'est-à-dire que l'on compare ce qui était prévu (planifié, budgétisé) avec ce qui a été effectivement réalisé. Il s'agit donc d'analyser les écarts et d'examiner si la performance du projet s'améliore ou se dégrade à l'aide par exemple de la technique de la valeur acquise.

1.1.5 Management de la qualité

Le projet doit répondre à certaines normes de qualité, soit à cause de la réglementation, ou soit à cause des exigences des parties prenantes. Ainsi, le management de la qualité se déroule en trois points : la planification de la qualité, la mise en oeuvre de l'assurance qualité et la mise en oeuvre du contrôle qualité.

On identifie donc tout d'abord les normes de qualité pertinentes pour le projet et on détermine les moyens de les respecter. Le *Guide PMBOK®* indique qu'il existe deux sources d'informations qui permettent de trouver ces normes : les facteurs environnementaux de l'entreprise (réglementations, normes ou directives spécifiques au champ d'application), et l'actif organisationnel (politique interne, procédures, données historiques et leçons apprises des projets précédents). Ceci suppose que le gestionnaire de projet ait un accès à une base de données partagée contenant au moins les informations concernant l'actif organisationnel. Ensuite, on planifie la qualité en faisant appel à des métriques, ou à des listes de contrôle qualité qui sont souvent modélisées sous forme de diagrammes de flux.

Le deuxième point concerne la mise en oeuvre de l'assurance qualité. Elle est "l'application d'activités planifiées et systématiques, afin d'assurer que le projet utilise tous les processus nécessaires pour répondre aux exigences" [1]. Pour cela, on peut utiliser les techniques vues précédemment, les audits qua-

lité, l'analyse des processus ou encore les outils et techniques du contrôle qualité dont il est question ci-après.

Le dernier point du management de la qualité implique la surveillance de résultats spécifiques du projet, pour déterminer s'ils sont conformes aux normes de qualité correspondantes, et trouver des moyens d'éliminer les causes de résultats insatisfaisants. Dans ce dernier volet du management de la qualité, le gestionnaire de projet peut s'aider des "sept outils de base de la qualité".

- le diagramme cause-effet, également appelé d'Ishikawa (ou en arête de poisson) ;
- le diagramme de contrôle qui permet de déterminer si un processus est stable ou non ;
- la représentation en diagramme de flux pour analyser comment les problèmes se produisent, en montrant les activités, les points de décisions et l'ordre de traitement ;
- l'histogramme qui est un diagramme à barres montrant une distribution de variables ;
- le diagramme de Pareto indiquant le nombre de défauts générés par type ou par catégorie de cause identifiée ;
- le relevé d'observations qui montre l'historique et le comportement de l'écart ;
- le diagramme de corrélation (ou en nuage de points) qui identifie le rapport possible entre des variations observées de deux variables.

Un tableur ou un logiciel de traitement de texte peut répondre au besoin de modélisation pour la plupart des diagrammes. On pourrait également envisager un logiciel de modélisation de diagrammes de flux à cause de leur complexité.

1.1.6 Management des ressources humaines

Le management des ressources humaines se décompose en quatre étapes selon le *Guide PMBOK®*.

Pour commencer, la planification des ressources humaines représente “l’identification et la documentation des rôles, responsabilités et relations d’autorité dans le cadre du projet” [1]. Celle-ci tient compte des facteurs environnementaux de l’entreprise qui peuvent être d’ordre organisationnels (organisations ou services impliqués), techniques (disciplines ou spécialités requises), interpersonnels (relations d’autorité, différences culturelles), logistiques (distance, fuseaux horaires) ou encore politiques (objectifs des parties prenantes). Dès lors, on a besoin d’outils pour modéliser cette planification. Les types d’outils proposés sont au nombre de trois : les diagrammes hiérarchiques, qui sont une représentation de la structure traditionnelle de l’organigramme ; les diagrammes ou grilles matricielles, pour illustrer les connexions entre travail à exécuter et les membres de l’équipe de projet ; et les formats type texte, qui sont généralement utilisés pour certains rôles qui nécessitent une description plus ou moins détaillée. A partir des informations citées ci-dessus, il est donc possible de construire le plan de management des ressources humaines qui explique tout d’abord d’où proviennent les ressources (de l’entreprise elle-même ou de l’extérieur via contrats). Ce plan contient également le calendrier des ressources qui décrit les périodes nécessaires pour les membres de l’équipe du projet, tant individuellement que collectivement. Le gestionnaire de projet utilise très souvent l’histogramme, un diagramme en barre qui permet de visualiser facilement les besoins de nivellement des ressources, c’est-à-dire lorsque le nombre maximum d’heures disponibles est dépassé. D’autres informations peuvent apparaître dans ce plan, comme les critères de désengagement, les besoins de formation, les critères de récompenses et

de sécurité. Il est donc à noter que pour ce plan, un traitement texte et un tableur sont requis. De plus, il serait intéressant qu'un logiciel de gestion de projet possède des fonctionnalités de planification et de nivellement des ressources.

Dans la deuxième étape, il s'agit de former l'équipe de projet, dans le sens où il faut affecter les personnes à cette équipe de projet, souvent à la suite de négociations ou d'acquisitions. Le *Guide PMBOK®* traite également des équipes virtuelles qu'il définit comme des groupes de personnes partageant un même objectif, mais qui ne se rencontrent physiquement que très rarement. Avec ce type d'équipe, les outils de communication modernes deviennent indispensables (courriel, visioconférence, base de données partagée, etc.).

Le processus intitulé "Développer l'équipe" constitue l'avant dernière étape de ce domaine de connaissance. Il est question de techniques managériales pour lesquelles un traitement de texte peut s'avérer utile. En effet, il est fait mention d'établir les règles de base clairement. Ainsi, le gestionnaire de projet peut proposer une version écrite de des règles à respecter en tout temps ou des comportements attendus.

Pour finir, diriger l'équipe consiste à suivre les performances des membres de l'équipe, leur fournir des informations en retour, résoudre les problèmes et coordonner les modifications en vue d'améliorer la performance du projet. Il s'agit là encore de techniques managériales, néanmoins, les évaluations peuvent être réalisées à l'aide des outils informatiques précédemment vus (traitement texte, tableur et logiciel de gestion de projet); et les résultats peuvent être communiqués au cours d'un rapport d'avancement dont il sera question dans la section suivante, par exemple avec un logiciel permettant de préparer une présentation. Enfin, toute connaissance apprise au cours du projet devrait être documentée et inscrite dans la base de données des leçons

appries.

1.1.7 Management des communications

Les communications sont d'une importance capitale dans tout projet, car elles permettent de mettre l'information nécessaire à la disposition des acteurs du projet en temps voulu. Ce domaine de connaissance contient quatre processus.

Il faut premièrement établir le plan des communications. Celui-ci détermine les besoins d'information et de communication des parties prenantes, grâce aux organigrammes réalisés précédemment ; ainsi que la technologie de communication à employer en fonction de l'urgence ou du nombre de personnes devant communiquer par exemple. Ce plan est donc constitué d'informations textuelles et d'informations de planification. Un traitement texte et un tableur ou un logiciel de planning devraient pouvoir permettre de le bâtir.

Deuxièmement, il faut organiser la diffusion de l'information. Cela signifie qu'il faut faire en sorte que les bonnes personnes obtiennent l'information appropriée au bon moment. Ici, on ne considère que les communications qui font appel à l'outil informatique. Le *Guide PMBOK®* signale qu'un ou plusieurs systèmes de collecte et de récupération des informations doivent être implantés. Ainsi, on peut y voir les bases de données partagées ou les logiciels de gestion de projet. Les méthodes de diffusion de l'informations peuvent être "les outils électroniques de communication et de conférence, tels que le courrier électronique ou la publication par Internet." [1]

Le troisième processus est celui de l'établissement du rapport d'avancement à travers lequel on collecte et on diffuse les informations sur la per-

formance. Cela englobe les rapports d'état, la mesure de l'avancement et les prévisions. Ce rapport fait appel aux fonctionnalités d'un traitement de textes, d'un tableur, d'un logiciel de planification et à celles d'une base de données partagée pour la collecte des informations.

Enfin, le management des communications ne saurait être complet sans la gestion des parties prenantes. En effet, satisfaire leurs exigences d'une part, et résoudre les problèmes majeurs avec elles d'autre part, sont deux activités qui évitent au projet de dévier de son objectif. Ainsi, les réunions en face à face sont sans aucun doute des plus efficaces, mais il ne faut pas oublier que lorsqu'on se trouve dans un contexte international, il n'est pas forcément aisé de réunir tous les acteurs dans une même pièce. Tous les outils cités précédemment peuvent donc jouer un rôle primordial.

1.1.8 Management des risques

Quand bien même tout semble organisé, planifié et maîtrisé, des risques inhérents à chaque projet peuvent apparaître. Un risque se définit comme un événement hors de contrôle de l'entreprise, mais qui lorsqu'ils se manifestent, peut avoir un impact négatif, voire destructeur sur le projet. Généralement, un risque influence négativement au moins l'une des trois exigences concurrentes de délais, de coûts ou de qualité. Les risques doivent donc être pris en considération. Et le guide propose une méthode en cinq étapes.

La première est celle de l'identification des risques. On détermine quels risques pourraient avoir un impact sur le projet et on documente leurs caractéristiques.

Ensuite, on effectue une analyse qualitative de ces risques en les hiérarchisant. Cela permet à l'équipe de projet, en combinant la probabilité d'oc-

currence et l'impact des risques, de définir les priorités pour la planification des réponses à ceux-ci et prépare l'étape suivante. Une matrice de probabilité et d'impact est généralement créée à l'aide d'un tableur.

La troisième étape consiste en une analyse quantitative. On affecte les effets des risques identifiés sur l'ensemble des objectifs du projet en termes de coûts, de délais, et de ressources. On en déduit plusieurs scénarios. A cette étape-ci, on voit l'intérêt d'une fonctionnalité avancée dans un logiciel de gestion de projet qui permettrait au gestionnaire de projet de simuler plusieurs situations. Il peut également faire usage d'outils de modélisation et de simulation qui implémenteraient la technique Monte-Carlo.

Il s'agit par la suite, de planifier des réponses aux risques. On cherche par là, à "élaborer des options et des actions pour améliorer les opportunités favorables aux objectifs du projet et réduire les menaces à leur rencontre" [1]. L'équipe décide lorsque cela lui est possible, d'éviter le risque en modifiant le plan de management, afin d'éliminer toute menace ; de transférer le risque vers un tiers ; ou d'atténuer son impact et sa probabilité d'occurrence. Si aucune de ces trois stratégies n'est valable, alors il faut accepter le risque et prévoir un plan de mitigation, communément appelé "plan B".

"Suivre les risques identifiés, surveiller les risques résiduels, identifier les risques nouveaux, exécuter les plans de réponse aux risques et évaluer leur efficacité au long du cycle de vie du projet" [1] sont les activités de la dernière étape. Dès lors, on comprend qu'il faut mettre en place des systèmes de mesures, collecter de l'information, et tenir à jour le plan de management des risques. Tous les outils informatiques identifiés depuis le début de ce chapitre seront encore une fois sollicités.

1.1.9 Management des approvisionnements

Le dernier domaine de connaissance concerne la gestion des approvisionnements. Il contient six processus : planifier les approvisionnements, planifier les contrats, solliciter des offres ou des propositions des fournisseurs, choisir les fournisseurs, administrer les contrats, et clôturer les contrats.

On peut immédiatement remarquer, sans entrer dans les détails de chacun des processus, qu'il s'agit encore une fois d'activités de planification et de sélection. Le gestionnaire de projet fera donc appel à des outils similaires que ceux identifiés jusqu'à présent. On apprend également, qu'à la suite de la sélection des fournisseurs, il faut négocier un contrat écrit avec chacun d'entre eux. Un logiciel de traitement de texte est tout à fait adapter à la rédaction de ces contrats. En ce qui concerne l'administration des contrats, il s'agit "de passer en revue et documenter les performances passées ou présentes d'un fournisseur". Il faut donc collecter des informations et entrer ces mesures dans un système afin de les exploiter. Une nouvelle fois, une base de données partagée semble être indispensable. Enfin, le dernier processus de clôture des contrats soutient le processus de clôture du projet dont il a été question dans le premier domaine de connaissance.

1.1.10 Collectes de données et méthodes d'estimations

Pour finir, il convient de se demander comment le gestionnaire de projet procède pour collecter les données dont il a été question plus avant et établir des estimations (budgétaires, de temps ou autres).

La collecte de données peut être envisagée de deux manières : le gestionnaire effectue la collecte lui-même ou bien, chaque acteur transmet ses

propres informations via un système plus ou moins intégré avec les outils de gestion de projet. On peut imaginer un système de communication texte comme le courriel ou un système plus strict et certainement plus efficace comme la feuille de temps électronique.

Concernant les estimations, il peut faire appel aux méthodes proposées par le *Guide PMBOK®* : le jugement d'expert, l'estimation par analogie, paramétrique, l'analyse de réserve ou encore l'estimation à trois points. Les deux premières méthodes font appel à un historique, c'est-à-dire qu'elles supposent un accès aux données des projets antérieurs, certainement classés par type ou par taille. Ainsi, un tableur peut permettre au gestionnaire de tenir des statistiques à jour sur la durée et/ou le coût de tel type d'activité par exemple. Les deux méthodes suivantes sont simples à mettre en oeuvre avec les outils déjà évoqués, puisque l'estimation paramétrique représente une simple multiplication entre la quantité de travail à exécuter et le taux de productivité ; et l'analyse de réserve est l'ajout d'une "provision pour aléas". L'estimation à trois points requiert quant à elle trois estimées : la plus probable, la plus optimiste et la plus pessimiste. Un tableur peut faire l'affaire pour ce genre d'applications en complexifiant légèrement le modèle de départ, mais si on pense en termes d'échéances, on pourrait avoir besoin d'une fonctionnalité supplémentaire dans l'outil de planification. Ainsi, le gestionnaire n'entre pas seulement une donnée pour chaque activité, mais trois, et le logiciel devrait être capable de calculer les trois scénarios différents. Comme il a été vu précédemment, cela peut être une fonctionnalité très utile pour envisager la gestion des risques.

1.2 Modèle de développement du CMMI

Le deuxième modèle retenu, pour comprendre ce qu'est la gestion de projet, est appelé le modèle de maturité CMMI (*Capability Maturity Model® Integration*). Il a été conçu pour devenir "la référence pour les activités de développement et de maintenance et couvre entre autres, la gestion de projet, le management de processus, l'ingénierie des systèmes, et le développement logiciel" [2]. De plus, il présente une vision à la fois légèrement différente et complémentaire à celle du *Guide PMBOK®*. Il est en effet possible de l'aborder sous deux angles : celui de la représentation continue d'une part, et celle du modèle étagé d'autre part.

La première permet d'identifier les processus préconisés en gestion de projet par le modèle, tandis que la deuxième invite à donner un ordre de priorité à ces processus. C'est exactement la démarche proposée dans cette section. Ainsi, il sera possible d'identifier d'éventuels nouveaux besoins en logiciels et surtout de les classer en leur attribuant une priorité.

1.2.1 Modèle continu

Le modèle continu offre un maximum de flexibilité selon la documentation du CMMI, car l'entreprise peut choisir un seul processus ou type de processus directement reliés aux objectifs d'amélioration de celle-ci. Il regroupe les processus en quatre catégories ou domaines (*process areas* :

- processus organisationnels,
- processus de gestion de projet,
- processus d'ingénierie,
- et processus de support.

En outre, seuls les six processus de gestion de projet sont à retenir pour le but poursuivi :

- planification de projet,
- suivi et contrôle de projet,
- gestion des ententes avec les fournisseurs,
- gestion de projet intégrée,
- management des risques,
- gestion quantitative de projet.

Il est indiqué qu'il faut dans un premier temps, développer le plan directeur, (processus de planification) puis favoriser l'implication appropriée des parties prenantes en obtenant leurs engagements par rapport à ce plan (processus de gestion des ententes avec les fournisseurs). Durant le projet, il s'agit dans un deuxième temps, de maintenir le plan directeur à jour. En d'autres termes, il faut suivre et contrôler l'avancement de ce qui a été réalisé (processus de suivi et de contrôle de projet). En parallèle de tout cela, la gestion de projet intégrée représente un processus défini et dirigé selon un ensemble de processus standards de l'entreprise. Il établit entre autres les conditions de travail standards du projet à partir de celles de la firme. Il facilite donc la mise en oeuvre d'équipes dites intégrées, qui établissent et font évoluer une vision partagée du projet. Le processus de management des risques permet quant à lui, d'identifier les risques, de les suivre et de développer des plans de mitigation. Enfin, lorsque des techniques quantitatives et statistiques sont mises en place tout au long du projet, pour mesurer sa performance et sa qualité, le CMMI appelle cela la gestion quantitative de projet.

Ainsi, la gestion de projet proposée par ce modèle ressemble fort à celle du *Guide PMBOK®*. Toutefois, le modèle étagé donne une vision complémentaire dans la section suivante.

1.2.2 Modèle étagé

Le modèle étagé présente une manière systématique et structurée pour l'amélioration des processus, et ce, une étape à la fois. Il reprend les mêmes catégories de processus que le modèle continu, mais les expose selon différents niveaux d'aptitude atteint par l'entreprise. La documentation indique qu'il existe six niveaux de maturité.

Le tableau 1.1 regroupe les niveaux maturité et une description du type d'aptitude à acquérir quant à la maîtrise du processus.

Cette représentation du modèle montre qu'en ce qui concerne la gestion de projet, trois processus appartiennent au niveau 2 et que les trois autres appartiennent à des niveaux supérieurs (3 et 4). Ainsi, on peut appeler les premiers, les processus basiques ; les autres, les processus avancés. La figure 1.2 illustre ces deux catégories de processus. On peut dès lors en déduire, dans la section suivante, les besoins primaires et avancés du gestionnaire de projet, en termes de logiciels ou fonctionnalités.

1.3 Besoins en logiciels

Il a été vu que la gestion de projet concerne plusieurs domaines de connaissances qui sont à peu de choses près identiques selon les différentes littératures. Ces groupes de processus peuvent être classés en deux catégories : les processus de base et les processus avancés. L'objectif est ici de construire un modèle qui sera utilisé lors de la phase d'évaluation des logiciels libres en gestion de projet. On peut donc imaginer, par analogie à la théorie d'Abraham Maslow, "la pyramide des besoins" d'un gestionnaire de projet. À partir

Tableau 1.1 – Niveaux d’aptitudes du modèle étagé CMMI

Niveaux	Le processus
0 - Incomplet	n’est pas exécuté ou que partiellement
1 - Exécuté	est exécuté et atteint ses objectifs spécifiques
2 - Géré	est exécuté et atteint ses objectifs spécifiques ainsi que les objectifs génériques de niveau 2
3 - Personnalisé	est personnalisé à partir des processus normalisés de l’organisation selon des critères et atteint les objectifs génériques du niveau 3
4 - Quantitatif	est contrôlé selon des techniques statistiques ou d’autres techniques quantitatives et atteint les objectifs génériques du niveau 4
5 - En optimisation	est optimisés en réduisant les variations des causes communes inhérentes au processus et atteint les objectifs génériques du niveau 5

de cette pyramide des besoins, il est possible de classer les différents types d’outils informatiques et les fonctionnalités par ordre croissant de nécessité.

1.3.1 La pyramide des besoins

En croisant les différentes informations recueillies lors de l’étude bibliographique sur la gestion de projet, l’élaboration du plan de management de

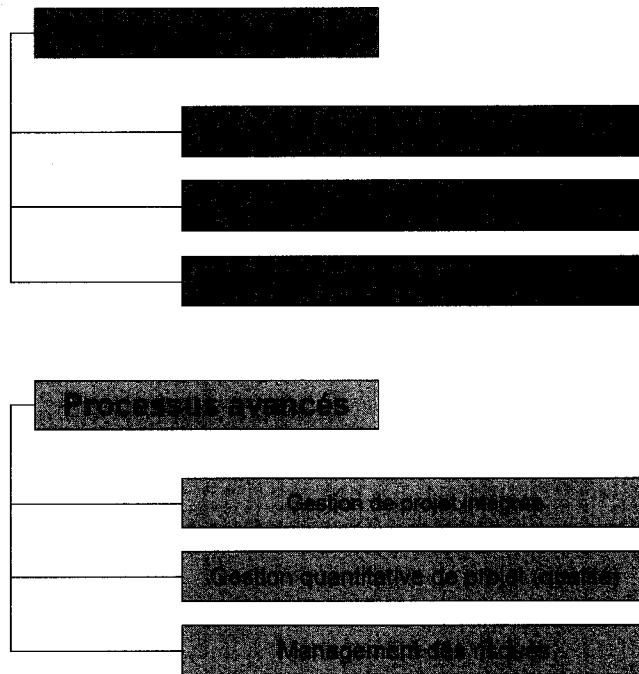


Figure 1.2 – Processus de gestion de projet (adapté du modèle CMMI)

projet, aussi appelé plan directeur de projet, ainsi que le suivi et le contrôle du projet apparaissent comme des besoins primaires. Ainsi, le gestionnaire a besoin tout d'abord de définir son projet (charte de projet, énoncé préliminaire du contenu), puis d'effectuer des activités de planification, suivi et contrôle en termes de délais, coûts, performances et ressources humaines et matérielles (plan directeur regroupant les plans de management des délais, des coûts, de qualité, de ressources humaines et d'approvisionnements). Une fois ces premières activités maîtrisées, il pourra dans un deuxième temps, faire de la gestion de projet intégrée et quantitative, tout en tenant compte de l'incertitude. En d'autres termes, il sera capable de mettre en place des processus standardisés, de mesurer la performance et la qualité du projet,

ainsi que de gérer les risques. Toutes les conditions seront dès lors réunies pour mettre en place “l’amélioration continue”. La pyramide des besoins du gestionnaire de projet se construit donc selon la figure 1.3.

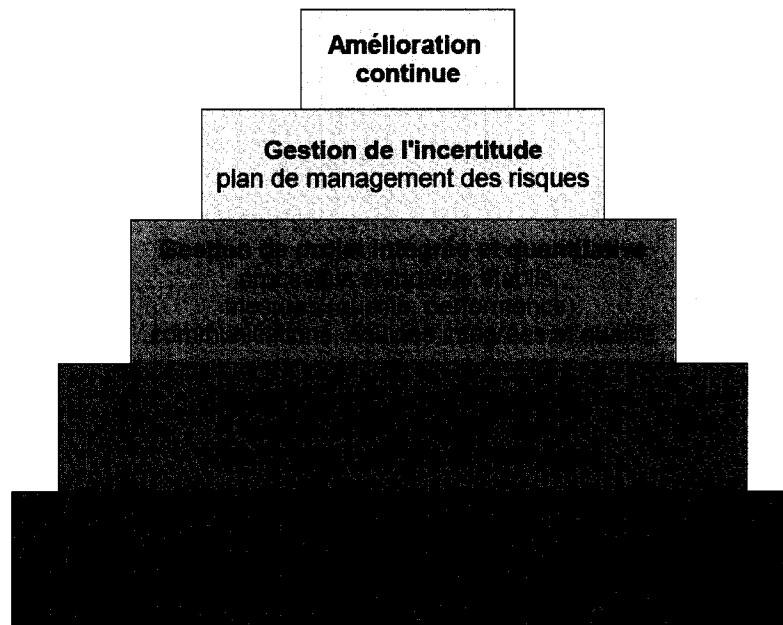


Figure 1.3 – Pyramide des besoins du gestionnaire de projet

1.3.2 Logiciels et fonctionnalités principales

Il s’agit désormais de reprendre les types de logiciels et les fonctionnalités identifiés tout au long de ce chapitre et d’en dresser la liste en les classant à l’aide de la pyramide des besoins du gestionnaire de projet. Le résultat de ce classement peut être présenté sous forme succincte, à la manière de la pyramide sur la figure 1.4.

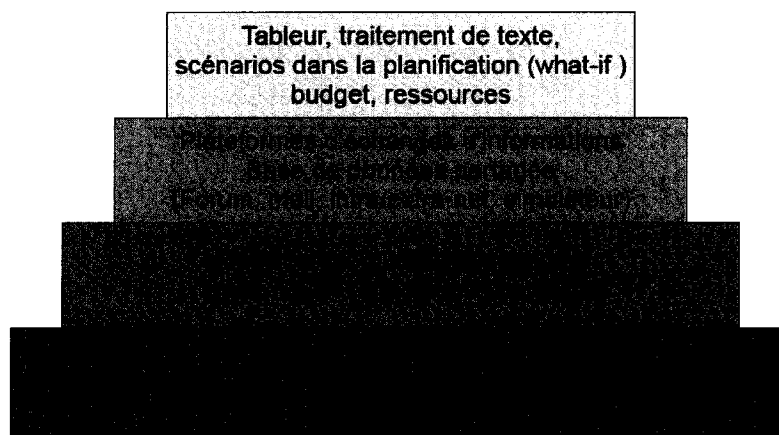


Figure 1.4 – Pyramide des besoins en logiciels

Chapitre 2

Les logiciels libres ou *open source*

“Parce qu’il n’y a pas un acteur de l’informatique qui ne soit pas impacté par le mouvement du logiciel libre” [39], le deuxième volet de cette étude bibliographique s’intéresse à le définir, le comprendre et à en saisir la teneur. À l’origine, deux philosophies, soutenues par la *Free Software Foundation* (FSF) d’une part et l’*Open Source Initiative* (OSI) d’autre part. Ces organisations posent les fondements du mouvement tel qu’il apparaît aujourd’hui. Cependant, différents enjeux entourent les logiciels libres. Ce chapitre expose donc les courants de pensées et propose une inspection des principaux enjeux que sont les licences, la sécurité, et les nouveaux modèles d’affaires.

2.1 Définitions

Pour comprendre ce que sont les logiciels libres, l’analogie proposée par Fogel et Moshe est intéressante. Les auteurs nous propulsent dans un monde imaginaire, dans lequel il existe un dispositif qui permet de reproduire tout objet physique à l’infini pour un coût nul ou quasi nul. Ils posent alors la question suivante : “si quelqu’un essaie de vous vendre un pneu pour votre voiture, pourquoi dans ce monde l’achèteriez-vous ?” [24] Il suffirait en effet de prendre n’importe quel pneu et de le passer dans le duplicateur.

Dans le monde réel, les logiciels libres peuvent être dupliqués à l’infini. Et

pour être précis, ce sont les fichiers source, également appelés codes sources qui seront dupliqués.

2.1.1 Code source

“Le *code source* est la version d’un programme qui est lisible et intelligible pour l’homme. C’est le code source qui est écrit par l’informaticien, le programmeur, et qui pourra être relu et modifié par d’autres. Les programmes peuvent être ensuite compilés, ce qui produit le code objet, ou binaire, ou encore exécutable, qui lui n’est pas compréhensible.” [8]

2.1.2 Deux philosophies : libre et *open source*

C’est l’accès au code source qui régit le fondement des deux écoles. N’importe qui peut, s’il le souhaite, obtenir les fichiers sources pour un prix quasi nul, soit en le téléchargeant sur Internet, soit en demandant une copie sur support cédérom et son envoi par courrier. De plus, afin de permettre la prolifération de programmes dont la qualité ne cesse de s’améliorer, les deux courants de pensées prônent la modification et la redistribution du code. Cependant, les motivations diffèrent.

“Richard Stallmann créateur de la *General Public Licence* (GPL) en 1989 et Linus Torvald concepteur en 1991 du noyau de Linux, système d’exploitation libre, multi tâche, multi plate-forme et multi utilisateur de type Unix” [39] incarnent tous deux, selon toute la littérature, le premier mouvement du logiciel libre, lequel a été reconnu à partir de la création de la *Free Software Foundation* (FSF) en 1985. Leur position s’appuie sur l’éthique, selon laquelle il existe “quatre libertés fondamentales : exécuter le programme,

l'étudier, l'adapter, et le redistribuer. Il faut souligner que le libre accès au code source est simplement rendu nécessaire par ces libertés fondamentales, et non une fin en soi" [39].

L'*Open Source Initiative* (OSI) quant à elle, a été créée en 1998, par le célèbre auteur du livre "la cathédrale et le bazar" [40], Eric S. Raymond, avec la complicité de Bruce Perens. Ces deux hommes ont participé au projet *Mozilla* qui a vu le jour suite à la libération du code du navigateur *Netscape*, concurrent direct d'*Internet Explorer*, le fureteur de *Microsoft*. On apprend d'ailleurs que la décision de libérer le code de *Netscape* a largement été influencée par les écrits de Raymond [22]. Pour l'OSI, "il ne s'agit guère d'éthique, ou même de philosophie, il est question avant tout de démontrer la supériorité des logiciels réalisés selon un modèle de développement *open source* communautaire, et de les faire entrer dans la sphère économique." [8] De plus, cette organisation met à disposition et promeut l'*open source definition*, composée de dix articles, lesquels doivent être respectés en tout point par la licence du logiciel, pour être dit *open source*.

On remarque donc que se sont les licences qui définissent si les logiciels sont libres (GPL) ou *open source* (licences compatibles OSI). C'est le thème abordé par la section suivante.

2.2 Licences publiques et gratuité

Afin de ne pas bafouer les droits d'auteurs des programmeurs proposant leurs créations en libre, il a fallu mettre en place un système de licences publiques. La licence choisie rend le logiciel *open source* ou libre. Il s'agira d'observer les licences publiques les plus populaires et de montrer que même si le terme anglais "*free*" ne signifie pas nécessairement "gratuit", dans la

pratique, c'est ce qui arrive la plupart du temps.

2.2.1 Licences publiques

L'*open source definition* est un bon point départ pour saisir les critères auxquels les licences publiques doivent se conformer pour être compatibles OSI. Comme indiqué plus haut, elle contient dix articles dont les trois premiers sont les plus importants.

1. Libre redistribution : la licence ne doit pas interdire à qui que ce soit de vendre ou donner le programme.
2. Code source : la licence doit permettre la distribution sous forme de code source, et si le code source n'accompagne pas le programme il doit être disponible de manière facile et pratiquement gratuite.
3. Travaux dérivés : la licence doit permettre des modifications et des travaux dérivés, et doit permettre que ces travaux soient distribués sous les mêmes termes de licence.

Le troisième point représente la base de la distinction entre deux grandes familles de licences, apprend-on dans la littérature.

La première famille, appelée *GNU* pour *GNU's Not Unix*, littéralement "GNU n'est pas UNIX" donne naissance à des licences très populaires. L'une d'entre elles, "utilisée par 70% des programmes" [8] libres, est la licence *GPL*. Son acronyme signifie *General Public Licence*. Cette licence est dite "virale" car elle possède un caractère héréditaire. En effet, tout travail dérivé d'un logiciel sous licence GPL doit être distribué sous la même licence.

La deuxième famille est appelée *BSD* pour *Berkley Software Distribution*. Elle prend le parti que la licence ne doit pas nécessairement obliger à ce que tout travail dérivé soit distribué sous les mêmes termes de licence et interprète le troisième point évoqué précédemment, de manière différente. En d'autres termes, les programmes sous licences BSD peuvent être modifiés et redistribués sous des licences de type propriétaire (privé) ou GNU (libre), pour ne donner que deux exemples.

Il existe par ailleurs de nombreuses autres licences publiques compatibles OSI comme la “licence MIT”, “la licence publique *IBM*” ou encore “la licence publique *Mozilla*” [22]. Cette dernière licence a vu le jour avec le projet *Mozilla* dont il a été question dans la section 2.1.2. Enfin, il est à noter que le site de l'*Open Source Initiative*, www.opensource.org, tient à jour la liste exhaustive des licences en conformité avec l'*open source definition*.

2.2.2 Gratuité

Le terme anglais “*free*” ne signifie pas nécessairement “gratuit”, mais plutôt “libre” selon le vocabulaire employé par la FSF. En effet, il lui arrive de vendre des logiciels GNU sur CD-ROM [22]. Malgré tout, on peut retenir qu'un logiciel libre est généralement gratuit, d'acquisition comme d'utilisation, puisqu'il s'avère difficile du point de vue de la licence, de vendre un programme que d'autres peuvent donner. Enfin, comme il sera développé dans la section 2.4, cela n'empêche pas qu'il soit accompagné d'une offre de services payants (intégration, support, formations, etc.) si bien que le coût total de possession est rarement nul [8].

2.3 Sécurité des logiciels libres

Lorsque le code source est disponible, la sécurité d'un programme informatique est une affaire inquiétante à première vue. De plus, le terme "sécurité" est difficile à cerner comme le prouve Richard Ford [26] dans un article publié en 2007. Ainsi, il est proposé de définir la sécurité logiciel dans un premier temps, puis de vérifier dans un deuxième temps, si cette inquiétude, quant à la sécurité des logiciels libres, est bien fondée. Peut-être est-elle même contradictoire ?

En effet, la Chine et l'Union Européenne ont demandé à l'entreprise *Microsoft* l'ouverture du code de son système d'exploitation *Windows* pour des raisons de sécurité et "pour mieux comprendre comment créer des produits qui peuvent être interopérables avec [celui-ci]" [19]. Certains dénoncent que *Microsoft* irait jusqu'à modifier ses propres logiciels afin que les produits concurrents ne fonctionnent plus convenablement [16]. Cela a pour effet de semer le doute chez les utilisateurs, en ce qui concerne la fiabilité et la sécurité des applications concurrentes. De plus, on peut mentionner les nombreux grands trous de sécurité des logiciels *Microsoft* ayant défrayé la chronique ces dernières années. Certes, comme l'explique Robert Di Cosmo, c'est parce qu'ils sont extrêmement répandus que les produits de la firme américaine font l'objet de nombreuses attaques, mais ils sont véritablement critiquables à juste titre. L'on comprend dès lors que l'enjeu de la sécurité est de grande envergure.

2.3.1 Définition complexe de la sécurité

Richard Ford a tenté de définir la sécurité d'un logiciel, un exercice plus complexe qu'il n'en a l'air. En effet, il explique qu'une définition naïve serait de simplement maintenir la confidentialité, l'intégrité et la disponibilité des informations. (*CIA* en anglais pour *Confidentiality, Integrity, Availability*). En réalité, il s'avère très difficile de mesurer la sécurité d'un logiciel informatique. Malgré tout, il y existe deux moyens connus :

- déterminer la probabilité que l'un des volets du *CIA* soit violé,
- déterminer le nombre effectif de vulnérabilités dans le produit.

La première mesure dépend du nombre d'attaques dirigées vers le logiciel assure l'auteur. "*MS-DOS* serait donc le système d'exploitation le plus sécuritaire, puisque personne ne cherche plus à l'attaquer" [26].

La deuxième mesure ne se fait qu'à posteriori, une fois que les vulnérabilités ont été rendues publiques. Il existe une variante à cette approche : compter le nombre de jours à risques, c'est-à-dire le temps qu'il faut à l'éditeur pour proposer un correctif, quand bien même il s'avère efficace. Là encore, cela dépend du nombre d'attaques connues.

2.3.2 La sécurité des logiciels libres

Les motivations des membres de la communauté du libre ainsi que le processus de développement lui-même jouent indéniablement en faveur de la sécurité d'après tous les auteurs qui se sont penchés cette question.

Un programmeur souhaitant contribuer en créant du code source s'applique très fort, car sa réputation de bon programmeur est en jeu. De plus, Feller et Fitzgerald rapportent les résultats de plusieurs études, qui montrent

que la productivité des programmeurs est radicalement différente dans le milieu professionnel. “Il est admis qu’un employé sur cinq ralentit les autres” [22]. Dans le monde du libre, beaucoup de novices collaborent bien sûr, mais à la hauteur de leurs compétences réelles, sans pour autant ralentir le développement global.

Le processus de développement est pour ainsi dire à l’envers du processus traditionnel. Les utilisateurs représentent la ressource la plus importante, puisque c’est une solution à leurs besoins qu’il s’agit de développer. Ainsi, ils contribuent activement aux phases de tests et à la documentation. En outre, le processus se différencie par une réelle revue du code impartiale. La littérature argue qu’il n’y a pas meilleur moyen pour augmenter la sécurité d’une application. La très célèbre phrase de l’auteur du livre *la Cathédrale et le bazar* : “*Given enough eyeballs, all bugs are shallow*” [40], citée de nombreuses fois ([21], [26]), explique la méthodologie dont la communauté du libre fait usage. Pour s’en convaincre, les débuts de la cryptographie sont racontés dans le livre blanc “Introduction à l’*open source* et au logiciel libre”. On pensait alors qu’un algorithme secret serait très sécuritaire, or il s’est avéré qu’il pouvait être défaillant puisque seulement connu et révisé que par trop peu d’experts. Dès lors, “la sécurité par l’obscurité” [26], une stratégie adoptée par les éditeurs traditionnels, à l’instar de *Microsoft*, ne semble pas être idéale.

2.4 Nouveaux modèles d’affaires

Si l’on en croit les dénonciations de Robert Di Cosmo, les demandes de plusieurs gouvernements du monde ainsi que les procès antitrust formulés à l’encontre de *Microsoft*, les logiciels libres ne représentent pas uniquement des

programmes plus sûrs, mais également et surtout, la sécurité pour les utilisateurs de préserver leur liberté de choisir. Cette liberté fondamentale aux yeux des partisans de la FSF s'exprime de plusieurs manières. La plus notoire est celle qui consiste à construire de nouveaux modèles d'affaires, en s'appuyant certainement sur les bases plus économiques lancées par Eric S. Raymond et Bruce Perens. C'est la raison pour laquelle cette section s'intéresse aux nouveaux modèles d'affaires.

“De fait, tout n'est pas gratuit dans l'*open source*, et il y a une vraie *économie de l'open source*, qui a ses particularités.” [8] Tous les auteurs s'accordent à dire qu'il existe 5 grandes typologies qu'il s'agit de présenter ci-après.

2.4.1 Les distributeurs

Il existe tout d'abord les distributeurs qui sont souvent considérés comme des “joueurs purs” [22]. Leurs activités consistent à

- “*sélectionner* des produits et versions, entourant le noyau Linux
- *valider* la maturité et la robustesse de ces produits
- *distribuer* ces produits et leurs mises à jour, c'est-à-dire assurer leur acheminement jusqu'aux utilisateurs-clients
- assurer le *support* de ces produits : hot-line, traitement des demandes, conseil, formation.” [8]

Elles ne détiennent donc pas les droits d'auteurs des logiciels qu'elles distribuent, mais trouvent leurs revenus dans leurs activités de support principalement. Les compagnies *Red Hat*, *VA Linux*, *Ubuntu* sont citées comme celles qui réussissent le mieux dans ce modèle. Pour donner un aperçu de sa pertinence, la firme *Red Hat* enregistre près de 300 millions de dollars de

chiffre d'affaires et emploie 2200 personnes dans le monde.

2.4.2 Les éditeurs *open source*

“C’est MySQL qui a ouvert la voie de la logique de l’éditeur *open source*” [8]. C’est une société commerciale classique, c’est-à-dire qu’elle cherche à réaliser des profits. Elle édite son propre système de gestion de base de données, mais le distribue sous licence *open source*. Elle va même jusqu’à le distribuer sous licence double en réalité. Ainsi, un éditeur comme MySQL tire ses revenus, de la vente de licences commerciales d’une part, lorsqu’il adopte le système de double licence, et de la vente de support et de prestations d’autre part. L’éditeur possédant une expertise totale sur les produits qu’il propose, apparaît comme idéal pour apporter des modifications, des personnalisations à ses clients. C’est pour cette raison que ce modèle fonctionne si bien.

2.4.3 Les fondations

Les fondations sont des organismes à but non lucratif qui proposent des logiciels ou composants sous licences publiques.

La *Free Software Foundation*, déjà évoquée plus haut pour ses actions militantes de défense des quatre libertés fondamentales et pour sa définition des licences GNU GPL, joue un rôle clé dans le développement de nombreux composants libres.

La fondation *Apache* qui a élaboré le serveur Web éponyme, est considérée aujourd’hui comme “l’un des temples de l’*open source*” [8]. Son serveur, un logiciel libre de grande qualité, détenait déjà 50% des parts du marché des serveur Web de la planète. [16]

Il existe d'autres fondations dont l'origine se trouve chez les concurrents de *Microsoft*. La fondation *Eclipse* par exemple, soutenue par IBM, s'est donné pour mission de développer et promouvoir une plate-forme de développement intégrée (IDE, *Integrated Development Environment*). "L'intérêt stratégique, pour IBM, est de contrer la plate-forme *Microsoft* dans les entreprises." [8]

2.4.4 Les firmes de conseil

Les firmes de conseil jouent un rôle d'intégrateurs de solutions libres, un peu à l'image des distributeurs, dans le sens où ils développent une expertise des différentes offres de la communauté et prépare souvent des projets clés en main. De plus, ils sont capables d'apporter un certain support puisque le code est ouvert. Mais leur différenciation réside dans le fait qu'elles sont plus généralistes que les distributeurs dont il a été question plus avant, car elles ne s'occupent pas de la distribution des logiciels. Elles peuvent donc se concentrer sur "le déploiement et le support de configurations multi-produits à base de logiciels *open source*". [8]

2.4.5 Éditeurs privés offrant une version libre

Enfin, il y a les éditeurs traditionnels ou privés, qui "se posent des questions sur l'ouverture de leur code, au moins pour des raisons tactiques visant à attirer de futurs acteurs avec une version dépouillée sous licence GPL permettant de tester le produit". [39] Il est à noter que de plus en plus d'éditeurs privés tentent ce genre de stratégie, car ils mesurent la force de la communauté du libre. Cependant, beaucoup échouent car ils retiennent le cœur de leurs produits en code fermé.

Deuxième partie

Les logiciels libres en gestion de projet

Les principaux besoins en logiciels informatiques du gestionnaire de projet ont été identifiés puis classés par ordre d'importance dans le chapitre 1, pour aboutir à la modélisation de ceux-ci en une pyramide des besoins, par analogie à la théorie de Maslow. Le chapitre 2 a montré que les logiciels libres jouent un rôle important dans l'évolution du poste de travail, car au-delà de la diminution des coûts qu'ils permettent, ils sont au moins autant sécuritaires sinon plus que leurs homologues propriétaires. Ainsi, cette partie cherche à présenter l'offre de solutions libres qui répond aux besoins du gestionnaire de projet et l'évalue selon certains critères qui seront détaillés dans la section 3.2. Suite à cela, viendra le temps de poser les hypothèses de recherche qui seront testées dans la partie III.

Chapitre 3

Méthode de recherche et d'évaluation

La communauté du libre est extrêmement active et propose de très nombreuses applications dont leur nombre ne cesse de grandir. Malheureusement, toutes ne sont pas adéquates ou performantes puisque leur processus de développement n'est pas terminé. Il faut donc se doter d'une méthode de recherche qui permette d'identifier les principaux logiciels arrivés à maturité, puis d'une méthode d'évaluation, afin de vérifier la pertinence des fonctionnalités proposées en rapport avec les besoins du gestionnaire de projet.

3.1 Méthode de recherche

La méthode de recherche employée pour ce travail se déroule en deux étapes. Dans la littérature sur le thème de la communauté *open source*, plusieurs répertoires de logiciels libres sont mentionnés, parmi lesquels *Source Forge* est reconnu pour être le meilleur. Le point de départ semble donc évident. Toutefois, le trop grand nombre d'applications répertoriées oblige à perfectionner la méthode dans une deuxième étape. Les écrits qui existent sur les solutions libres en gestion de projet sont certes peu nombreux, mais demeurent tout à fait appropriés.

3.1.1 Répertoires de logiciels libres

Le plus grand et le plus respecté des répertoires en ligne de logiciels libres à ce jour, est celui de la communauté *Source Forge* (<http://www.sourceforge.net>). En date du 20 mai 2008, il regroupait en effet plus de 310 000 projets.

En outre, il existe une section intitulée *Project Management* qui répertoriait à la même date environ 1600 produits. Dès lors, on comprend que la tâche serait trop fastidieuse de chercher parmi toutes ces applications si bien que la littérature sera un guide utile.

3.1.2 Littérature

Peu de gens se sont penchés sur le problème des logiciels libres en gestion de projet. Malgré tout, le journal du net a publié quelques articles intéressants sur le sujet, et parle de solutions “qui commencent à apparaître et à se faire une place sur le marché” [7], auxquelles on peut ajouter quelques suggestions d’ERP (*Enterprise Resource Planning*) trouvés dans les livres blancs publiés par l’entreprise SMILE.

3.2 Méthode d’évaluation

Afin de développer une méthode d’évaluation pertinente pour le présent projet, il est seyant dans un premier temps, de prendre connaissance de ce qui se fait.

Le modèle de maturité proposé par Capgemini ou encore le *Navica open source maturity model* [45] sont deux exemples de modèles d’évaluation des

logiciels libres qui ont été développés ces dernières années.

Le premier permet de trouver le produit *open source* le plus adapté en 7 étapes. Dans les grandes lignes, il s'agit selon le modèle d'effectuer présélection, puis d'utiliser des indicateurs pour noter les produits et déterminer l'importance de ces indicateurs pour enfin, obtenir une carte d'évaluation pour chacun des produits sélectionnés [45].

Le second modèle "utilise six éléments du produit en trois phases : évaluer la maturité, assigner un poids aux facteurs, et calculer la note de maturité pour le produit" [45].

D'autres ont tenté de définir les critères de succès et d'échec d'un logiciel libre [20]. D'après une étude empirique basée sur le répertoire *Source Forge*, les auteurs ont montré que le nombre de publications et l'activité de la communauté jouent un rôle primordial. En effet si à la troisième publication, le logiciel ne convient toujours pas, c'est-à-dire qu'il n'intéresse personne, alors le projet va être abandonné par ses auteurs.

D'autres encore ont mis sur pieds des grilles d'évaluation multi critères pour les environnements de développement *open source* [47].

Tous s'intéressent donc principalement au processus de développement, à la communauté entourant le logiciel, à la fréquence des activités de mises à jour, et à l'existence d'un support technique et de son efficacité. Ces critères sont tout à fait pertinents et ont été utilisés par la littérature pour la sélection des logiciels libres arrivés à maturité.

Dans un deuxième temps, il est question d'évaluer les produits selon les besoins réels du gestionnaire de projet. Ainsi, la pyramide développée dans la partie I nous est utile afin de créer une grille d'évaluation des besoins en logiciels. Il ne faut cependant pas se limiter aux offres spécialisées en gestion

de projet, mais bien couvrir tous les besoins identifiés. Malgré tout, à partir de la sélection des logiciels libres de la littérature, il s'agira de procéder à une vérification des licences, car elles doivent être compatibles OSI ou libres, puis il faudra télécharger et installer chacun des logiciels retenus, afin de tester les fonctionnalités décrites sur son site Internet et procéder à son évaluation grâce à la grille d'évaluation.

Cette grille, construite à partir de la pyramide des besoins est présentée par la figure 3.1 pour les besoins primaires, et la figure 3.2 pour les besoins avancés. Elle indique en colonnes les besoins identifiés et en lignes les logiciels à évaluer. Une marque verte en forme de "V" indique que le logiciel évalué intègre la fonctionnalité et répond ainsi, de manière adéquate au besoin identifié ; tandis qu'un rond rouge barré indique que le logiciel n'intègre pas la fonctionnalité souhaitée.

Logiciels	Définition du projet	Planification, suivi et contrôle					
		Coûts				Ressources	
		Valeur planifiée	Calcul budget	Feuilles de temps	Utilisation ressources	Néologie	
	✓						

Figure 3.1 – Grille d'évaluation des besoins primaires

Logiciels	Gestion de projet intégrée et quantitative					Gestion de l'incertitude	
	Logiciel	Logiciel	Logiciel	Logiciel	Logiciel	Durées alternatives	Génération de scénarios
1	✓	⊗					

Figure 3.2 – Grille d'évaluation des besoins avancés

Par ailleurs, un lien avec l'amélioration continue proposée par le CMMI peut être établi dès à présent. Comment est-il possible d'améliorer les outils selon des besoins qui évoluent ? La réponse est la structure modulaire de tout logiciel libre convenablement conçu, car elle permettra d'améliorer les outils en suivant l'évolution des besoins. Ainsi, on possède flexibilité et scalabilité (possibilité d'extensions) tant dans les fonctionnalités du logiciel que dans son déploiement. Les logiciels libres ne respectant pas une certaine modularité seront donc écartés de la présente évaluation puisqu'ils seraient trop limités.

Enfin, une confrontation avec les logiciels propriétaires les plus populaires est très intéressante, car s'ils sont tant utilisés, c'est parce qu'ils doivent répondre à la plupart des besoins du gestionnaire de projet.

Chapitre 4

Évaluation des outils libres

En utilisant les méthodes de recherche et d'évaluation décrites au chapitre 3, il s'agit maintenant de proposer les logiciels libres ou *open source* les plus matures qui répondent aux besoins du gestionnaire de projet.

4.1 Suite *Open Office*

Lors de la rédaction du présent document, la suite *Open Office* est disponible dans sa version 3.0 sous licence LGPL (*Lesser General Public License*). Cette suite bureautique contient cinq outils principaux :

- un traitement de texte (*Writer*)
- un tableur (*Calc*)
- un outil de création de diaporamas (*Impress*)
- un logiciel de dessin vectoriel (*Draw*)
- un gestionnaire de bases de données (*Base*)

Le traitement de texte permet la création et l'édition de textes et d'images pour les courriers, les rapports, documents ou les pages Web.

Le tableur permet d'effectuer des calculs, de l'analyse d'informations en générant des diagrammes (en barres, en lignes, en zones, en secteurs, en toile, etc.) et de gérer des listes grâce aux feuilles de calcul. Il intègre même la pos-

sibilité d'un travail collaboratif sur les feuilles de calcul. Plusieurs utilisateurs peuvent ainsi travailler sur une même feuille de calcul et lors de l'enregistrement, toutes les modifications seront visibles avec des couleurs différentes. L'auteur peut par la suite gérer les conflits au sein d'une boîte de dialogue afin d'optimiser les modifications.

L'outil *Impress* permet quant à lui de créer et d'éditer des présentations ou diaporamas pour préparer les réunions ou toute autre communication du projet.

Le logiciel de dessin vectoriel permet la création et l'édition de dessins, d'organigrammes et de logos.

Enfin, le gestionnaire de bases de données permet de créer des requêtes et des rapports pour mieux gérer les informations.

Ces outils répondent donc à plusieurs types de besoins du gestionnaire de projet. Si l'on se réfère à la pyramide des besoins développée dans la partie I, ils couvrent une partie des besoins primaires (définition, planification, suivi et contrôle du projet) et une partie des besoins de niveau 3 (gestion de projet intégrée et quantitative). En effet, comme il a été vu lors de la revue bibliographie, la charte, le plan de management de projet et l'énoncé préliminaire du contenu sont tout d'abord des documents texte. La planification des coûts est particulièrement aisée à l'aide d'une feuille de calcul et l'outil de base de données peut être envisagé pour faire de l'archivage de données historiques et servir de base aux décisions prises dans l'amélioration des processus.

4.2 Logiciels en gestion de projets

Dans cette section sont présentés six logiciels matures en gestion de projet, suivis d'un récapitulatif illustrant leur capacité respective à répondre aux besoins du gestionnaire de projet.

4.2.1 DotProject

[http ://www.dotproject.net](http://www.dotproject.net) *DotProject* est une application Web, distribuée sous licence GPL et supportée par une équipe de volontaires et les utilisateurs eux-mêmes. En effet, aucune entreprise ou fondation ne se trouve derrière ce projet. De plus, le logiciel est disponible dans 24 langues.

DotProject intègre quelques outils certes basiques, mais intéressants, en ce qui concerne la définition du projet. En effet, il est tout d'abord possible d'ajouter une description textuelle au projet et à chaque tâche. Mais ce qui semble plus utile encore, c'est la possibilité d'ajouter un lien hypertexte dans les propriétés du projet. Ce lien peut diriger sur un site Internet contenant tous les documents principaux qui doivent être produits dans ce domaine de connaissance. Il est également possible d'ajouter directement des fichiers dans le projet. Ces fichiers seront partagés à tous les membres de l'équipe de projet. Ensuite, *DotProject* possède toutes les fonctionnalités de planification et suivi des délais identifiées par la pyramide des besoins. Il possède également quelques modules correspondant à certains besoins avancés du gestionnaire de projet comme la gestion multi projets et le forum. Enfin, comme il a été dit plus tôt, des fonctionnalités de *groupware* (travail de groupe) sont disponibles avec le partage de documents, disponible également au niveau de chaque tâche.

4.2.2 Ganttproject

<http://www.ganttproject.biz> *Ganttproject* a été initialisé par le français Thomas Alexandre et est distribué sous licence GPL. L'application a été programmée en Java, c'est-à-dire qu'elle est indépendante de la plate-forme sur laquelle elle est exécutée. De plus, elle est disponible dans 19 langues.

Cette solution répond aux besoins de planification de base, incluant la gestion des ressources. De plus, il est possible d'exporter les chartes de Gantt ou les diagrammes d'utilisation des ressources aux formats HTML, PDF ou encore sous forme d'images PNG.

4.2.3 GNOME Planer

<http://live.gnome.org/Planner> Richard Hult et Mikael Hallendal, deux employés de l'entreprise Imendio ont développé une application de planification et de suivi de projets qui s'appelait ImmendioPlaner. Quelques années plus tard, leur création a été offerte à la fondation *GNOME* dont le but est de proposer un ensemble d'applications de très haute qualité pour l'environnement GNU/Linux et Unix. L'application d'Imendio a donc été rebaptisée *GNOME Planer* et est maintenant distribuée sous licence GPL version 2. Une version pour Microsoft Windows est également disponible. *GNOME Planer* possède les mêmes fonctionnalités que *Ganttproject* et propose en plus une fonctionnalité d'attribution du coût unitaire d'une ressource, ce qui permet de connaître le coût d'une tâche, mais ne propose pas de calcul budgétaire global.

4.2.4 Open Workbench

<http://www.openworkbench.org> *Open Workbench*, de l'éditeur Californien *Niku*, vise des besoins de planification élevés et s'affiche clairement comme l'alternative libre à *Microsoft Project*. Il est en effet possible de gérer plusieurs projets et sous-projets tout en conservant les dépendances entre eux. Il intègre toutes les fonctionnalités correspondantes aux besoins primaires, les feuilles de temps mises à part. En ce qui concerne le suivi du projet, plusieurs fonctionnalités de comparaison, entre ce qui a été planifié et ce qui a été effectivement réalisé, sont disponibles. Une fonction du calcul de la valeur acquise est également présente.

Par ailleurs, il faut savoir que cette application est disponible sous licence Mozilla Publique (MPL 1.1) et traduite dans trois langues (Anglais, Français et Allemand).

4.2.5 PhpCollab

<http://www.php-collab.com/blog> *PhpCollab* est une solution *groupware* de l'éditeur français Neuronnexion, distribuée sous licence GPL et disponible en trois langues (Français, Anglais et Italien). Il s'agit d'une application Web qui offre la plupart des fonctionnalités en réponse aux besoins primaires identifiés, malgré une gestion des coûts totalement absente. (définition, planification, suivi et contrôle de projet). En effet, tout comme *Dot-Project*, *PhpCollab* propose des outils *groupware* qui permettent de déposer des fichiers directement dans le projet ou sur les tâches pour leur définition plus fine. De plus, des besoins avancés, comme la génération de graphiques à partir de données statistiques ou encore la prise en charge des forums et la notification par courriel, font de cette application, une solution performante

à la gestion de projet distribuée.

4.2.6 TaskJuggler

<http://www.taskjuggler.org> *Task Juggler* est un outil distribué sous licence GPL version 2, et développé initialement pour Linux et Unix, mais certains développeurs l'ont porté sur d'autres plate-formes comme Windows ou MacOS, bien que l'équipe principale ne s'y implique que très peu. Ce programme semble néanmoins le plus avancé en termes de planification, suivi, contrôle et gestion de l'incertitude. En effet, la gestion des coûts et des ressources y est intégrée, comprenant un outil automatique de nivelage des ressources. De plus, on peut générer un nombre illimité de scénarios à partir du même projet et effectuer de l'analyse de cas (*What-if analysis*). En revanche, la prise en main du logiciel est plus complexe que les autres puisqu'il faut littéralement "programmer" le projet et les tâches. La syntaxe est certes très facile à appréhender puisque l'interface guide beaucoup l'utilisateur, la documentation très bien fournie, mais cela peut surprendre au départ.

4.2.7 Résultats

Il s'agit désormais de présenter ces premiers résultats grâce à la grille d'évaluation développée à la section 3.2. Pour se faire, la figure 4.1 reprend la liste des applications et les besoins primaires du gestionnaire de projet. Elle met donc en évidence si les applications couvrent ou non ces besoins primaires. De la même manière, la figure 4.2 met en évidence les réponses aux besoins avancés du gestionnaire de projet.

Gantt Project et *GNOME Planer* sont équivalents et apparaissent comme

des outils permettant de faire de la planification et du suivi de base. Il est à noter que ces deux logiciels se ressemblent tant dans les fonctionnalités que dans leur apparence et leur simplicité d'utilisation.

Pour des besoins plus avancés en termes de planification ou de suivi, *Open Workbench* est très bon. Il est également facile à prendre en mains et permet même de générer des scénarios pour la prise en charge des risques. En revanche, son fonctionnement en *stand alone* fait qu'il n'intègre aucune fonctionnalité pour faciliter la communication.

Si l'on vise des performances encore plus élevées en termes de planification et de gestion de l'incertitude, sans pour autant avoir besoin de fonctionnalités de communications, il faudra se tourner vers *Task Juggler*. Mais ce sera au coût d'un apprentissage plus long, car ce logiciel n'est pas simple à appréhender.

Par ailleurs, si l'on souhaite mettre en place une application Web performante, afin de jouir pleinement de fonctionnalités des environnements collaboratifs, *Dott Project* ou *Php Collab* sont deux très bonnes solutions. Néanmoins, *Php Collab* répond à plusieurs besoins avancés du gestionnaire de projet que *Dott Project* ne prend pas du tout en charge. Dès lors, si une installation sur un serveur Web ne présente aucune difficulté particulière lors de son déploiement, il est certainement préférable de choisir *Php Collab*.

Enfin, il faut noter qu'aucune de ces solutions *open source* ne propose de réponse à la feuille de temps. Peut-être trouvera-t-on la réponse dans les ERP libres ?

Planification, suivi et contrôle												
Logiciels	Définition du projet						Coûts			Ressources		
							Valeur planifiée	Calcul budget	Feuilles de temps	Utilisation ressources	Nécessité	
	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	○	✓	✓	○	✓	✓	○	○	○	○	○	○
	○	✓	✓	✓	✓	✓	✓	○	○	✓	✓	✓
	○	✓	✓	✓	✓	✓	✓	✓	○	✓	✓	✓
	✓	✓	✓	✓	✓	✓	○	○	○	✓	○	○
	○	✓	✓	✓	✓	✓	✓	✓	○	✓	✓	✓

Figure 4.1 – Résultats de l'évaluation des besoins primaires - logiciels libres en gestion de projet

4.3 Les ERP libres

Dans la littérature qui s'intéresse aux logiciels libres, plusieurs ERP libres (*Enterprise Resource Planning*) sont cités. En Français, on dit que ce sont des logiciels de planification des ressources de l'entreprise. Il est intéressant pour ce travail, de noter pour trois d'entre eux, les fonctionnalités proposées qui permettent de répondre aux besoins en gestion de projet.

4.3.1 Open ERP

Open ERP, anciennement appelé *TinyERP* [3], est une application multi-plateforme développée par *Tiny SPRL* et distribuée sous licence GPL version 2.

On retrouve “la notion de tableaux de bord personnalisés pour le monitoring par les chefs de projets (dont diagramme de Gantt) ou la saisie des imputations par les employés. *TinyERP* offre même une intégration des plannings avec *Outlook* ou *Thunderbird*” [48]. Si l'on prend en considération que le fait de pouvoir attacher des documents et ajouter des notes aux projets et aux tâches, on peut dire que ce progiciel couvre tous les besoins primaires du gestionnaire de projet. De plus, il est capable de faire la gestion multi-projets et propose plusieurs fonctionnalités de *reporting* et d'analyse. Il couvre donc également une partie non négligeable des besoins avancés. Notons toutefois l'absence de fonctionnalités permettant d'appréhender l'incertitude.

4.3.2 Neogia

Neogia est distribué sous licence LGPL qui couvre presque la totalité des besoins primaires. En effet, selon Smile, “dans Neogia ou TinyERP, la gestion des RH [ressources humaines], incluant les feuilles de temps et les notes de frais, permettent une assignation fine des tâches et projets. ” [48] En revanche, la prise en charge des besoins avancés reste assez limitée.

4.3.3 Project-Open

Project-Open est une application Web, distribuée sous double licence GPL et propriétaire. En effet, il existe une offre de base ouverte et des modules supplémentaires dont le code reste fermé. Pour cette étude, seule l’offre de base est à prendre en considération.

Si l’on en croit la description du progiciel, il sait tout faire et répond à tous les besoins qui ont été identifiés par la pyramide. Après étude, il s’avère que c’est presque un sans faute.

En effet, il propose sept modules principaux dont les cinq premiers sont les plus intéressants :

- Gestion de projets / porte-folio
- Gestion des relations clients
- Gestion des finances
- Gestion des ressources humaines
- Gestion des savoirs

Le module gestion de projets / porte-folio se décompose en 16 sous-modules et on peut dire qu’ils couvrent la totalité des besoins primaires et avancés du gestionnaire de projet à l’exception d’une fonctionnalité, l’af-

fectionation de durées alternatives pour les tâches. Toutefois, il est à noter que *Project-Open* propose une charte graphique de gestion des risques qui n'existe dans aucune autre solution libre étudiée.

4.3.4 Résultats

De la même manière que pour les logiciels libres en gestion de projet, la figure 4.3 reprend la liste des applications ERP et les besoins primaires du gestionnaire de projet. Elle met donc en évidence si les ERP libres couvrent ou non ces besoins primaires. La figure 4.4 met, quant à elle en évidence, les réponses aux besoins avancés du gestionnaire de projet.

La première remarque répond à la question de la section 4.2.7 : les feuilles de temps sont bien prises en charge dans les ERP libres. De plus, la gestion des coûts, qui laissait parfois à désirer dans les logiciels précédents, est désormais pleinement intégrée. Enfin, la solution *Project-Open* apparaît nettement supérieure à toutes les autres offres en termes de fonctionnalités en réponse aux besoins du gestionnaire de projet.

Logiciels	Définition du projet	Planification, suivi et contrôle									
		Coûts					Ressources				
		Valeur planifiée	Calcul budget	Feuilles de temps	Utilisation ressources	Mixage					
	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	⊖	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Figure 4.3 – Résultats de l'évaluation des besoins primaires - ERP libres

Logiciels	Gestion de projet intégrée et quantitative										Gestion de l'incertitude	
											Durées alternatives	Génération de scénarios
	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	⊖	⊖
	✓	✓	⊖	✓	✓	✓	✓	✓	✓	✓	⊖	⊖
	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

Figure 4.4 – Résultats de l'évaluation des besoins avancés - ERP libres

Chapitre 5

Confrontation avec logiciels propriétaires

Dans le chapitre précédent, les logiciels libres ont été évalués à l'aide de la grille d'évaluation développée dans la section 3.2. Pour compléter cette évaluation, il faut effectuer la même analyse avec les logiciels propriétaires en gestion de projet les plus reconnus, à savoir, *MS Project 2007*, *Primavera* et *PS Next*.

5.1 MS Project 2007

“L'écosystème de *Project* est riche et fidèle. Il compte plus de 20 millions d'utilisateurs et représente la sixième source de revenus de *Microsoft*, après *Exchange*.” [28] Pourtant, dès l'introduction de son livre, Paul Eastwood Harris affiche clairement que la version de *MS Project 2007* ne permettait pas de prendre en charge tous les processus préconisés par le *Guide PMBOK*® [29]. Pourtant il a écrit ce livre dans le but de proposer les solutions les plus concrètes possible afin de mieux appréhender les fonctionnalités de *MS Project* en relation avec les recommandations du *Guide PMBOK*®.

En reprenant le modèle développé pour cette recherche, on montre que la plupart des besoins primaires sont bien couverts par l'application. En

revanche, même si la publicité se vante de la fonctionnalité “multiple-level undo” (retours en arrière multiples) qui selon elle, serait une implémentation de l’outil “What-if cases” et permettrait donc la gestion des risques, on ne peut pas, de manière objective, attribuer la gestion de l’incertitude à *MS Project 2007*.

5.2 Primavera 6

Sur le site officiel de *Primavera*, on peut lire que ce logiciel satisfait plus de 76 000 clients dans le monde entier. Dans sa version 6, *Primavera* répond à tous les besoins primaires à l’exception de la définition du projet et des feuilles de temps. En revanche, à la différence de *MS Project 2007*, il est possible de faire la gestion multi-projets, d’afficher des statistiques, sans oublier une réelle prise en charge de scénarios alternatifs.

5.3 PS Next

Ce logiciel est intéressant car il se base sur une interface Web. Il intègre ainsi des fonctionnalités plus avancées que ses deux concurrents comme la notification par courriel et le partage de documents. Il va même un peu plus loin que le simple partage, puisqu’il dispose de fonctionnalités de gestion de la documentation. En ce qui concerne les besoins primaires, il possède un module “Ma page” qui permet aux membres de l’équipe de projet de remplir des feuilles de temps. Enfin, il est à noter que *PS Next* permet de générer des scénarios différents et d’en sauvegarder cinq d’entre eux.

5.4 Résultats

La figure 5.1 reprend la liste des applications propriétaires étudiées et les besoins primaires du gestionnaire de projet. Elle met donc en évidence si ces logiciels couvrent ou non ces besoins primaires. La figure 5.2 met, quant à elle en évidence, les réponses aux besoins avancés du gestionnaire de projet.

On remarque immédiatement que *MS Project 2007* est réellement en compétition avec la solution libre *Open Workbench*. Ce dernier cherche effectivement à faire mieux que la solution proposée par *Microsoft*. En termes de fonctionnalités utiles au gestionnaire de projets, c'est tout à fait exact. Cette analyse nous permet de confirmer les affirmations de la communauté entourant *Open Workbench*.

Primavera trouve un compétiteur à partir de *Task Juggler* et reste visiblement meilleur que son homologue *open source*. Mais les solutions ERP libres dépassent largement les possibilités de *Primavera*.

Enfin, l'application Web *PS Next* ne trouve d'égal que dans les ERP libres, tant ses fonctionnalités sont avancées. Et malgré cette étude, il est difficile de départager *Project-Open* qui intègre les forums de la solution propriétaire *PS Next* qui n'en intègre pas. Toutefois, on peut affirmer que le coût total d'acquisition devrait faire la différence.

Planification, suivi et contrôle											
Logiciels	Définition du projet						Coûts			Ressources	
							Valeur planifiée	Calcul budget	Feuilles de temps	Utilisation ressources	Verdoye
	⊗	✓	✓	✓	✓	✓	✓	✓	⊗	✓	✓
	⊗	✓	✓	✓	✓	✓	✓	✓	⊗	✓	✓
	✓	✓	✓	✓	✓	✓	✓	⊗	✓	✓	✓

Figure 5.1 – Résultats de l'évaluation des besoins primaires - Logiciels propriétaires

Gestion de projet intégrée et quantitative											
Logiciels	Gestion de projet intégrée et quantitative						Gestion de l'incertitude				
							Durées alternatives	Génération de scénarios			
	⊗	⊗	✓	✓	✓	✓	⊗	⊗	⊗	⊗	⊗
	✓	✓	✓	✓	✓	✓	⊗	⊗	⊗	✓	✓
	✓	✓	✓	✓	✓	✓	⊗	⊗	⊗	✓	✓

Figure 5.2 – Résultats de l'évaluation des besoins avancés - Logiciels propriétaires

Chapitre 6

Hypothèses de recherche

L'offre de la communauté du libre dans le domaine de la gestion de projet arrive à maturité et concurrence, voir surpasse les offres propriétaires. De plus, sachant que la gestion de projet distribuée est rendue possible grâce à l'Internet, et que l'Internet existe grâce aux logiciels libres et standards ouverts comme l'indique toute la littérature [16], [41] et [40], il est judicieux d'émettre la première hypothèse suivante, selon laquelle, les logiciels libres ou *open source* sont des outils très adaptés à la gestion de projet distribuée.

Leur code source est disponible, et sa modification est autorisée, tout comme sa redistribution, selon les termes de la licence publique. On obtient donc, en deuxième hypothèse, une grande flexibilité, car on peut adapter les outils de gestion de projet aux besoins qui évoluent sans cesse. Cette deuxième hypothèse répond adéquatement au concept "d'amélioration continue" développé par le modèle de maturité CMMI dont il a été question dans la section 1.2. Et quand bien même, on ne possède aucune compétence pour améliorer soi-même le logiciel, on a, comme il a été mentionné plus haut plusieurs fois, un certain contrôle sur les nouvelles fonctionnalités devant apparaître dans les versions ultérieures du logiciel. Ainsi, une firme souhaitant atteindre le dernier niveau de maturité proposé par le CMMI, celui de l'optimisation, aurait un grand intérêt à utiliser ses ressources ou partenaires informatiques pour trouver, adapter, et faire évoluer suivant ses propres besoins, des solutions libres au lieu d'acheter des solutions propriétaires en

gestion de projets.

“Les pouvoirs publics ne s’y sont pas trompés, tant en France où la vague du logiciel libre est irrésistible dans les administrations, que dans l’Union Européenne, l’État du Massachussetts, ou le Brésil, l’Inde, le Venezuela et le Pérou qui mènent des politiques volontaristes en faveur du logiciel libre.” [39]

C’est également l’aventure dans laquelle s’est lancée l’École Polytechnique de Montréal, en adoptant *Moodle*, un outil libre de gestion des apprentissages pour remplacer *WebCT*, une application propriétaire. *Moodle* doit pouvoir répondre aux besoins en gestion de projets pour les quelques 44 cours intégrateurs donnés à Polytechnique et certainement plusieurs autres cours. Cependant un besoin n’est pas satisfait dans la version de base, celui de la base de données partagée. Le même besoin s’est fait sentir à l’Université de Franche-Comté en France. Ceci a créé l’opportunité de mettre en place un projet international au service de la présente recherche afin de tester les hypothèses qui viennent d’être formulées et fera l’objet de la partie III.

Troisième partie

Test des hypothèses sur un cas réel

Dans cette partie, il s'agit de tester les hypothèses formulées précédemment. La méthode adoptée se déroule en deux étapes, chacune correspondant à un chapitre.

La première étape consiste à mettre en place un projet international en utilisant un besoin réel de l'École Polytechnique de Montréal et de l'Université française de Franche-Comté. Il s'agit d'un projet de redéveloppement d'un module de partage de fichiers pour la plate-forme d'apprentissage *Moodle*. Bien entendu, la gestion de ce projet, nécessairement distribuée, doit être réalisée à l'aide de logiciels libres afin d'expérimenter les hypothèses.

La deuxième étape est celle de l'observation, car elle est primordiale si l'on souhaite obtenir des résultats exploitables.

Chapitre 7

Projet international de réingénierie

L'École Polytechnique de Montréal a choisit de changer de plate-forme d'apprentissage en ligne. Elle utilisait *WebCT*, une application propriétaire, et a souhaité adopter *Moodle*, un logiciel sous licence *open source*. Cette migration a fait naître un nouveau besoin et a créé l'opportunité de mettre en place un projet international d'amélioration d'un module pour *Moodle*. Ce chapitre décrit comment cette opportunité a été mise au service de la présente recherche et propose une description détaillée du projet.

7.1 L'implantation de *Moodle* à Polytechnique créé un besoin

À l'École Polytechnique de Montréal, 44 cours intégrateurs sont actuellement dispensés. On les appellent aussi projets intégrateurs, car il y a, durant la session, un projet à réaliser en équipe. Ce projet représente un outil pédagogique pour faire vivre aux étudiants une expérience de gestion de projet. En outre, l'enseignant doit pouvoir évaluer l'avancement du travail de chaque équipe. Une base de données partagée, à travers laquelle les étudiants peuvent échanger des documents, répond souvent à ce besoin de suivi et donne l'oc-

casion aux étudiants d'utiliser un outil avancé pour la gestion de projet.

Or, cette fonctionnalité n'était pas incluse directement dans la nouvelle plate-forme *Moodle*. Cependant, comme toute solution libre, elle est construite de manière modulaire, si bien qu'il est possible d'ajouter un ou plusieurs modules en fonction des nouveaux besoins, sans être obligé de modifier le code source du coeur de la plate-forme. Une recherche préliminaire dans les différentes modules inclus dans *Moodle* a montré qu'il n'y avait aucune possibilité de répondre de manière satisfaisante au besoin identifié (wiki, base de données). Un module externe, presque adéquat, a été identifié, mais il a fallu l'adapter à la version actuelle de *Moodle* et y ajouter plusieurs fonctionnalités.

Suite à une présentation de la plate-forme *Moodle*, le module fera l'objet d'une étude de reverse-engineering pour identifier ce qu'il fait exactement et ce qu'il faut modifier ou y ajouter.

7.1.1 *Moodle*

On dit de *Moodle* que c'est un LMS (*Learning Management System*) ou en français une plate-forme d'apprentissage en ligne. Il s'agit d'une plate-forme Web, c'est-à-dire qu'elle tourne sur un serveur et on y accède à l'aide d'un fureteur. Cette plate-forme "offre au corps enseignant les outils pour créer un site de cours et permet un contrôle des accès si bien que seuls les étudiants inscrits au cours peuvent le voir" [13]. Il met également à disposition plusieurs autres outils pour déposer et partager le matériel de cours, tenir des conversations électroniques, proposer des tests sous forme de quiz par exemple, et tenir les carnets de notes.

Partage de contenu Au lieu d'utiliser un éditeur HTML pour créer une page Web, puis de l'envoyer sur un serveur à l'aide d'un client FTP, *Moodle* propose de remplir un formulaire Web, directement accessible sur la plateforme et permet ainsi de partager du contenu de manière très simple. Aucune connaissance en langage HTML n'est requise pour créer sa propre page Web sur le site d'un cours.

De plus, il est possible de déposer des fichiers et de mettre à la disposition des étudiants les documents nécessaires ou complémentaires à la compréhension du contenu du cours. Les étudiants peuvent dès lors y accéder quand ils le souhaitent.

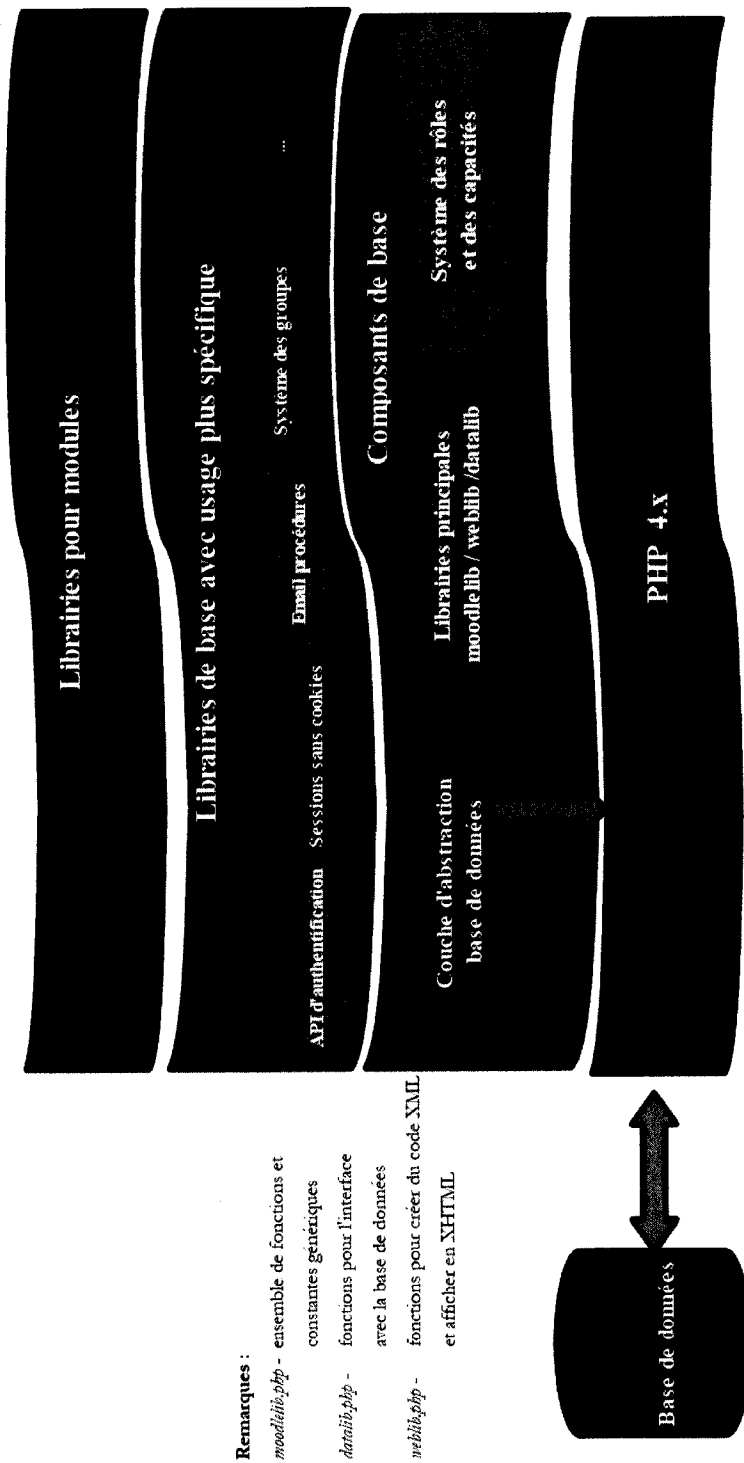
Cependant, ils ne peuvent pas, comme il a été expliqué en introduction, partager des documents entre eux. Cette fonctionnalité est nécessaire pour faire vivre aux étudiants l'expérience de gestion de projet telle que définie précédemment.

Construction modulaire Fort heureusement, comme la plupart des logiciels libres, *Moodle* est construit de manière modulaire si bien qu'il est possible d'y ajouter des fonctionnalités sans être obligé de modifier le code source du coeur de la plateforme. Il existe un grand nombre de modules et autres *plugins* pour *Moodle*, répondant à différents types de besoins. Une recherche approfondie dans la base des modules a permis d'identifier le module *File Manager* (gestionnaire de fichiers) qui est présenté dans la sous-section 7.1.2.

Pour bien comprendre le travail à effectuer pendant ce projet, la figure 7.1 représente la structure des bibliothèques standards de *Moodle*. On appelle ces bibliothèques des APIs, pour *Application Programming Interface*. Elles sont un ensemble de classes et fonctions utiles pour le développement de nouveaux composants. Elles créent des couches d'abstractions pour faciliter le travail

du programmeur souhaitant ajouter de nouvelles fonctionnalités.

Par exemple, il est inutile d'écrire du code pour aller sélectionner des enregistrements dans la base de données car il existe une simple fonction prévue à cet effet. Peu importe le gestionnaire de base de données utilisé, le programmeur n'a pas besoin de le connaître, puisqu'il lui suffit d'écrire la requête SQL (*Structured Query Language*), de la donner à une fonction de la librairie qui s'occupe d'interroger la base de données avec les paramètres adéquats et de récupérer les résultats. On comprend dès lors le terme de "couche d'abstraction".

Figure 7.1 – Structure de *Moodle*

7.1.2 Module gestionnaire de fichiers

Le module gestionnaire de fichiers ou *File Manager* a été initialement développé par Michael Avelar. Il permet l'échange de fichiers de manière très souple. En effet, chaque utilisateur inscrit dans le cours possède un espace personnel et peut y créer une arborescence de répertoires, fichiers, URLs (*Uniform Resource Locator*) et catégories, ainsi que de partager à souhait toutes ces ressources à n'importe quels utilisateurs inscrits au cours. Il est même possible de créer des archives compressées d'un ou plusieurs répertoires ou fichiers.

Définitions des ressources Voici pour chaque type de ressource, une courte définition afin de mieux cerner l'utilité du module gestionnaire de fichiers :

Fichiers on peut déposer des fichiers de tout type dans l'espace personnel.

URL une URL (*Uniform Resource Locator*) est un lien hypertexte donnant accès à une page internet dans le langage commun. C'est cette définition qu'il faut comprendre pour le module.

Répertoires un répertoire est un espace dans lequel peuvent être rangés plusieurs fichiers, URL ou autre répertoires.

Catégories les catégories sont des vues de l'espace de fichiers qui peuvent différer de celle créée par l'arborescence. Une catégorie n'est ni un fichier, ni un répertoire. Un fichier ou un répertoire peut appartenir à une catégorie. Ainsi, peu importe la place d'un fichier ou d'un répertoire dans l'arborescence, on peut le faire appartenir à une catégorie. Tout l'intérêt d'une catégorie réside dans le fait que l'on peut ensuite la partager : dès lors, tous les fichiers et répertoires appartenant à cette

catégorie seront partagés au même moment, peu importe leurs emplacements respectifs.

Archives ZIP une archive est un format compressé de fichier contenant un assemblage de plusieurs fichiers, et/ou répertoires et/ou d'autres archives ZIP. (*Zone Information Protocol*)

Cas d'utilisations L'analyse des cas d'utilisations du module gestionnaire de fichiers permet de dire qu'il existe trois acteurs principaux : l'étudiant, le professeur et l'administrateur. La figure 7.2 illustre pour ces trois types d'acteurs leurs cas d'utilisations. Pour comprendre ce schéma, voici quelques informations complémentaires :

- l'acteur "professeur" hérite des fonctionnalités de l'"étudiant" et l'acteur "administrateur" hérite des fonctionnalités du "professeur"
- gérer un objet signifie : ajouter, modifier ou supprimer un objet de ce type
- administrer le module signifie : configurer les options du module qui sont :
 - la taille des fichiers que l'on peut télécharger sur le serveur,
 - la taille maximale d'un répertoire,
 - l'activation du partage de fichiers ou de catégories à d'autres membres,
 - l'activation du partage à tout le monde depuis la page principale,
 - et l'activation du module *File Manager*.

7.2 Objectifs du projet

Le projet lancé par un professeur en TIC (Techniques d'Informations et de Communications), en Informatique et en Sociologie numérique à l'EISTI (École Internationale des Sciences du Traitement de l'Information) entend offrir une version finalisée du module *File Manager* pour *Moodle* version 1.8 et plus.

Ce professeur a créé, suite à la demande de l'Université de Franche-Comté, un groupe de travail, composé d'étudiants de première année de l'EISTI, rejoint par moi-même peu de temps après le début du projet puisque Polytechnique souhaitait également obtenir une version fonctionnelle du module pour *Moodle* 1.9. L'équipe du projet est présentée dans la section 7.3.

Les objectifs poursuivis ont été de deux ordres :

- le nettoyage du code source et son débogage afin de réactiver le partage de fichiers entre utilisateurs,
- et l'ajout d'une fonctionnalité non encore implémentée : l'espace de stockage par groupe.

7.2.1 Réactivation du partage de fichiers entre utilisateurs

Afin de réactiver le partage de fichiers entre utilisateurs, il s'agit de procéder à ce que l'on appelle du *reverse engineering* afin de comprendre l'architecture du module et de son code source. En effet, il est lui-même construit à l'aide de bibliothèques de fonctions, chacune dédiée à des tâches spécifiques. De plus, il a été identifié que la structure de la base de données a été modifiée depuis *Moodle* 1.6, ce qui explique le dysfonctionnement des requêtes SQL

lancées par le module. Dès lors, il faut procéder également à du *bug tracking*, c'est-à-dire que l'on cherche à identifier et corriger tous les défauts présents dans la version de base du module.

7.2.2 Création d'espace de stockage par groupe

Le fonctionnement du module *File Manager* a été repensé afin qu'il puisse fonctionner également avec les groupes. Effectivement, il a été convenu dans un premier temps qu'il fallait prévoir un lien "GroupFiles" à la manière de "MyFiles" pour afficher dans cette vue les fichiers du groupe auquel appartient l'utilisateur. Ainsi, la distinction est claire entre les fichiers personnels et les fichiers du groupe. Tous les membres du groupe accéderaient donc désormais aux mêmes fichiers. Cela représenterait la forme de partage utilisée pour les groupes. Cependant, un utilisateur peut appartenir à plusieurs groupes à partir de *Moodle* version 1.9. Il faut donc prévoir un lien "NomDuGroupe" pour chaque groupe auquel appartient l'utilisateur.

Ensuite, il faut prendre en compte le mode dans lequel le cours se trouve, en ce qui concerne les groupes :

- mode pas de groupe : aucun accès supplémentaire que le répertoire personnel,
- mode groupes séparés : accès à tous les groupes auxquels appartient l'utilisateur,
- mode groupes visibles : accès à tous les groupes du cours mais uniquement en lecture pour les groupes dont l'utilisateur ne fait pas parti.

De plus, les professeurs responsables du cours doivent pouvoir accéder en lecture et en écriture aux répertoires de tous les groupes.

7.3 Équipe de projet

L'équipe de projet est constituée du professeur de l'EISTI, initiateur du projet, deux de ses étudiants en informatique à l'EISTI (École Internationale des Sciences du Traitement de l'Information) en France, d'un étudiant de l'École Polytechnique de Montréal, moi-même, et d'une personne ressource du service de soutien pédagogique de l'École Polytechnique de Montréal.

Le tableau 7.1 présente le rôle que chacun des membres a joué dans ce projet.

Tableau 7.1 – Équipe de projet

Nom	Pays	Rôle
DAUDIGNON Michaël	Montréal	Gestionnaire du projet et programmeur
Personne ressource	Montréal	Programmeur
Professeur EISTI	France	Initiateur du projet et programmeur
Étudiant 1 EISTI	France	Programmeur
Étudiant 2 EISTI	France	Programmeur

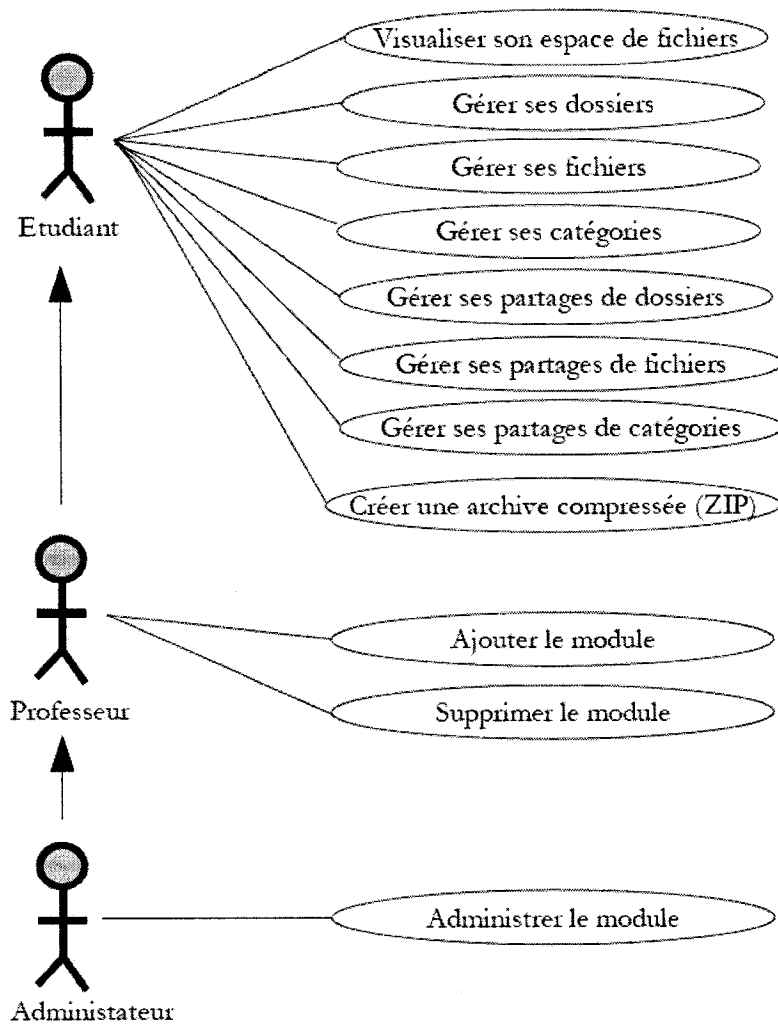


Figure 7.2 – Cas d'utilisation du module gestionnaire de fichiers

Chapitre 8

Observation du projet

Le présent chapitre se consacre à l'observation du projet international présenté dans le chapitre précédent. La première section décrit la méthode d'observation tandis que les sections suivantes explicitent les logiciels libres utilisés tant pour la gestion que pour la réalisation du projet, pour terminer sur l'exposition des résultats de cette observation.

8.1 Méthode d'observation

La méthode d'observation choisie se compose de trois volets : la gestion du projet, son exécution et son impact. Pour observer la gestion du projet, on peut par exemple prendre en compte le facteur des logiciels libres utilisés et en évaluer ses fonctionnalités sur la base du modèle développé pour cette recherche. En ce qui concerne l'exécution du projet, les logiciels utilisés pour le développement sont intéressants à observer. Enfin, l'impact du projet peut être envisagé de manière personnelle et collective. Ainsi, on peut mettre en lumière les leçons apprises de ce projet et élaborer des statistiques d'utilisation du module, à Polytechnique ou ailleurs dans le monde.

8.2 La gestion du projet

Le projet mis en place nécessite une gestion de projet distribuée et l'on souhaite bien entendu utiliser les logiciels libres pour gérer le projet. Comme il a été dit dans la section 7.1, dans le cadre des projets intégrateurs, ce sera la plate-forme *Moodle* qui sera utilisée pour faire vivre aux étudiants leur expérience en gestion de projet. Dès lors, on souhaite pour ce projet informatique utiliser au maximum les possibilités de la plate-forme.

8.2.1 *Moodle* plate-forme de gestion de projet distribuée

Il se trouve que la plate-forme *Moodle* répond à un grand nombre de besoins primaires et avancés identifiés, grâce à l'installation judicieuse de modules complémentaires. Ces modules sont facilement accessibles via le répertoire officiel des modules et *plugins*. Dès lors, un site *Moodle* a été mis en place auquel quelques modules complémentaires ont été ajoutés. Le site qui a servi de plate-forme de gestion de projet distribuée est disponible à l'adresse suivante : <http://www.ethnoinformatique.fr/course/view.php?id=132>. La figure 8.1 est une capture d'écran du site en question.

On remarque immédiatement que, dans sa version de base, *Moodle* permet d'effectuer la définition du projet puisqu'il est possible d'éditer du contenu. Cependant, à la différence d'un cours normal, tous les membres de l'équipe doivent avoir un accès "professeur" au cours afin de pouvoir éditer le contenu. Ici, des liens vers des pages Web ont été utilisés dans l'encadré 1.

Ensuite, il existe un outil de planification efficace pour les projets de développement. Il s'agit du module *Technical project*, disponible à l'adresse sui-

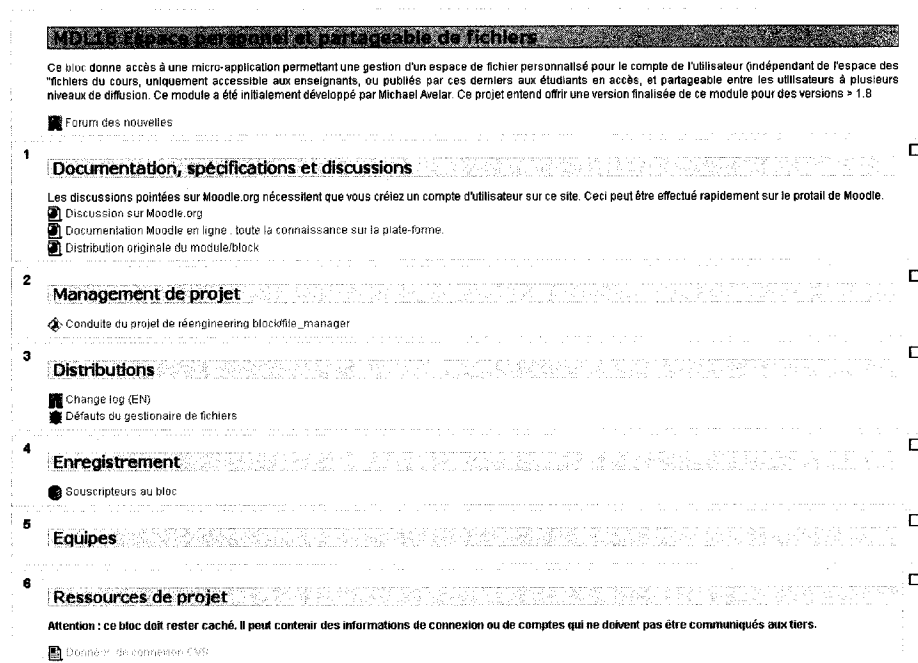


Figure 8.1 – Capture d'écran du site plate-forme de gestion de projet distribuée

vante : <http://moodle.org/mod/data/view.php?d=13&rid=799&filter=1>.

Il propose tout d'abord de décrire le projet de manière textuelle, puis il permet de rédiger les spécifications, que l'on appelle aussi les exigences. L'ordonnancement des tâches est rendu possible dans l'onglet "tâches". Il faut savoir que l'on peut hiérarchiser les tâches et afficher leur état d'avancement. La figure 8.2 est un exemple du type de planification possible à l'aide du module *Technical project*.

Dès lors, si l'on dresse la carte de réponses aux besoins du gestionnaire de projet, on peut dire que *Moodle* permet d'effectuer la définition du projet, sa planification et permet également un certain suivi lorsque l'on installe le module complémentaire *Technical project*.

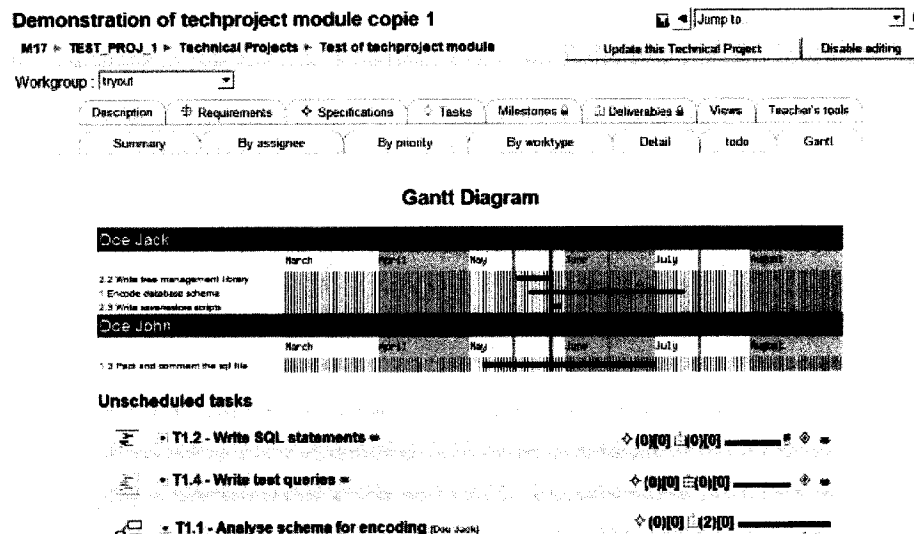


Figure 8.2 – Capture d’écran du module *Technical project*

8.2.2 Suite *Open Office* 2.3 pour la génération de documents

En ce qui concerne la génération de documents, de rapports d’avancements ou autres documents décrivant les jeux de tests par exemple, c’est la Suite *Open Office* 2.3 qui a été utilisée dans le cadre de ce projet. Comme il a été vu dans la section 4.1, cette suite de logiciel répond à un grand nombre de besoins à la base de la communication entre membres d’une équipe de projet.

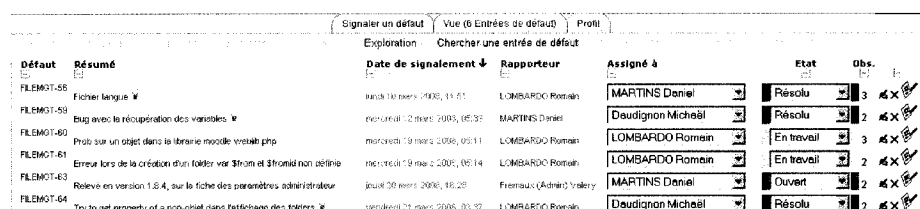
8.3 L’exécution du projet

Il est également intéressant d’observer l’exécution du projet qui s’est déroulée à l’aide de logiciels libres et *open source*. D’après les objectifs du pro-

jet, on comprend qu'il a fallu d'une part, se doter d'un outil de *bug tracking*, puisque l'on souhaitait améliorer une version déjà existante d'un module, et d'autre part, d'un environnement de développement Web pour réaliser les corrections et l'ajout des nouvelles fonctionnalités.

8.3.1 Bug tracking

Il existe un module dans *Moodle* qui permet de faire du *bug tracking*, c'est-à-dire qu'il aide à lister les bogues (ou défauts), et permet d'effectuer le suivi de leur correction. La figure 8.3 est une capture d'écran du module de traqueur de défauts. On peut y remarquer que chaque défaut est reconnu de manière unique par une référence, puis est décrit par du texte. Enfin, on peut voir à qui est affecté la correction du défaut et son état d'avancement.



Défaut	Résumé	Date de signalement	Rapporteur	Assigné à	Etat	Obs.
FLEMO7-55	Fichier langue	jeudi 14 mars 2008, 14:51	LOMBARDO Romain	MARTINS Daniel	Résolu	3
FLEMO7-59	bug avec la récupération des variables	jeudi 14 mars 2008, 05:32	MARTINS Daniel	Daudignon Michaël	Résolu	2
FLEMO7-60	Probl sur un objet dans la librairie moodle version php	mercredi 19 mars 2008, 05:11	LOMBARDO Romain	LOMBARDO Romain	En travail	3
FLEMO7-61	Erreur lors de la création d'un folder via \$from et \$fromid non définie	mercredi 19 mars 2008, 05:14	LOMBARDO Romain	LOMBARDO Romain	En travail	2
FLEMO7-63	Relève en version 1.5.4, sur la fiche des paramètres administrateur	jeudi 20 mars 2008, 16:20	Fremault (Adrien) Valéry	MARTINS Daniel	Ouvvert	2
FLEMO7-64	Try to get property of a non objet dans l'affichage des totaux	vendredi 21 mars 2008, 03:37	LOMBARDO Romain	Daudignon Michaël	Résolu	2

Figure 8.3 – Capture d'écran du module de traqueur de défauts

8.3.2 Environnement de développement Web

Une fois les défauts identifiés, il faut les corriger, puis développer les nouvelles fonctionnalités en groupes dont il a été question lors de la définition des objectifs du projet dans la section 7.2.

L'environnement de développement Web sélectionné pour ce projet a été *ApacheFriends XAMPPlite* version 1.5.5, que l'on appelle communément

Xampp. Il s'agit en fait d'un ensemble de logiciels libres qui permettent de mettre en place un serveur Web, le serveur *Apache*, un serveur de base de données, *MySQL*, ainsi qu'un environnement *PHP* qui permet d'interpréter les scripts. Ces derniers représentent l'application *Moodle* et le module gestionnaire de fichiers.

Cet environnement de développement a été agrémenté d'un éditeur de textes puissant, sous licence GPL, *Notepad++*, version 4.5.

Enfin, pour faciliter les tâches de synchronisation entre les développeurs, un serveur CVS (*Concurrent Versions System*) a été utilisé. Cet outil est un système de gestion des versions. En d'autres termes, il résout les conflits entre différentes versions d'un même fichier, ce qui survient fréquemment dans un environnement de développement distribué.

8.4 Impact du projet

Tout d'abord, on peut dire que le projet est un réel succès, puisque le module *File Manager* a été corrigé, puis amélioré avec les fonctionnalités décrites dans la section 7.1. Il a également réussi une série de tests de conformité qui visaient à vérifier ses fonctionnalités, sa stabilité, sa sécurité et sa robustesse. Cette section s'attache à décrire les leçons apprises du projet, les perspectives souhaitées pour le module et les statistiques que l'on peut relever quant à l'utilisation de ce dernier à Polytechnique et ailleurs.

8.4.1 Leçons apprises et perspectives

Ce projet international permet de mettre en lumière quelques leçons apprises, qui sont en fait des conséquences de l'environnement de travail que représente la communauté du libre.

Documentation incomplète et parfois inexistante Tout d'abord, le code source est certes facilement disponible, mais sa documentation laisse parfois à désirer, si bien qu'il est assez difficile de comprendre ce que le programmeur précédent a voulu faire. Dès lors, il est bien important de prendre conscience qu'il est nécessaire d'avoir du temps pour devenir opérationnel lorsque l'on reprend le code de quelqu'un d'autre.

Revue par les pairs et reconnaissance Par ailleurs, il a été prouvé à travers ce projet que la revue par les pairs s'avère très efficace lorsqu'il s'agit d'identifier un défaut et de le corriger. De plus, la reconnaissance du travail effectué se déroule de manière différente que dans un processus de développement classique. En effet, les développeurs retirent une certaine fierté de voir ce qu'ils ont fait, repris, testé et amélioré par d'autres. C'est le cas du module *File Manager*, puisqu'il a été question sur les forums de discussions officiels de *Moodle* de faire évoluer le module pour son intégration dans la version de base à partir de *Moodle 2.0*.

8.4.2 Statistiques d'utilisation du module à Polytechnique

Le module tel que développé, n'inscrit aucune entrée d'utilisation dans les journaux de *Moodle* (aussi appelés fichiers logs). Cependant, il est possible de connaître le nombre d'occurrences du module sur le serveur de Polytechnique. Ainsi, en admettant que lorsqu'un professeur met à disposition le module gestionnaire de fichiers dans son cours, c'est qu'il en a l'utilité et que par conséquent, ses étudiants s'en servent régulièrement.

Un relevé a été effectué le 03 octobre 2008, un mois après la mise en production du module sur le serveur de l'école. Le tableau 8.1 présente les résultats du relevé.

Tableau 8.1 – Statistiques d'utilisation du module à Polytechnique Montréal

Type de mesure	Quantité
Cours actifs sur <i>Moodle</i> (incluant les sites de formation)	268
Cours utilisant le gestionnaire de fichiers	29
personnes inscrites dans le rôle "étudiant" dans les sites qui utilisent le gestionnaire	1732

Il est clair qu'il s'agit d'un taux d'utilisation remarquable puisqu'aucune publicité particulière n'a été effectuée.

8.4.3 Module utilisé dans le cadre du projet *Teach Advanced Online*

Le module *File Manager* est utilisé dans le cadre d'un projet réalisé entre Intel et l'Éducation Nationale française. Le professeur qui est à l'origine du projet de redéveloppement a gentiment accepté de m'expliquer à travers un courriel de quoi il retourne. En voici quelques extraits choisis pour comprendre l'essentiel :

“Le gouvernement français (Ministère de l'Éducation Nationale) et Intel ont signé en 2006 un accord de partenariat par lequel l'Éducation Nationale appuie institutionnellement le déploiement d'une offre pédagogique conçue et développée par Intel, dans le cadre de ses activités citoyennes (*non-profit operations*). Intel intervient à ce titre dans deux programmes majeurs, liés à leur secteur d'activité :

- Un programme de recherche/développement sur la réduction de l'impact thermique des processeurs, (recherche ayant conduit notamment au passage au multicœur, au détriment d'une solution d'augmentation de fréquence beaucoup plus impactante sur le réchauffement climatique.
- Un ensemble de programmes éducatifs et pédagogiques destinés à diffuser largement la culture technologique générale permettant l'accès à la société de l'information et favorisant l'utilisation des TICE.

Nous intervenons dans le deuxième programme, qui consiste à déployer une offre de méta-formation (formation de formateurs) destinée à accroître la compétence naturelle des enseignants dans la production d'artefacts pédagogiques utilisant le système d'information généralisé. Il s'agit donc de fournir les pratiques et techniques de base aux enseignants pour qu'ils soient capables d'utiliser naturellement les ENT (Espaces Numériques de Travail) et plates-formes d'*e-Learning*, mais également produire les contenus, scénar-

rios et ressources qui permettent de constituer ces volumes et de baser ces pratiques.

Le programme d'Intel (*Teach Advanced Online* - TAO) est réalisé à échelle Euro-méditerranéenne (Irlande, Allemagne, Italie, France, UK, Jordanie, Israël). Le programme s'appuie sur l'hypothèse que la future production de valeur continuera à basculer de l'économie physique (dont la croissance ne dépend que linéairement de la croissance en nombre du marché d'équipement du à l'arrivée des nouveaux pays — les tailles du marché des pays industrialisés étant stables depuis plus de 20 ans) à une économie de la connaissance et de la virtualisation des services (on voit aujourd'hui le problème que des grains de sables dans la machinerie de confiance sur laquelle repose toute cette économie peut poser). Malgré les soubresauts et toussotements de cette mutation (il faut regarder à l'échelle de -15 à + 15 ans, dans une prospective très rapide et mobile), cette hypothèse reste viable, même aujourd'hui. De ce fait, l'investissement sur les compétences permettant de massivement donner l'accès aux pratiques et compétences de la société de l'information sont un secteur clef d'intervention. Plutôt que de viser une action directe sur les futurs utilisateurs, ce que le marketing actuel des firmes de nouvelles technologies s'empresse de faire, Intel préfère viser l'amont de la chaîne, réduisant l'effet de fracture qui augmente entre les générations d'élèves à venir et la génération d'enseignants actuelle. Même si le développement des TICE est dans toutes les bouches institutionnelles et opérationnelles dans l'éducation nationale, l'existence d'une fracture numérique à ce niveau ne peut être ignorée. Cette fracture pose un problème majeur de maintien de l'appétence à apprendre, lorsque les méthodes pédagogiques de la "classe enseignante" ne sont plus cohérentes avec l'état des pratiques généralisées dans la société de la communication actuelle.

Dans ce cadre, l'utilisation du module file manager est prévue pour offrir

un *container* personnel de ressources propres des stagiaires, leur permettant de partager et de se communiquer des ressources de base engagées dans un projet de construction d'un artefact. Nous prévoyons également de l'utiliser comme endroit de collecte des liens distribués à travers toute la plate-forme (un peu comme un gestionnaire personnel partageable de signets)".

Quatrième partie

Résultats, recommandations, et perspectives

Cette dernière partie présente les résultats de ce travail et cherche à formuler des recommandations quant à l'utilisation de ces derniers. Enfin, on souhaite proposer quelques suggestions de recherches complémentaires pour l'avenir.

Chapitre 9

Résultats

Comme le titre de la présente recherche l'indique, il s'agit d'étudier les forces et faiblesses des logiciels libres en gestion de projet distribuée. Dès lors, les résultats sont présentés en deux sections ; la première s'intéressant aux forces, et la secondes aux faiblesses.

9.1 Les forces des logiciels libres en gestion de projets

Au-delà de la diminution des coûts que représente l'utilisation de logiciels libres, (voir section 2.2), ces derniers possèdent d'autres forces qu'il s'agit de qualifier dans cette section.

9.1.1 Une offre mature et durable

Tout d'abord, on peut dire que les logiciels libres et *open source* forment une offre mature, tant en termes de fonctionnalités, que de sécurité. De plus, on peut penser qu'il est bon d'investir dans la communauté du libre puisqu'elle semble durable. Rappelons par exemple, que le libre échange des codes sources a débuté avec l'informatique. Il existe de nombreux acteurs, reconnus

à l'international, qui font la promotion de cette communauté et y parviennent avec brio : la *Free Software Foundation* (FSF), l'*Open Source Initiative* (OSI) ou encore la *Mozilla Foundation* pour n'en nommer que trois d'entre eux. Enfin, il est important de noter que les modèles d'affaires développés par la communauté, sont d'une part viables économiquement, et que d'autre part, ils intéressent de plus en plus des acteurs oeuvrant dans le modèle traditionnel privé.

9.1.2 Une réponse adéquate aux besoins en gestion de projet distribuée

Si l'on regarde à travers la pyramide des besoins du gestionnaire de projet, modèle développé pour cette recherche dans la section 1.3.1, on conclut que les logiciels libres offrent une réponse adéquate. En effet, la combinaison de la suite *Open Office* avec l'outil ERP *Project-Open* par exemple, permet de couvrir l'ensemble des besoins en gestion de projet formulés par la littérature. Toutefois, la mise en place d'un ERP peut s'avérer complexe. Dès lors, si l'on ne possède ni les moyens, ni les compétences, il faut savoir que d'autres solutions existent, beaucoup plus simples à mettre en oeuvre. Pour finir, il est important de comprendre qu'en règle générale, plus la solution est simple, moins elle couvre de besoins.

9.1.3 Flexibilité, contrôle et évolution rapide

Grâce à la modularité, imposée par le processus de développement distribué de la plupart des logiciels libres, leur flexibilité est réelle. Le projet international mis en place durant cette recherche a permis de le démontrer. Si la fonctionnalité n'est pas incluse directement dans la version originale

d'un produit, il est très souvent possible d'y ajouter un ou plusieurs modules complémentaires si bien que les possibilités sont très nombreuses.

Par ailleurs, comme l'indique la littérature qui s'intéresse aux logiciels libres, les utilisateurs possède un certain contrôle sur l'évolution des produits. Dès lors, on comprend qu'il s'agit là d'un énorme avantage sur les solutions privées, puisque la simple formulation d'un souhait sur un forum, peut rapidement aboutir à l'intégration effective de la fonctionnalité dans la publication suivante du logiciel. Et quand bien même cette fonctionnalité tardait à venir, il est toujours possible d'apporter les modifications soit-même. L'évolution des solutions libres est donc très rapide au vu des contributions volontaires du monde entier.

9.2 Les faiblesses des logiciels libres en gestion de projets

La section précédente fait état de l'offre mature et adéquate aux besoins du gestionnaire de projet. Malgré tout, certaines faiblesses demeurent.

9.2.1 Trouver ce qui nous convient

Tout d'abord, on peut affirmer qu'il n'est pas toujours facile de trouver le logiciel qui convienne à nos besoins. Il existe certes des répertoires de logiciels libres et de nombreuses publications qui évaluent ces produits, mais trop souvent en termes de maturité et oublie d'évaluer la réponse aux besoins de tel ou tel secteur. C'est la raison pour laquelle, cette présente recherche s'est attardée sur la réponse aux besoins du gestionnaire de projet.

9.2.2 Temps d'apprentissage et documentation

Malgré tous les efforts des fondations, des organismes militant pour la communauté du libre et autres entreprises gravitant autour du logiciel libre, la documentation est souvent incomplète et parfois inexistante. Et quand bien même tout est bien documenté, des compétences spécifiques en informatique et un certain temps d'apprentissage sont nécessaires pour être capable de valider l'hypothèse formulée au chapitre 6 concernant l'amélioration continue des outils.

Cette faiblesse a été prise en compte depuis plusieurs années par toute la communauté, et elle cherche activement à résorber le fossé qui sépare les membres de celle-ci du grand public. Cette recherche a permis de mettre en lumière qu'il reste encore du chemin à faire.

Chapitre 10

Recommandations et perspectives

Le dernier chapitre se consacre à la formulation de quelques recommandations quant à l'utilisation de la plate-forme d'apprentissage en ligne *Moodle*, ainsi que d'envisager certaines recherches complémentaires intéressantes, suite à ce présent travail.

10.1 Recommandations pour la plate-forme *Moodle*

La plate-forme d'apprentissage en ligne *Moodle* comporte tous les avantages d'une solution libre. Dès lors, il apparaît intéressant de consacrer des ressources ayant de bonnes compétences en informatique et en programmation Web, à son amélioration. Les nouvelles contributions étant nombreuses, il ne faut pas relâcher les efforts de veille concernant les prochaines publications et les modules complémentaires proposés. En effet, ces derniers peuvent apporter la réponse adéquate ou presque aux éventuels nouveaux besoins formulés par le corps enseignant, les employés ou les étudiants.

En ce qui concerne le module *File Manager*, il serait intéressant le faire évoluer et d'aider à son acceptation par la communauté *Moodle*, si bien qu'il serait possible de profiter de conseils avisés et de support. En effet, si le

module venait à être intégré dans une version ultérieure de la plate-forme, l'École Polytechnique n'aurait plus besoin de mettre autant d'efforts qu'elle ne l'a fait par le passé, pour maintenir à jour le module d'une part, ni même pour maintenir sa rétro-compatibilité d'autre part. La rétro-compatibilité suggère de faire en sorte que le module fonctionne également sur des versions antérieures de *Moodle*.

10.2 Perspectives

Après cette étude des forces et faiblesses des logiciels libres en gestion de projet distribuée, il est possible d'imaginer ce que pourraient devenir ces produits dans un avenir plus ou moins proche.

On a vu que la réponse globale aux besoins du gestionnaire de projet est proche avec *Project-Open* par exemple et qu'il est possible de la compléter avec des outils de traitement de textes, tableur et présentation, rivalisant avec les solutions propriétaires. Tous ces outils libres utilisent les standards ouverts définis par les recommandations de l'ODF (*Open Document Format*). Dès lors, on peut penser que l'avenir prépare une plus grande intégration des fonctionnalités. Parce que toutes les applications sont construites de manière modulaires, il est assez naturel d'entrevoir le développement d'interfaces capables de faire interagir les outils ERP avec les suites bureautiques. Il serait donc intéressant de réfléchir à ces interfaces et de regrouper un ensemble de logiciels *open source* en une solution globale, complète et ouverte sur l'avenir.

Cet avenir devra très certainement prendre en considération les aspects environnementaux dans le cycle de vie de tout projet et de tout produit. On peut donc penser que les gestionnaires de projet iront s'enquérir de telles fonctionnalités puisqu'un nouveau besoin émergera très prochainement. Mais

avant de pouvoir mettre en place de tels outils, il faut d'abord avoir une meilleure compréhension afin qu'il soit possible de prendre en compte les aspects environnementaux depuis le début du cycle de vie d'un projet ou d'un produit, jusqu'à son démantèlement.

Conclusion

Pour conclure, on peut affirmer qu'il existe dans l'offre des solutions libres toutes les réponses aux besoins du gestionnaire de projet. On peut également leur faire confiance, car on leur fait déjà confiance, pour l'Internet notamment. On utilise en réalité beaucoup de logiciels libres sans le savoir. De plus, la sécurité est souvent meilleure, la conception est mieux faite que pour leurs homologues privés. De surcroît, leur code source peut être vérifié, inspecté, modifié. Ces logiciels peuvent être moins chers, mais pas toujours, car ce que l'on ne paie pas en licence d'utilisation, on le dépense dans la formation et la maintenance du logiciel. Cependant, on gagne surtout en flexibilité et en contrôle sur l'avenir du produit. Ainsi, le logiciel libre deviendra ce que les utilisateurs voudront en faire.

Une réelle prise de conscience des citoyens de leur responsabilité et de leur pouvoir sur le futur s'opère graduellement grâce à la visibilité qui est donnée à la communauté du libre. Elle pourrait être le sujet d'une étude sociologique remarquable et très intéressante sur le plan humain.

Bibliographie

- [1] *Guide du corpus des connaissances en management de projet : guide PMBOK*. Project Management Institute, Newtown Square, Pa., 3e édition, 2004.
- [2] *CMMI for Development, Version 1.2*. Université Carnegie Mellon Software Engineering Institute, Pittsburgh, 2006. <http://www.sei.cmu.edu/publications/documents/06.reports/06tr008.html>.
- [3] Bien choisir son erp open source. *Journal du net*, 2008. <http://www.journaldunet.com/solutions/intranetextranet/dossier/08/0528-livre-blanc-erp-open-source/1.shtml>.
- [4] Les solutions de gestion de portefeuille de projets. *Journal du net*, 2008. <http://www.journaldunet.com/solutions/0605/060524-panorama-gestion-portefeuille-projets-1.shtml>.
- [5] ALHAZMI, O.H., MALAIYA, Y.K. et RAY, I. : Measuring, analyzing and predicting security vulnerabilities in software systems. *Computers and Security*, 26(3), 2007. <http://dx.doi.org/10.1016/j.cose.2006.10.002>.
- [6] BACHMANN, M. et BELL, M. : Open-source in der verwaltung. <http://edoc.hu-berlin.de/cmsj/28/bachmann-michail-37/PDF/bachmann.pdf>.

- [7] BARDOUL, L. : Les solutions de gestion de projets open source. *Journal du net*, 2004. [http : // www . journaldunet . com /solutions / 0409/ 040917 _panorama _gestion projets OS..shtml](http://www.journaldunet.com/solutions/0409/040917_panorama_gestion_projets_OS.shtml).
- [8] BERTRAND, P. : Introduction à l'open source. 2008. [http : // www . smile . fr / publications / livres-blancs /open-source-etlogiciel-libre](http://www.smile.fr/publications/livres-blancs/open-source-etlogiciel-libre).
- [9] BONACCORSI, A. et ROSSI, C. : Altruistic individuals, selfish firms ?, the structure of motivation in open source software. *First Monday*, 9(1), 2004. [http :// www. Firstmonday .org/issues/issue9_1/bonaccorsi/index.html](http://www.Firstmonday.org/issues/issue9_1/bonaccorsi/index.html).
- [10] BRODKIN, J. : Open source impossible to avoid, gartner says. *Computerworld*, 2007. [http ://www. Computerworld .com.au /index.php/id ;1025846193](http://www.Computerworld.com.au/index.php/id;1025846193). [Consulté le 2 octobre 2008].
- [11] BURGELMAN, R. A., WHEELWRIGHT, S. C. et CHRISTENSEN, C. M. : *Strategic management of technology and innovation*. McGraw-Hill Irwin, Boston ; Montreal, 4e édition, 2004.
- [12] CLELAND, D. I. et IRELAND, L. R. : *Project management : strategic design and implementation*. McGraw-Hill, New York; Toronto, 4e édition, 2002.
- [13] COLE, J. R. et FOSTER, H. : *Using Moodle : teaching with the popular open source course management system*. O'Reilly, Beijing, 2^e édition, 2008.
- [14] DEEK, F. P. et MC HUGH, J. A. : *Open source : technology and policy*. Cambridge University Press, Cambridge ; New York, 2008. [http ://www.books24x7.com/marc.asp ?bookid=13906](http://www.books24x7.com/marc.asp?bookid=13906).
- [15] DEPROW, J. : Technical evaluation report 25 : Open source software : Two learning management systems. 2003. [http ://www.irrodl.org/index.php/irrodl/article/view/135/215](http://www.irrodl.org/index.php/irrodl/article/view/135/215).

- [16] DI COSMO, R. et NORA, D. : *Le hold-up planétaire : la face cachée de Microsoft*. Calmann-Lévy, 1998. <http://www.dicosmo.org/HoldUp/HoldUpPlanetaire.pdf>.
- [17] DIBONA, C., OCKMAN, S. et STONE, M. : *Open sources : voices from the open source revolution*. O'Reilly, Beijing ; Sebastopol, 1999. <http://www.books24x7.com/marc.asp?isbn=1565925823>.
- [18] DIXON, R. : *Open source software law*. Artech House telecommunications library. Artech House, Boston, 2004. <http://www.books24x7.com/marc.asp?isbn=1580537197>.
- [19] DUMONT, E. : Microsoft ouvre une partie du code source de windows pour calmer bruxelles. *ZDNet*, 2006. <http://www.zdnet.fr/actualites/informatique/0,39040745,39305718,00.htm>. [Consulté le 12 juillet 2008].
- [20] ENGLISH, R. et SCHWEIK, C. : Identifying success and tragedy of floss commons : A preliminary classification of sourceforge.net projects , 2007. <http://dx.doi.org/10.1109/FLOSS.2007.9>.
- [21] FELLER J. : *Perspectives on free and open source software*. MIT Press, Cambridge, Mass., 2005. <http://www.books24x7.com/marc.asp?bookid=10751>.
- [22] FELLER, J. et FITZGERALD, B. : *Understanding Open Source Software development*. Addison-Wesley, London ; Toronto, 2002.
- [23] FILIPPONE, D. : Les outils de gestion de projet open source : cap sur le collaboratif. *Journal du net*, 2007. http://www.journaldunet.com/solutions/0610/061025-panorama_gestion_projets-open-source/1.shtml.
- [24] FOGEL, K. et BAR, M. : *Open source development with CVS*. Coriolis Group Books, Scottsdale, Ariz., 2e édition, 2001. <http://www.books24x7.com/marc.asp?isbn=1932111204>.

- [25] FORAY, D. : *L'économie de la connaissance*. Éditions La Découverte, Paris, 2000.
- [26] FORD, R. : Open vs. closed : Which source is more secure ? *Queue*, 5(1):32–38, 2007. <http://dx.doi.org/10.1145/1217256.1217267>.
- [27] GALLEGO, M. D., LUNA, P. et BUENO, S. : User Acceptance model of open source software. *Computers in Human Behavior*, 24(5):2199 – 2216, 2008. <http://dx.doi.org/10.1016/j.chb.2007.10.006>.
- [28] GRANDMONTAGNE, Y. : Microsoft project epm 2007 : mode web et gestion de portefeuille, avant le plm. *Silicon.fr*, 2007. <http://www.silicon.fr/fr/silicon/news/2007/02/08/microsoft-projectepm-2007-mode>. [Consulté le 14 avril 2008].
- [29] HARRIS, P. E. : *Planning and control using Microsoft Project and PMBOK R guide : updated for Microsoft Office Project 2007*. Eastwood Harris Pty. Ltd., Victoria, Australia, 3e édition, 2007. <http://www.books24x7.com/marc.asp?bookid=20852>.
- [30] HARRIS, P. E. : *Project planning & scheduling using Primavera P6 for all industries including versions 4 to 6 planning and progressing project schedules with and without roles and resources in an established enterprise environment*. Eastwood Harris Pty. Ltd., Victoria, Australia, 2008. <http://www.books24x7.com/marc.asp?bookid=24926>.
- [31] LEVESQUE, M. : Fundamental issues with open source software development. *First Monday*, 9(4), 2004. http://www.firstmonday.org/issues/issue9_4/levesque/index.html.
- [32] LYTRAS, M. D. et NAEVE, A. : *Open source for knowledge and learning management : strategies beyond tools*. Idea Group Pub., Hershey, 2007. <http://www.books24x7.com/marc.asp?bookid=16484>.

- [33] MARMEL, E. J. : *Microsoft Project 2007 bible*. Wiley Pub., Inc, Indianapolis, Ind., 2007. [http ://www. books24x7 .com/marc .asp ?bookid =16967](http://www.books24x7.com/marc.asp?bookid=16967).
- [34] MIUR, S. P. et LEGGOTT, M. : *Open source software*. Emerald Group Publishing, [Bradford, England], 2005. [http :// www .books 24x7.com/ marc.asp ?bookid=13083](http://www.books24x7.com/marc.asp?bookid=13083).
- [35] MSUTAR, P., PENAN, H. et ALTER, N. : *Encyclopédie de l'innovation*. Economica, Paris, 2003.
- [36] NICHOLS, D. M. et TWIDALE, M. B. : The usability of open source software. *First Monday*, 8(1), 2003. [http:// www . firstmonday. org/issues /issue8_1 /nichols / index .html](http://www.firstmonday.org/issues/issue8_1/nichols/index.html).
- [37] OEZBEK, C. et PRECHELT, L. : On understanding how to introduce an innovation to an open source project. pages 4273083 –, 2007. [http ://dx.doi.org /10.1109/ FLOSS .2007.11](http://dx.doi.org/10.1109/FLOSS.2007.11).
- [38] PAWLAK, M., BRYCE, C. et LAURIÈRE, S. : The practice of free and open source software processes, 2008. [http ://hal.inria.fr/inria-00274823/en/](http://hal.inria.fr/inria-00274823/en/) ; [http ://hal. Archivesouvertes. fr/ docs/00/27/48/23/PDF/PI-1891.pdf](http://hal.archives-ouvertes.fr/docs/00/27/48/23/PDF/PI-1891.pdf).
- [39] RASTETTER, Y. : *Le logiciel libre dans la mondialisation*. Études et logiciels informatiques. Hermès, Paris, 2006.
- [40] RAYMOND, E. S. : *The Cathedral & the Bazaar*. O'Reilly, 2e édition, 2001.
- [41] ROSENBERG, D. K. : *Open Source : the unauthorized white papers*. M&T Books, Foster City, CA, 2000. [http ://www. books24x7. com/marc.asp ?isbn=0764546600](http://www.books24x7.com/marc.asp?isbn=0764546600).

- [42] SCACCHI, W. : Free/open source software development : Recent research results and emerging opportunities. pages 459 – 468, Dubrovnik, Croatia, 2007. <http://dx.doi.org/10.1145/1295014.1295019>.
- [43] SCHWEIK, C. M. et SEMENOV, A. : The institutional design of open source programming, implications for addressing complex public policy and management problems. *First Monday*, 8(1), 2003. http://www.firstmonday.org/issues/issue8_1/schweik/index.html.
- [44] SOWE, S. K., STAMELOS, I. G. et SAMOLADAS, I. M. : *Emerging free and open source software practices*. IGI Pub., Hershey, 2008. <http://www.books24x7.com/marc.asp?bookid=20793>.
- [45] ST. AMANT, K. et STILL, B. : *Handbook of research on open source software : technological, economic, and social perspectives*. Information Science Reference, Hershey, PA, 2007. <http://www.books24x7.com/marc.asp?bookid=20783>.
- [46] STALDER, F. et HIRSH, J. : Open source intelligence. *First Monday*, 7(6), 2002. http://www.firstmonday.org/issues/issue7_6/stalder/index.html.
- [47] TRUCOTTE, É. : Liste de critères d'évaluation des logiciels libres / bouquiner activité de synthèse présentée comme exigence partielle de la maîtrise en informatique de gestion, 2003.
- [48] VALYUI, R. : *ERP open source*. 2008. <http://www.smile.fr/publications/livres-blancs/erp-open-source>.
- [49] WASSERMAN, A. I. et CAPRA, E. : Evaluating software engineering processes in commercial and community open source projects. pages 4273072 –, 2007. <http://dx.doi.org/10.1109/FLOSS.2007.6>.

- [50] WATSON, R. T., BOURDEAU, M.C., YORK, P. T., Martina GREINER, E. et WYNN JR, D. : The business of open source. *Communications of the ACM*, 51(4):41–46, 2008. <http://dx.doi.org/10.1145/1330311.1330321>.