PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Processus de quantification des attributs logiciels (PQAL) basé sur une méthode de prise de décisions multicritères
Auteur: Author:	John Slavich
Date:	2000
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Slavich, J. (2000). Processus de quantification des attributs logiciels (PQAL) basé sur une méthode de prise de décisions multicritères [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8905/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/8905/
Directeurs de recherche: Advisors:	Pierre N. Robillard
Programme: Program:	Non spécifié

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

PROCESSUS DE QUANTIFICATION DES ATTRIBUTS LOGICIELS (PQAL) BASÉ SUR UNE MÉTHODE DE PRISE DE DÉCISIONS MULTICRITÈRES

JOHN SLAVICH
DÉPARTEMENT DE GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉLECTRIQUE)
NOVEMBRE 2000

©John Slavich, 2001

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

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

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

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

0-612-60922-7

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

PROCESSUS DE QUANTIFICATION DES ATTRIBUTS LOGICIELS (PQAL) BASÉ SUR UNE MÉTHODE DE PRISE DE DÉCISIONS MULTICRITÈRES

Présenté par: Slavich John
En vue de l'obtention du diplôme: Maîtrise ès sciences appliquées
A été dûment accepté par le jury d'examen constitué de:

M. Pierre-N. Robillard, Ph.D., directeur de recherche
M. John Mullins, Ph.D., membre et président
M. Patrick D'Astous, Ph.D., membre

DÉDICACE

Je désire dédicacer ce mémoire à ma chère épouse, Chantal, pour sa confiance en mon potentiel, ainsi que son support continu tout au long de mon cheminement. J'aimerais aussi dédier cet ouvrage à ma fille Stéphany qui, par sa présence a su enrichir ma vie et me fournir la motivation nécessaire pour parfaire cette recherche.

REMERCIEMENTS

Je remercie chaleureusement tous ceux et celles qui ont participés à la réalisation de ce mémoire. Tout d'abord, mes sincères remerciements à mon directeur, Dr. Pierre N. Robillard, pour son enseignement et ses précieux commentaires tout au long de ce mémoire. Son support constant a su contribuer à toutes les démarches nécessaires à l'élaboration et la réalisation de ce projet.

Je désire aussi souligner l'apport considérable des membres du jury, Dr. John Mullins et Dr. Patrick D'Astous, pour leurs commentaires constructifs à l'égard de cet ouvrage.

Je tiens également à remercier Dr. Leblanc pour ses conseils judicieux quant à l'utilisation du modèle théorique et Dr. Dagenais pour avoir cru en mon potentiel de réussite au moment de mon admission.

À mes collègues étudiants, un grand merci pour avoir contribué à un climat propice à l'apprentissage.

Je tiens à remercier très sincèrement mes proches. Tout d'abord, à mes parents, Pina et Giovanni, de même qu'à ma sœur Anne-Marie, je désire exprimer mon appréciation pour leur intérêt, leur confiance et leur soutien inconditionnels. À mes beaux-parents, Thérèse et Martial, j'exprime ma gratitude pour leur support constant.

Enfin, je désire remercier mon épouse, Chantal pour ses encouragements, son soutien et son amour qui ont su promouvoir ma motivation tout au long de ce projet. Un grand merci également pour ses précieux commentaires qui ont contribué à faciliter la lecture de ce mémoire.

RÉSUMÉ

Définir et mesurer la qualité logicielle s'avèrent une activité complexe. Bien qu'il existe une multitude de métriques pouvant évaluer les attributs logiciels, on dénote des difficultés à s'entendre sur le choix des métriques et sur l'interprétation des résultats recueillis, en raison des lois de la théorie de la mesure et de l'inhabilité d'agréger les résultats issus d'échelles de mesure différentes. En effet, à notre connaissance, aucun des articles recensés portant sur les métriques en génie logiciel n'offre une solution pour l'agglomération des résultats de métriques.

Ce mémoire avait pour but de développer un modèle permettant l'agglomération des résultats de diverses métriques, et ce indépendamment des échelles de mesure utilisées, permettant ainsi la quantification des attributs de la qualité logicielle. Le modèle, surnommé le Processus de Quantification des Attributs Logiciels (PQAL), s'inspire du modèle théorique *Analytical Hierarchy Process* (AHP) de Saaty (1996), lequel s'appuie sur la théorie de prise de décisions multicritères (PDMC). Comme attribut de la qualité, nous avons utilisé le processus de développement logiciel, tel qu'élaboré dans le modèle de qualité de Boloix et Robillard (1995). Afin d'appliquer notre modèle à l'évaluation d'un processus de développement logiciel, nous avons initialement validé, à partir des critères de vérification et de validation suggérés par Florac et Carleton (1999), la base de données générée par six équipes d'étudiants travaillant sur le même projet dans le cadre d'un cours de premier cycle, Atelier de Génie Logiciel. Finalement, nous avons procédé à l'application du modèle et à l'évaluation du processus de développement logiciel en utilisant deux métriques, celle de la distribution de l'effort et celle de l'évaluation des artefacts.

L'application du PQAL nous a permis de constater que la «signature» graphique de la distribution de l'effort pour chacun des workflows différait pour chacune des équipes,

bien que le projet était essentiellement identique pour toutes les équipes d'étudiants. Ainsi, la distribution de l'effort semblait dépendre peu du projet, mais plutôt des équipiers. À l'aide de notre modèle, il nous a été possible d'observer que toutes les équipes, mise à part l'équipe 5, présentaient une faible évaluation de la distribution globale de l'effort. Cette observation nous laissait présumer une mauvaise compréhension du processus UPEDU et/ou une gestion inefficace au niveau du temps et des tâches pour la plupart des équipes. Toutefois, il s'avérait surprenant de constater un faible résultat pour l'équipe 3, considérant sa performance initialement établie par le professeur et le chargé de laboratoire. Nous pouvions donc supposer que l'évaluation, initialement offerte par le professeur et le chargé de laboratoire, s'inspirait davantage des artefacts générés par les équipes que de leur effort.

À la lumière de la seconde métrique, le PQAL nous a permis de noter, qu'en comparaison, l'évaluation des artefacts s'avérait significativement plus élevée que celle de la distribution de l'effort. Mise à part l'équipe 5, l'évaluation des artefacts semblait fortement favoriser toutes les équipes, lesquelles présentaient de faibles évaluations concernant la distribution de l'effort. Ainsi, l'application de notre modèle laissait croire que les évaluations de ces deux métriques étaient indépendantes l'une de l'autre. En d'autres mots, il semblait peu probable de prédire l'effort qu'à partir de la qualité des artefacts générés.

Afin de pouvoir évaluer le processus de développement logiciel dans sa globalité, l'agglomération des résultats des deux métriques faisait l'objet de la prochaine étape. À la lumière des résultats obtenus, nous pouvions observer une concordance entre le rang attribué aux équipes à l'aide du PQAL et celui déterminé par le professeur. Une telle corroboration nous portait à croire en l'utilité future de notre modèle. Les résultats de cette recherche ont donc démontré que le PQAL a pour objectif de "mesurer," de façon quantifiable, les attributs de la qualité logicielle et ce, tout en respectant la théorie de la mesure. Bien qu'elle existe depuis plusieurs décennies dans les domaines mathématique

et économique, la théorie de prise de décisions multicritères (PDMC) demeure, jusqu'à ce jour, peu utilisée dans notre discipline. De plus, son utilisation se restreint au classement plutôt qu'à l'évaluation. Le PQAL, au contraire, se voulant un modèle d'évaluation, nous croyons que l'application d'une théorie «empruntée» à une discipline connexe peut s'avérer une contribution inédite en génie logiciel.

Le modèle développé dans le cadre de cette recherche, le Processus de Quantification des Attributs Logiciels (PQAL), pourra contribuer à l'avancement des connaissances au niveau académique et au renouvellement des pratiques en milieu industriel. Ainsi, l'utilisation du PQAL pourrait promouvoir la compréhension des étudiants, de même que des ingénieurs en regard de l'implication et de l'interaction des métriques lors de l'évaluation des attributs de la qualité logicielle. Enfin, nous recommandons la réalisation de futures recherches visant la validation du PQAL dans le milieu industriel afin de poursuivre le raffinement du modèle dans le cadre d'activités d'assurance qualité.

ABSTRACT

Trying to define and measure software quality has always been considered a complex activity within the software engineering arena. Despite the amount of metrics designed to evaluate software attributes, difficulties remains as to the choice and numbers of metrics required to evaluate an attribute as well as to the interpretation of their results. One of the causes to these dilemma lies with measurement theory that prevents us from aggregating results issued from different types of measurement scales. In fact, to our knowledge, present software engineering literature related to metrics, do not appear to offer a solution to this problem.

The purpose of this master's thesis consists in developing a model that will enable the quantification the quality of software attributes by agglomerating various metrics' results issued from different measurement scales. Moreover, all the calculations associated with this quantification, will respect the confines of measurement theory. This model, referred henceforth as the Processus de Quantification des Attributs Logiciels (PQAL), was inspired by Saaty's (1996) *Analytical Hierarchy Process* (AHP), which is based on the multicriteria decision making theory. In order to validate the PQAL, the quality attribute chosen for this thesis was the software development process, based on Boloix and Robillard's (1995) quality model. The data used by our model for evaluation, was generated by six teams of students working on the same project within an undergraduate software engineering class called Software Engineering Workshop. Two different metrics results were used by the PQAL, the first was related to the effort distribution and the other, to the artifacts' evaluation. Before using the raw data related to the effort distribution of each team, the database was verified and validated based on several criteria suggested by Florac and Carleton (1999). Once completed, the results of both metrics were then compiled and introduced to our model.

The application of the PQAL has allowed us to observe that the graphical «signature» of the effort distribution for each workflow was different for each teams, even though they worked on the same project. Consequently, the effort distribution seems to depend mostly on the workers rather than the project itself. Through the use of the PQAL, we also noted that every team, except for team #5, had a weak evaluation as to the effort distribution during the software process. Such observation seems to indicate a lack of comprehension regarding the UPEDU process and/or an inefficient management related to time and task management for most teams. This result was even more surprising when we realized that team #3 did not faired well in this evaluation, especially considering that the teacher and laboratory assistant ranked their final product has the second best, after team #5. This has lead us to believe that the initial evaluation offered by the teacher and laboratory assistant for each team, was based mostly on the final product rather than the actual effort distribution.

Following the application of the second metric to the PQAL (i.e.: the artifacts' evaluation), our model showed us that most teams' evaluation were significantly improved from that of the effort distribution evaluation. Excluding Team #5, the artifacts' evaluation favored every team, even though their scores during the effort distribution evaluation were low. Therefore, the application of our tool seem to indicate that both metrics' evaluations are independent of each others. In other words, it does not seem likely that we can predict the "quality" of the effort distribution from the "quality" of the generated artifacts.

In order to evaluate the software development process in its entirety it was necessary to agglomerate both metrics. Based on our results, we were able to observe a correlation between the ranks attributed to the teams by the PQAL and the one given to them by the teacher. Moreover, our research findings has shown that the PQAL can « measure » in a quantifiable manner, the software quality attributes while respecting measurement theory.

Despite its widespread use in mathematical and economical domains, multicriteria decision making theory remains rarely used in our discipline. Moreover, its use has been mainly restrained for the classification of elements. We believe that the PQAL, being rather an evaluation tool, can provide an interesting contribution to software engineering. Such knowledge contribution could have implications in both the academia and industrial practices. For example, the PQAL utilization could enhance students as well as engineers' comprehension of metrics' implication and interaction surrounding the evaluation of software quality attributes. Finally, future research is needed to validate the PQAL within the industrial arena in order to refine the model and to increase its contribution to the field of software quality assurance.

TABLE DES MATIÈRES

DÉDICACE iv
REMERCIEMENTS v
RÉSUMÉ vi
ABSTRACT ix
LISTE DES TABLEAUX xvi
LISTE DES FIGURES xviii
CHAPITRE 1 : PROBLÉMATIQUE 1
1.1. Le rôle de la mesure en génie logiciel 2
1.2. Le paradigme Goal-Question-Metrics 3
1.3. Le dilemme de combiner les métriques 4
1.4. But et Questions de recherche 7
CHAPITRE 2 : MODÈLE D'ÉVALUATION DE LA QUALITÉ LOGICIELLE 8
2.1. Échelles de Mesure 10
2.2. Terminologie utilisée par les MPDMCs 14
2.2.1. Définition d'un attribut 14
2.2.2. Définition d'un objectif 15
2.2.3. Définition d'un but 16
2.2.4. Définition d'un critère 16
2.3. Introduction au Analytical Hierarchy Process (AHP) 17
2.3.1. La Hiérarchie d'après Saaty 18
2.3.1.1. Création des Sous-Niveaux 18
2.3.1.2. Homogénéité des Éléments 21
2.3.1.3. Assurer une bonne représentation hiérarchique 23
2.3.2. Échelle de mesure utilisée par l'AHP 25
2.3.3. L'Échelle est de Type Absolu 28
2.3.4. L'Échelle de Ratio et la Normalisation 29
2.3.5. Les priorités 31

2.3.5.1. Création d'une matrice de priorité 31
2.3.5.2. Calcul des priorités pour un critère 33
2.3.5.3. Calcul des priorités en présence de plusieurs sous-niveaux 37
2.3.6. Cohérence 39
2.4. La mesure comparative absolue 41
2.5. L'impact de l'intensité d'un critère sur le but 43
2.5.1. Signification des résultats 48
2.6. Controverses associées au modèle AHP 49
2.6.1. Inversion des rangs 52
2.6.2. Intransitivité 54
2.7. Modifications au modèle AHP 56
CHAPITRE 3 : VÉRIFICATION ET VALIDATION DE LA BASE DE DONNÉES 59
3.1. Introduction 59
3.2. UPEDU 60
3.3. Description de la Base de Données 61
3.3.1. Objectif du Projet 61
3.3.2. Description des équipes 62
3.3.3. Structure des Équipes 62
3.3.4. Les Jetons 64
3.4. Critères de Vérification et de Validation de la Base de Données 66
3.4.1. Vérité 66
3.4.1.1. Est-ce que les données correspondent à un type numérique ou alphanumérique approprié? 67
3.4.1.2. Est-ce ce que le format est approprié? 67
3.4.1.3. Est-ce ce que les données sont à l'intérieur des plages? 68
3.4.1.4. Est-ce ce que les données sont complètes? 69
3.4.1.5. Est-ce ce que les équations arithmétiques sont appropriées? 69

3.4.2. Duplication....70
3.4.3. Consistance....72
3.4.4. Validité....72
3.5. Correction et Élimination des Jetons....74
3.6. Application du Processus de Vérification/Validation (V&V) de la Base de Données....75
3.7. Agglomération des Jetons....76
3.8. Résultats et Analyses Statistiques de la V&V par Équipe....81
3.8.1. Nombre Total Travaillé....81
3.8.2. Nombre Total Accepté....81
3.8.3. Nombre Total Corrigé et Accepté....84
3.8.4. Nombre D'erreurs de Chevauchement....84
3.8.5. Nombre D'erreurs de Workflows et/ou d'Artefacts....85
3.8.6. Nombre D'erreurs d'Artefacts....85
3.8.7. Nombre Total de CF Refusé....86
3.8.8. Nombre D'erreurs de Duplicata....86
3.8.9. Nombre d'Erreurs de Chevauchement....87
3.8.10. Nombre d'Erreurs : Artefacts/Activités/Workflow....87
3.8.11. Nombre d'Erreurs de Temps et Autres....87
3.9. Vue d'Ensemble de l'Analyse V&V....87
3.9.1. Distribution des Heures Acceptées et Refusées par Étudiant....88
3.9.2. Répartition des Heures Totales de Chaque Équipe....91
3.10. Discussion du Processus de Vérification et de Validation....92
CHAPITRE 4 : APPLICATION DU PQAL....94
4.1. Introduction....94
4.2. Application du PQAL à la Métrique de l'Effort....94
4.2.1. Quelle hiérarchie devrait-on utiliser?....95

4.2.2. Quel poids devrait-on accorder aux priorités pour chacun des éléments de la hiérarchie? 96
4.2.3. Quelle période de temps devrait-on utiliser pour évaluer le processus ? .. 100
4.2.3.1. Analyse de la distribution hebdomadaire de l'effort. 102
4.2.3.2. Calcul des Priorités des Artefacts et Workflows 107
4.2.4. Quels seront nos critères d'intensité? 108
4.2.5. Quelles sont les plages et les priorités des critères d'intensité? 116
4.3. Calcul et Analyse des Résultats de l'Évaluation de la Globalité des Distributions Hebdomadaires de l'Effort. 119
4.4. Calcul et Analyse des Résultats de l'Évaluation de la Distribution Globale de l'Effort 123
4.5. Métrique Additionnelle 124
4.5.1. Évaluation Globale des Artefacts 125
4.6. Calcul et Analyse de l'Évaluation Globale du Processus 130
CHAPITRE 5 : CONCLUSION 133
5.1. Conclusion 133
5.2. Contributions 136
5.2.1. Contributions au Niveau Théorique et de Recherche 136
5.2.2. Applications Futures 136
5.2.2.1. Mesure Progressive de l'Effort 136
5.2.2.2. Mesure Prédictive de l'Effort 140
5.2.3. Contributions dans le Milieu Industriel 143
5.3. Recommandations 143
5.3.1. Recommandations sur le Plan Pratique 143
5.3.2. Recommandations sur le Plan Recherche 143
5.4. Limites de l'Étude 144
5.5. Mot de la Fin 144
REFERENCES 146

LISTE DES TABLEAUX

Tableau 2.1: Types d'échelles de mesure 11
Tableau 2.2: Relation entre les types d'échelles de mesure et les statistiques permissives 12
Tableau 2.3: Conversions d'ensembles pour différents types d'échelles 12
Tableau 2.4: Comparaisons de ratio entre les éléments des ensembles *M* et *M'* du Tableau 2.3 13
Tableau 2.5: Échelle de Comparaison Binaire (Saaty, 1984). 26
Tableau 2.6: Normalisation d'ensembles provenant de différents types d'échelles. 29
Tableau 2.7: Normalisation des d'ensembles *M* et *M'* du Tableau 2.6. 30
Tableau 2.8 Valeurs de l'Index Randomisé pour Différentes Matrices. 40
Tableau 2.9: Valeurs Accordées aux Éléments d'Intensités 45
Tableau 2.10: Comparaisons d'Intensités pour le DD 45
Tableau 2.11: Mode d'intensité idéal pour le DD 45
Tableau 2.12: Trois scénarios comportant des résultats de métriques différents 47
Tableau 2.13: Impact des intensités sur la fiabilité 47
Tableau 3.1: Abréviations des Champs de la Base de Données. 65
Tableau 3.2: Vérification de la Caractéristique: Type. 67
Tableau 3.3: Vérification de la Caractéristique: Format. 68
Tableau 3.4: Vérification de la Caractéristique: Plages. 68
Tableau 3.5: Vérification de la Caractéristique: Complet 69
Tableau 3.6: Vérification de la Caractéristique: Équation. 70
Tableau 3.7: Vérification de la Caractéristique: Duplication: Duplicata. 71
Tableau 3.8: Vérification de la Caractéristique : Duplication: Chevauchement 71
Tableau 3.9: Vérification de la Caractéristique: Validité. 74
Tableau 3.10: Distribution des Éléments du Processus UPEDU, 1ère V&V. 78
Tableau 3.11: Artefacts Nécessaires au Processus 78

Tableau 3.12: Règles combinatoires. 79
Tableau 3.13: Distribution des Éléments du Processus UPEDU, 2ième V&V....81
Tableau 3.14: Résultats Statistiques de la V&V Finales. 83
Tableau 3.15: Distribution du Temps Travaillé par Équipe....88
Tableau 4.1: Distribution de l'Effort des Éléments de Modélisation 97
Tableau 4.2: Priorités des Éléments de Modélisation: Workflows. 99
Tableau 4.3: Priorités des Éléments de Modélisation: Artefacts/Activités. 100
Tableau 4.4: Priorités Hebdomadaires des Workflows. 108
Tableau 4.5: Priorités Hebdomadaires des Artefacts/Activités. 108
Tableau 4.6: Exemples de Plages d'intensités Basées sur la distance en Écart type. 111
Tableau 4.7: Valeurs Normalisées Hebdomadaires pour les Équipes 2, 3 et 4....112
Tableau 4.8: Valeurs Normalisées Hebdomadaires pour les Équipes 5, 6 et 7....113
Tableau 4.9: Règles d'Élimination de Valeurs Extrêmes 113
Tableau 4.10: Valeurs des Moyennes Utilisées pour Déterminer les Intensités 114
Tableau 4.11: Valeurs des Écarts Types Utilisés pour Déterminer les Intensités....114
Tableau 4.12: Distance en Écart Type pour les Équipes 2, 3 et 4....115
Tableau 4.13: Distance en Écart Type pour les Équipes 5, 6 et 7....116
Tableau 4.14: Plages d'Intensités pour Utilisation du PQAL....117
Tableau 4.15: Poids Accordés aux Intensités de Chaque Valeur Hebdomadaire 118
Tableau 4.16: Valeurs des Priorités Hebdomadaires de la Distribution de l'Effort....119
Tableau 4.17: Valeur Globale de l'Évaluation de la Distribution de l'Effort. 123
Tableau 4.18: Priorités des Workflows pour l'Évaluation des Artefacts. 126
Tableau 4.19: Priorités des Artefacts pour l'Évaluation des Artefacts. 126
Tableau 4.20: Notes Accordées à Chacun des Artefacts....127
Tableau 4.21: Comparaison entre les Résultats du PQAL et ceux des Notes Finales. 132

LISTE DES FIGURES

Figure 1.1: Exemple d'un diagramme de Kiviat (Drake, 1996) 5
Figure 2.1: Représentation Hiérarchique pour Déterminer les Rangs 20
Figure 2.2: Représentation Hiérarchique pour l'Évaluation 21
Figure 2.3: Représentation Hiérarchique de plusieurs critères 38
Figure 2.4 : Algorithme du AHP pour des mesures relatives (Leblanc et al., 1993).....41
Figure 2.5: Hiérarchie pour la fiabilité logicielle.....44
Figure 2.6: Hiérarchie incluant les niveaux d'intensités pour la fiabilité logicielle.....46
Figure 2.7: Algorithme du AHP pour des mesures relatives (Leblanc et al, 1993).....58
Figure 3.1: Enchaînements d'activités reliées aux UPEDU.
(Adapté de Krutchen, 2000.).....61
Figure 3.2: Interface usager de l'outil de jeton MS Access 65
Figure 3.3: Représentation des Heures Acceptées et Refusées par Étudiant. 90
Figure 3.4: Répartition des Heures Normalisées Totales par Équipe 92
Figure 4.1: Hiérarchie pour l'Évaluation du Processus 96
Figure 4.2: Distribution de l'Effort par Workflows par Équipe.....98
Figure 4.3: Évaluation Globale du Processus. 101
Figure 4.4: Signature du Processus de l'Effort : Équipe 2.....103
Figure 4.5: Signature du Processus de l'Effort : Équipe 3.....103
Figure 4.6: Signature du Processus de l'Effort : Équipe 4.....104
Figure 4.7: Signature du Processus de l'Effort : Équipe 5.....104
Figure 4.8: Signature du Processus de l'Effort : Équipe 6.....105
Figure 4.9: Signature du Processus de l'Effort : Équipe 7.....105
Figure 4.10: Application du PQAL pour la Semaine du 2000/02/21. 120
Figure 4.11: Distribution Graphique de l'Évaluation Hebdomadaire de l'Effort. 121
Figure 4.12: Hiérarchie pour l'évaluation globale des Artefacts. 125
Figure 4.13: Représentation du PQAL à l'évaluation globale des artefacts. 129

Figure 4.14: Hiérarchie pour l'évaluation globale du processus. 130
Figure 5.1: Distribution Progressive de l'effort. 139
Figure 5.2: Représentation prédictive de l'effort. 142

CHAPITRE 1
PROBLÉMATIQUE

Un des objectifs principaux du génie logiciel consiste à créer des produits logiciels de qualité (Fenton & Pfleeger, 1997; Pressman, 1997). Bien qu'il existe plusieurs définitions du concept «qualité» dans les écrits portant sur la qualité logicielle (Tervonen, 1996), il s'avère intéressant de noter qu'il y a peu de consensus quant à ce terme. Pressman (1997) explique qu'il est difficile de définir la «qualité logicielle» en raison de la nature intangible du produit. Contrairement à un objet tangible, le logiciel correspond à une entité intellectuelle, ce qui rend sa qualité plus difficile à définir et à mesurer. De plus, une caractéristique subjective semble s'associer rapidement à ce concept. Bien que l'on peut croire que tous les individus (e.g., programmeurs, chefs de projet, utilisateurs) impliqués dans le cycle de vie d'un logiciel visent ou exigent que celui-ci soit de qualité, on ne peut espérer qu'ils se basent sur les même critères d'évaluation pour en déterminer la qualité. Ces individus, pour la plupart, perçoivent le logiciel de divers points de vue, puisque chacun a des besoins ou des attentes différentes. Par exemple, un critère de qualité logicielle pour un utilisateur pourrait être la facilité d'utilisation du logiciel, tandis que pour un chef de projet, un critère serait davantage la distribution de l'effort durant le processus de développement logiciel. Si nous acceptons l'idée que les critères d'évaluation de qualité logicielle dépendent du point de vue de chaque individu, il semble donc nécessaire que la définition de «qualité logicielle» englobe ces différentes perspectives. En raison de sa perspective globale, nous utilisons, dans le cadre de ce mémoire, la définition de qualité du *International Standard Organization* (*ISO*) (8402-1986): « *The totality of features and characteristics of a software product that bear on its ability to satisfy stated or implied needs* ». Cette définition conceptuelle, à elle seule, n'est toutefois pas suffisante pour évaluer la qualité d'un logiciel. En effet, l'utilisation de mesures s'avère nécessaire afin de procéder à cette évaluation.

1.1. Le rôle de la mesure en génie logiciel

L'évaluation de la qualité d'un logiciel implique, selon Fenton et Pfleeger (1997), les mesures des attributs externes et internes du logiciel. Ils expliquent qu'un attribut externe correspond à l'interrelation entre le logiciel et son environnement. Par exemple, l'effort, c'est-à-dire le nombre d'heures par personne, nécessaire pour compléter un artefact peut représenter la mesure d'un attribut externe. Un attribut interne, quant à lui, peut être mesuré directement, tel que le nombre de lignes de code d'un logiciel. C'est ainsi que pour plusieurs auteurs dont, Fenton et Pfleeger (1997), de même que Pressman (1997), la «mesure» des attributs correspond à une activité fondamentale dans le monde du génie logiciel. Malheureusement, tout comme le concept de «qualité», la «mesure» s'avère un concept qui demeure souvent incompris et/ou ignoré (Fenton & Pfleeger, 1997; Pressman 1997). On se rappelle qu'une des raisons d'être du domaine du génie logiciel est de promouvoir la qualité. Afin de réaliser cet objectif, l'évaluation de logiciels doit être entreprise à partir de différentes mesures, lesquelles font partie intégrante de ce que l'on appelle «métriques». Pressman (1997) affirme que plusieurs personnes éprouvent des difficultés à différencier entre les termes «mesure» et «métrique». La mesure, selon Pressman, est définie comme un indicateur quantitatif en regard de l'étendue, la quantité, les dimensions ou la grandeur de certains attributs du produit ou du processus logiciel. En ce qui a trait à la définition de «métrique», il cite celle mentionnée par le *International Electrical and Electronic Engineer* (*IEEE*): *«a quantitative measure of the degree to which a system, components, or process possesses a given attribute* » (p. 77). En d'autres mots, une métrique inclut toujours des mesures, mais une mesure ne correspond pas nécessairement à une métrique.

En dépit de l'importance accordée aux métriques logicielles, on dénote, encore de nos jours, des difficultés à s'entendre sur l'éventail des attributs à mesurer, le choix des métriques à utiliser, de même que sur l'interprétation des résultats recueillis (Fenton & Pfleeger, 1997; Pressman, 1997; Kitchenham et al, 1995). De plus, la relation entre les

métriques et les attributs logiciels ne nous semble pas toujours explicite. À notre connaissance, il n'existe aucun outil ou standard qui identifie toutes les métriques devant être utilisées pour évaluer un attribut particulier.

Le IEEE (1992) a tenté d'offrir une méthodologie d'évaluation de la qualité logicielle à l'aide de métriques basées sur le standard IEEE 1061-1992. Toutefois, ils concluent que les métriques offertes ne sont que des «exemples». Ceci nous porte à croire que les métriques dépendent, entre autres, de la nature des logiciels, de leurs attributs respectifs et de la maturité de l'organisation de réalisation. En d'autres mots, le choix des métriques pour évaluer ou prédire un attribut de la qualité logicielle est laissé à chaque utilisateur. Bien que ceci soit compréhensible, en considérant chaque projet logiciel comme unique (e.g., différents langages de programmation, personnel, budget, application, etc.), cela ne facilite guère la tâche aux utilisateurs désirant évaluer la qualité du logiciel.

1.2. Le paradigme Goal-Question-Metrics

Afin de guider les utilisateurs dans leurs activités de mesure, Basili et Weiss (1984) ont introduit le paradigme *Goal-Question-Metrics* (GQM). Le but du GQM consiste à permettre à chaque utilisateur d'établir des objectifs liés à un projet logiciel. À l'aide de questions reliées aux objectifs visés (e.g., fiabilité), il s'avère alors possible de spécifier les métriques appropriées. En s'appuyant sur leurs expériences quant à la mesure et à l'évaluation des processus de génie logiciel, Basili et Rombach (1988) ont observé que «Most aspects of software processes and products are too complicated to be captured by a single metric» (p. 790). Ainsi, ils concluent également que plusieurs métriques s'avèrent nécessaires pour répondre aux questions du modèle *GQM*.

Ces auteurs admettent que de choisir tous les objectifs, questions et métriques appropriés au projet, correspond à une activité complexe, laquelle dépend du niveau d'expérience de l'utilisateur. Tout comme Basili et Rombach (1988), plusieurs auteurs (Basili et Weiss, 1984; Ebert, 1999; Hall & Fenton, 1997; Hauser & Katz, 1998; Kitchenham et al., 1995;

Pfleeger, 1995; Pfleeger et al., 1997) sont d'avis que chaque programme de métriques doit être créé à partir des besoins et objectifs spécifiques des utilisateurs. Ceci semble donc impliquer que les utilisateurs doivent concevoir leur propre plan de métriques. Toutefois, pour les utilisateurs novices, cette démarche peut s'avérer complexe. Drake (1996), quant à lui, énumère une série de 20 métriques utilisées par le *Software Engineering Applied Technology Center* (*SEATC*) du *National Security Agency* afin d'évaluer la qualité logicielle. Bien que les résultats s'avèrent intéressants, l'auteur n'avance aucune explication quant aux raisons sous-jacentes au nombre de métriques utilisées par le *SEATC*. On peut alors se questionner sur la nature des résultats si le nombre de métriques avait été différent. Ainsi, jusqu'à ce jour, il n'existe aucune règle précise pouvant guider le choix de métriques appropriées à l'évaluation d'un attribut quelconque. Il semble également qu'aucune règle, spécifiant le nombre requis de métriques nécessaires à l'évaluation des objectifs fixés, n'est disponible à l'heure actuelle.

1.3. Le dilemme de combiner les métriques

Un problème additionnel se pose lorsqu'on admet qu'il s'avère nécessaire d'utiliser plusieurs métriques afin d'évaluer les attributs de la qualité logicielle. En effet, l'agglomération de deux ou plusieurs métriques peut entraîner une violation des principes sous-jacents à la théorie de la mesure. Fenton et Pfleeger (1997) expliquent que la plupart des métriques proviennent d'une échelle ordinale, limitant, par le fait même, la majorité des opérations statistiques. Ainsi, le regroupement de valeurs ordinales, intervalles et de ratio, selon lui, devient problématique puisque les lois sur la théorie de la mesure ne sont pas respectées. De plus, suite à une recherche exhaustive sur le sujet, aucun écrit aborde le problème d'agrégation des métriques afin d'offrir une solution à ce dilemme, en raison que les métriques ne sont généralement pas caractérisées par le même type d'échelles de mesure. Une limitation importante aux regroupements des métriques persiste donc dans le domaine du génie logiciel.

De façon à contourner ce problème associé à la théorie de la mesure, plusieurs auteurs (Daskalantonakis, 1994; Drake, 1996; Dutoit & Bruegge, 1998; Hollenbach et al, 1997; Oman & Pfleeger, 1997; Pfleeger et al, 1992) utilisent le diagramme Kiviat (Figure 1.1) qui consiste à présenter visuellement les résultats de toutes les métriques employées pour évaluer la qualité d'un logiciel. Toutefois, ces métriques sont comparées à partir de valeurs historiques plutôt que d'être comparées entre elles, et ce par rapport au logiciel en question. De plus, le diagramme ne permet pas d'attribuer une valeur numérique ou un pourcentage à la qualité du logiciel. Bien que ce diagramme indique les résultats excédant les valeurs minimales et maximales imposées par les utilisateurs, aucune inférence ne s'avère possible quant à la nature ou le niveau de qualité du produit.

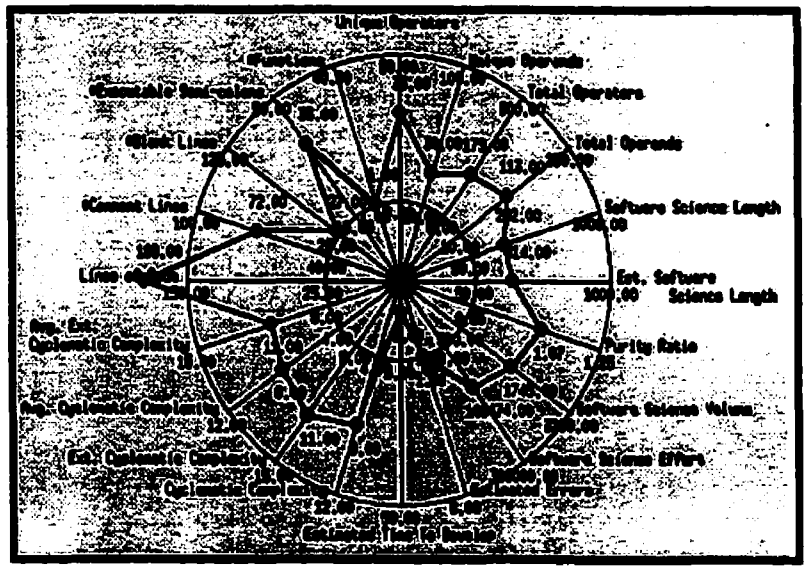


Figure 1.1: Exemple d'un diagramme de Kiviat (Drake, 1996)

En s'appuyant sur les théories de la prise de décisions multicritères, le modèle *Analytical Hierarchy Process* (AHP), introduit en 1972 par le physicien américain Saaty (1972), propose que pour simplifier un système complexe de plusieurs éléments différents et connaître l'interaction de chaque élément sur ce système, nous devons commencer par faire un agrégat de tous les éléments en créant une hiérarchie. Ensuite, la relation entre les

éléments de chaque niveau de la hiérarchie est déterminée et interprétée par des priorités en terme d'importance. Cet auteur utilise des comparaisons binaires afin de déterminer ces priorités. Il s'avère également possible, avec ce modèle, d'attribuer des priorités aux intensités ou aux standards de chaque élément de la hiérarchie. Bien que plusieurs auteurs (Boloix et Robillard, 1995a; Finnie, Wittig et Petkov 1993; Ho-Won, 1998; Karlsson et Ryan, 1998; Karlsson et Ryan, 1997; Lee et Xue, 1999; Rebstock et Kaula, 1996) ont utilisé le AHP, ou d'autres méthodes multicritères, dans le domaine du génie logiciel, jusqu'à maintenant son application s'est limitée à la classification, plutôt qu'à l'évaluation et l'agglomération d'éléments.

S'inspirant de ce modèle, nous croyons, qu'une agrégation des données pourrait ainsi permettre de générer, à l'aide d'un algorithme, une valeur représentative de toutes les métriques qui sont utilisées pour évaluer des attributs logiciels. Par exemple, plusieurs métriques évaluant l'attribut de fiabilité d'un logiciel pourrait conclure que celui-ci est fiable à 85%. Un tel paradigme d'agglomération de différents attributs pourrait alors favoriser une vision globale de l'entité mesurée.

À la lumière du modèle AHP de Saaty (1996), nous nous proposons, suite à l'élaboration d'un modèle visant l'agglomération de métriques, d'utiliser une base de données accessible afin d'évaluer un attribut de la qualité logicielle. Dans le cadre de ce mémoire, nous utiliserons comme attribut de qualité, le processus de développement logiciel, tel qu'identifié dans le modèle de qualité de Boloix et Robillard (1995a). Ainsi, deux métriques, celle de la distribution de l'effort, représentant un élément qui est absorbé par le processus, et celle de l'évaluation des artefacts, correspondant à un élément produit suite au processus, seront retenues afin de mesurer le processus de développement logiciel.

Nous croyons donc que l'adaptation de la théorie de Saaty au domaine du génie logiciel pourrait s'avérer, à la fois, prometteuse et novatrice dans la résolution de la problématique exposée ci-haut.

1.4. But et Questions de recherche

Le but visé par cette recherche universitaire consiste à développer un modèle permettant l'agglomération des résultats de diverses métriques, et ce indépendamment des échelles de mesure utilisées, permettant ainsi la quantification des attributs de la qualité logicielle. Générant ainsi une valeur représentative des attributs de la qualité logicielle, le Processus de Quantification des Attributs Logiciels (PQAL) pourra, nous croyons, faciliter l'activité d'assurance qualité en milieux académique et industriel. Voici les questions que nous tenterons de répondre dans le cadre de ce projet de deuxième cycle:

- Comment pouvons-nous adapter le modèle AHP de Saaty (1996) afin d'élaborer un modèle de quantification de la qualité logicielle?
- Comment pouvons-nous générer une valeur représentative de l'évaluation d'un attribut logiciel en agglomérant un amalgame de valeurs de métriques?
- Quel sera le niveau de validité du modèle?
- Quel sera le niveau de complexité relatif à l'utilisation du modèle?

Finalement, nous croyons qu'une telle étude pourra contribuer à promouvoir le développement des connaissances dans le domaine du génie logiciel et, dans un deuxième temps, faciliter le travail de l'utilisateur en dirigeant celui-ci lors des prises de décisions relatives à l'amélioration de la qualité logicielle. Les écrits importants, de même que le modèle théorique à la base de l'élaboration de notre modèle feront l'objet du prochain chapitre.

CHAPITRE 2
MODÈLE D'ÉVALUATION DE LA QUALITÉ LOGICIELLE

Certaines décisions sont faciles à prendre, tandis que d'autres ne le sont pas. Le problème existe lorsque nous devons choisir entre plusieurs alternatives (e.g.: des voitures de marques différentes). Afin de choisir la meilleure alternative, nous devons nous appuyer sur différents critères (e.g. prix, couleur, consommation d'essence). Toutefois, plus les critères sont nombreux, plus il est difficile de prendre une décision.

Déjà en 1772, Benjamin Franklin introduit une des premières méthodologies pour solutionner des problèmes multicritères, tel qu'on le constate dans une de ses lettres à Joseph Priestly. Devant choisir entre deux alternatives, Priestly demande l'opinion à Franklin afin de résoudre son dilemme. Franklin réalise que pour prendre la meilleure décision, il faut considérer la multiplicité des aspects (Zeleny, 1982). Il reconnaît que l'exercice de choisir la meilleure alternative parmi plusieurs autres est une activité complexe, puisqu'il est difficile de visualiser simultanément toutes les raisons "pour" ou "contre" une alternative. Il a toutefois fallu attendre 180 ans avant que les mathématiciens s'intéressent aux problèmes de décisions multicritères (DMC). Un des premiers pionniers de ce domaine fut T.C. Koopman en 1951 et depuis, nous retrouvons plus d'une trentaine d'outils différents pouvant être utilisés pour résoudre des problèmes de DMC (Al-Shemmeri et al., 1997; Zanakis et al., 1998). Malgré tout, le champ de DMC est loin d'être considéré comme une science exacte.

Voogd (1983) a démontré que si nous appliquons plusieurs méthodes de prise de décisions multicritères (MPDMC) au même problème, les résultats auront une variance de plus de 40%. Zanakis et al. (1998) en font également l'observation. En effet, plusieurs facteurs peuvent influencer les résultats de ces méthodes: la nature et la robustesse de l'algorithme employé, le nombre de critères et d'alternatives utilisés, l'expérience du ou

des décideurs, le nombre de décideurs impliqués dans le choix des critères, etc. Selon la littérature existante, il n'existe aucun consensus supportant une MPDMC idéale. Ceci peut être expliqué par le fait qu'il existe peu d'études comparant les différentes méthodes (Zanakis et al, 1998).

D'après Al-Shemmeri et al. (1997), chaque MPDMC possède des points forts ainsi que des faiblesses. Ils reconnaissent l'importance de choisir le bon modèle pour l'application donnée. Pour des raisons qui seront élaborées tout au long de ce chapitre, nous avons décidé de privilégier la méthode de Saaty (1972) qui s'intitule Analytical Hierarchy Process (AHP).

L'objectif de ce chapitre consiste à introduire le modèle de prise de décisions proposé dans le cadre de ce mémoire de maîtrise. Permettant aux utilisateurs de quantifier l'évaluation des attributs de qualité logiciel, cette méthode vise à faciliter les prises de décisions des acteurs impliqués à cet égard. L'analyse de prise de décision est un champ d'étude très vaste. De par la limite imposée par un travail de deuxième cycle, il nous est impossible de l'aborder en détail.

Ainsi, nous débuterons en offrant un bref résumé des échelles de mesure les plus fréquemment utilisées par les MPDMC. L'introduction au langage et aux concepts utilisés en DMC et la description du modèle *Analytical Hierarchy Process* (AHP) de Saaty (1996) suivront. La section suivante portera sur l'utilisation du modèle AHP pour l'évaluation des attributs de la qualité logicielle. Pour terminer, nous discuterons sur les problèmes inhérents aux modèles de type AHP, leurs contre-mesures, ainsi que certaines modifications apportées au modèle.

2.1. Échelles de Mesure

Tel que nous allons voir dans les sections subséquentes, la mesure joue un rôle essentiel dans les MPDMCs. Pour cette raison, nous débutons ce chapitre en faisant un bref survol de la théorie des échelles de mesure.

Fenton et Pfleeger (1997) définissent l'action de mesurer comme étant le "mapping from the empirical world to the formal world" et la mesure comme étant le "number or symbol assigned to an entity by this mapping in order to characterize an attribute" (p. 28). En d'autres mots, la mesure directe d'un attribut[1] assigne une représentation *M* ("mapping") d'un système empirique (observable) à un système de relation numérique. Par exemple, le poids, l'attribut, de cette personne est de 50 kilogrammes. Cette représentation *M* nous permet ainsi de manipuler les données du monde mathématique et ensuite d'utiliser ces résultats afin de mieux comprendre, contrôler et améliorer les actions sur un attribut à l'intérieur du monde observable (Fenton & Pfleeger, 1997; Pfleeger et al., 1997). Dans ce contexte, la représentation *M* réfère à une échelle de mesure. Il existe plusieurs échelles de mesure, les cinq plus importantes utilisées sont décrites par ordre de *richesse* dans le Tableau 2.1.

Si un type d'échelle A, contient toutes les transformations admissibles de l'échelle B, nous disons que l'échelle A est plus riche que l'échelle B. C'est-à-dire, plus que le système de relation empirique s'avère riche, plus l'échelle de mesure est sophistiquée. En possédant la mesure d'un attribut quelconque, nous désirons connaître quelles autres mesures acceptables il nous est possible d'appliquer. En d'autres mots, nous voulons représenter une mesure acceptable par une autre. Par exemple, nous pouvons convertir le nombre de kilomètres en nombre de milles, $M(x) = a*M(y)$ (e.g.: 1 mille = 1.6*1 kilomètre), étant donné que la mesure de la longueur provient d'une échelle de ratio. D'un autre coté, nous ne pouvons calculer le ratio de mesure provenant d'une échelle nominale, ordinale ou

d'intervalle. Ainsi, les analyses mathématiques et statistiques des mesures demeurent limitées par le type de leur échelle.

Le Tableau 2.2, basé sur Fenton et Pfleeger (1997, p. 59), identifie les opérations statistiques admissibles aux types d'échelle de mesure. En examinant ce tableau, on dénote clairement que le type d'échelle limite les manipulations mathématiques de ses données et que le niveau de qualité d'information est fort supérieur avec une échelle de ratio. Prenons par exemple le Tableau 2.3 qui démontre deux ensembles de valeurs (i.e., ensemble M : A, B, C et D) appartenant à deux types d'échelle différents (i.e., intervalle et ratio). Suite à des transformations admissibles différentes, il devient possible de créer de nouveaux ensembles de valeurs (i.e., ensemble M' : A', B', C' et D').

Tableau 2.1: Types d'échelles de mesure

Nominal	$M' = F(M)$ Représentation 1-1 de M à M'	Étiquetage, classification d'entités
Ordinal	Fonction monotonique croissante de M à M', i.e., $M(x) >= M(y) =>$ $M'(x) >= M'(y)$	Préférence, dureté, qualité de l'air, dimension des vêtements.
Intervalle	$M' = a M + b$ $(a > 0)$	Temps relatif, années, température (Celsius, Fahrenheit).
Ratio	$M' = a M$ $(a > 0)$	Intervalle de temps, longueur, température (Kelvin), vitesse
Absolu	$M' = M$	Comptage (e.g. nombres de lignes de code)

[1] Un attribut est une propriété d'un entité (i.e. un objet ou événement dans le monde réel), e.g. la couleur (d'une chambre), la grandeur (d'une personne).

Tableau 2.2: Relation entre les types d'échelles de mesure et les statistiques permissives

Nominal	1. Équivalence	Mode Fréquence	Non-paramétriques
Ordinal	1. Équivalence 2. Plus grand que	Mode Percentile Spearman *r* Kendall *t*, *W*	Non-paramétriques
Intervalle	1. Équivalence 2. Plus grand que 3. Ratio connu de n'importe quelle intervalle	Moyenne Déviation standard Corrélation produit-moment de Pearson Multiple corrélation produit-moment	Non-paramétriques et paramétriques
Ratio	1. Équivalence 2. Plus grand que 3. Ratio connu de n'importe quelle intervalle 4. Ratio connu de deux valeurs d'échelle	Moyenne géométrique Coefficient de variation	Non-paramétriques et paramétriques

Tableau 2.3: Conversions d'ensembles pour différents types d'échelles

Type d'échelle	Ensemble *M*				Conversions Admissibles	Ensemble *M'*			
	A	B	C	D		A'	B'	C'	D'
Intervalle	5	10	15	20	$M' = a M + b$ ($a = 2$, $b = 25$)	35	45	55	65
Ratio	2	4	8	12	$M' = a M$ ($a = 5$)	10	20	40	60

D'après le Tableau 2.4, nous pouvons observer ce qui survient lorsque nous comparons les ratios des valeurs de *M* (e.g., A/B) avec ceux de *M'* (e.g., A'/B') de chaque échelle.

Tableau 2.4: Comparaisons de ratio entre les éléments des ensembles *M* et *M'* du Tableau 2.3

Type d'échelle	Conversions Admissibles	B/A = B'/A'	C/A = C'/A'	D/A = D'/A'	C/B = B'/C'	D/B = D'/B'	D/C = D'/C'
Intervalle							
	$M' = a\,M + b$ ($a = 2, b = 25$)	Faux	Faux	Faux	Faux	Faux	Faux
Ratio							
	$M' = a\,M$ ($a = 5$)	2 = 2 Vrai	4 = 4 Vrai	6 = 6 Vrai	2 = 2 Vrai	3 = 3 Vrai	2/3 = 2/3 Vrai

Il est clair que nous perdons de l'information avec l'échelle d'intervalle puisque les ratios ne sont pas similaires. D'autre part, la fraction des ensembles de l'échelle ratio étant toutes identiques, aucune information n'est perdue. Ceci s'avère compréhensible parce que la constante "*a*" est éliminée suite à la fraction de deux valeurs *M'*, i.e.:

$$\frac{a\ A'}{a\ B'} = \frac{A'}{B'} \qquad (1)$$

L'objectif de ce chapitre n'étant pas d'offrir une discussion détaillée sur la théorie de la mesure, il importe toutefois de souligner quelques points importants à retenir lors de la collecte, la manipulation et l'interprétation des données provenant d'échelles de mesure.

1. Lorsque nous voulons mesurer un attribut, il s'avère important de bien comprendre le système empirique afin de choisir le type d'échelle de mesure approprié le plus sophistiqué possible (i.e. ratio).

2. Lors de l'interprétation des données de mesure, il est capital de connaître l'origine du type d'échelle afin de valider les résultats. Par exemple, quelle est la signification des résultats si un modèle de prise de décision calcule la moyenne des données provenant d'une échelle ordinale?

3. Si nous possédons des données provenant de différents types de mesure et que nous voulons combiner tous les résultats, nous devons utiliser les opérations statistiques appropriées au type d'échelle le plus primitif. Par exemple, si un formulaire contient des données ordinales, d'intervalle et de ratio, nous sommes limités aux opérations mathématiques reliées à l'échelle ordinale.

Ceci résume notre survol sur les échelles de mesure, toutefois, lors de l'introduction du modèle Analytical Hierarchy Process (AHP) de Saaty (1996), nous discuterons davantage de l'application des échelles de mesure utilisées par le AHP.

2.2. Terminologie utilisée par les MPDMCs

Le but de cette section consiste à introduire les termes qui sont fréquemment utilisés par les MPDMCs. Ceci s'avère d'autant plus nécessaire puisqu'il n'existe, à l'heure actuelle, aucun standard pour définir ces termes.

2.2.1. Définition d'un attribut

D'après Zeleny (1982), ce terme réfère aux descripteurs d'une réalité objective, c'est-à-dire, ce sont des caractéristiques d'objets dans le monde "réel". Malgré que ces caractéristiques ne peuvent être séparées des valeurs du décideurs et de la réalité, elles

peuvent toutefois être identifiées et mesurées indépendamment des besoins et désirs du décideur. Par exemple, un décideur choisit une voiture d'après la couleur, le modèle, le coût, etc. Dans ce cas-ci, la couleur rouge demeure la couleur rouge et ni le décideur, ni la voiture ne peuvent changer cette couleur. Cet attribut est objectif, mais d'autres peuvent être subjectifs, par exemple le style ou la beauté de la voiture. Étant donné qu'ils sont moins bien définis que les attributs objectifs, les attributs subjectifs s'avèrent plus difficiles à mesurer de façon précise. Toutefois, grâce aux processus de prise de décision que nous allons voir sous peu, il devient possible de mesurer ces attributs subjectifs tout en limitant l'influence des valeurs et des besoins du décideur.

Leblanc (1999) explique qu'un attribut "est une caractéristique dont les performances sont mesurées directement sur un projet. Ces caractéristiques décrivent le projet comme un moyen pour satisfaire les objectifs du décideur". De plus, il souligne l'importance de demeurer prudent lors du choix des caractéristiques utilisées pour caractériser un projet en regard d'un objectif. La section , sur la discussion quant aux problèmes inhérents à certains modèles de prise de décision, précisera cette mise en garde.

2.2.2. Définition d'un objectif

D'après Zeleny (1982), une fois les attributs décrits et mesurés, le décideur doit déterminer à quel degré chaque attribut devra être maximisé ou minimisé. C'est à ce point que les valeurs et les besoins du décideur entrent en ligne de compte (Leblanc, 1999; Zeleny, 1982). Par exemple, parmi plusieurs voitures, est-ce que le décideur va choisir la voiture qui est la moins coûteuse, qui possède la plus belle couleur ou qui a le plus beau style? En d'autre mots, est-ce que le décideur va maximiser la couleur et le style alors qu'il va minimiser le coût? Afin de répondre à ces questions, nous devons spécifier les objectifs du décideur.

Zeleny (1982) souligne qu'un objectif en soi n'est pas un attribut, mais qu'il découle d'un ou plusieurs attributs. Ainsi, la grandeur peut être un attribut, mais trouver la plus grande personne dans un groupe (i.e. maximiser la grandeur) s'avère un objectif. De plus, un objectif peut aussi être une combinaison de deux ou plusieurs objectifs. À ce moment, les sous-objectifs deviennent des attributs pour le super-objectif. Ceci prend la forme d'une hiérarchie permettant alors d'articuler les objectifs poursuivis en termes d'attributs considérés (Leblanc, 1999). Finalement, un objectif, nous dit Leblanc (1999), correspond à une finalité pour le décideur. Cet auteur souligne aussi que "son [décideur] système de besoins et valeurs identifient ses objectifs et pondèrent leurs importances relatives en prenant aussi en considération le degré de satisfaction qui est atteint pour chacun".

2.2.3. Définition d'un but

Un but, quant à lui, représente la racine de la hiérarchie d'où tous les objectifs et attributs découlent. Zeleny (1982) mentionne qu'on élabore le but à la lumière des besoins et des valeurs du décideur. Malgré que la distinction entre les buts et les objectifs peut être ambiguë à cause de la nature de la hiérarchie, cet auteur explique que les buts réfèrent, de manière sans équivoque, à des cibles de niveau de réalisation définies par ces attributs et objectifs. Par exemple, un objectif pourrait être de maximiser l'économie d'essence et le but serait d'atteindre une consommation de 10 litres au 100 km.

2.2.4. Définition d'un critère

Leblanc (1999) définit un critère comme "un terme générique qui recouvre les éléments de jugements dans un choix qui peut être attribué à un objectif ou un attribut". Zeleny (1982), pour sa part, qualifie les critères comme étant des mesures, des règles et des standards qui guident les prises de décisions. Puisque les prises de décisions sont basées sur la sélection de différents attributs, objectifs ou buts, ces trois catégories peuvent être considérées comme des critères. En d'autres mots, les critères correspondent tous à des

attributs, objectifs ou buts jugés pertinents par un décideur dans une situation de prise de décisions. Donc, le terme prise de décisions multicritères représente une classe générale de problèmes qui inclut des *multi-attributs*, des *multi-objectifs* et des *multi-buts*. Il s'avère toutefois nécessaire de mentionner que le modèle AHP de Saaty (1996) remplace les termes "attribut" et "objectif" par les termes "critère" et "sous-critère".

Ceci termine notre section sur la terminologie utilisée par les MPDMCs. L'Analytical Hierarchy Process de Saaty (1996), modèle sur lequel nous avons basé notre modèle de quantification des attributs logiciels, fera l'objet de la prochaine section.

2.3. Introduction au Analytical Hierarchy Process (AHP)

Tel qu'il sera présenté ci-dessous, les MPDMC s'avèrent d'excellents outils pour répondre à un des objectifs de ce mémoire concernant l'agrégation des résultats des métriques. De tous les modèles de prise de décisions qui existent, celui répondant le mieux à nos besoins s'avère le modèle AHP de Saaty (1996), puisqu'il nous permet de faire l'agrégation de plusieurs critères et de les classer par ordre d'importance, contrairement à la méthode ELECTRE (Roy et Bouyssou, 1993), par exemple, qui procure seulement le meilleur critère. De plus, le modèle AHP nous permet de vérifier la cohérence des jugements du décideur lors de l'évaluation des poids d'importance. Ce modèle prend également en considération l'évaluation de la performance des attributs au niveau du but de la hiérarchie. Par exemple, si le résultat d'une métrique est négatif, quel sera sont impact sur la qualité du logiciel? Enfin, il s'avère également important que le modèle soit facile à comprendre et à utiliser afin de permettre à n'importe quel usager de l'employer. D'ailleurs, comme le mentionne Leblanc, (1999), depuis l'introduction du AHP par Saaty durant dans les années 70, le modèle fait "l'objet d'une littérature particulièrement fournie sur ses applications et limites du modèle".

Pour débuter, nous introduirons la notion de hiérarchie, un concept qui est à la base du modèle de Saaty. La section subséquente portera sur l'échelle de mesure utilisée par

l'AHP, suivi par une discussion sur l'importance des priorités pour le AHP. Une fois le modèle introduit, nous présenterons les principales faiblesses de ce modèle ainsi que les modifications qui seront apportées à l'AHP. Pour terminer, nous discuterons de l'application du modèle AHP à notre projet.

2.3.1. La Hiérarchie d'après Saaty

Saaty (1996) décrit le monde comme étant un système complexe composé d'éléments (e.g. attributs, objectifs) qui sont tous interactifs. Par exemple, l'économie est dépendante des ressources naturelles, de la situation politique, de la position géographique, etc. Afin de mieux comprendre ce système complexe , Saaty (1996) suggère de le décomposer et de créer une hiérarchie (i.e., une structure d'arbre comprenant des racines, des branches et des nœuds) où chaque niveau est composé de ses éléments primaires. Ensuite, la relation entre les éléments de chaque niveau de la hiérarchie est déterminée et interprétée par des priorités ou des poids d'importance. Il s'avère donc, nous dit Saaty, essentiel de bien comprendre la structure hiérarchique. Malgré qu'il n'existe aucune règle spécifique à la construction d'une hiérarchie, il demeure toutefois important de considérer certains facteurs pouvant assurer une bonne représentation d'un système complexe (Saaty, 1995; Saaty, 1996).

2.3.1.1. Création des Sous-Niveaux

Les hiérarchies utilisées par Saaty pour créer le AHP sont de type fonctionnel. Ceci signifie une décomposition d'un système complexe à partir des ses éléments constituants et en fonction de leurs relations essentielles. La hiérarchie du modèle AHP peut être représentée de deux façons différentes, ce qui dépend du mandat du décideur. La première représentation hiérarchique est utilisée lorsque nous voulons connaître le classement de plusieurs alternatives. Le premier niveau, le but (e.g. élément A de la Figure 2.1), se compose d'un seul élément qui représente l'objectif global du système.

En ce qui concerne les niveaux subséquents, Saaty recommande de maximiser le nombre de critères et de sous-critères à 9 éléments homogènes. Si un niveau contient plus que 9 éléments, ces éléments doivent être subdivisés en d'autres sous-niveaux. Pour aider à la création des sous-niveaux, Saaty (1996) suggère de poser la question suivante: Est-ce que je peux comparer les éléments du sous-niveau en termes d'un ou de plusieurs éléments du niveau supérieur? En d'autres mots, est-ce que le critère Crit_B1 (Figure 2.1) peut être décrit en terme de l'alternative Crit_C1, ou encore, peut-on comparer les items Crit_C1 et Crit_B1? Il faut donc s'assurer que les éléments (e.g., Crit_C1, Crit_C2, Crit_C3 et Crit_C4) d'un même niveau soient homogènes, car un des critères importants du AHP consiste à être en mesure de comparer les éléments d'un sous-niveau avec l'élément du niveau supérieur correspondant (i.e., Crit_B1, Crit_B2, Crit_B3 et Crit_B4). Les éléments d'un niveau qui partagent les mêmes propriétés que leur élément au niveau supérieur sont considérés complets. Par contre, si les éléments supérieurs ne partagent pas les mêmes propriétés, ils sont alors identifiés comme incomplets. Le niveau des alternatives (e.g., Alt_1) reflète un groupe d'éléments dont nous désirons évaluer le classement (i.e.: l'importance) final selon les critères supérieurs.

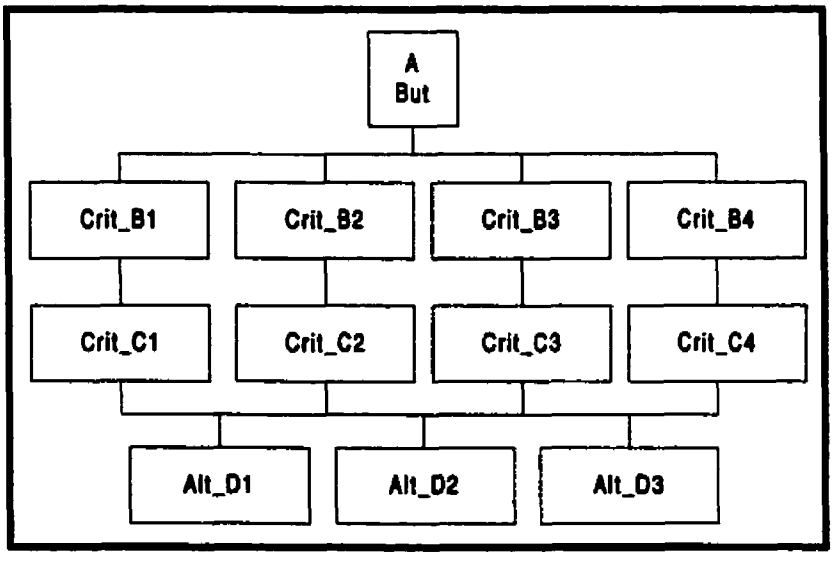


Figure 2.1: Représentation Hiérarchique pour Déterminer les Rangs

La deuxième représentation hiérarchique (Figure 2.2) est celle qui sera employée dans le cadre de notre mémoire. Elle est utilisée puisque nous désirons connaître l'impact des sous-critères sur le but. Par exemple, nous voulons connaître l'impact sur l'attribut de qualité logiciel (i.e.: le but) si le résultat de la Métrique_1 est pauvre. Ce concept sera détaillé au cours des prochaines sections.

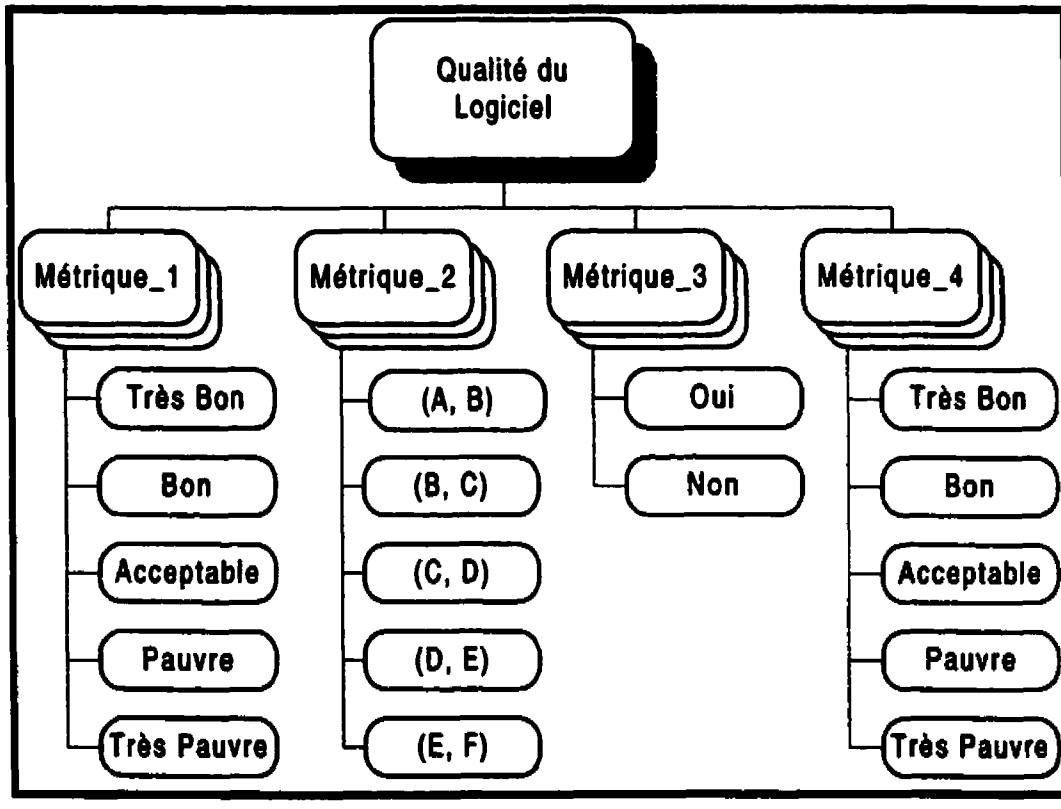


Figure 2.2: Représentation Hiérarchique pour l'Évaluation

Il apparaît important de noter que dans les deux représentations hiérarchiques, il n'est pas nécessaire de balancer l'arbre (i.e., une branche peut posséder plus de nœuds que d'autres branches) et le ou les chevauchements entre les niveaux s'avèrent également permis.

2.3.1.2. Homogénéité des Éléments

Un des rôles du AHP consiste à classifier, par ordre d'importance, tous les éléments de la hiérarchie en accordant une priorité à chaque élément. Tel qu'il sera discuté dans la section 2.3.5, la détermination des priorités se fait par des comparaisons binaires, c'est-à-dire que deux sous-critères d'un certain niveau sont comparés par rapport au critère du niveau supérieur. Il s'avère donc essentiel que tous les éléments d'un sous-niveau soient homogènes, sinon aucune comparaison ne peut être entreprise. Ainsi, les priorités sont calculées à partir de valeurs que nous avons accordées durant la comparaison binaire. Ces chiffres proviennent d'une échelle de 1 à 9 qui représente le degré d'importance d'un élément face à un autre (section 2.3.2). Par exemple, si nous comparons deux éléments identiques, la valeur 1 est accordée. Si un élément est deux fois plus important qu'un autre élément, nous donnons la valeur 2 et ainsi de suite. Il faut noter que la valeur 0 n'est pas présente dans cette échelle. En effet, une telle valeur nous indiquerait que les deux éléments sont complètement différents et ne peuvent être comparés, ils ne devraient donc pas se classer au même niveau.

En raison de nos limitations psychologiques lors des comparaisons binaires, Saaty (1996) suggère de limiter le degré d'importance à neuf. Il explique que la précision de nos comparaisons diminue lorsque le degré d'importance entre deux éléments augmente. Par exemple, il s'avère difficile de comparer la grosseur (l'attribut) d'un bleuet avec celui d'un melon d'eau, où le degré d'importance est supérieur à neuf. Nous devons faire une transition plus graduelle du bleuet au melon d'eau. Pour remédier à ce problème, nous devons dissocier ces deux items dans des groupes séparés. Même si nous sommes limités à seulement deux éléments, Saaty (1996) recommande de créer plusieurs ensembles et de rajouter d'autres éléments de grosseurs différentes. Par exemple, nous pouvons placer dans le premier groupe un bleuet, un raisin et une prune. Le deuxième groupe peut être composé d'éléments tels que: une prune, une pomme et un pamplemousse. Le troisième groupe se caractérise alors par un pamplemousse, un melon miel, et finalement, le melon

d'eau. Nous pouvons maintenant comparer la grosseur des éléments de chaque ensemble avec l'échelle de 1 à 9. Éventuellement, l'AHP nous permet de comparer la priorité du bleuet avec celle du melon d'eau. Il apparaît important de noter, lors de la transition d'un groupe à un autre, que le plus gros élément du groupe est présent dans le prochain (i.e., la prune et le pamplemousse), en devenant le plus petit élément. Ces éléments sont considérés comme des pivots (Saaty, 1996).

Cet auteur recommande donc d'utiliser des groupes homogènes dont leurs degrés d'importance sont moins que 9, afin d'assurer une plus grande stabilité des priorités face aux jugements (lors de la comparaison binaire), sinon les priorités risquent d'être inconsistantes. Il ne faut pas oublier que le but de la hiérarchie consiste à prendre un système complexe et à le simplifier, pour ce faire nous devons aussi simplifier la comparaison entre chaque élément.

2.3.1.3. Assurer une bonne représentation hiérarchique

Afin d'assurer une bonne représentation hiérarchique, il est nécessaire de tenir compte de certains facteurs importants. L'approche utilisée pour créer une hiérarchie dépend aussi du type de décision qui doit être prise (Saaty, 1996). Par exemple, sommes-nous intéressés à choisir la meilleure alternative (e.g. nous voulons choisir la voiture qui répond le mieux à nos critères) ou à mesurer l'influence de certains attributs sur le reste du système?

Un autre facteur important à considérer correspond au nombre de personnes nécessaire pour créer une hiérarchie. Ce facteur dépend du niveau de complexité du système (e.g.: une hiérarchie à deux niveaux et cinq éléments contre une hiérarchie à quinze niveaux et 80 éléments) (Saaty, 1996). La création d'une hiérarchie complexe doit être complétée par un groupe d'experts au niveau du système. Un échange d'idées, d'opinions et d'expérience facilite une meilleure représentation du système à analyser au niveau des facteurs et des alternatives. La coopération d'un groupe d'experts peut également

contribuer à clarifier les définitions des objectifs et attributs utilisés afin d'éliminer toute ambiguïté lors des comparaisons binaires.

Saaty (1996) souligne que la hiérarchie, une fois complétée, peut être modifiée de nouveau afin de répondre aux différents besoins. Par exemple, si nous ne sommes pas satisfaits d'une priorité assignée à une alternative, nous pouvons modifier la hiérarchie en éliminant ou en ajoutant certains éléments. Si l'alternative reçoit encore la plus forte priorité après ces modifications, nous savons que cette alternative doit être le meilleur choix. Nous pouvons modifier une hiérarchie tout et aussi longtemps que les éléments de chaque niveau sont homogènes et que nous pouvons les comparer avec l'élément du niveau supérieur. Malheureusement, cet exercice s'avère à l'origine de plusieurs critiques vis-à-vis l'AHP (Belton et Gear, 1983; Dyer 1990; Dyer et Ravinder, 1983; Dyer et Wendell, 1985; Leblanc et al, 1993; Schoner et Wedley, 1989) où l'ajout et le retrait d'un attribut ou d'un objectif peuvent causer une inversion des rangs. Ce phénomène fera partie d'une discussion à la section 2.6.

Afin d'aider à l'élaboration du design hiérarchique, Saaty (1996) offre quelques suggestions:

1) Identifier le but global de la hiérarchie. Quelle est la principale question?
2) Identifier les critères (i.e. attributs, objectifs) qui vont rencontrer le but global.
3) Si approprié, identifier les sous-critères pouvant décrire les critères.
4) Identifier les alternatives qui seront évaluées en termes de sous-critères.
5) Si les objectifs peuvent être déterminés en termes d'attributs, la hiérarchie est terminée.
6) Sinon, identifier et évaluer un niveau de facteurs au-dessus des attributs.
7) Continuer à insérer des niveaux jusqu'au moment où tous les niveaux sont branchés, ensuite, compléter la comparaison binaire et établir les priorités.

Le but de la hiérarchie consiste à simplifier un système complexe. Toutefois, pour qu'il soit bien représentatif, certains critères doivent être respectés. Malgré que ces sections se veulent un résumé de la méthodologie de la construction des hiérarchies proposée par Saaty (1996), nous avons jugé nécessaire d'expliquer les points les plus importants afin d'assurer une bonne représentation des priorités.

Toutefois, avant de discuter des priorités, il est primordial d'aborder, une autre fois, le sujet d'échelle de mesure. Tel que nous avons discuté au début de ce chapitre, le choix d'échelles de mesure doit se faire judicieusement. En effet, le type d'échelles que nous choisissons pour mesurer un attribut, détermine la qualité d'information que nous pouvons en retirer (Fenton et Pfleeger, 1997). À la lecture de ses travaux, il s'avère possible de constater que Saaty est conscient de l'importance du type d'échelles choisi pour les comparaisons binaires et les calculs des priorités. Pour ces raisons, nous discuterons maintenant du type d'échelles créé par Saaty pour son modèle AHP.

2.3.2. Échelle de mesure utilisée par l'AHP

D'après Saaty (1996), les psychologues intéressés au domaine cognitif reconnaissent que les gens peuvent faire deux types de comparaisons: absolue et relative. Lorsque nous comparons des alternatives à partir d'un standard basé sur nos expériences, nous faisons une comparaison absolue. Ainsi, nous utilisons la comparaison absolue lorsque nous voulons classer une alternative d'après l'intensité de chaque critère de comparaison (e.g. très bon, bon, moyen, pauvre, et très pauvre). La comparaison absolue jouera un rôle très important lors des calculs des priorités pour le modèle de qualité que nous verrons plus tard.

La comparaison relative, quant à elle, est une comparaison binaire entre deux éléments d'un même sous-niveau en se basant sur l'objectif qu'il partage au niveau supérieur. Par exemple, nous pouvons comparer la différence entre une prune et une pomme (des

éléments d'un même niveau) par rapport à leur grosseur (l'objectif ou le critère, l'élément supérieur). Le niveau d'importance peut être exprimé de façon qualitative — la pomme est faiblement plus importante que la prune — ou de façon quantitative — la pomme est deux fois plus grosse que la prune.

Saaty (1998) a créé une échelle relative de comparaison binaire où l'intensité d'importance entre deux éléments est représentée par une échelle fondamentale de valeurs absolues. Les valeurs de degrés d'importance sont réelles et appartiennent à l'intervalle (1, 9) (Tableau 2.5).

Tableau 2.5: Échelle de Comparaison Binaire (Saaty, 1984).

Degré D'importance	Définition	Explication
1	Importance égale des deux éléments	Deux éléments contribuent autant à la propriété.
3	Faible importance d'un élément par rapport à un autre	L'expérience et l'appréciation personnelles favorisant légèrement un élément par rapport à un autre.
5	Importance forte ou déterminante d'un élément par rapport à un autre.	L'expérience et l'appréciation personnelles favorisent fortement un élément par rapport à un autre.
7	Importance attestée d'un élément par rapport à un autre.	Un élément est fortement favorisé et sa dominance est attestée dans la pratique.
9	Importance extrême d'un élément par rapport à un autre.	Les preuves favorisant un élément par rapport à un autre sont aussi convaincantes que possible.
2, 4, 6, 8	Valeurs intermédiaires entre deux appréciations voisines.	Un compromis est nécessaire entre deux appréciations
Réciproques	Si l'activité *i* se voit attribuer l'un des chiffres précédents lorsqu'elle est comparée à l'activité *j*, *j* a donc la valeur réciproque lorsqu'on la compare à *i*.	

Selon Leblanc (1999), cette échelle relative nécessite aucune information sur la valeur absolue des éléments comparés. En d'autres mots, si nous disons que la pomme est deux fois plus pesante que la prune, il n'est pas nécessaire de connaître le poids de chaque élément. Si le poids de la pomme est w_i et celle de la prune est w_j, nous disons que:

$$a_{ij} = w_i/w_j = 2 = \alpha \tag{2}$$

Cette équation provient de l'échelle de ratio et identifie à quel point un élément *i* est préférable, ou non, à l'élément *j* (Leblanc, 1999). De plus, l'équation démontre que les poids de chaque élément (w_i et w_j) ne sont pas nécessaires pour la comparaison. Ceci a un énorme avantage lorsqu'un décideur doit comparer des éléments dont les attributs ne peuvent être mesurés de façon qualitative (e.g., beauté, style).

Les règles de a_{ij} sont définies de façon suivante (Saaty, 1996):

Règle 1: Si $a_{ij} = \alpha$, alors $a_{ji} = 1/\alpha$, $\alpha \neq 0$ ($a_{ji} = w_j/w_i$). Par exemple, si l'élément A est 3 fois plus important que l'élément B, alors B est 1/3 aussi important que A.

Règle 2: Si les deux éléments sont jugés comme ayant le même ordre d'importance alors $a_{ij} = 1$ et $a_{ji} = 1$.

La règle 1 nous aide à limiter le nombre de comparaisons possibles entre les éléments. Si nous comparons l'élément A avec l'élément B, il n'est pas nécessaire de comparer l'élément B avec l'élément A, puisque nous savons que la comparaison a_{ij} est le réciproque de a_{ji}. Alors si nous avons n éléments à comparer, le nombre total de comparaison binaire est de $n(n-1)/2$ au lieu de n^2. Malheureusement, si le nombre de n est large, le temps nécessaire à faire toutes ces comparaisons sera aussi considérable. Nous pourrions toutefois diminuer ce nombre de comparaisons à n si nous utilisons une échelle d'intervalle où une seule réponse par élément est nécessaire pour son l'évaluation (Leblanc, 1999) éliminant ainsi la valeur réciproque. Mais ce que nous gagnons en temps,

avec un échelle d'intervalle, nous pourrions le perdre en précision, et ce, à cause des règles statistiques permissives aux deux échelles (voir section 2.1 et 2.3.4).

Il est aussi important de noter que l'échelle d'importance du Tableau 2.5 assume que la ou les personnes faisant la comparaison binaire entre deux éléments sont familières avec ces éléments et leurs attributs. Par exemple, nous voulons comparer deux villes, Montréal et Londres, par rapport à leur distance à une troisième ville, Istanbul. Si nous n'avons aucune idée où est situé Istanbul, il n'existe aucune échelle pouvant nous être utile.

2.3.3. L'Échelle est de Type Absolu.

Roberts (1979) explique que l'échelle de type absolu est le type d'échelle le plus « pure » puisque la seule transformation admissible est :

$$\phi(x) = x \tag{3}$$

L'échelle absolue est un ensemble de valeurs qui sont invariantes sous une "*identity transformation.*" En d'autres mots, nous ne pouvons manipuler ces valeurs de façon à ce qu'elles représentent la même chose. Par exemple, l'échelle de ratio est invariante sous la transformation de similarité. Ainsi, nous pouvons multiplier un ratio par une constante positive et obtenir le même ratio (e.g., transformation de centimètre en pouce). Avec l'échelle absolue, il n'y a qu'une seule façon de mesurer les choses. Lorsque nous comparons deux objets ensemble et nous jugeons que l'objet A est 3 fois l'objet B, l'objet A ne sera jamais 4 fois B, ou 10 fois B, ou 100 fois B. La valeur 3 ne peut être transformée ou manipulée de façon à garder la même signification, la valeur 3 est donc absolue. Si nous prenons le ratio de deux chiffres, par exemple on divise six kg de bananes par trois kg de bananes la réponse est 2. La valeur 2 est absolue, car elle ne peut être remplacée de façon à dire la même chose (Saaty, 1996).

En se basant sur le Tableau 2.1, on se rappelle que la transformation admissible pour l'échelle de type ratio est (a > 0):

$$M' = aM \tag{4}$$

$$\frac{M'}{M} = a \tag{5}$$

La valeur "*a*" représente donc une valeur absolue du ratio (a > 0):

2.3.4. L'Échelle de Ratio et la Normalisation

Tel que nous venons de voir, les valeurs de comparaison binaire du Tableau 2.5 proviennent d'une échelle absolue (Saaty, 1996). Cette caractéristique est essentielle pour l'AHP, car toutes les priorités doivent éventuellement être normalisées et la normalisation de toutes autres échelles inférieures à celui de ratio (e.g. ordinale et intervalle), n'est ni valable, ni significative (Saaty, 1996). Une normalisation représente la division de chaque valeur dans un ensemble, par la somme totale de cet ensemble. Ce calcul est attrayant car il crée un nouvel ensemble de chiffres situé dans l'intervalle [0,1] dont la structure ressemble à des probabilités. De plus, Saaty est d'avis que la normalisation est indépendante de l'origine de l'échelle de mesure (e.g. ordinale, intervalle, ratio), c'est à dire qu'il est possible de normaliser des valeurs provenant de ces trois échelles de mesure (voir Tableau 2.6).

On peut être tenté d'appliquer ces valeurs normalisées de type ordinal et intervalle à des statistiques au même titre que les valeurs de type ratio. Toutefois, tel que présenté à la section 2.1, la division n'est pas une opération mathématique valable pour les échelles de type ordinal et intervalle. Donc, la signification de la normalisation peut être douteuse lorsque nous utilisons une échelle inférieure à celle de ratio.

Tableau 2.6: Normalisation d'ensembles provenant de différents types d'échelles.

Type d'échelle	Ensemble	Valeur Normalisée
Ordinal	1, 3, 5, 7	0.06, 0.19, 0.31, 0.44
Intervalle	3, 6, 12, 14	0.09, 0.17, 0.34, 0.40
Ratio	4, 8, 14, 20	0.09, 0.17, 0.30, 0.43

Voici un autre exemple, prenons les ensembles *M* et *M'* du Tableau 2.3 et normalisons les valeurs. Par exemple, chaque valeur (e.g., A) est divisée par la somme de leur ensemble (e.g., A+B+C+D). Les résultats du Tableau 2.7 affirment que seule l'échelle de ratio préserve la nature des valeurs initiales.

Tableau 2.7: Normalisation des d'ensembles *M* et *M'* du Tableau 2.6.

Type d'échelle	Normalisation des Ensembles	
	Ensembles *M*	Ensembles *M'*
Intervalle	{5/50, 10/50, 15/50, 20/50} ou {0.1, 0.2, 0.3, 0.4}	{7/40, 9/40, 11/40, 13/40} ou {0.18, 0.23, 0.28, 0.33 }
Ratio	{1/13, 2/13, 4/13, 6/13}	{1/13, 2/13, 4/13, 6/13}

Saaty (1996) souligne que le résultat suite aux opérations de calculs des priorités (section 2.3.5) à partir des chiffres provenant d'échelles ordinales ou d'intervalles normalisées n'a aucune signification, puisqu'il serait facile de produire des résultats différents avec un nouvel ensemble de chiffres normalisés provenant d'une différente échelle ordinale ou intervalle. En somme, une échelle de ratio s'avère unique et maintient sa proportionnalité

avant et après la normalisation. De plus, Saaty reconnaît qu'il est dangereux d'utiliser une échelle inférieure à celui de ratio durant le calcul des priorités. Il faut donc être judicieux dans notre choix d'échelle de mesure.

Pour terminer cette section, il s'avère important de noter à nouveau que l'échelle d'importance du assume que la ou les personnes faisant la comparaison binaire entre deux éléments sont familières avec ces éléments et leurs attributs.

2.3.5. Les priorités

Une fois que nous avons simplifié un système complexe à l'aide d'une hiérarchie, les prochaines étapes de l'AHP consistent à faire des comparaisons binaires (section 2.3.2) afin de calculer les priorités de tous les éléments. Les priorités reflètent l'ordre d'importance des éléments à l'intérieur de la hiérarchie.

2.3.5.1. Création d'une matrice de priorité

Afin de nous aider à faire les comparaisons binaires, nous utiliserons une matrice. Prenons par exemple un sous-niveau de quatre éléments, A_1, A_2, A_3 et A_4, avec leur attribut commun, C, lequel est situé au niveau supérieur :

C	A_1'	A_2'	A_3'	A_4'
A_1				
A_2				
A_3				
A_4				

où $A_i = A_i'$ (6)

Pour effectuer la comparaison binaire, l'échelle d'importance créée par Saaty sera employée (Tableau 2.5). Premièrement, nous allons comparer le premier élément, A_1, que l'on retrouve dans la colonne, avec tous les autres éléments situés dans la rangée, en fonction de l'attribut C, i.e.:

A_1', A_2', A_3' et A_4' (7)

La question suivante peut être posée pour aider à faire les comparaisons: Par rapport à l'attribut C, quelle est l'importance de l'élément A_x face à l'élément A_y'? La question utilisée pour faire les comparaisons s'avère très importante car elle doit refléter la bonne relation entre les deux éléments d'un certain niveau et l'élément au niveau supérieur. La première comparaison correspond à l'ensemble (A_1, A_1') et contient deux éléments identiques. D'après la *règle 2* (section 2.3.2), la valeur d'importance pour cet ensemble est de 1. Alors tous les ensembles qui comparent deux éléments identiques ([A_a, A_b'] , ou a = b) ont une valeur d'importance égale à 1, donc toutes les valeurs diagonales de la matrice sont égales à 1 :

C	A_1'	A_2'	A_3'	A_4'	
A_1	1				
A_2		1			où $A_i = A_i'$ (8)
A_3			1		
A_4				1	

Ensuite, nous comparons l'élément, A_1, avec tous les autres éléments A_x' (où x = {2,3,4}) et accordons à chaque ensemble une valeur d'importance (Karlsson & Ryan, 1997):

C	A_1'	A_2'	A_3'	A_4'	
A_1	1	1/3	2	4	
A_2		1			où $A_i = A_i'$ (9)
A_3			1		
A_4				1	

La matrice nous indique que d'après l'attribut C, l'élément A_1 a une "forte importance" par rapport à l'élément A_4' et une "faible importance" par rapport à l'élément A_2' (Tableau 2.5).

Maintenant que nous avons terminé avec l'élément A_1, nous continuons nos comparaisons avec le deuxième élément de la colonne, soit A_2. Nous remarquons très rapidement que le premier ensemble de comparaisons (A_2, A_1') est l'inverse de l'ensemble (A_1, A_2') (section 2.3.2). Dans une situation idéale, si une pierre, W_1, est deux (2) fois plus pesante que la pierre, W_2', nous pourrions aussi dire que la pierre W_2' est deux fois moins (1/2) lourde que la pierre, W_1. Alors, le degré d'importance accordé à l'ensemble (A_2, A_1') est le réciproque du degré d'importance pour l'ensemble (A_1, A_2'). Nous continuons les comparaisons de l'élément A_2 avec tous les éléments A_x' (où x = {3,4}). Les ensembles de comparaisons peuvent donc être résumés comme suit:

$$(A_i, A_j') \qquad 0 < (j \text{ et } i) < 5 \tag{10}$$

À cause de la relation de réciprocité, nous avons seulement à compléter la moitié de la matrice :

C	A_1'	A_2'	A_3'	A_4'
A_1	1	1/3	2	4
A_2	3	1	5	3
A_3	1/2	1/5	1	1/3
A_4	1/4	1/3	3	1

où $A_i = A_i'$ (11)

2.3.5.2. Calcul des priorités pour un critère

Maintenant, nous sommes prêts à calculer le poids relatif de chacun des n éléments. La représentation formelle de la matrice de comparaisons par paires peut s'exprimer comme suit :

$$A = \begin{bmatrix} 1 & a_{12} & \cdots & a_{1n} \\ 1/a_{12} & 1 & \cdots & a_{2n} \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ \cdot & \cdot & \cdot & \\ 1/a_{1n} & 1/a_{2n} & \cdots & 1 \end{bmatrix} \quad a_{ij} \ (i, j = 1, 2, \ldots, n) \tag{12}$$

À l'aide de la matrice *A*, nous pouvons maintenant calculer le poids relatif de chaque élément A_x. Pour résoudre ce problème, Saaty (1996) considère une situation idéale où toutes les comparaisons s'avèrent parfaitement cohérentes. Par exemple, prenons deux pierres ayant des poids différents, w_i et w_j, le degré d'importance de chaque élément par rapport à leur poids est représenté par l'équation suivante :

$$w_i / w_j = a_{ij} \quad (\text{où } i, j = 1,2,\ldots, n) \tag{13}$$

La matrice A est représentée de la façon suivante:

$$A = \begin{bmatrix} w_1/w_1 & w_1/w_2 \ldots & w_1/w_n \\ w_2/w_1 & w_2/w_2 \ldots & w_2/w_n \\ . & . & . \\ . & . & . \\ w_n/w_1 & w_n/w_2 \ldots & w_n/w_n \end{bmatrix} \tag{14}$$

Considérons la rangée *i* qui contient *n* valeurs:

$$a_{i1}, a_{i2}, \ldots, a_{ij}, \ldots, a_{in} = w_i/w_1, w_i/w_2, \ldots w_i/w_j, \ldots w_i/w_n \tag{15}$$

Donc, dans une situation idéale, si nous multiplions la première valeur de cette rangée avec w_1 et la deuxième valeur avec w_2, et ainsi de suite, nous obtenons :

$$(w_i/w_1)\, w_1 = w_i, (w_i/w_2)\, w_2 = w_i, \ldots, (w_i/w_j)\, w_j = w_i, (w_i/w_n)\, w_n = w_i \tag{16}$$

Le résultat est maintenant une rangée de valeurs identiques:

$$w_i, w_i, \ldots, w_i \tag{17}$$

La représentation matricielle de ce que nous venons de faire est:

$$Aw = \begin{bmatrix} w_1/w_1 & w_1/w_2 \ldots & w_1/w_n \\ . & . & . \\ w_i/w_1 & w_i/w_2 \ldots & w_i/w_n w_i \\ . & . & . \\ w_n/w_1 & w_n/w_2 \ldots & w_n/w_n \end{bmatrix} \begin{bmatrix} w_1 \\ . \\ . \\ . \\ w_n \end{bmatrix} \qquad (18)$$

Jusqu'à maintenant, nous avons considéré une situation idéale, mais pour les cas généraux, Saaty (1996) démontre que w_i peut être représenté par l'équation :

$$w_i = \frac{1}{\lambda \text{ max}} \sum_{j=1}^{n} a_{ij} w_j \qquad i = 1, 2, ..., n \qquad (19)$$

Cette équation décrit un système linéaire qui peut aussi être énoncé de la façon suivante :

$$A\, w = \lambda_{max}\, w \qquad (20)$$

La constante λ_{max} désigne la valeur propre de la matrice A et est utilisée pour la normalisation. La valeur w, quant à elle, correspond au vecteur propre. Ce vecteur propre devient le vecteur de priorité des éléments de la matrice A. Pour une matrice positive qui est réciproque et dont la diagonale est égale à 1, Saaty (1996) prouve que :

$$\lambda_{max} \geq \max \Sigma(a_{ij}) \geq n \qquad (21)$$

Prenons par exemple la matrice de priorité suivante dont trois projets sont comparés au critère C:

C	A1'	A2'	A3'
A1	1	3	7
A2	1/3	1	5
A3	1/7	1/5	1
Σ	1.48	4.20	13.00

(22)

La dernière rangée représente la sommation des comparaisons binaires de chaque colonne. Maintenant toutes les valeurs sont normalisées par la sommation de leur colonne respective:

C	A1'	A2'	A3'
A1	0.68	0.71	0.54
A2	0.23	0.24	0.38
A3	0.10	0.05	0.08
Σ	1.00	1.00	1.00

(23)

La prochaine étape consiste à calculer le vecteur propre de la matrice. Pour ce faire, nous additionnons les valeurs de chaque rangée et ensuite nous divisons ce résultat par le nombre d'éléments ($n = 3$):

C	A1'	A2'	A3'		
A1	0.68	0.71	0.54		0.64
A2	0.23	0.24	0.38	$w=$	0.28
A3	0.10	0.05	0.08		0.07
Σ	1.00	1.00	1.00		1.00

(24)

Le vecteur propre, w, indique l'importance de chaque élément par rapport à l'attribut C et démontre la distribution d'un élément vis-à-vis un autre à l'intérieur de l'ensemble. Nous pouvons donc dire que l'élément A2 (w_2) est deux fois moins important que l'élément A1 (w_1). La prochaine étape consiste à calculer la valeur propre, λ_{max}. Pour ce faire, nous

calculons la valeur $\lambda i'max$ dans chacune des rangées i du produit matriciel $Bw = \lambda max$ (Leblanc, 1999):

$$\begin{vmatrix} 1 & 3 & 7 \\ 1/3 & 1 & 5 \\ 1/7 & 1/5 & 1 \end{vmatrix} \times \begin{vmatrix} 3.67 \\ 2.11 \\ 0.45 \end{vmatrix} = \begin{vmatrix} 13.13 \\ 5.57 \\ 1.39 \end{vmatrix} \quad (25)$$

Ensuite, nous prenons pour λmax la moyenne des n valeurs $\lambda i'max$ ainsi obtenues, i.e. (Leblanc, 1999):

$$\lambda \text{ max} = \frac{\sum_{t=1,\ldots,n} \lambda'_{t \text{ max}}}{n} \quad (26)$$

Les 3 solutions de $\lambda i'max$ sont:

$$[13.13/3.67, 5.57/2.11, 1.39/0.45] = [3.58, 2.64, 3.09] \quad (27)$$

La valeur moyenne estimée de λmax est:

$$\lambda max = (3.58 + 2.64 + 3.09)/3 = 3.10 \quad (28)$$

Nous remarquons que la valeur de λmax est très proche du nombre total d'éléments, $n = 3$. Mais quelle serait la signification si la valeur de λmax s'avérait supérieure à n ? Ce point sera abordé à la section 2.3.6.

2.3.5.3. Calcul des priorités en présence de plusieurs sous-niveaux

Nous allons maintenant considérer une hiérarchie possédant plusieurs sous-niveaux. La Figure 2.3 représente une hiérarchie contenant trois projets (A, B et C), trois critères (C1, C2 et C3) et un objectif (Q).

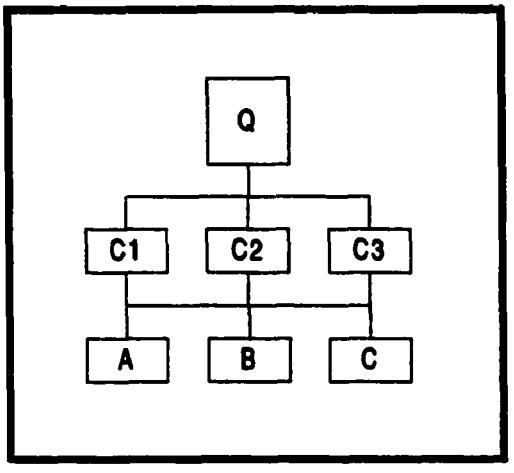


Figure 2.3: Représentation Hiérarchique de plusieurs critères

Premièrement, afin de calculer les poids de chacun des critères (C1, C2 et C3) nous devons procéder à des comparaisons binaires en rapport avec l'objectif Q. Cette exercice est similaire à ce que l'on a traité à la section 2.3.5.1. Afin de faciliter le travail, nous allons accorder des poids égaux aux trois critères (i.e. w = 1/3). Ensuite, nous comparerons chaque projet, A, B et C, selon chacun des critères. La matrice de priorité est représentée comme suit:

C1	A	B	C
A	1	1/4	1/2
B	4	1	3
C	2	1/3	1
Σ	7	2.58	4.5

C2	A	B	C
A	1	1/4	1/5
B	4	1	1/2
C	5	2	1
Σ	10	3.25	1.7

C3	A	B	C
A	1	3	1/3
B	1/3	1	1
C	3	1	1
Σ	4.33	5	2.33

(29)

Par la suite, le vecteur propre (w) de chaque critère est calculé:

C1	A	B	C	w
A	0.14	0.16	0.11	0.14
B	0.57	0.63	0.67	0.62
C	0.29	0.21	0.22	0.24

C2	A	B	C	w
A	0.1	0.08	0.12	0.10
B	0.4	0.31	0.29	0.33
C	0.5	0.62	0.59	0.57

C3	A	B	C	w
A	0.23	0.6	0.14	0.32
B	0.08	0.2	0.43	0.24
C	0.69	0.2	0.43	0.44

(30)

L'étape finale consiste à évaluer le poids final de chaque projet par rapport à l'objectif Q. Ainsi, le poids de chaque projet est multiplié avec le poids des critères respectifs (i.e. 1/3) et sont ensuite additionnés ensemble:

$$w(\text{Projet A}) = (0.14*0.33) + (0.10*0.33) + (0.32*0.33) = 0.1848 \quad (31)$$

$$w(\text{Projet B}) = (0.62*0.33) + (0.33*0.33) + (0.24*0.33) = 0.3927 \quad (32)$$

$$w(\text{Projet C}) = (0.24*0.33) + (0.57*0.33) + (0.44*0.33) = 0.4125 \quad (33)$$

Le résultat final démontre que, par rapport à l'objectif Q, le projet C domine légèrement sur B et que B domine sur A: C(0.41)>B(0.39)>A(0.18). Les priorités confirment également que les projets C et B sont deux fois plus importants que le projet A.

2.3.6. Cohérence

Une matrice de comparaison est généralement élaborée à partir de l'expertise professionnelle des usagers. À l'occasion, il arrive que le degré d'importance accordé durant la comparaison binaire soit incohérent. Par exemple, si l'élément A présente un degré d'intensité plus fort que B et que B a un degré d'intensité plus fort que C, il serait donc logique de dire que l'élément A possède un degré d'intensité plus fort que C. Si par mégarde ou par un moment d'inattention, nous indiquons que l'élément A présente un degré d'intensité moins fort que C, la matrice risque d'être incohérente. Un λ_{max} supérieur au nombre total d'éléments, n, s'avère un des premiers indicatifs d'une matrice incohérente (Saaty, 1996). Donc, plus que la valeur de λ_{max} se rapproche de n, plus on peut estimer nos résultats comme cohérents. Saaty (1996) propose l'utilisation d'un Index de Cohérence (IC) afin de calculer la déviation de la cohérence :

$$IC = (\lambda_{max} - n) / (n - 1) \quad (34)$$

Lorsqu'une matrice dont tous les jugements de comparaisons binaires sont faits de façon aléatoire (en prenant une échelle de 1/9, 1/8, 1/7, ...1, 2, 3, ..., 9), s'avère réciproque, l'index de cohérence moyenne peut être représenté par l'Index Randomisé (IR). Le Tableau 2.8 présente les valeurs de l'IR pour les matrices de grandeur n où ($0<n<10$).

Tableau 2.8 Valeurs de l'Index Randomisé pour Différentes Matrices.

Grandeurs de la matrices	1	2	3	4	5	6	7	8	9	10
Index Randomisé	0.00	0.00	0.52	0.89	1. 11	1.25	1.35	1.40	1.45	1.49

Si nous comparons l'IC de notre matrice contre l'IC d'une matrice dont toutes les comparaisons binaires sont incohérentes, nous aurions le Ratio de Cohérence:

$$RC = IC / IR \tag{35}$$

D'après Saaty (1996), un RC inférieur à 0.10 (ou 10%) est considéré acceptable, c'est à dire que les jugements des degrés d'importance s'avèrent cohérents. Il faut toutefois faire preuve de vigilance, puisqu'une grande cohérence ne signifie pas nécessairement une grande précision (i.e., deux matrices différentes peuvent avoir la même cohérence sans pour autant avoir le même vecteur de priorité). Lorsqu'un RC est élevé, la matrice de comparaison devrait être réétudiée afin d'apporter des changements de degrés d'intensité appropriés. Lorsque le RC persiste à demeurer élevé, il est recommandé que la ou les personnes ayant complété la comparaison binaire puissent mieux comprendre la signification des éléments et réexaminer la structure de la hiérarchie.

Afin de récapituler les sections précédentes, l'algorithme développé par Leblanc et collaborateurs (1993) est présenté à la Figure 2.4. Cet algorithme se veut un résumé de la méthode AHP pour les mesures relatives (i.e. comparaisons par pairs).

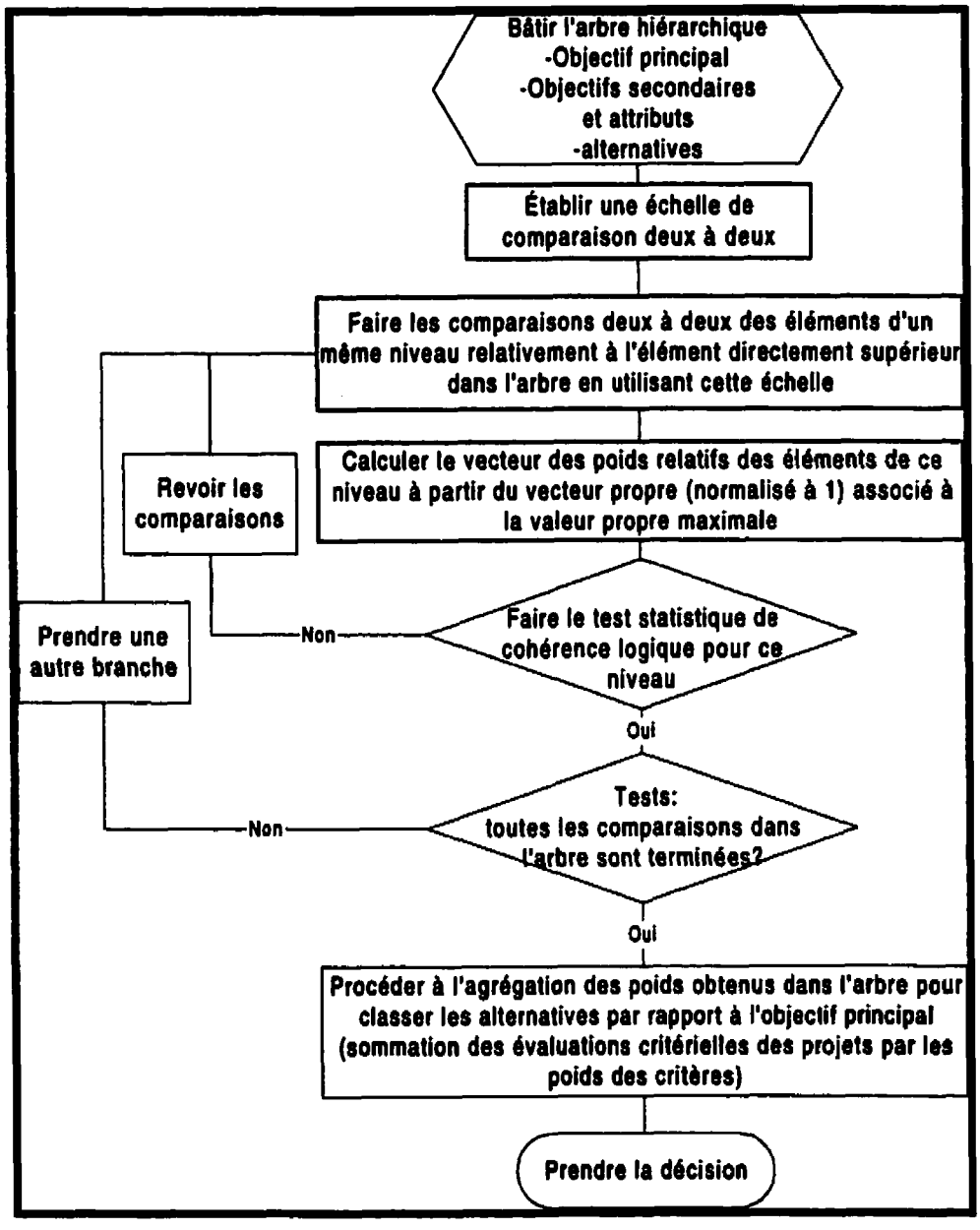


Figure 2.4 : Algorithme du AHP pour des mesures relatives (Leblanc et al., 1993)

2.4. La mesure comparative absolue

Les sections précédentes avaient pour but d'introduire les bases fondamentales de la méthode AHP afin de classer par ordre d'importance tous les critères à l'intérieur d'une hiérarchie. Maintenant, nous allons utiliser ces connaissances afin d'évaluer l'impact des

critères sur le but global. Cette caractéristique importante sera employée afin de répondre à un des objectifs de ce projet: créer un modèle qui peut évaluer l'impact des résultats de métriques sur les attributs de la qualité logicielle.

Tel que nous l'avons mentionné à la section 2.3.2, il existe, d'après certains psychologues cognitifs (Saaty, 1996), deux types de mesures comparatives: relative et absolue. La comparaison relative reflète une comparaison binaire faite à partir d'un attribut commun aux deux éléments. La comparaison absolue, d'autre part, évalue et classe par ordre d'importance, les critères à partir des standards basés sur notre expertise. Par exemple, les résultats des métriques peuvent être représentés par une échelle d'intensité ou de standards (e.g., excellent, très bon, bon, moyen, moins bon, faible et très faible).

Les priorités sont attribuées à chaque intensité ou standard en faisant des comparaisons binaires. Ces valeurs sont ensuite normalisées en divisant par le plus grand élément de l'ensemble. C'est-à-dire, le plus grand élément d'intensité se voit maintenant attribuer une valeur de 1 et tous les autres éléments possèdent des valeurs inférieures à 1. Saaty (1995) réfère à cette normalisation comme étant un "mode idéal."

Lorsque la valeur du critère d'intensité est choisie (e.g., par le résultat d'une métrique), la priorité (en mode idéal) de cette intensité multiplie la priorité normalisée (i.e. le total des priorités correspond à un) de l'attribut en question. Donc, si le résultat des métriques correspond à l'intensité la plus élevée, la priorité idéale du critère d'intensité est 1 et la priorité de l'attribut ne se verra pas influencée. Toutefois, si la priorité de la valeur du critère d'intensité est inférieure à 1, le total des priorités pour ce niveau sera moins de 1. Comme nous le verrons dans la prochaine section, une telle situation aura un impact direct sur le but global de la hiérarchie.

2.5. L'impact de l'intensité d'un critère sur le but

Pour visualiser l'impact des intensités sur le but, nous allons créer une hiérarchie sur la fiabilité du logiciel, laquelle dépend de trois métriques. Les métriques que nous avons choisies proviennent du standard sur la fiabilité logicielle IEEE 982.1-1998 (IEEE, 1998):

1) **Densité de défaut** (DD): Fenton et Pfleeger (1997) décrivent cette métrique comme étant une mesure standard pour évaluer la qualité logicielle. Le densité de défaut est calculée en divisant le nombre de défauts connus par le nombre de lignes de code. Malgré la simplicité de l'équation, il demeure que la signification du résultat peut s'avérer ambiguë. Un manque de consensus sur ce qu'est un défaut et sur la façon de compter le nombre de lignes de code explique cette ambiguïté. Toutefois, le DD est considéré comme étant une mesure indispensable dans un programme de métrique. Le DD utilise une échelle de ratio.

2) "***Mean Time to Failure***" (MTTF): La fonction MTTF représente le temps anticipé du prochain défaut observé. Selon le IEEE 982.1-1998 (IEEE, 1998), le MTTF "is the basic parameter required by most software reliability models (p. 28)." Cette métrique est basée sur une échelle ratio.

3) **Complexité cyclomatique** (CC): Cette mesure, s'appuyant sur une échelle absolue, a initialement été proposée par McCabe en 1976 afin de représenter la *complexité* d'un algorithme logiciel. La complexité cyclomatique reflète le nombre de chemins indépendants à travers le logiciel. Une valeur de CC excédant la valeur 10, signale au programmeur que le logiciel pourrait nécessiter plus de temps à tester et à réparer (Fenton et Pfleeger, 1997).

Maintenant que nous avons défini tous les éléments de la hiérarchie (Figure 2.5), nous effectuerons des comparaisons par paires des éléments afin d'établir leur priorité.

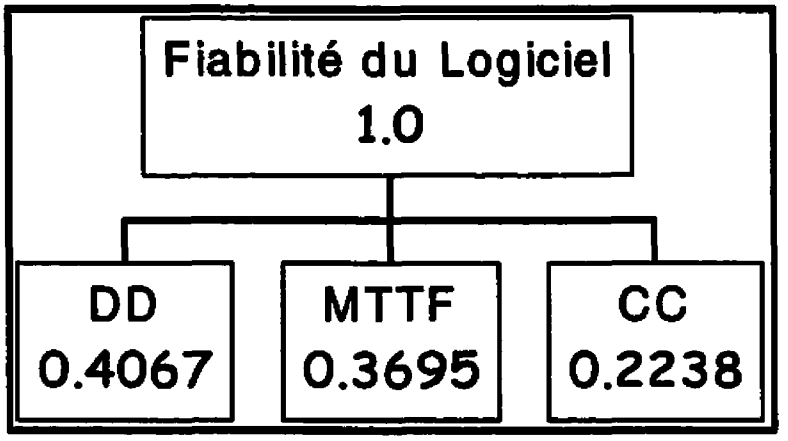


Figure 2.5: Hiérarchie pour la fiabilité logicielle

L'objectif de cet exercice est de nous permettre d'évaluer l'impact des résultats de chacune des métriques sur la fiabilité du logiciel. Par conséquent, nous allons créer un autre sous-niveau composé des intensités (e.g., idéal, très bon, moyen, faible et très faible) qui reflète les résultats de chaque métrique. La plage des résultats pour les intensités de chaque métrique se retrouve au Tableau 2.9.

La prochaine étape consiste à calculer les priorités de ces intensités en effectuant des comparaisons binaires. Le Tableau 2.10 démontre la matrice des priorités d'intensités de la métrique de défaut de densité (DD).

D'après Saaty (1996), les priorités d'intensité sont ensuite "pesées" (multipliées) avec la priorité de leur critère respectif (Tableau 2.11), puis le mode d'intensité idéal est calculé en divisant les nouvelles priorités par la plus grande valeur de l'ensemble (e.g., 0170).

Tableau 2.9: Valeurs Accordées aux Éléments d'Intensités

	DD	MTTF	CC
Idéal	< 0.1	1600 à 5000	0 à 3
Très Bon	0.1 à 0.5	160 à 1600	4 à 5
Moyen	0.5 à 1	50 à 160	6 à 7
Faible	1 à 3	5 à 50	8 à 9
Très Faible	> 3	1.6 à 5	>10

Tableau 2.10: Comparaisons d'Intensités pour le DD

	Idéal	Très Bon	Moyen	Faible	Très Faible	Priorités
Idéal	1.0	2.0	3.0	4.0	5.0	0.419
Très Bon	1/2	1.0	2.0	3.0	4.0	0.263
Moyen	1/3	1/2	1.0	2.0	3.0	0.160
Faible	1/4	1/3	1/2	1.0	2.0	0.097
Très Faible	1/5	1/4	1/3	1/2	1.0	0.062

R.C.=0.015

Tableau 2.11: Mode d'intensité idéal pour le DD

	Priorité Pesée par 0.4067	Mode Idéal
Idéal	0.170	1.000
Très Bon	0.107	0.629
Moyen	0.065	0.382
Faible	0.039	0.229
Très Faible	0.025	0.147

La Figure 2.6 représente la nouvelle hiérarchie pour la fiabilité du logiciel incluant le niveau des intensités de chaque métrique, les valeurs des priorités (entre parenthèses) ainsi que les priorités en mode idéal.

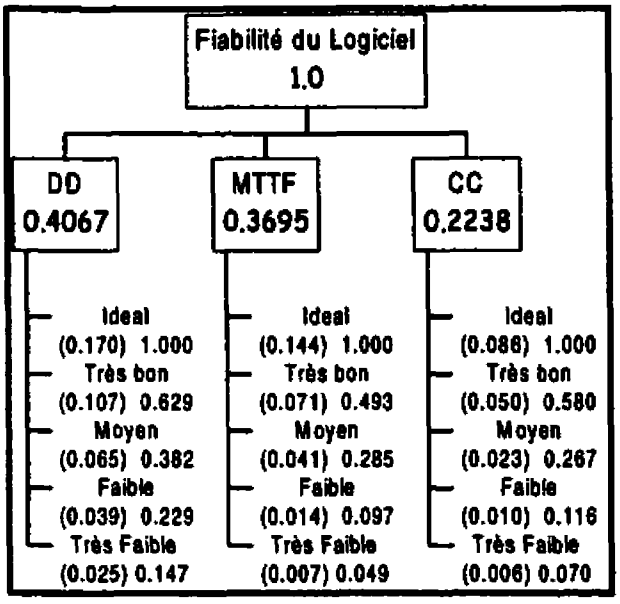


Figure 2.6: Hiérarchie incluant les niveaux d'intensités pour la fiabilité logicielle

Cette hiérarchie peut maintenant être utilisée dans un contexte d'évaluation, ce qui nous permettra de visualiser l'influence des intensités sur le but global de la hiérarchie (i.e.: la fiabilité). Par exemple, si nous considérons les deux premiers niveaux de la hiérarchie (Figure 2.6), le but est égal à 1 puisqu'il s'avère dépendant des priorités des métriques sous-jacentes et que l'addition des ces priorités normalisées égale à 1 ou 100%:

$$1 = 0.4067 + 0.3695 + 0.2238. \tag{36}$$

Donc, une fiabilité de 100% présume que les priorités des métriques demeurent inchangées. De la même façon que le but dépend du niveau sous-jacent, les métriques sont aussi dépendantes du dernier niveau d'intensité. Toutefois, contrairement au but, cette dépendance relève directement de la valeur de l'intensité choisie par le décideur, puisque la priorité en mode idéal de cette intensité multiplie la priorité de la métrique

correspondante. Ainsi, si les intensités des résultats de métriques sont toutes catégorisées comme "Idéal," (i.e. la priorité idéale est de 1), les priorités des métriques demeurent inchangées et la valeur de la priorité de fiabilité sera égale à 1:

$$\begin{aligned} \text{Fiabilité du logiciel} &= \Sigma\,[\text{priorité (X)} * \text{priorité idéale de l'intensité (X)}] \\ &= 0.4067*1\ (\text{DD}) + 0.3695*1\ (\text{MTTF}) + 0.2238*1\ (\text{CC}) \\ &= 1 \end{aligned} \qquad (37)$$

Nous pouvons alors nous questionner quant à l'impact sur le but si une ou plusieurs intensités étaient inférieures à 1. Le Tableau 2.12 propose trois scénarios comportant différents résultats de métriques. L'impact sur la fiabilité associé aux différentes intensités est illustré dans le Tableau 2.13.

Tableau 2.12: Trois scénarios comportant des résultats de métriques différents.

	DD	MTTF	CC
Test 1	< 0.1 (Idéal)	1600 (Idéal)	3 (Idéal)
Test 2	0.3 (Très Bon)	50 (Moyen)	12 (Très Faible)
Test 3	0.4 (Très Bon)	160 (Très Bon)	7 (Moyen)

Tableau 2.13: Impact des intensités sur la fiabilité

	DD (0.4067)	**MTTF (0.3695)**	**CC (0.2238)**	**Total***
Test 1	Idéal (1.000)	Idéal (1.000)	Idéal (1.000)	**1.000**
Test 2	Très bon (0.629)	Moyen (0.285)	Très faible (0.070)	**0.376**
Test 3	Très bon (0.629)	Très bon (0.493)	Moyen (0.267)	**0.496**

*Total = $A\,w^t$ = (Test X) Transposé (Vecteur de Priorité des Critères)

À la lumière de ce dernier tableau, il apparaît évident que lorsque l'intensité de la métrique est au-dessous de la valeur idéale, le but de la hiérarchie peut devenir fortement affecté. La signification d'une diminution de la priorité du but indique au décideur que le logiciel n'a pas répondu aux standards. De plus, l'impact de l'intensité sur le but dépend également de la priorité même de la métrique. Par exemple, une forte diminution de la priorité du DD affectera d'avantage le but, qu'une forte diminution de la CC, car la priorité du DD représente 40% du but total contrairement à 22% pour la CC. Ce phénomène semble donc pouvoir assister le décideur à justifier la quantité d'efforts ou de coûts requis à investir afin de promouvoir la valeur du but global. En d'autres mots, le décideur devrait être tenté d'investir davantage dans l'amélioration du résultat de la métrique DD (possédant la plus haute priorité) que de la métrique CC (représentant la plus basse priorité).

2.5.1. Signification des résultats

Tel que nous venons de voir, l'influence des intensités sur la qualité est clairement dominée par la priorité des critères. Maintenant que nous savons que la priorité du but de la hiérarchie peut diminuer, il s'agit de trouver la signification lorsqu'elle est inférieure à 1. Un but possédant une priorité optimale de 1 ou 100% signifie que toutes les intensités répondent aux standards que nous lui avons imposés (i.e., intensité idéale = 1). Alors, considérant la nature de l'échelle utilisée par le AHP, si le but diminue à 0.5 ou 50%, nous pourrions dire que la fiabilité du logiciel s'avère deux fois moins importante que la fiabilité optimale. Si la fiabilité du logiciel s'évalue à 0.25 ou 25%, nous pourrions dire que cette fiabilité est quatre fois moins importante qu'une fiabilité idéale, et ainsi de suite. Puisque le pourcentage dépend des éléments de la hiérarchie ainsi que des priorités qui leur ont été imposées par le décideurs, la valeur du but global reflète alors les intensités du système idéal tel que décrit initialement par le décideur. Par conséquent, il semble que ce pourcentage ne peut être généralisable à l'évaluation de d'autres systèmes.

Donc, ce type d'analyse s'avère un indicatif de la fiabilité logicielle mesurée. Grâce aux concepts de priorité, il nous semble possible d'augmenter la fiabilité logicielle en se concentrant à améliorer la valeur des métriques dont la priorité des critères est élevée. Ainsi, les autres critères, ayant une priorité moins importante, pourraient être négligés ou discutés plus tard.

Il demeure important de noter que les standards ainsi que les priorités des critères doivent être calculés à partir des comparaisons binaires complétées par un groupe d'experts dans leur domaine (e.g., utilisateurs, réalisateurs et maîtres d'œuvres). Dans ce cas, nous pouvons présumer que les priorités reflètent l'opinion de tous les experts du domaine. D'autre part, si les comparaisons binaires avaient été complétées par seulement un individu, les priorités seraient probablement différentes de ceux du groupe et la généralisation à l'ensemble de la population deviendrait encore moins probable.

2.6. Controverses associées au modèle AHP

Malgré que le AHP possède plusieurs caractéristiques avantageuses (e.g. utilisation d'une échelle de ratio, indice de cohérence, évaluation de l'objectif selon l'intensité des critères), il demeure que cette méthode possède certaines faiblesses qui ne peuvent être laissées sous silence. Par exemple, certains auteurs ont critiqué l'échelle utilisée par le modèle AHP (Dyer 1990; Dyer et Ravinder, 1983; Dyer et Wendell, 1985). Ces critiques portent, entre autres, sur la limite de 9 proposée par Saaty et sur le fait qu'il n'existe pas de valeur "0" sur l'échelle (considérant que l'échelle est de type absolu). Examinons ces points un à un. D'après Forman (cité par Saaty, 1996), l'échelle de 9 points n'est pas nécessairement requise par les axiomes du AHP. Saaty (1996) a basé cette limite sur des études empiriques, de plus, cette échelle aide à structurer la complexité du problème en limitant le nombre d'éléments à 7±2. Toutefois, si l'élément A s'avère 9 fois plus important que B et que B est 5 fois plus important que C, à cause de la mesure de ratio, nous sommes obligés de dire que A est 45 fois plus important que C. Un tel arrangement peut causer

une élévation du ratio de cohérence (section 2.3.6), mais le décideur pourrait accepter un ratio élevé (même plus de 10%), s'il reconnaît que la cause est provoquée par des jugements extrêmes (Saaty, 1996). En ce qui concerne le second point, c'est-à-dire l'absence de la valeur "0" de l'échelle, cette valeur n'est pas vraiment absente, mais plutôt inutile. Par exemple, si deux éléments ne sont pas homogènes et ne partagent aucune relation avec le critère du niveau supérieur, la valeur de la comparaison par paire devrait être "0". Cependant, ce problème ne devrait pas survenir puisque chaque niveau de la hiérarchie comprend des éléments qui s'avèrent homogènes et la hiérarchie ne devrait pas être acceptable lorsque l'homogénéité n'est pas maintenue.

Durant les années 80 et le début des années 90, le modèle AHP fut le centre de critiques par plusieurs auteurs (Belton, & Gear, 1983; Boucher, & MacStravic, 1991; Dyer, & Ravinder, 1983; Dyer, & Wendell, 1985; Dyer, 1990a,b; Kamenetsky, 1982; Leblanc et al, 1993; Schoner, & Wedley, 1989). La nature de ces critiques sont principalement basées sur ce qui est communément appelé *l'inversion des rangs*. Leblanc et al (1993) expliquent: "par ce phénomène, certaines alternatives, lorsque rajoutées à une série d'alternatives ayant déjà été analysées et classées, provoquent une inversion des rangs relatifs de ces alternatives" (p. 3). Ce problème se présente lors de mesures relatives (section 2.3.2) et non en présence de mesures absolues (Saaty, 1996). Ainsi, durant une mesure absolue, même si nous ajoutons une alternative, il ne devrait pas y avoir une inversion de rangs parmi le groupe d'alternatives. Toutefois, si nous décidons de faire l'ajout de critères additionnels dans notre hiérarchie, nous risquons de provoquer une inversion des rangs parmi les critères, puisque leurs rangs ont été déterminés par la mesure relative. Étant donné la portée de ce problème, la prochaine sous-section y sera consacrée dans le but d'offrir une solution.

2.6.1. Inversion des rangs

Afin de clarifier et de bien comprendre le problème d'inversion des rangs, Leblanc et al. (1993) offrent l'exemple proposé initialement par Dyer (1990) utilisant les données de l'exemple original de Belton et Gear (1983).

Trois projets A, B et C sont comparés selon trois critères différents C1, C2 et C3 ayant des poids égaux. Les comparaisons par paires donnent les résultats suivants:

C1	A	B	C	*w*
A	1	1/9	1	1/11
B	9	1	9	9/11
C	1	1/9	1	1/11

C2	A	B	C	*w*
A	1	9	9	9/11
B	1/9	1	1	1/11
C	1/9	1	1	1/11

C3	A	B	C	*w*
A	1	8/9	8	8/18
B	9/8	1	9	9/18
C	1/8	1/9	1	1/18

(38)

Avec la méthode AHP classique, on obtient le classement [de priorités] B(0.47)>A(0.45)>C(0.08). Les évaluations par critère correspondent aux vecteurs *w*. Dyer (1990), reprenant les données de l'exemple de Belton et Gear (1983), ajoute une quatrième alternative D qui obtient les résultats 8, 1 et 8 sur les critères C1, C2 et C3. Les comparaisons par paires prennent donc l'allure suivante:

C1	A	B	C	D	*w*
A	1	1/9	1	1/8	1/19
B	9	1	9	1/9	9/19
C	1	1/9	1	1/8	1/19
D	8	9	8	1	8/19

C2	A	B	C	D	*w*
A	1	9	9	9	9/12
B	1/9	1	1	1	1/11
C	1/9	1	1	1	1/12
D	1/9	1	1	1	1/12

C3	A	B	C	D	*w*
A	1	8/9	8	1	8/26
B	9/8	1	9	9/8	9/26
C	1/8	1/9	1	1/8	1/26
D	1	8/9	8	1	8/26

(39)

Le résultat final par AHP donne alors un classement A(0.37) > B(0.30) > D(0.27) >C(0.006). Ainsi, le projet A est passé en tête, devant le projet B, par le seul fait que l'on a ajouter D. (Leblanc et al , 1993, p. 3).

Ce phénomène d'inversion des rangs est principalement causé par la mécanique de la méthode AHP et non par une incohérence dans les comparaisons binaires. Dyer (1990) explique que l'inversion des rangs survient surtout lorsque un élément (e.g., attribut) s'ajoute ou s'avère éliminé de la hiérarchie. La caractéristique de cet élément correspond à sa similarité avec un ou plusieurs autres du groupe. La cause correspond à une "dilution" des pointages de comparaisons. En d'autres mots, la présence de "copie proche" diminue la dominance relative des autres alternatives. Saaty (1987) a étudié les caractéristiques des "copies proches" qui suscitent une inversion des rangs. Il a conclu qu'une alternative doit être éliminée si son score est à l'intérieur de 10% d'une autre alternative. On pourrait se questionner sur l'utilité d'une méthode multicritère de prise de décisions lorsque la dominance des alternatives s'avère évidente. Leblanc (1999) explique que l'ordre d'importance est essentiel à connaître surtout dans les situations où les alternatives possèdent une dominance rapprochée. De plus, Saaty (1990) mentionne que l'élimination ou l'ajout d'un élément de la hiérarchie, démontrent un manque de compréhension des intensités de la part du décideur. Malheureusement, ce commentaire ne reflète pas toujours la réalité, puisque dans certaines situations des éléments, initialement absents, doivent être rajoutés.

Leblanc et al. (1993) soulignent la possibilité que le classement obtenu avec le AHP soit incohérent du point de vue logique. Les auteurs offrent l'exemple suivant afin d'expliquer ce phénomène:

> Trois projets A, B et C sont comparés selon trois critères différents C1, C2 et C3 ayant des poids égaux [i.e., 1/3]. Les comparaisons par paires donnent les résultats suivants:

C1	A	B	C	*w*
A	1	8	1	8/17
B	1/8	1	1/8	1/17
C	1	8	1	8/17

C2	A	B	C	*w*
A	1	1/6	1/2	1/9
B	6	1	3	6/9
C	2	1/3	1	2/9

C3	A	B	C	*w*
A	1	1	3	3/7
B	1	1	3	3/7
C	1/3	1/3	1	1/7

(40)

> Par la méthode AHP, nous avons le rang suivant B(0.38) > A(0.34) > C(0.28). Toutefois, la comparaison directe des projets A et B selon AHP conclut que A(0.51) > B(0.49). De même, on obtient A(0.53) > C(0.47) et B(0.54) > C(0.46). Le classement final devrait donc être A > B > C. (L'inspection des données montre bien aussi que A domine B). (p. 6).

De plus, Leblanc et al. (1993) ont aussi identifié plusieurs écrits: "qui utilisent AHP et échappent à ce problème, ne sont pas vraiment multicritères, ils présentent un projet qui est supérieur aux autres de façon évidente" (p. 6). Il est recommandé que le décideur soit vigilant lors de l'interprétation des données.

2.6.2. Intransitivité

Leblanc et al. (1993) ont découvert une autre situation pouvant survenir lors de l'utilisation du AHP, celle d'intransitivité, i.e. "où l'agrégation du premier niveau débouche sur une intransitivité avec A > C, B > C et C > A, que le préordre approché du second niveau conclut à un préordre du type A > B > C" (Leblanc, et al., 1993, p. 10). Ces auteurs offrent l'exemple ultérieur qui peut survenir avec le AHP:

> Trois projet A, B et C sont comparés selon trois critères différents C1, C2 et C3 ayant des poids égaux [i.e., 1/3]. Les comparaisons par paires donnent les résultats suivants:

C1	A	B	C	w
A	1	3	9	9/13
B	1/3	1	3	3/13
C	1/9	1/3	1	1/13

C2	A	B	C	w
A	1	1/8	1/4	1/13
B	8	1	2	8/13
C	4	1/2	1	4/13

C3	A	B	C	w
A	1	4	1/2	4/13
B	1/4	1	1/8	1/13
C	2	8	1	8/13

(41)

La méthode AHP donne comme résultat A(0.36) > C(0.31) > B(0.31). Toutefois la comparaison des projets deux à deux donnent les résultats suivants, A(0.55) > B(0.45), B(0.51) > C(0.49) et A(0.52) > C(0.48). Malgré la parfaite cohérence des matrices de comparaisons par paires, il y a intransitivité dans la classement des projets.

Leblanc et ses collaborateurs (1993) distinguent trois situations différentes en présence de l'intransitivité :

- L'intransitivité n'est pas acceptable pour le décideur. Dès lors, il faut remonter aux évaluations initiales ou changer de méthode d'agrégation des comparaisons par paires afin de corriger la situation. (p. 10)

- L'intransitivité est acceptable (pour des décisions collectives par exemple), et un projet domine les autres. Dans ce cas, A est préféré à B, C, et D dans les comparaisons par paires, alors que B, C et D se surclassent l'un et l'autre. Puisque les projets sont mutuellement exclusifs, le projet dominant peut être retenu. (p. 11)

- L'intransitivité est acceptable et aucun projet n'est dominant. Pour cette éventualité, il est nécessaire d'ajouter de l'information pour parvenir à discriminer entre les projets. (Notons cependant que l'on peut souvent partager l'ensemble des projets en deux classes distinctes: celle à retenir pour analyse ultérieure et celle à rejeter. Cette démarche, qui est au cœur de la méthode ELECTRE (Roy, 1993), peut également être utile à l'approche proposée dans ce travail) (p. 11).

2.7. Modifications au modèle AHP

Tel que nous venons de le voir, le AHP peut être susceptible à certaines controverses. De plus, fréquemment, le décideur n'est pas alerté aux problèmes pouvant survenir lors de l'utilisation du AHP ou de d'autres modèles de prise de décision (i.e., inversion des rangs, intransitivité, classement illogique). Leblanc (1999) explique que ces problèmes demeurent habituellement dissimulés à l'intérieur des résultats. Donc, le décideur ne possède aucune façon de valider le classement final. Afin de résoudre ce problème, Leblanc et al. (1993) offrent une modification à l'algorithme AHP (Figure 2.4). Cette méthode "agrège les comparaisons multicritères par paires de projets plutôt que les préordres complets établis sur les projets pour chaque critère" (p. 11). L'algorithme de Leblanc et al. (1993) n'apporte aucun changement à la méthodologie même du AHP. Cependant, elle introduit des étapes ayant pour but d'informer le décideur et d'anticiper les sources de problèmes pouvant survenir.

Les premières étapes du modèle de Leblanc et al. (1993) s'avèrent identiques au modèle AHP (i.e., élaboration d'une hiérarchie et de la matrice de priorités, le calcul du vecteur propre et le ratio de cohérence). Ensuite:

> Lorsque les poids relatifs des critères et sous-critères de tous les niveaux ont été calculés, on fait l'agrégation des niveaux en multipliant les poids des sous-critères par le poids des critères associés. Par ces calculs, on réussit à obtenir des poids pour tous les sous-critères du dernier niveau qui tiennent compte de toute la hiérarchie. Une fois que cette base est établie, tous les couples d'alternatives sont évalués individuellement (ab, ac, ad, ae, bc, bd, be, etc...). Les deux alternatives de chaque couple sont alors comparés et leurs poids relatif trouvé pour chaque sous-critères de dernier rang. À partir de ces poids relatifs et des poids des sous-critères, il est possible d'établir un classement relatif des deux alternatives par une simple sommation. Comme on peut le voir, cette comparaison par paires de deux

> alternatives quelconques est totalement insensible à l'ajout et au retrait d'une alternative dans le groupe de projets. Une fois que tous les couples d'alternatives ont été ainsi comparés, la construction d'un préordre sur les projets peut alors être entrepris. Ceci sera direct s'il n'y a aucune intransitivité, c'est-à-dire de situation telle que A > B, B > C et C > A. (Leblanc et al., 1993, p. 13).

D'après ces auteurs, le modèle peut, non seulement, prévenir l'inversion des rangs et les cas d'intransitivité, mais il s'avère également robuste aux altérations faites sur l'ensemble des projets (i.e. ajout ou retrait de projets).

En appliquant le modèle de Leblanc et al. (1993) dans leurs prises de décision, les décideurs peuvent maintenant disposer d'une meilleure assurance sur la validation des résultats de classement. Autrement, le décideur devra toujours se demander si le classement final n'a pas été compromis par des problèmes intrinsèques au AHP ou à toute autre méthode de classement (Leblanc, 1999).

Ayant considéré les écrits exposées précédemment, nous avons réussi à créer un modèle visant à permettre la quantification d'attributs logiciels. La prochaine étape correspond à l'application du PQAL afin de pouvoir le valider et apporter les ajustements nécessaires. Dans le but de procéder à l'application de notre modèle, une validation et une vérification de la base de données utilisée sera amorcée dans le cadre du prochain chapitre.

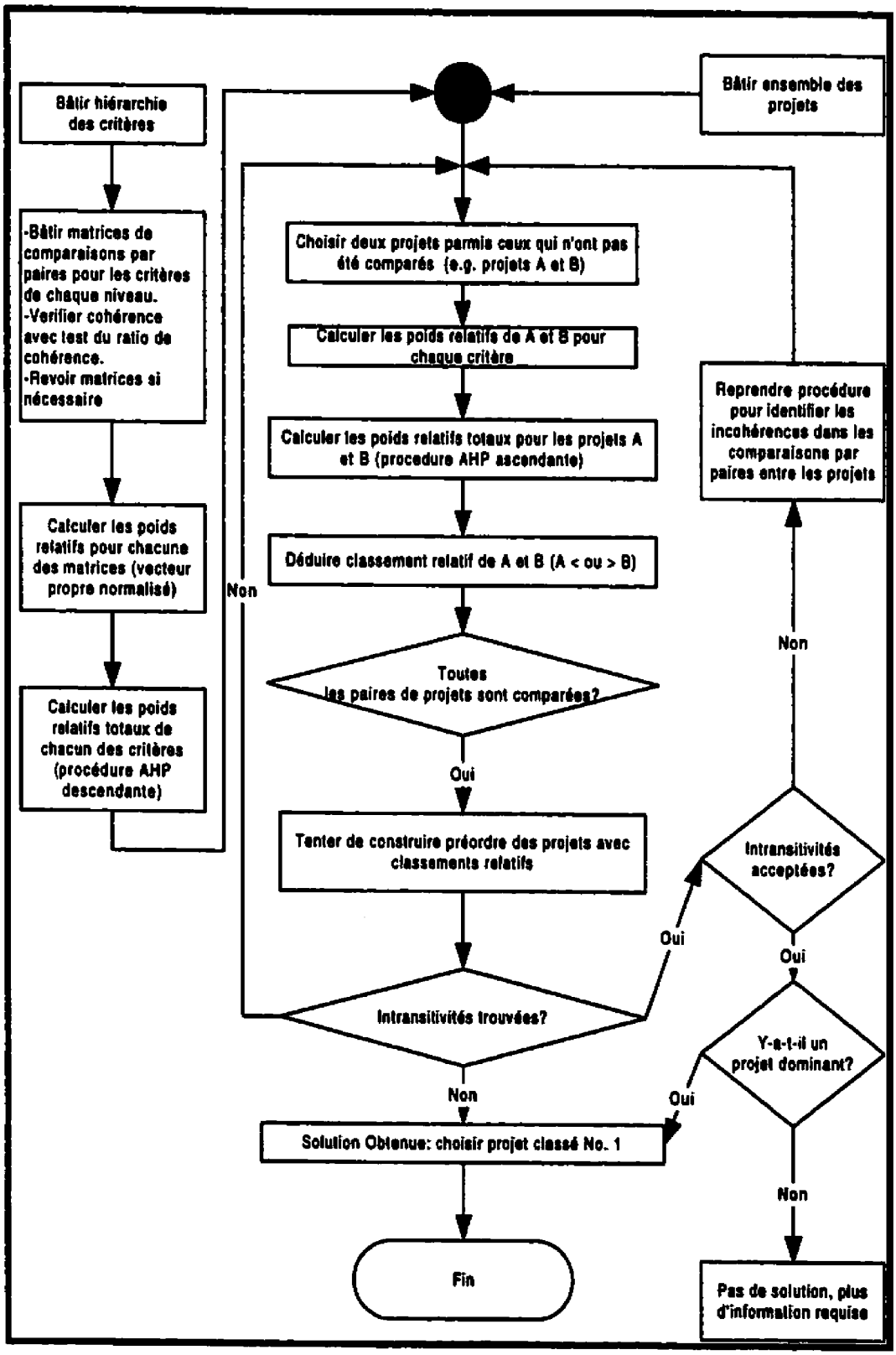


Figure 2.7: Algorithme du AHP pour des mesures relatives (Leblanc et al, 1993)

CHAPITRE 3
VÉRIFICATION ET VALIDATION DE LA BASE DE DONNÉES

3.1. Introduction

Afin de pouvoir procéder à l'application de notre modèle, le Processus de Quantification des Attributs Logiciels (PQAL), il devenait nécessaire de valider d'abord la base de données mise à notre disposition. Ainsi, cette validation fera l'objet de ce chapitre. Cette base de données déjà existante provient des travaux de six groupes d'étudiants qui ont eu pour objectif de développer un produit logiciel en s'inspirant du processus de développement logiciel UPEDU (*Unified Process for Education*), lequel correspond à une adaptation du processus commercial RUP (*Rational Unified Process*) pour des fins pédagogiques, dans le cadre du cours IF501- Atelier de Génie Logiciel à l'École Polytechnique de Montréal. Au cours de leur projet, chaque étudiant avait la responsabilité de remplir une feuille de temps en identifiant la quantité de temps accordée à chacune des activités du processus. Il demeure toutefois important de noter que l'objectif de ce mémoire n'est pas d'évaluer le processus même, mais d'examiner si le suivi d'un processus s'avère directement relié à la qualité du produit. Puisque nous connaissons déjà la qualité finale du travail créé par chaque groupe, il nous sera davantage possible de confirmer si notre analyse du processus est significative.

Pour les besoins de ce mémoire, nous présumerons que le lecteur est familier avec le concept du processus de développement logiciel. Toutefois, dans la prochaine section, nous allons tout de même identifier deux aspects importants du processus UPEDU. Par la suite, nous décrirons l'outil employé pour créer la base de données utilisée pour l'analyse du processus. La section suivante portera sur les critères de vérification et de validation de la base de données. Pour terminer, les résultats de cette validation seront présentés et une discussion sera offerte.

3.2. UPEDU

Tel que mentionné plus haut, les étudiants se sont inspirés du processus UPEDU pour développer le logiciel. Dans le but d'analyser le déroulement du processus, nous avons utilisé deux aspects importants de la structure logique du UPEDU.

Kruchten (2000) caractérise principalement l'architecture générale de UPEDU par deux dimensions: le temps et les enchaînements d'activités (voir). Le temps correspond à la première dimension, représenté par l'axe horizontal, qui démontre le déroulement du cycle de vie du processus, lequel est exprimé en termes de cycles, de phases, d'itération et de jalons (Kruchten, 2000). La deuxième dimension concerne les enchaînements d'activités, représentés sur l'axe vertical, qui regroupent les activités selon leur nature. Ceci représente l'aspect statique du processus qui s'exprime en termes de composants, de processus, d'activités, d'enchaînements, d'artefacts et de travailleurs. Les enchaînements d'activités, ou «workflows», ont pour but de produire un résultat observable ou un artefact (i.e., un élément d'information fabriqué, modifié ou utilisé). Les six principaux enchaînements reliés au processus UPEDU sont : (a) les requis ou exigences, (b) l'analyse et le design, (c) l'implémentation, (d) le test, (e) la gestion de la configuration et des changements, et finalement, (f) la gestion de projet. La Figure 3.1 illustre la distribution de l'effort (e.g., nombre d'heures travaillées par personne) pour les six enchaînements ou workflows durant le cycle de vie du produit logiciel (Kruchten, 2000).

Puisque les étudiants se sont inspirés du processus UPEDU durant leur projet, ces deux dimensions se retrouvent également dans les jetons qui seront présentés dans la section suivante.

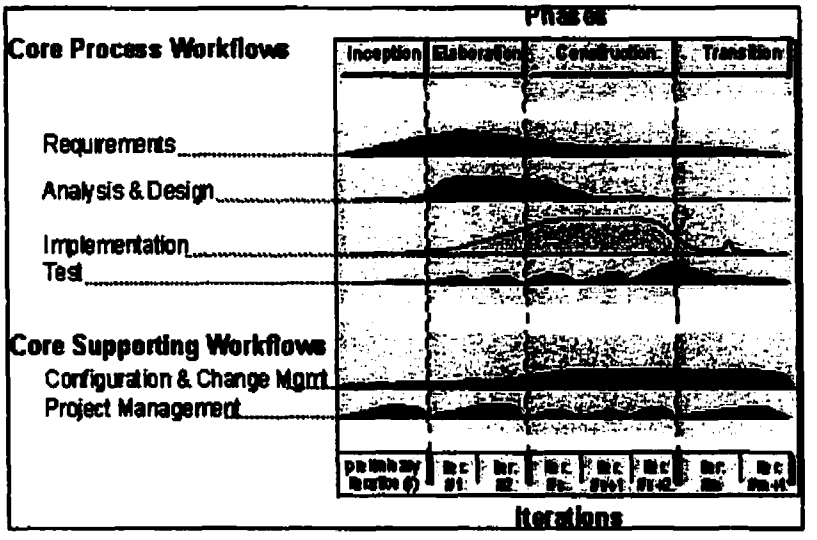


Figure 3.1: Enchaînements d'activités reliées aux UPEDU. (Adapté de Krutchen, 2000.)

3.3. Description de la Base de Données

Cette section est consacrée à la description de la base de données utilisée pour l'évaluation du processus de développement logiciel. Soulignons, que les données sont déjà existantes et n'ont pas été recueillies en fonction de ce mémoire. Premièrement, nous expliquerons l'objectif poursuivi par les étudiants dans leur projet, suivi des critères reliés au projet. En terminant, nous décrirons le logiciel que les étudiants ont utilisé pour insérer les jetons ainsi que le produit final.

3.3.1. Objectif du Projet

En équipe, les étudiants avaient pour but de créer un logiciel, en langage Java, représentant un outil de feuille de temps. Le projet a duré 13 semaines et chaque étudiant devait y consacrer au moins 10 heures par semaine, incluant la présence en classe.

Les objectifs reliés à la Spécification des Requis du Système (SRS) afférent à l'outil de feuille de temps étaient :

- ❖ Intégration et suivi de la composante 'budget' pour un projet donné;
- ❖ Enregistrement précis du temps passé sur les différentes tâches/activités pour un projet donné;
- ❖ Système doit permettre de fournir suffisamment d'informations au processus d'évaluation (soumission et validation de tâche de projet en cours) ainsi qu'à l'analyse de la fiabilité logicielle de nos produits;
- ❖ Création d'un rapport d'avancement hebdomadaire individuel.

3.3.2. Description des équipes

Les étudiants, dans le cadre du cours IF501- Atelier de Génie Logiciel, font partie du programme de Génie Logiciel au niveau baccalauréat. La classe fut composée de six équipes, toutes en compétition sur le même projet. Chaque équipe comporte le nombre d'étudiants suivant:

- Équipe 2 = 4 étudiants;
- Équipe 3 = 6 étudiants;
- Équipe 4 = 6 étudiants;
- Équipe 5 = 5 étudiants;
- Équipe 6 = 5 étudiants;
- Équipe 7 = 6 étudiants;

3.3.3. Structure des Équipes

Il fut recommandé aux étudiants d'adapter les rôles suivants pour chacun des membres de l'équipe:

Coordonnateur

- ❖ Assurer la coordination entre les membres de l'équipe
- ❖ Fixer la date et le lieu de la réunion
- ❖ Fixer la durée de la réunion

- Préparer l'ordre du jour de la réunion
- Planifier et maintenir la discipline durant la réunion (l'efficacité)

Responsable des mesures

- Recevoir de chaque membre de l'équipe les jetons
- Enregistrer les jetons dans une base de données «Access».
- Gérer quotidiennement la base de données
- Communiquer, sur une base hebdomadaire, la mise à jour au chargé de laboratoire

Responsable de la documentation

- S'assurer que le contenu des artefacts est concordant avec l'activité qu'il a généré. ***Exemple*** : S'assurer que le document «use-case» est concordant avec l'activité de la détermination des «use-cases»
- S'assurer que les gabarits des artefacts sont respectés selon le UPEDU
- S'assurer que les mises à jour des documents sont effectuées après les «walkthroughs» (i.e.: revues formelles).

Secrétaire des revues

- Enregistrer la présence des membres à chaque réunion
- Produire les rapports des réunions
- S'assurer que les points soulevés durant une réunion ont été traités

Responsable de l'environnement Physique (Répertoires)

- Maintenir l'organisation mémoire des répertoires et fichiers
- Gérer les versions des artefacts

- S'assurer que les dernières versions des artefacts sont présentes sur les répertoires de travail et ce, au bon endroit
- Exécuter les sauvegardes de secours

Responsable de la gestion de projet

- Planifier les activités et les tâches en fonction des dates de livraisons, des ressources humaines (membres de l'équipe) et des ressources matérielles
- Produire les diagrammes de PERT et de GANTT
- Présenter hebdomadairement l'état du projet au chargé de laboratoire

3.3.4. Les Jetons

Essentiellement, un jeton associe la structure du processus {par exemple, Nom_Projet, Nom_Tâche, Nom_Activité, Nom_sous_activité} à un nombre de minutes ou d'heures. Cette association représente la durée passée sur une activité ou une sous activité. Chaque membre de l'équipe avait la responsabilité de communiquer leurs jetons au Responsable des Mesures, lequel enregistrait les jetons dans une base de données «Access de Microsoft». Il était également important que les jetons soient gérés de façon quotidienne afin de minimiser les entrées erronées.

La Figure 3.2 représente l'interface usagée (IU) du logiciel Access utilisée par les étudiants pour entrer leurs jetons. L'IU comprend neufs champs: Le nom du projet, du travailleur, du workflow, de l'activité, de l'artefact, la date, les heures de début et de fin, la durée de l'activité et enfin, un champ permettant à l'étudiant de fournir la description de leur travail. Il est à noter que chaque champ était indépendant, c'est à dire, pour un artefact donné, l'utilisateur avait la possibilité d'entrer n'importe quelle activité et workflow. Malheureusement, cet aspect permettait l'introduction d'erreurs dans la base de données (i.e., une activité et/ou un workflow qui ne concorde pas avec l'artefact enregistré). Une discussion plus approfondie sur ce sujet sera présentée ultérieurement. La base de données fut ensuite exportée en format Excel pour fin d'analyse.

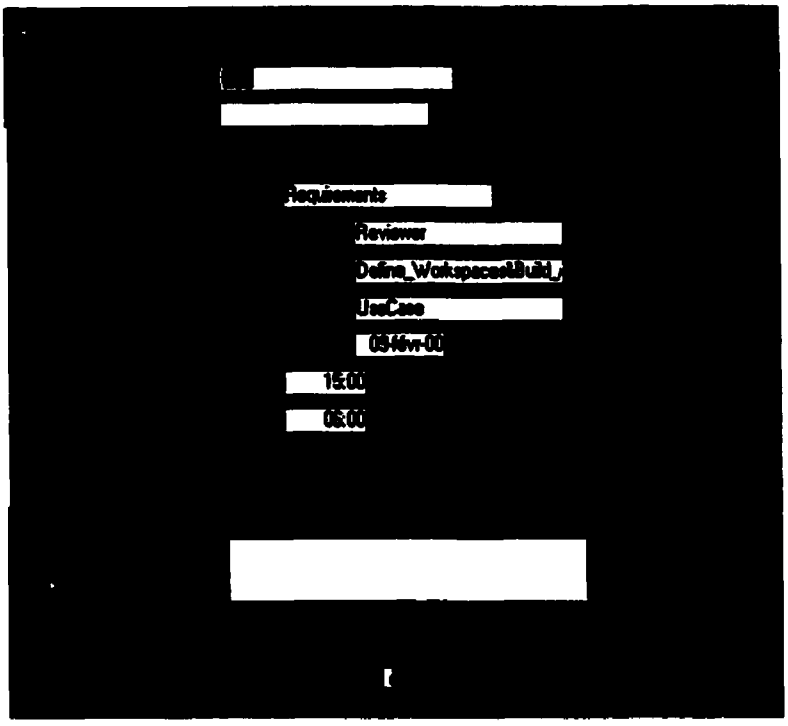


Figure 3.2: Interface usager de l'outil de jeton MS Access

Pour les besoins de ce mémoire, nous allons utiliser les abréviations suivantes pour décrire chacun des champs (voir Tableau 3.1).

Tableau 3.1: Abréviations des Champs de la Base de Données.

Travailleur	WK
Workflow	WF
Activité	AC
Artefact	AR
Date	DT
Heures de Début	ST
Heures de Fin	ET
Durée	DR
Description	DS

3.4. Critères de Vérification et de Validation de la Base de Données

Tel que Florac et Carleton (1999) le mentionnent, avant de poursuivre avec l'analyse des données, il existe certains critères que les données doivent satisfaire afin que les conclusions de l'analyse soit crédibles et significatives. Les auteurs recommandent quatre critères: vérité, synchronisation, consistance et validité. D'après ces auteurs, la synchronisation est présente lorsque les valeurs de deux ou plusieurs attributs sont reliés en fonction du temps. Comme nos mesures dépendent que d'un seul attribut à la fois, il est davantage justifiable de remplacer ce critère par un nouveau critère, celui de «duplication». La prochaine section portera donc sur la vérification et la revue de la base de données et ce, à partir de l'adaptation des critères suggérés par Florac et Carleton (1999).

3.4.1. Vérité

La vérification des données a été réalisée afin d'assurer qu'elles ont été compilées selon les spécifications fournies par le professeur et le chargé de laboratoire et qu'elles ne contiennent aucune erreur. Nous avons corrigé le jeton lorsqu'une erreur était identifiée. Lorsqu'une correction s'avérait impossible, le jeton était rejeté de la base de données. Voici les caractéristiques suggérées par les auteurs Florac et Carleton (1999) qui ont guidé le processus de vérité.

Les valeurs d'échelles de ratio identifiées dans les prochains tableaux représentent le pourcentage d'erreur ne répondant pas correctement à la caractéristique visée pour un champ donné. Les pourcentages sont basés sur le nombre total de jetons. Par exemple, un pourcentage de 10% dans la colonne AC, nous indique que plus de 10% de toutes les données d'activités sont erronées. De plus, le symbole « -- » dans une cellule signifie qu'aucune erreur n'a été observée. À la fin de cette section un résumé de toutes les erreurs pour chacun des groupes sera offert.

3.4.1.1. Est-ce que les données correspondent à un type numérique ou alphanumérique approprié?

Cette caractéristique permet de déterminer, pour un champ donné, si chacune des données est de type numérique ou alphanumérique. Par exemple, pour les champs d'activité, les données doivent être de type alphanumérique tandis que, pour le champ de durée, le type doit être numérique. Tel que nous pouvons le constater à la lumière du , les résultats de la vérification démontrent qu'aucune erreur associée à cette caractéristique a été identifiée.

Tableau 3.2: Vérification de la Caractéristique: Type.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	--	--	--	--	--	--	--	--	--	
3	--	--	--	--	--	--	--	--	--	
4	--	--	--	--	--	--	--	--	--	
5	--	--	--	--	--	--	--	--	--	
6	--	--	--	--	--	--	--	--	--	
7	--	--	--	--	--	--	--	--	--	

3.4.1.2. Est-ce ce que le format est approprié?

Cette caractéristique permet de vérifier que les données possèdent le format approprié. Par exemple, le champ des noms doit contenir les noms de chacun des travailleurs, le champ des workflows doit contenir seulement les workflows, le champ des dates doit contenir seulement les dates, et ainsi de suite. Les résultats de cette caractéristique sont représentés dans le Tableau 3.3 et démontrent que la majorité des bases de données possède aucune ou très peu d'erreurs. Notons toutefois, que plus de 20% des données du champ d'activités (i.e.: AC) de l'équipe 4 sont erronées. La source de ces erreurs provient du fait que plusieurs des données ne correspondaient pas aux activités telles que retrouvées dans UPEDU.

Tableau 3.3: Vérification de la Caractéristique: Format.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	--	--	--	--	--	--	--	--	--	
3	--	--	--	--	--	--	--	0.24%	--	
4	--	--	21.6%	--	--	--	--	--	--	
5	2.4%	4.1%	--	--	--	--	--	--	--	
6	--	--	1.4%	--	--	--	--	--	--	
7	--	--	--	--	--	--	--	--	--	

3.4.1.3. Est-ce ce que les données sont à l'intérieur des plages?

Ici nous désirons savoir si les valeurs numériques et alphanumériques sont à l'intérieur des plages qui ont été préétablies. Par exemple, est-ce que le nombre de travailleurs correspond au nombre d'étudiants dans l'équipe ou est-ce qu'il manque un travailleur ou est-ce qu'un travailleur a été rajouté sans notification, ou encore, est-ce que les dates dépassent la date de remise du projet, etc. Les résultats de cette caractéristique apparaissent dans le Tableau 3.4.

Tableau 3.4: Vérification de la Caractéristique: Plages.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	--	--	--	--	2.61%	--	--	--	--	*
3	--	--	--	--	0.96%	--	--	--	--	*
4	--	--	--	--	6.57%	--	--	--	--	*
5	--	--	--	--	1.08%	--	--	--	--	*
6	--	--	--	--	1.36%	--	--	--	--	*; On a dépassé la date limite.
7	--	--	--	--	2.64%	--	--	--	--	*; Idem

* Les dates de la présentation du projet final par les étudiants ont été omises.

3.4.1.4. Est-ce ce que les données sont complètes?

La base de données doit contenir tous les éléments essentiels et textuels nécessaires pour comprendre et interpréter les données. Ainsi, nous voulons savoir si tous les champs sont

complets. Les résultats de cette caractéristique sont présentés dans Tableau 3.5. Il s'avère intéressant de noter que le champ de Description, pour les équipes 2, 4 et 6, sont vides et ne possèdent aucune donnée quelconque (i.e.: 100% erreurs). Malheureusement, ayant utilisé une base de données déjà existante, il s'avère impossible de connaître l'origine de ces erreurs. En d'autres mots, nous ignorons si les champs sont vides à cause d'une erreur dans le transfert des données de MS Access à MS Excel ou si les équipes ont négligé d'offrir une description.

Tableau 3.5: Vérification de la Caractéristique: Complet.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	--	--	--	--	--	0.87%	0.87%	0.87%	100%	*
3	1.20%	--	0.48%	8.17%	--	--	--	--	42.6%	*
4			4.69%	14.6%	3.76%	0.94%	0.94%	100%	100%	*
5	0.22%	--	--	--	32.5%	32.5%	32.5%	--	32.5%	*
6	--	--	0.45%	--	--	0.91%	0.91%	100%	100%	*
7	0.38%					--	--	17.4%	58.1%	*

*Toutes les erreurs représentent des champs vides

3.4.1.5. Est-ce ce que les équations arithmétiques sont appropriées?

Toutes les données provenant d'une équation arithmétique doivent être examinées de façon à vérifier la nature de l'équation. La seule valeur provenant d'une manipulation arithmétique est celle de la durée. Afin de vérifier ces données, nous avons calculé la durée en soustrayant le temps final moins le temps de début. Tel que nous allons le constater dans le Tableau 3.6, la plupart des groupes présentaient des erreurs de durée. La principale raison est que l'équation originale de l'outil de jeton ne tenait pas compte si le temps final du jeton avait dépassé 00:00 (minuit). En d'autres mots, l'équation originale soustrayait le temps final du temps de début:

$$\text{Durée (pour toutes les heures)} = \text{Temps Final} - \text{Temps Début} \quad (42)$$

Toutefois lorsque le temps final est égale ou dépasse 00:00, l'équation doit être:

Durée (si temps final ≥ 0:00) = 24:00 – (Temps Début – Temps Final) (43)

Tel que nous le démontre le Tableau 3.6, plusieurs corrections ont été apportées aux données relatives au champ de durée.

Tableau 3.6: Vérification de la Caractéristique: Équation.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	n/a	n/a	n/a	n/a	n/a	n/a	n/a	–	n/a	
3	n/a	n/a	n/a	n/a	n/a	n/a	n/a	1.20%	n/a	
4	n/a	n/a	n/a	n/a	n/a	n/a	n/a	100%	n/a	Champ DR est vide
5	n/a	n/a	n/a	n/a	n/a	n/a	n/a	–	n/a	
6	n/a	n/a	n/a	n/a	n/a	n/a	n/a	100%	n/a	Champ DR est vide
7	n/a	n/a	n/a	n/a	n/a	n/a	n/a	4.2%	n/a	

3.4.2. Duplication

Rappelons ici que nous avons ajouté cette caractéristique puisque nous avons remarqué, suite à l'examen de la base des données, que certains jetons étaient en fait des duplicata, c'est à dire, deux ou plusieurs jetons avaient tous les mêmes champs d'entrées. Nous avons identifié deux causes possibles à ce phénomène, la première étant une erreur de transfert de données entre les logiciels Microsoft Access et Microsoft Excel (ceci était confirmé par le chargé de laboratoire), et la deuxième étant une erreur provenant de l'utilisateur lorsque le même jeton est entré deux fois ou plus par inadvertance. Pour le besoin de l'analyse, il n'était pas nécessaire de distinguer entre les deux sources d'erreurs, mais seulement de les identifier et de les retirer de la base de données. Une autre erreur de duplication correspond au chevauchement des données. Cette erreur survient lorsque l'utilisateur, par inattention, entre deux ou plusieurs jetons dont les heures se chevauchent. Par exemple, jeton A débute à 10:00 et se termine 12:00 et le jeton B commence à 11:00 et se termine 15:00. La correction de ces erreurs a été faite de façon subjective afin d'éliminer le chevauchement. Pour reprendre notre exemple, le temps de

départ pour le jeton B aurait été modifiée de 11:00 à 12:00. Toutefois, lorsque les heures de début et de fin du jeton B se trouvaient toutes les deux à l'intérieur des heures du jeton A, le Jeton B était éliminé. Il est clair que cette manipulation diminue la fiabilité des données, toutefois, ce problème ne représente qu'un faible pourcentage des données (voir Tableau 3.14).

Les deux prochains tableaux démontrent la présence des duplicata et chevauchements pour les six équipes. Notons que seule l'équipe 4 ne possède aucune erreur de duplicata.

Tableau 3.7: Vérification de la Caractéristique: Duplication: Duplicata.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	5.26%	5.26%	5.26%	5.26%	5.26%	5.26%	5.26%	5.26%	5.26%	
3	14.2%	14.2%	14.2%	14.2%	14.2%	14.2%	14.2%	14.2%	14.2%	
4	--	--	--	--	--	--	--	--	--	
5	8.93%	8.93%	8.93%	8.93%	8.93%	8.93%	8.93%	8.93%	8.93%	
6	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	0.91%	
7	3.04%	3.04%	3.04%	3.04%	3.04%	3.04%	3.04%	3.04%	3.04%	

Tableau 3.8: Vérification de la Caractéristique : Duplication: Chevauchement.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	n/a	n/a	n/a	n/a	n/a	2.63%	2.63%	n/a	n/a	0.88% jetons corrigés 1.75% jetons éliminés
3	n/a	n/a	n/a	n/a	n/a	2.16%	2.16%	n/a	n/a	1.2% jetons corrigés 0.96% jetons éliminés
4	n/a	n/a	n/a	n/a	n/a	8%	8%	n/a	n/a	3.76% jetons corrigés 4.23% jetons éliminés
5	n/a	n/a	n/a	n/a	n/a	0	0	n/a	n/a	
6	n/a	n/a	n/a	n/a	n/a	2.74%	2.74%	n/a	n/a	1.83% jetons corrigés 0.91% jetons éliminés
7	n/a	n/a	n/a	n/a	n/a	4.18%	4.18%	n/a	n/a	0.76% jetons corrigés 3.42% jetons éliminés

3.4.3. Consistance

Afin d'éviter des erreurs d'analyses, Florac et Carleton (1999) recommandent que toutes les valeurs considérées comme étant improbables ou étranges soient investiguées de façon

à vérifier leur exactitude. Cette investigation peut parfois exiger à l'analyste de prendre contact avec la source des données afin de vérifier l'exactitude des entrées. En ce qui concerne notre recherche, il ne s'avérait pas possible d'entrer en contact avec les étudiants pour confirmer que tous comprenaient la façon d'entrer les données. Toutefois, en ce qui concerne la façon d'entrer les données, nous savons que les mêmes directives ont été fournies par le professeur et le chargé de laboratoire à tous les étudiants. De plus, chaque équipe avait accès au même outil de jetons Microsoft Access.

À priori, il semble justifiable de ne pas remettre en cause la consistance des données. À posteriori, par contre, tel que nous allons voir, plusieurs erreurs sont survenues, ce qui nous laisse croire que la compréhension du processus parmi les équipes était inconsistante. En d'autres mots, nous croyons que chaque équipe avait leur propre interprétation des différents aspects associés aux trois éléments de modélisations du UPEDU (i.e.: les workflows, les activités et les artefacts). Puisque nous ne pouvons interroger tous les étudiants, il demeure difficile de mesurer ou même d'établir le degré d'inconsistance des jetons. Il sera donc important, lors de l'analyse des données, de prendre en considération la présence d'inconsistance des jetons entre les étudiants ou les équipes.

3.4.4. Validité

Le dernier critère consiste à démontrer que les valeurs recueillies pour décrire un attribut sont vraiment représentatives de l'attribut en question. En effet, lorsqu'un étudiant insère un jeton, il est nécessaire de s'assurer que les artefacts, activités et workflows représentent ce que l'étudiant a accomplit durant la période donnée. Puisque le processus utilisé par les étudiants devait être basé sur UPEDU, les règles d'associations entrent les artefacts, activités et workflows s'avèrent bien définies. Ainsi, nous pouvons associer, pour chaque artefact, une ou plusieurs activités données, qui sont elles même associées à des workflows prédéterminés. En d'autres mots, une activité A ne peut être associée avec

n'importe quel artefact ou workflow. Pour cette raison, il demeure important que tous les artefacts, activités et workflows choisis par les utilisateurs respectent la structure logique du UPEDU.

Lors de l'analyse de validation, nous avons examiné la relation entre les workflows, les activités et les artefacts. La pertinence des artefacts dépendait directement des commentaires offerts par les étudiants. Lorsque les commentaires ne correspondaient pas à l'artefact choisi, l'artefact fut remplacé par celui correspondant le mieux aux commentaires. Par la suite, les activités et workflows furent modifiés en concordance avec l'artefact choisi. Toutefois, tel que mentionné dans le Tableau 3.5, pour certaines équipes, le champ de discussion était vide, à ce moment, seule la pertinence des activités et des workflows reliés à l'artefact a pu être vérifiée.

S'avérant impossible de contacter chaque étudiant, les changements des artefacts basés sur les commentaires peuvent être considérés «subjectifs». Ainsi, l'introduction d'erreurs externes demeure présente et doit être prise en considération lors de l'analyse finale des données. Afin de rendre l'exercice de vérification des activités et des workflows plus rigoureux, une macro écrit en Visual Basic pour Excel fut créée. Le Tableau 3.9, démontre la présence d'erreurs quant à la relation entre les artefacts, activité et workflows, et ce pour chaque équipe.

Tableau 3.9: Vérification de la Caractéristique: Validité.

Équipe No.	WK	WF	AC	AR	DT	ST	ET	DR	DS	Commentaires
2	n/a	–	5.26%	51.8%	n/a	n/a	n/a	n/a	n/a	
3	n/a	0.48%	2.88%	51.8%	n/a	n/a	n/a	n/a	n/a	
4	n/a	9.39%	8.5%	26.3%	n/a	n/a	n/a	n/a	n/a	
5	n/a	4.79%	13.1%	43.6%	n/a	n/a	n/a	n/a	n/a	
6	n/a	3.20%	6.85%	46.6%	n/a	n/a	n/a	n/a	n/a	
7	n/a	2.28%	9.89%	36.5%	n/a	n/a	n/a	n/a	n/a	

3.5. Correction et Élimination des Jetons

La question suivante survient fréquemment dans le domaine des sciences expérimentales: Pourquoi devrait-on éliminer des données que nous doutons erronées? Bevington et Robinson (1992) répondent à cette question par une autre question: Pourquoi devrait-on risquer de contaminer notre échantillonnage avec des données douteuses ou erronées, puisqu'elles contribuent souvent bien peu au résultat final? Malheureusement, il n'existe aucune règle unique qui puisse définir si telle donnée doit être acceptée, corrigée ou éliminée. Voilà pourquoi nous avons utilisés les critères de validation et de vérification énoncés précédemment. Toutefois, contrairement à des données provenant de recherches empiriques, la majorité de nos jetons contiennent des valeurs qualitatives, alors, même si nous créons des règles de validation et de vérification pour ce type de données, il demeure que certaines corrections seront basées sur des notions subjectives, ou «biais naturel», du correcteur. Tel que Bevington et Robinson (1992) le mentionnent, nous pouvons retirer un plus grand profit si nous utilisons un échantillonnage contenant peu de valeurs extrêmes. Ainsi, lors de la vérification et validation de la base de données, toutes données douteuses ou ne répondant pas aux critères de validation et de vérification, ont été corrigées ou éliminées. Toutefois, il s'avère important d'insister que l'introduction d'un biais naturel peut, en lui-même, introduire d'autres erreurs à la base de données. Cette observation sera donc prise en considération lors de l'analyse finale des données.

En d'autres mots, pour fin d'analyse, nous allons considérer le terme «erreur» comme étant de nature objective (e.g.: la différence entre l'heure de début et de fin ne correspondait pas à la durée) et de nature subjective (e.g.: selon notre avis, la valeur de l'artefact ne correspondait pas à la description faite par l'utilisateur). Le processus de vérification et de validation des données a été complété de façon manuelle (e.g. : validation des artefacts par l'entremise des commentaires) et automatique, c'est à dire par l'entremise des macros Visual Basic (e.g. : une macro a été créée pour vérifier si les activités correspondaient à l'artefact et si le workflow correspondait à l'activité).

3.6. Application du Processus de Vérification/Validation (V&V) de la Base de Données

Nous avons donc procédé à la vérification et validation de la base de données à la lumière des critères mentionnés à la section 3.4. Lors de la première validation, nous avons décidé de nous baser sur les artefacts pour corriger les activités et les workflows. Cette approche présumait que les valeurs des artefacts ne nécessitaient aucune correction. Une fois la vérification et la correction terminées, nous avons réalisé que, d'après le champ de discussion, certains artefacts ne correspondaient pas aux commentaires des étudiants, donc une deuxième validation fut nécessaire. Cette fois-ci, le champ de discussion (i.e.: les commentaires des étudiants) ont servi de base à la validation des artefacts, activités et workflows. Évidemment, seulement les artefacts possédant des commentaires ont pu être modifiés.

Durant le processus de V&V, afin de faciliter l'analyse de données, nous avons décidé d'éliminer de la base de données tous les jetons faisant référence au workflow Cours et Formation (CF). La raison pour cette décision est que ce workflow, faisant référence aux périodes de cours et de laboratoire disponibles aux étudiants, devait être identique pour toutes les équipes. La décision de retirer tous les jetons référant au workflow de CF semblait appropriée, puisque nous désirons comparer les éléments du processus qui ne sont pas identiques pour chacune des équipes.

3.7. Agglomération des Jetons

À la fin du premier processus de V&V, nous avons regroupé tous les jetons en ordre d'artefact, d'activité et de workflow, et ce, pour chacune des équipes. Le Tableau 3.6 représente la distribution, en nombre de jetons, des éléments du processus UPEDU pour chacune des équipes. Il est intéressant de noter que la somme de tous les artefacts utilisés par toute les équipes se chiffre à plus de 50. De plus, lorsque nous comparons les artefacts

de chacune des équipes entre elles, nous remarquons, que, pour le même projet, chaque équipe a utilisé une combinaison différente d'artefacts et d'activités. Tel que nous l'avons mentionné au début de ce chapitre, les étudiants ont eux accès à l'outil d'enregistrement de jetons créé avec le logiciel Access. À l'intérieur de cet outil, les étudiants avaient accès à 23 artefacts différents (voir Tableau 3.11). Cette liste représentait tous les artefacts nécessaires au processus du projet. Toutefois, en comparant les artefacts du Tableau 3.10 et du Tableau 3.11, nous remarquons que plusieurs artefacts ont été rajoutés à la base de données Access (i.e., 50 versus 23 artefacts). Ainsi, durant le déroulement du projet, certains étudiants ont décidé d'ajouter des artefacts additionnels retrouvés dans UPEDU ou encore, d'inventer de nouveaux artefacts, croyant probablement mieux décrire les documents sur lesquels ils travaillaient.

Étant donné que nous voulons comparer le processus de développement logiciel de chaque équipe, il s'avérait donc important d'avoir une base de comparaison similaire pour toutes les équipes. C'est-à-dire, il nous faut pouvoir comparer les mêmes ensembles d'artefacts entre les différentes équipes. En conséquence, au moment du second processus de V&V, nous avons décidé d'agglomérer certains artefacts afin d'harmoniser la base de données. Cette agglomération consiste à altérer un artefact par un autre afin de les regrouper ensembles. Cette manœuvre a été réalisée selon les règles combinatoires illustrées dans le Tableau 3.12. Cependant, les artefacts suivants: *User help Design*, *Vision*, *Implementation Model* et *CM Plan*, ont dû être éliminés puisqu'ils ne pouvaient être combinés.

Tableau 3.10: Distribution des Éléments du Processus UPEDU, 1ère V&V.

A&D	Architectural Design	Design Class					1		0.2
A&D	Architectural Design	Design Package			4	3	2		1.5
A&D	Architecture Analysis	Use-Case Realizations				1	1		0.3
A&D	Class Design	Design Class	10	38	5	34	14	4	17.5
A&D	Identify Design Elements	Design Package		1					0.2
A&D	Identify Design Elements	Interface	1	8		4		4	2.8
A&D	Review the Design	Review-AD		8	7	2	6		3.8
A&D	Subsystem Design	Interface			3	1			0.7
A&D	Use-Case Analysis	Use-Case Realizations		1		7	1		1.5
A&D	Use-Case Design	Use-Case Realizations		8	14	23	13	14	11.7
CM	Write the CM Plan	CM Plan						1	0.2
Implementation	Fix a Defect	Component		10			1	1	2.0
Implementation	Implement Classes	Component				3			0.5
Implementation	Implement Components	Component	35	100	60	99	56	62	68.5
Implementation	Integrate System	Build	1	9		7	3	15	5.8
Implementation	Perform Unit Test	Component				1			0.2
Implementation	Perform Unit Tests	Unit Tests		1	1	3			0.8
Implementation	Plan System Integration	Integration Build Plan					1	3	0.7
Implementation	Review Code	Component	1		10	4	1		2.7
Inconnue	Inconnue	User help Design	1						0.2
PM	Define Monitoring & Ctrl Proc.	Measurement Plan				3			0.5
PM	Develop Iteration Plan	Iteration Plan		4		5	7	8	4.0
PM	Develop Measurement Plan	Measurement Plan			1	4			0.8
PM	Develop Project Plan	Project Plan		2	1	21	1	4	4.8
PM	Evaluate the Iteration	Iteration Plan				2			0.3
PM	Execute Iteration Plan	Implementation Model				1			0.2
Requirements	Capture a Common Vocabulary	Glossary			1				0.2
Requirements	Detail the Software Requirements	Requirements Attributes				5		6	1.8
Requirements	Detail Use-Case	Requirements Attributes	1				2	1	0.7
Requirements	Detail Use-Case	Supplementary Specification				4	2		1.0
Requirements	Detail Use-Case	Use-Cases			6	59	4	11	13.3
Requirements	Find Actors and Use-Cases	Supplementary Specification		8					1.0
Requirements	Find Actors and Use-Cases	Use-Case Model	1			1		1	0.5
Requirements	Find Actors and Use-Cases	Use-Cases			12	8			3.3
Requirements	Manage Dependencies	Vision					1		0.2
Requirements	Obtain Client Requirements	Glossary					1		0.2
Requirements	Prioritize Use-Cases	Glossary	1			8	1		1.7
Requirements	Review Requirements	Glossary	2						0.3
Requirements	Review Requirements	Software Requirements Spe.						2	0.3
Requirements	Review Requirements	Use-Case Model	1			2			0.5
Requirements	Review Requirements	Use-Cases	1			1			0.3
Requirements	Structure The Use-Case Model	Use-Case Model		12	3	15	6	6	7.0
Test	Design Test	Test Acceptation		1	2	2			0.8
Test	Design Test	Test Procedures	3	6	1	5	2		2.8
Test	Execute Test	Test Acceptation		1				1	0.3
Test	Implement Test	Test Component				7	2	1	1.7
Test	Implement Test	Test Procedures			2	7	19	4	5.3
Test	Implement Test	Test Script				1		1	0.3
Test	Implement Test Comp. and Subs.	Test Component					1		0.2
Test	Perform Unit Tests	Test Component		15		1	3	1	3.3
CF	Any Activity-CF	Any Artifact-CF	40	59	61	64	56	76	63.0
*Moyenne=Somme des Jetons/6									

Tableau 3.11: Artefacts Nécessaires au Processus.

1. Any Artifact	9. Implementation Model	17. Test Component
2. CM Plan	10. Integration Build Plan	18. Test Procedures
3. Component	11. Measurement Plan	19. Test Script
4. Design Class	12. Project Plan	20. Use-Case
5. Design Package	13. Requirements Attributes	21. Use-Case Model
6. Development Case	14. Risk List	22. Use-Case Realization
7. Development-Organization Assessment	15. Status Assessment	23. Vision
8. Glossary	16. Supplementary Use-Case	

Tableau 3.12: Règles combinatoires.

A&D	Architectural Design	Design Class	A&D	Class Design	Design Class
A&D	Architecture Analysis	Use-Case Realizations	A&D	Use-Case Design	Use-Case Realizations
A&D	Identify Design Elements	Design Package	A&D	Class Design	Design Class
A&D	Identify Design Elements	Interface	A&D	Class Design	Use-Case Realizations
A&D	Review the Design	Review-AD	A&D	Class Design	Design Class
A&D	Subsystem Design	Interface	A&D	Use-Case Design	Use-Case Realizations
A&D	Use-Case Analysis	Use-Case Realizations	A&D	Use-Case Design	Use-Case Realizations
Implementation	Fix Defect	Component	Test	Implement Components	Component
Implementation	Implement Classes	Component	Implementation	Implement Components	Component
Implementation	Implement User Interface	Interface	Test	Implement Components	Component
Implementation	Integrate System	Build	Test	Implement Components	Component
Implementation	Perform Unit Test	Unit Tests	Test	Implement Components	Component
Implementation	Plan System Integration	Integration Build Plan	Implementation	Implement Components	Component
Implementation	Review Code	Component	Implementation	Implement Components	Component
PM	Define Monit. & Ctrl Procedures	Project Plan	PM	Develop Project Plan	Project Plan
PM	Develop Iteration Plan	Iteration Plan	PM	Develop Project Plan	Project Plan
PM	Develop Meas. Plan	Project Plan	PM	Develop Project Plan	Project Plan
PM	Develop Meas. Plan	Project Plan	PM	Develop Project Plan	Project Plan
Requirements	Capture a Common Vocabulary	Glossary	Requirements	Detail Use-Case	Use-Cases
Requirements	Detail the Soft. Requirements	Requirements Attributes	Requirements	Detail Use-Case	Supplementary Specification
Requirements	Detail Use-Case	Requirements Attributes	Requirements	Detail Use-Case	Supplementary Specification
Requirements	Find Actors & Use-Cases	Supplementary Specification	Requirements	Detail Use-Case	Supplementary Specification
Requirements	Find Actors & Use-Cases	Use-Case Model	Requirements	Structure The Use-Case Model	Use-Case Model
Requirements	Find Actors &	Use-Cases	Requirements	Detail Use-Case	Use-Cases

	Use-Cases				
Requirements	Obtain Client Requirements	Glossary	Requirements	Detail Use-Case	Use-Cases
Requirements	Prioritize Use-Cases	Glossary	Requirements	Detail Use-Case	Use-Cases
Requirements	Review Requirements	Glossary	Requirements	Detail Use-Case	Use-Cases
Requirements	Review requirements	Supplementary Specification	Requirements	Detail Use-Case	Supplementary Specification
Requirements	Review Requirements	Use-Case Model	Requirements	Structure The Use-Case Model	Use-Case Model
Test	Design Test	Test Acceptation	Test	Design Test	Test Plan
Test	Design Test	Test Procedures	Test	Design Test	Test Plan
Test	Execute Test	Test Acceptation	Test	Execute Test	Test Plan
Test	Impl. Test Compts & Subs.	Test Component	Test	Design Test	Test Plan
Test	Implement Test	Test Component	Test	Design Test	Test Plan
Test	Implement Test	Test Procedures	Test	Design Test	Test Plan
Test	Implement Test	Test Script	Test	Design Test	Test Plan
Test	Perform Unit Tests	Test Component	Test	Execute Test	Test Plan
Test	Plan Test	Test Plan	Test	Design Test	Test Plan

En dépit du nombre considérable de règles de combinaison d'artefacts, seule une faible proportion de ces règles a été utilisée. Ceci s'explique par le fait que lorsque le deuxième processus de V&V a été entrepris, plusieurs des artefacts ont été corrigés par l'entremise du champ de discussion.

Le Tableau 3.13 illustre le regroupement de tous les artefacts suite au deuxième processus de V&V. Suite à ces modifications, on peut noter une base de comparaison accrue au niveau des artefacts de chaque équipe, visant ainsi à faciliter l'analyse des données. Le deuxième processus de V&V a donc nécessité plusieurs manipulations des données, ce qui peut être considéré comme une source d'erreurs puisque ces changements ont été entrepris de façon subjective. Cette observation sera prise en considération lors de l'analyse finale des données.

Tableau 3.13: Distribution des Éléments du Processus UPEDU, 2ième V&V.

Requirements	Detail Use-Case	Supplementary Specification	4	6	7	9	7	9	7.0
Requirements	Detail Use-Case	Use-Cases	2	12	27	61	7	19	12.8
Requirements	Structure The Use-Case Model	Use-Case Model	2	12	5	9	7	3	6.3
A&D	Class Design	Design Class	13	43	16	44	23	4	17.0
A&D	Use-Case Design	Use-Case Realizations	1	16	15	45	19	16	15.0
Implementation	Implement Components	Component	38	124	70	117	60	80	59.7
Test	Design Test	Test Plan	3	10	6	11	19	1	4.3
Test	Execute Test	Test Plan	0	17	3	12	6	5	4.0
PM	Develop Project Plan	Project Plan	0	13	2	27	11	16	4.2

*Moyenne=Somme des Jetons/6

3.8. Résultats et Analyses Statistiques de la V&V par Équipe

Cette section porte sur les résultats et analyses statistiques des jetons de chacune des équipes suite au deuxième processus de V&V. Le

Tableau 3.14 illustre ces résultats statistiques. Les prochaines sous-sections sont consacrées à l'interprétation du tableau et des résultats obtenus.

3.8.1. Nombre Total Travaillé

Ce champ représente le nombre total d'heures ou de jetons travaillés brutes, c'est-à-dire avant correction ou modification, faisant partie de la base de données initiale des équipes. Ces valeurs sont utilisées comme référence pour calculer les pourcentages des autres champs.

3.8.2. Nombre Total Accepté

Ces valeurs représentent les nombres d'heures ou de jetons qui ont été acceptés suite au processus de V&V et qui seront utilisés pour fin d'analyse. Rappelons que ce champ peut inclure des jetons qui ont été modifiés et/ou corrigés. Tel que mentionné dans une section

précédente, les heures consacrées au workflow Cours et Formation ne sont pas incluses dans ce champ.

En consultant le tableau, il s'avère intéressant de noter que pour un même projet, le nombre d'heures consacré par étudiant à la réalisation du produit varie entre 47 heures pour l'équipe 2 et 226 heures pour l'équipe 5. Seules les équipes 4, 6 et 7 semblent avoir nécessité un nombre similaire d'heures de travail pour accomplir le projet, soit: 90, 114 et 99 heures respectivement. Étant donné que nous présumons que tous les étudiants ont le même niveau d'expertise dans le domaine du génie logiciel, il semble douteux d'assumer la qualité du produit de l'équipe 2 comme étant acceptable. À l'autre extrême, l'investissement de l'équipe 5 se chiffre à 226 heures, ce qui représente une différence de plus de 480% par rapport à l'équipe 2 et approximativement 220% par rapport aux équipes 4, 6 et 7. Clairement, la contribution de l'équipe 5 à l'élaboration du produit est considérable. Il aurait été intéressant ici de connaître la charge académique de chaque étudiant afin d'évaluer si cette charge a un impact sur le nombre d'heures consacrées au processus.

Tableau 3.14: Résultats Statistiques de la V&V Finales.

Nombre Total Travaillé:	325:30	61:22		114	29	
Nombre Total Accepté:	180:45	47:41	59%	63	16	55%
Nombre d'erreurs de Chevauchement :	01:00	00:15	1%	1	0	2%
Nombre d'erreurs de Workflows et/ou Activités:	14:00	03:30	7%	5	1	6%
Nombre d'erreurs d'artefacts:	145:15	39:18	79%	49	12	79%
Nombre Total de CF:	98:00	24:00	29%	37	9	32%
Nombre d'erreurs de duplicata:	18:30	04:37	6%	6	2	5%
Nombre d'erreurs de Chevauchement :	06:30	01:22	2%	2	1	2%
Nombre d'erreurs: Artefacts/Activités/Workflows :	11:45	02:56	4%	5	1	4%
Nombre d'erreurs de temps ou autres:	03:00	00:45	1%	1	0	1%

Nombre Total Travaillé:	1404:12	234:02		417	70	
Nombre Total Accepté:	912:07	152:01	65%	253	42	61%
Nombre d'erreurs de Chevauchement :	20:35	03:25	2%	5	1	2%
Nombre d'erreurs de Workflows et/ou Activités:	80:48	10:08	7%	14	2	6%
Nombre d'erreurs d'artefacts:	578:35	96:25	69%	163	26	60%
Nombre Total de CF:	174:45	29:07	12%	94	16	23%
Nombre d'erreurs de duplicata:	251:20	41:53	18%	58	10	14%
Nombre d'erreurs de Chevauchement :	21:00	03:30	1%	4	1	1%
Nombre d'erreurs: Artefacts/Activités/Workflows :	03:30	00:35	0%	2	0	0%
Nombre d'erreurs de temps ou autres:	41:30	06:55	3%	5	1	1%

Nombre Total Travaillé:	708:15	117:02		215	36	
Nombre Total Accepté:	542:45	90:27	77%	151	25	71%
Nombre d'erreurs de Chevauchement :	19:00	03:10	4%	8	1	5%
Nombre d'erreurs de Workflows et/ou Activités:	97:30	16:15	18%	36	5	24%
Nombre d'erreurs d'artefacts:	115:30	19:15	21%	44	7	29%
Nombre Total de CF:	123:00	20:30	17%	45	8	21%
Nombre d'erreurs de duplicata:	00:00	00:00	0%	0	0	0%
Nombre d'erreurs de Chevauchement :	16:30	02:45	2%	9	2	4%
Nombre d'erreurs: Artefacts/Activités/Workflows :	00:00	00:00	0%	0	0	0%
Nombre d'erreurs de temps ou autres:	24:00	04:00	3%	8	1	4%

Nombre Total Travaillé:	1464:30	261:54		459	82	
Nombre Total Accepté:	1131:05	226:13	77%	335	67	73%
Nombre d'erreurs de Chevauchement :	00:00	00:00	0%	0	0	0%
Nombre d'erreurs de Workflows et/ou Activités:	206:00	41:12	18%	73	15	22%
Nombre d'erreurs d'artefacts:	697:15	139:27	62%	163	33	49%
Nombre Total de CF:	155:40	31:08	11%	66	13	14%
Nombre d'erreurs de duplicata:	144:05	28:49	10%	41	9	9%
Nombre d'erreurs de Chevauchement :	00:00	00:00	0%	0	0	0%
Nombre d'erreurs: Artefacts/Activités/Workflows :	12:10	02:26	1%	11	2	2%
Nombre d'erreurs de temps ou autres:	16:30	03:18	1%	7	1	2%

Nombre Total Travaillé:	758:32	151:06		219	44	
Nombre Total Accepté:	572:47	114:33	76%	159	32	73%
Nombre d'erreurs de Chevauchement :	08:00	01:36	1%	4	1	3%
Nombre d'erreurs de Workflows et/ou Activités:	50:05	10:01	9%	21	4	13%
Nombre d'erreurs d'artefacts:	157:29	31:29	27%	52	10	33%
Nombre Total de CF Relus:	147:15	29:27	19%	50	10	23%
Nombre d'erreurs de duplicata:	04:00	00:48	1%	2	0	1%
Nombre d'erreurs de Chevauchement :	08:00	01:36	1%	2	0	1%
Nombre d'erreurs: Artefacts/Activités/Workflows :	03:00	00:36	0%	1	0	0%
Nombre d'erreurs de temps ou autres:	20:30	04:06	3%	5	1	2%

Nombre Total Travaillé:	695:50	149:18		263	44	
Nombre Total Accepté:	595:55	99:19	67%	153	26	58%
Nombre d'erreurs de Chevauchement :	03:15	00:32	1%	2	0	1%
Nombre d'erreurs de Workflows et/ou Activités:	65:30	10:55	11%	28	4	17%
Nombre d'erreurs d'artefacts:	322:55	53:49	54%	82	14	54%
Nombre Total de CF Relus:	208:10	34:51	23%	79	13	30%
Nombre d'erreurs de duplicata:	22:45	03:47	3%	8	1	3%
Nombre d'erreurs de Chevauchement :	33:15	05:32	4%	9	2	3%
Nombre d'erreurs: Artefacts/Activités/Workflows :	08:30	01:05	1%	6	1	2%
Nombre d'erreurs de temps ou autres:	28:15	04:42	3%	8	1	3%

3.8.3. Nombre Total Corrigé et Accepté

Ces champs (nombre total corrigé et accepté #1 et nombre total corrigé et accepté #2) représentent tous les jetons qui ont été corrigés puis acceptés. Tel que nous pouvons le constater dans le tableau, bien que l'appellation s'avère identique pour ces deux champs, ils possèdent des pourcentages différents. Le calcul du premier pourcentage (#1) tient compte du nombre total d'heures travaillées tel que mentionné par les étudiants (Nombre Total Travaillé), tandis que le deuxième (#2) s'appuie plutôt sur le nombre total des heures acceptées suite à notre processus de vérification et de validation (Nombre Total Accepté).

À première vue, nous remarquons qu'en moyenne plus de 50% des jetons de chaque équipe ont du être corrigés. En effet, les équipes 4 et 6 possèdent plus de 43% et 38% d'erreurs respectivement et les équipes 2, 3, 5, et 7 ont en moyenne plus de 75% erreurs. En d'autres mots, un très large pourcentage des données exigeait une modification, ce qui est très significatif. Tel que nous l'avons mentionné dans une section précédente, il paraît probable que l'introduction d'erreurs additionnelles causées par le biais naturel puisse contribuer à expliquer ce phénomène. De plus, en prenant en considération tous ces hauts pourcentages d'erreurs, il semble indiqué de présumer que le processus UPEDU n'était pas bien compris par les étudiants. En conséquence, il sera fondamental d'être très judicieux lors de l'interprétation des résultats au moment de l'application du PQAL à cette base des données.

La nature des erreurs émises par les étudiants est représentée par les trois prochains champs: le Nombre D'erreurs de Chevauchement, le Nombre D'erreurs de Workflows et/ou d'artefacts et le Nombre D'erreurs d'artefacts.

3.8.4. Nombre D'erreurs de Chevauchement

Comme nous l'avons expliqué ultérieurement, lorsque l'on inscrit 12:00 comme heure de départ et 15:00 comme heure de fin pour un jeton et que nous insérons 14:00 comme heure de départ et 17:00 comme heure de fin pour un autre jeton, et ce pour la même date, nous

sommes en présence d'un problème de chevauchement. Puisqu'il est peu probable qu'un étudiant puisse exécuter deux activités en même temps, ce problème devait être rectifié. La solution entreprise est celle de modifier l'heure de départ du deuxième jeton par l'heure de fin du premier jeton (e.g.: l'heure de départ du second jeton deviendrait 15:00).

L'inattention de la part des étudiants explique probablement une telle erreur. Tel que nous pouvons le constater dans le Tableau 3.14, ce type d'erreurs ne représente pas un pourcentage significatif de toutes les erreurs corrigées et acceptées.

3.8.5. Nombre D'erreurs de Workflows et/ou d'Artefacts.

Lors du processus de V&V, si un artefact était jugé acceptable mais que le choix de l'activité et/ou du workflow ne correspondait pas à l'artefact, l'activité et/ou le workflow étaient alors corrigés. En s'appuyant sur le processus UPEDU et sur une macro écrit en Visual Basic pour Excel, la correction de ces erreurs devenait entièrement automatisée, visant ainsi à réduire au maximum l'impact du biais naturel. Le tableau indique que ce type d'erreurs était plus significatif pour les équipes 4 et 6 puisque plus de 22% de tous leurs jetons comportait ce problème. Seule l'équipe 3 semble avoir bien compris la relation entre les artefacts, les activités et les workflows puisqu'ils affichent seulement 6% d'erreurs pour ce champ. Nous pouvons présumer que les erreurs de ce type représente un manque de compréhension, de la part des étudiants, du processus UPEDU.

3.8.6. Nombre D'erreurs d'Artefacts

Ce champ représente tous les artefacts qui ont été modifiés selon le processus de V&V décrit dans section précédente. Suite à cette modification, une vérification supplémentaire des activités et workflows a été réalisée afin d'assurer une conformité conformes avec le processus UPEDU. Ironiquement, l'équipe 3, qui détenait le meilleur pourcentage d'erreurs en ce qui concerne le nombre d'erreurs de workflows et/ou d'artefacts laissant sous-entendre une bonne compréhension du processus UPEDU, a présenté le plus grand

nombre d'erreurs d'artefacts, i.e., 60%. Cette valeur semble signifier, qu'en dépit d'une bonne compréhension de la relation entre les artefacts/activités/workflows, que la sélection des artefacts était mal saisie par les étudiants de cette équipe. En consultant le tableau, ce commentaire paraît être applicable pour chacune des équipes. Il est à noter que les équipes 4 et 6 possèdent le plus bas pourcentage, toutefois, il faut réitérer que le champ de Description (e.g.: DS) était vide. Ce pourcentage aurait possiblement augmenté en présence de commentaires.

Contrairement au champ précédent, l'impact du biais naturel lors des corrections des artefacts est significatif. En effet, tous ces changements ont été réalisés de façons manuelle et subjective. Même si plusieurs artefacts étaient clairement erronés, il est donc important d'être vigilant lors de l'interprétation de résultats statistiques pour ce champ. Néanmoins, même en tenant compte du biais, le pourcentage d'erreurs demeure significatif, ce qui nous porte à croire que les étudiants semblent avoir manifesté une difficulté à sélectionner correctement les artefacts appropriés. Il va sans dire que ce facteur devra être pris en considération lors de l'interprétation des résultats de l'application du PQAL.

3.8.7. Nombre Total de CF Refusé

Suite au processus de V&V, tous les jetons faisant référence au workflow Cours et Formation (e.g.: les cours et laboratoires) ont été éliminés de la base de données puisque ce workflow devait être similaire pour chacune des équipes. Toutefois, en examinant les pourcentages pour ce champ, nous remarquons que les valeurs ne sont pas identiques pour chaque équipe. Ceci s'explique par le fait que la présence des étudiants aux cours et laboratoires n'était pas obligatoire.

3.8.8. Nombre D'erreurs de Duplicata

Tel que, d'écrit dans le processus de V&V, tout duplicata était éliminé de la base de données. Ce problème s'avérait particulièrement significatif pour les équipes 3 et 5. Une

erreur dans le transfert de la base de données pour ces équipes ou un manque d'attention de la part des étudiants pourraient être à l'origine de ce problème.

3.8.9. Nombre d'Erreurs de Chevauchement

Contrairement aux erreurs de chevauchement décrites à la section 3.8.4, si les heures de départ et de fin d'un jeton A se trouvaient toutes deux à l'intérieur d'un autre jeton B, et ce pour la même journée, ce jeton A était éliminé. Tel que nous pouvons le constater, cette erreur, qui était probablement associée à l'inattention de la part des étudiants, ne compte que pour une faible proportion des jetons refusés.

3.8.10. Nombre d'Erreurs : Artefacts/Activités/Workflow

Parmi les artefacts, activités ou workflows, quelques uns ne faisaient pas partie du processus UPEDU ou ne pouvaient être modifiés selon les règles d'agglomération, ces jetons étaient alors éliminés. En effet, certains étudiants ont réussi à modifier le choix d'artefacts, d'activités ou de workflows de l'outil de jetons afin d'inclure des choix ne faisant pas partie du processus UPEDU. Toutefois, cette catégorie d'erreurs demeure négligeable.

3.8.11. Nombre d'Erreurs de Temps et Autres

Cette catégorie consistait à éliminer tout jeton possédant des erreurs de temps (e.g.: les champs de Date et/ou Durée manquent) ou des erreurs qui ne pouvait être rectifiées par les règles du processus de V&V. Selon les statistiques, nous remarquons encore une fois, que cette catégorie d'erreurs n'était pas significative.

3.9. Vue d'Ensemble de l'Analyse V&V

La section précédente illustrait les valeurs statistiques du processus de V&V, maintenant, nous allons poursuivre l'analyse des données en offrant quelques représentations graphiques de ces données. Deux graphiques ont été générés: la distribution des heures

acceptées et refusées par étudiant et la répartition des heures totales pour chacune des équipes.

3.9.1. Distribution des Heures Acceptées et Refusées par Étudiant

Afin de mieux comprendre la participation de chaque étudiant durant le projet, nous avons créé les graphiques qui sont illustrés dans la Figure 3.3. À partir des heures totales acceptées nous avons calculé la contribution (en pourcentage) au projet de chaque étudiant, de plus, nous avons estimé le pourcentage des heures qui ont été rejetées par étudiants. Ce dernier pourcentage est représenté d'une part, par le nombre de duplicata et d'autre part, par les erreurs de chevauchements, artefacts/activités/workflows et autres.

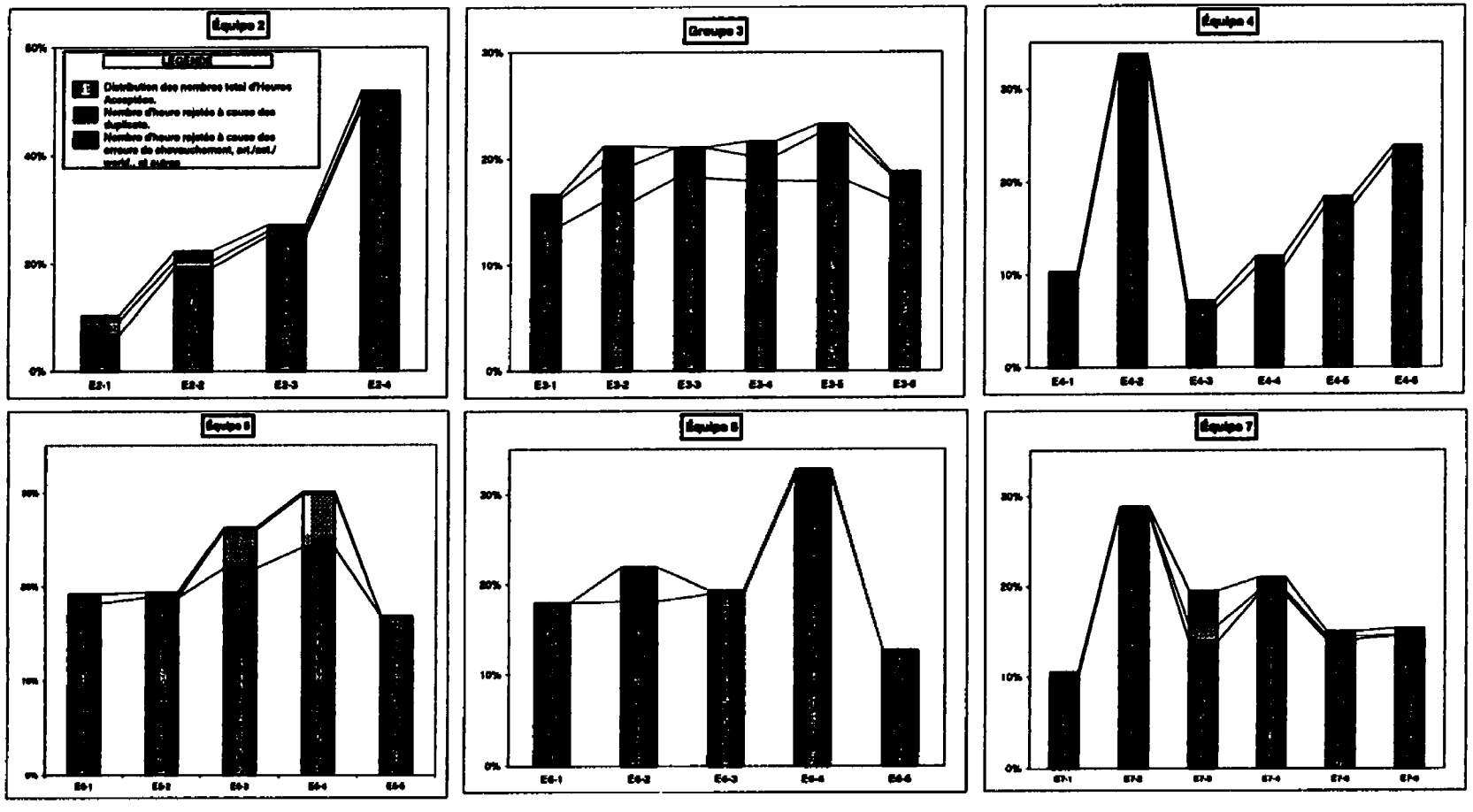
Afin de mieux comprendre la distribution du temps travaillé parmi les étudiants d'une même équipe, nous avons utilisé l'opération statistique de niveau de confiance à 5%. Les résultats sont représentés dans le Tableau 3.15.

Tableau 3.15: Distribution du Temps Travaillé par Équipe.

Équipe	Niveau de Confiance (0.05)
2	14.2%
3	1.4%
4	8%
5	2.6%
6	6.4%
7	5.1%

Ce tableau nous démontre que le nombre d'heures investies par les étudiants des équipes 3 et 5 est uniforme. En d'autre mots, les étudiants de ces équipes ont participé équitablement au projet. Toutefois, cette similitude n'est pas partagée par les autres équipes, et ce particulièrement pour les équipes 2 et 4, où nous remarquons, d'après la Figure 3.3, que certains étudiants ont investi plus de temps que d'autres sur le projet. Cette distribution sera utilisée à la fin de notre étude afin de voir s'il existe en fait une corrélation entre la distribution de l'effort parmi les membres d'une équipe et la qualité finale du produit.

Figure 3.3: Représentation des Heures Acceptées et Refusées par Étudiant.



Tel que nous l'avons souligné à la section précédente, le problème de duplicata s'avérait significatif pour les équipes 3 et 5. En examinant la Figure 3.3, la distribution des erreurs pour l'équipe 3 est plutôt uniforme pour tous les étudiants, ce qui pourrait signifier un problème dans le transfert de la base de données. Toutefois, en ce qui concerne l'équipe 5, la distribution des duplicata est partagée seulement entre deux étudiants, à ce moment nous pourrions aussi stipuler que les duplicata ont été entraînés par un problème de transfert, mais il est également probable que l'erreur a été causée par l'étudiant même. Finalement, la distribution des erreurs causées par la catégorie des chevauchements, artefacts/activités/workflows ou autres, semble homogène parmi les étudiants.

3.9.2. Répartition des Heures Totales de Chaque Équipe

La Figure 3.4 représente la distribution, en heures, des jetons sans erreur, corrigés et refusés inclus dans la base de données initiale de chacune des équipes. De plus, pour fin de comparaison, cette distribution est normalisée par le nombre d'étudiants dans l'équipe et n'inclut pas les jetons du workflow Cours et Formation.

Tel que nous l'avons mentionné à la section précédente, la Figure 3.4 démontre clairement que les équipes 3 et 5 sont celles qui ont investi le plus de temps au projet. Toutefois, contrairement aux équipes 4, 6 et 7, dont le nombre d'heures investies dans le projet semble être représentatif d'une moyenne, les équipes 2, 3 et 5, quant à elles, représentent les valeurs extrêmes de la distribution (i.e.: inférieur pour l'équipe 2 et supérieur pour les équipes 3 et 5). À la lumière de ce graphique, nous pouvons questionner les raisons qui ont engendré ces valeurs extrêmes. Sans une discussion avec les étudiants, il s'avère toutefois difficile d'expliquer pourquoi certains étudiants ont travaillé plus fort que d'autres.

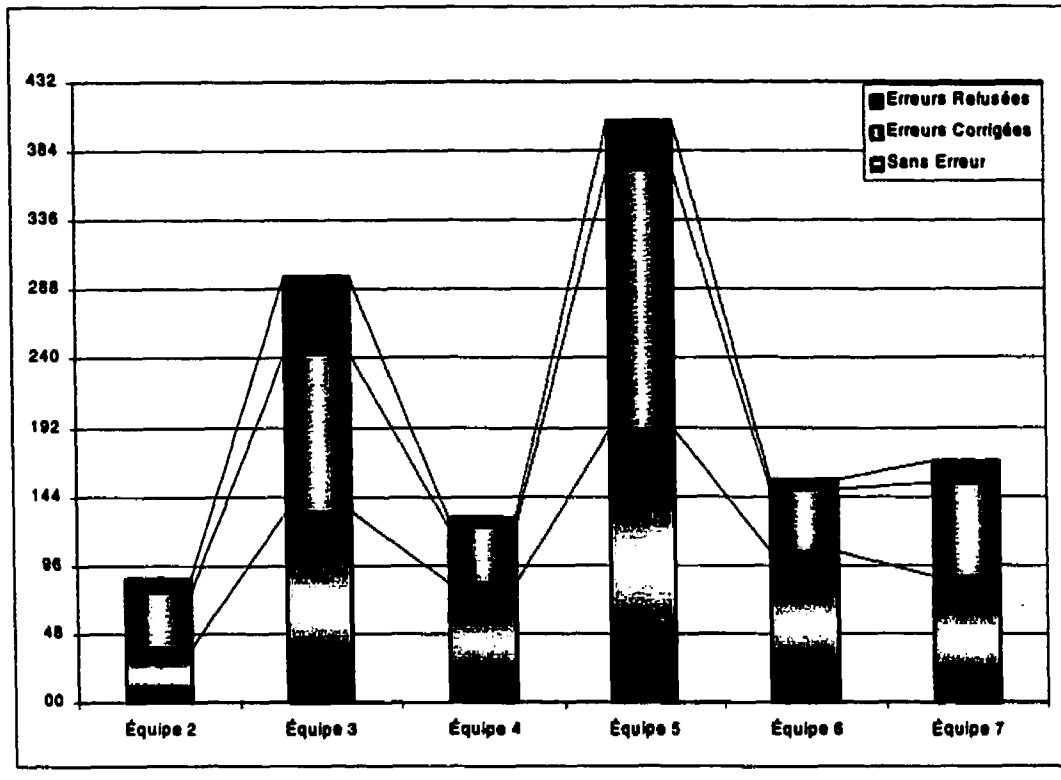


Figure 3.4: Répartition des Heures Normalisées Totales par Équipe

3.10. Discussion du Processus de Vérification et de Validation

Ainsi, ce chapitre portait sur le processus de vérification et de validation de la base des données utilisée en vue de l'application du PQAL. S'inspirant des critères émis par Florac et Carleton (1999), nous avons démontré le processus de vérification et de validation entrepris en vue de rendre l'analyse de la base de données plus crédible et significative.

Tel que mentionné, plusieurs corrections et modifications ont été nécessaires afin de respecter les critères de V&V. À la lumière des statistiques résumant le processus de V&V, nous pouvons constater un manque flagrant de compréhension, par toutes équipes, du processus UPEDU, lequel devait servir de base aux étudiants dans l'élaboration de leur projet. Nous avons également remarqué, et ce pour toutes les équipes, une difficulté associée à l'utilisation de l'outil de jetons créé pour les étudiants. Ces observations

corroborent les conclusions de d'autres auteurs (Bradac et al., 1994; Perry et al., 1994; Robillard, 2000) qui ont également relevé des erreurs multiples reliées à l'enregistrement de jetons relatifs à l'effort. Par exemple, Robillard (2000) soulève des problèmes relatifs à l'ambiguïté des jetons, l'invalidité des données, de même que la non-conformité de l'utilisation des jetons et des intervalles de temps. Rappelons que les mêmes problèmes se sont présentés pour notre base de données.

Bien que de telles corrections et modifications s'avèrent essentielles pour réduire au maximum les erreurs contenues dans la base de données brutes, elles peuvent néanmoins entraîner d'autres types d'erreurs (i.e., biais naturel). Soulignons que l'utilisation des critères suggérés par Florac et Carleton visait à promouvoir l'objectivité et la rigueur associées au processus de V&V. Toutefois, les modifications subjectives des artéfacts suites aux commentaires des étudiants ainsi que la nécessité d'agglomérer les résultats pour fins de comparaison, ont entraîné quelques biais qui ne pourraient être validés que par les étudiants mêmes. Comme cette étape s'avère impossible dans le cadre de cette étude, nous ne pouvons que tenir compte de ces limites lors de l'utilisation de la base de données au moment de l'application du PQAL, réalisée dans le prochain chapitre. En effet, en présence de plusieurs erreurs, Robillard (2000) recommande d'être vigilant quant à l'interprétation de la base de données.

CHAPITRE 4
APPLICATION DU PQAL

4.1. Introduction

Tel que détaillé au chapitre antérieur, la base de données utilisée pour l'application du PQAL provient des travaux de six groupes d'étudiants dans le cadre du cours IF501-Atelier de Génie Logiciel. Afin de mesurer le processus de développement logiciel entrepris par les équipes à l'aide du PQAL, deux métriques ont été retenues, celle de la distribution de l'effort et celle de l'évaluation des artefacts. Rappelons que l'effort se caractérise par la distribution des jetons dans le temps par personne, tandis que les artefacts représentent les documents produits suite à des activités, lesquelles ont été remis au professeur à la fin de la session pour fin d'évaluation.

Ce chapitre porte donc sur l'application du PQAL en regard de l'évaluation du processus de développement logiciel à partir de ces deux métriques. D'abord, nous décrirons les étapes entreprises pour utiliser notre modèle face à la métrique de l'effort. Ensuite, nous présenterons, interpréterons et discuterons des résultats de l'analyse de cette première métrique. Nous poursuivrons avec le même procédé pour la seconde métrique. Finalement, nous discuterons de la méthode utilisée afin de considérer la valeur globale du processus de développement logiciel.

4.2. Application du PQAL à la Métrique de l'Effort.

Une fois le processus de Vérification et de Validation des jetons complété, il était alors possible d'utiliser la base de données afin d'évaluer le processus logiciel. Toutefois, avant même d'appliquer le PQAL, il s'avérait nécessaire de répondre aux questions qui suivent:

Quelle hiérarchie devrait-on utiliser?

Quel poids devrait-on accorder aux priorités pour chacun des éléments de la hiérarchie?

Quelle période de temps devrait-on utiliser pour évaluer le processus?

Quels seront nos critères d'intensité?

Quelles sont les plages et les priorités des critères d'intensité?

Les prochaines sous-sections porteront sur la démarche utilisée afin de répondre à toutes ces questions.

4.2.1. Quelle hiérarchie devrait-on utiliser?

Afin d'utiliser le PQAL, il s'avère nécessaire de créer une hiérarchie représentant le système que nous désirons analyser (voir Chapitre 2). La structure du processus UPEDU, servant de base au système en question, comporte, entre autres, trois éléments de modélisations: les workflows, les activités et les artefacts. Puisque ces trois éléments ont été utilisés de façon explicite par toutes les équipes et que les jetons décrivent le temps accordé à chacun de ces éléments, nous avons donc décidé de les utiliser afin de créer notre hiérarchie. La Figure 4.1 représente la hiérarchie qui sera utilisée dans le cadre de ce mémoire.

Le premier niveau représente la vision globale de la hiérarchie, désirant ainsi évaluer le processus de développement logiciel. Le niveau suivant correspond à la distribution des workflows qui composent le processus UPEDU. Le dernier niveau, quant à lui, caractérise la distribution des activités et de leurs artefacts, ces derniers sont placés entre parenthèses, pour chacun des workflows. Puisque nous retrouvons une relation 1:1 (voir chapitre précédent) entre les activités et les artefacts, nous avons opté pour combiner ces deux éléments dans un même niveau. Pour ceux qui connaissent bien le processus UPEDU, ils noteront que le nombre d'activités et d'artefacts est supérieur à celui représenté dans la Figure 4.1. Cette dissonance s'explique par le fait que nous avons

seulement représenté les éléments qui ont été utilisés par toutes les équipes. Bien qu'il aurait été possible de créer une hiérarchie tenant compte de tous les éléments du UPEDU, la valeur des priorités des éléments non utilisés par les équipes aurait été de zéro. En d'autres mots, tout élément ayant un poids de zéro n'apporterait aucune contribution à la hiérarchie.

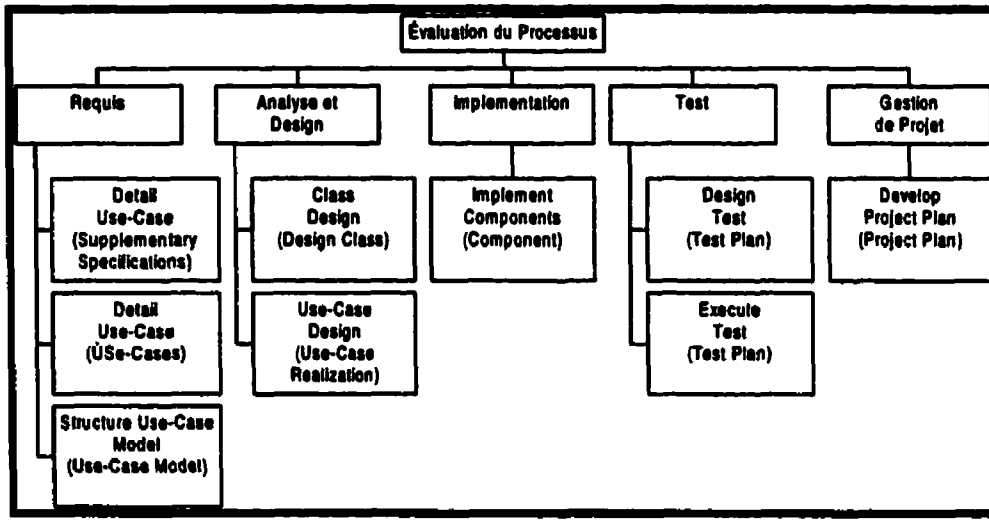


Figure 4.1: Hiérarchie pour l'Évaluation du Processus

4.2.2. Quel poids devrait-on accorder aux priorités pour chacun des éléments de la hiérarchie?

Les jetons représentent la distribution de l'effort pour chaque individu et ce, pour chacun des éléments de modélisation du UPEDU. La distribution des jetons durant le processus du développement logiciel permet d'avoir une vue d'ensemble de l'effort pour chacun des trois éléments de modélisation. Le Tableau 4.1 illustre cette distribution pour chaque équipe.

Tableau 4.1: Distribution de l'Effort des Éléments de Modélisation

Requis	Detail Use-Case	Supplementary Specification	5.8%	2.6%	3.5%	2.5%	3.2%	4.0%
	Detail Use-Case	Use-Cases	2.1%	3.3%	11.1%	10.6%	2.7%	7.5%
	Structure The Use-Case Model	Use-Case Model	2.1%	2.6%	3.1%	1.6%	2.6%	1.1%
Analyse & Design	Class Design	Design Class	19.8%	9.5%	9.4%	11.4%	12.6%	5.5%
	Use-Case Design	Use-Case Realizations	1.0%	5.8%	7.2%	12.5%	9.7%	5.6%
Implémentation	Implement Components	Component	65.0%	66.1%	60.3%	41.9%	52.8%	68.8%
Test	Design Test	Test Plan	4.2%	2.3%	2.6%	7.0%	6.7%	0.3%
	Execute Test	Test Plan	0.0%	5.6%	2.3%	5.6%	4.7%	1.8%
Gestion de Projet	Develop Project Plan	Project Plan	0.0%	2.1%	0.5%	7.0%	5.0%	5.4%
		TOTAL	100%	100%	100%	100%	100%	100%

Sous un autre angle, la Figure 4.2 démontre la distribution graphique des workflows de chaque équipe. Tel qu'il s'avère possible de le constater, la distribution de l'effort n'est pas uniforme entre les workflows. Ceci corrobore l'opinion de Royce (1998) qui explique que la distribution de l'effort varie entre les activités durant le processus de développement logiciel. Toutefois, à notre connaissance, il ne semble exister aucune étude scientifique validant cette déclaration. Même si nous avions accès à de telles études, il demeure peu probable que l'on puisse comparer leurs conclusions à notre distribution puisque plusieurs variables dépendantes (e.g.: la nature du projet, l'expérience des employés, les ressources disponibles, etc.) risquent de modifier ce type de distribution.

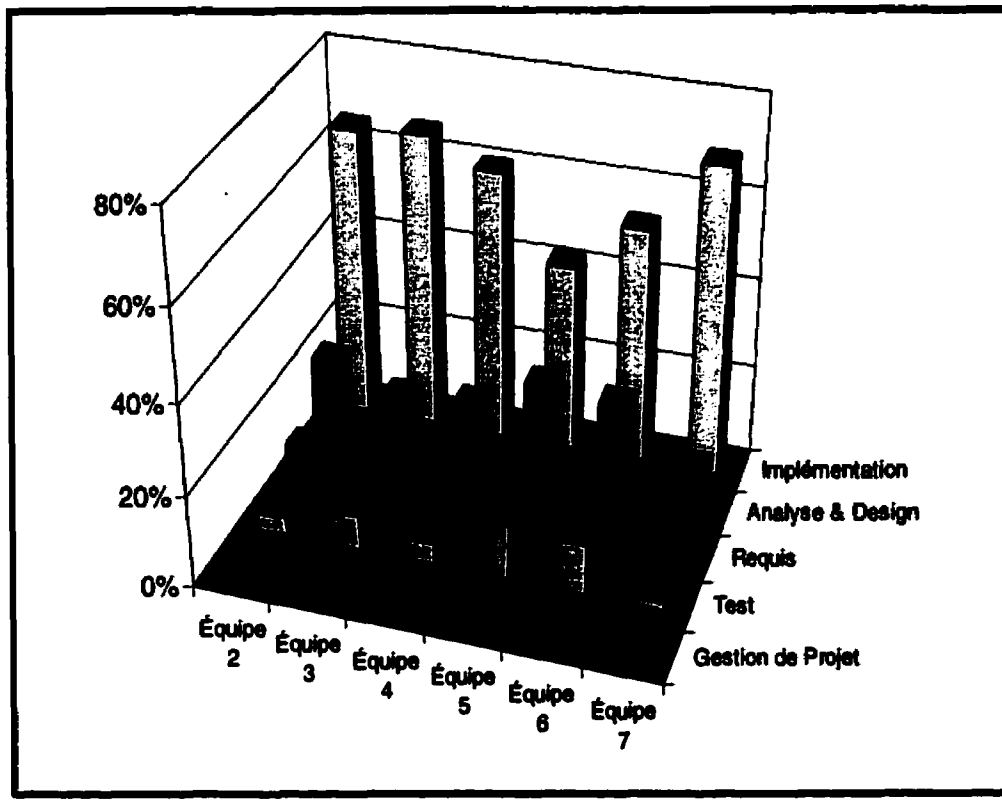


Figure 4.2: Distribution de l'Effort par Workflows par Équipe

Comme l'illustre la Figure 4.2, les pourcentages attribués à ces distributions représentent, en fait, l'importance de chaque workflow durant le processus, c'est-à-dire, que ces pourcentages pourraient être considérés comme étant le poids d'importance de chacun des workflows durant le projet. Tel que nous l'avons mentionné en début de chapitre, afin d'utiliser le PQAL pour l'évaluation du processus, il s'avère fondamental d'accorder des poids relatifs à l'importance que chacun des éléments possède par rapport aux autres. Il serait toujours possible, tel que Saaty (1996) suggère, de faire des comparaisons binaires entre les éléments. Cependant, d'après le Tableau 4.1, l'attribution des poids d'importance de chaque élément semble avoir déjà été déterminée, à posteriori, par les équipes lors de la répartition des éléments de modélisation. Ainsi, nous avons décidé de nous baser sur les données fournies par les étudiants afin d'établir les poids d'importance dans la hiérarchie. Nous avons donc calculé, pour une période donnée, la moyenne des

workflows et des activités/artefacts des six équipes dans le but de déterminer les poids de ces éléments. Par exemple, les priorités des éléments pour la durée complète du processus sont représentées dans le Tableau 4.2 et le Tableau 4.3. On peut noter, à la lumière de ces tableaux, que plus de 50% du temps des étudiants fut consacré à l'écriture du code et moins de 3% à la gestion de projet, ce qui est normal considérant que peu de gestion de projet était requise de la part des étudiants dans le travail exigé. Il est également intéressant d'observer que quelques écarts types s'avèrent très élevés, c'est-à-dire, représentant, dans certains cas, plus de 80% de la moyenne. Ceci semble indiquer la non-homogénéité de notre échantillon.

En résumé, les priorités des éléments de modélisation se feront à partir de la distribution des jetons pour la période de temps que nous désirons analyser. Cette «période» sera déterminée à la prochaine sous-section.

Tableau 4.2: Priorités des Éléments de Modélisation: Workflows.

Workflows	Priorités	
	Moyenne	Écart Type
Requis	12.0%	3.7%
Analyse et Design	18.3%	4.8%
Implémentation	59.2%	10.1%
Test	7.2%	4.2%
Gestion de Projet	3.3%	2.9%

Tableau 4.3: Priorités des Éléments de Modélisation: Artefacts/Activités.

Artefacts	Priorités	
	Moyenne	Écart Type
Supplementary Specification	3.6%	1.2%
Use-Cases	6.2%	4.1%
Use-Case Model	2.2%	0.8%
Design Class	11.4%	4.8%
Use-Case Realizations	7.0%	3.9%
Component	59.2%	10.1%
Test Plan (Design Test)	3.8%	2.6%
Test Plan (Execute Test)	3.3%	2.3%
Project Plan	3.3%	2.9%

4.2.3. Quelle période de temps devrait-on utiliser pour évaluer le processus ?

Une fois la hiérarchie du PQAL et les priorités des éléments déterminées, il devenait important de décider si on allait évaluer le processus de façon globale ou de façon temporelle. Une évaluation globale consiste principalement à regrouper tous les jetons, du début jusqu'à la fin du processus. L'évaluation temporelle, quant à elle, implique une subdivision du cycle de vie en plusieurs sections (e.g., journalière, hebdomadaire ou mensuelle) afin d'évaluer le processus séparément pour chacune de ces sections. L'avantage de cette dernière permettrait d'évaluer l'évolution du processus dans le temps. Puisque le processus s'avère un enchaînement d'activités durant le cycle de vie, où certaines activités peuvent précéder (e.g.: les activités des requis précèdent ceux de l'implémentation) et/ou avoir plus d'importance que d'autres activités, une évaluation globale ne tiendrait pas compte de tels aspects du processus. Pour cette raison, nous avons

opté pour l'évaluation du processus en tenant compte du facteur temporel afin de mesurer l'évolution des éléments de modélisation.

Afin de pouvoir conserver l'aspect temporel tout en examinant la globalité du processus, il serait intéressant d'intégrer ces multiples évaluations dans une autre hiérarchie, telle que nous le démontre la Figure 4.3. Dans un tel scénario, la priorité accordée à chacune de ces évaluations pourrait alors être déterminée par des comparaisons binaires. Dans notre contexte, puisque nous considérerons chaque évaluation comme étant d'égale importance dans le cycle de vie du processus, nous avons décidé d'accorder le même poids à chacune de ces évaluations. La priorité, à ce moment, sera représentée par le ratio du nombre de sections (i.e.: chaque section aura une priorité = 1/n, où n représente le nombre de sections).

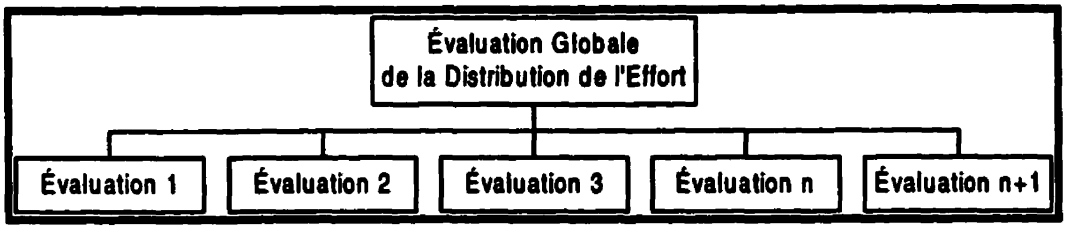


Figure 4.3: Évaluation Globale du Processus.

Une fois notre choix du type d'évaluation fixé, nous devions décider quant à la fréquence du facteur temporel. La granularité d'une évaluation journalière risquerait d'être trop fine puisque les étudiants ne travaillaient pas quotidiennement sur le projet. D'autre part, la granularité d'une évaluation mensuelle pourrait s'avérer trop grande étant donné que la durée globale du projet n'a été que de trois mois. Ainsi, une évaluation mensuelle ne tiendrait pas compte de plusieurs détails importants. Pour ces raisons, nous avons opté pour une granularité hebdomadaire. Tout d'abord, nous avons sectionné le projet en parties égales de 13 semaines, pour ensuite évaluer le processus pour chacune de ces périodes. Nous avons donc regroupé les jetons de façon hebdomadaire pour toutes les équipes. Une analyse graphique de la distribution de l'effort pour chaque équipe nous a

permise d'identifier quelques observations que nous croyons pertinentes de partager lors de la prochaine sous-section.

4.2.3.1. Analyse de la distribution hebdomadaire de l'effort.

Les prochaines six figures démontrent la distribution hebdomadaire des workflows normalisés pour chacune des équipes. À partir de ces graphiques, nous pouvons déjà faire quelques observations quant au déroulement du processus. Par exemple, bien que la distribution de l'effort entre les workflows varie d'une équipe à l'autre, il n'en demeure pas moins que toutes les équipes ont suivi un certain cheminement associés aux différentes étapes de UPEDU. En d'autres mots, ils ont tous débuté par le workflow des requis, suivi de l'analyse et design, puis ont terminé avec le workflow de l'implémentation.

Malgré que le projet est essentiellement le même pour toutes les équipes, il demeure que la «signature» graphique de la distribution de l'effort pour chacun des workflows, s'avère différente pour chacune des équipes. En d'autres mots, chaque équipe a utilisé une combinaison différente d'activités durant le déroulement du processus d'un même projet. Plusieurs facteurs peuvent contribuer à expliquer ce phénomène, par exemple, le nombre d'étudiants par équipe, l'expérience de vie de chaque individu, le nombre de requis implantés, les différentes interprétations des requis entre les équipes, la présence ou non d'un leader, le niveau de compréhension du processus par chaque membre de l'équipe, etc. Bien que la compréhension de ce phénomène et de tous les facteurs pouvant l'influencer surpasse clairement les limites de ce mémoire, nous pourrions tout de même émettre l'hypothèse, même en présence d'un échantillonnage limité, que la distribution de l'effort est unique, non pas au projet, mais à chacune des équipes.

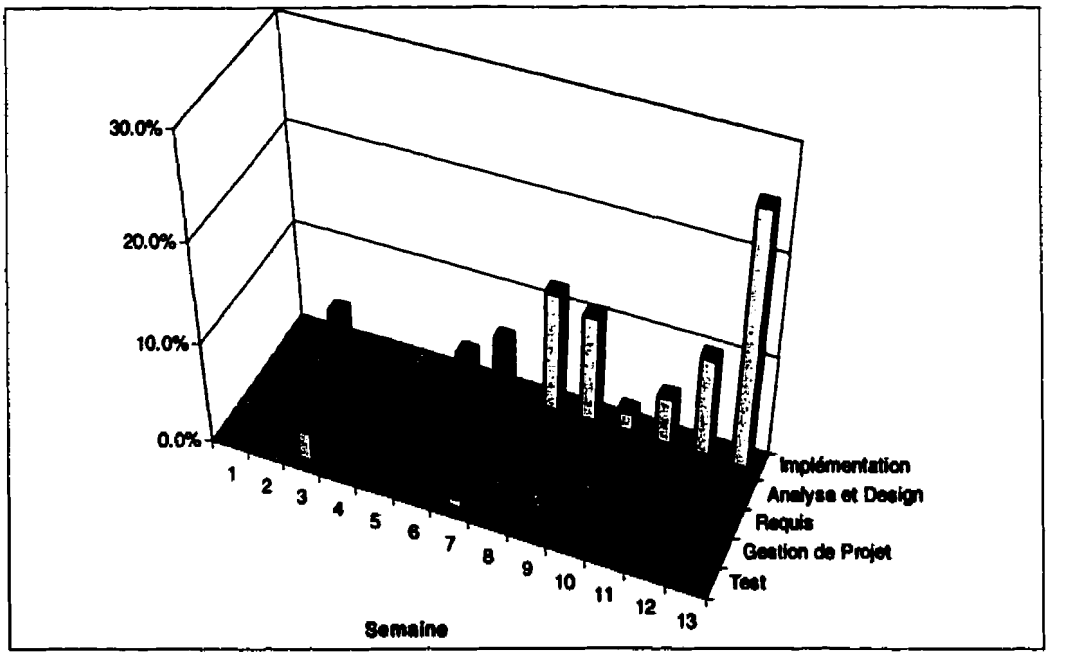


Figure 4.4: Signature du Processus de l'Effort : Équipe 2

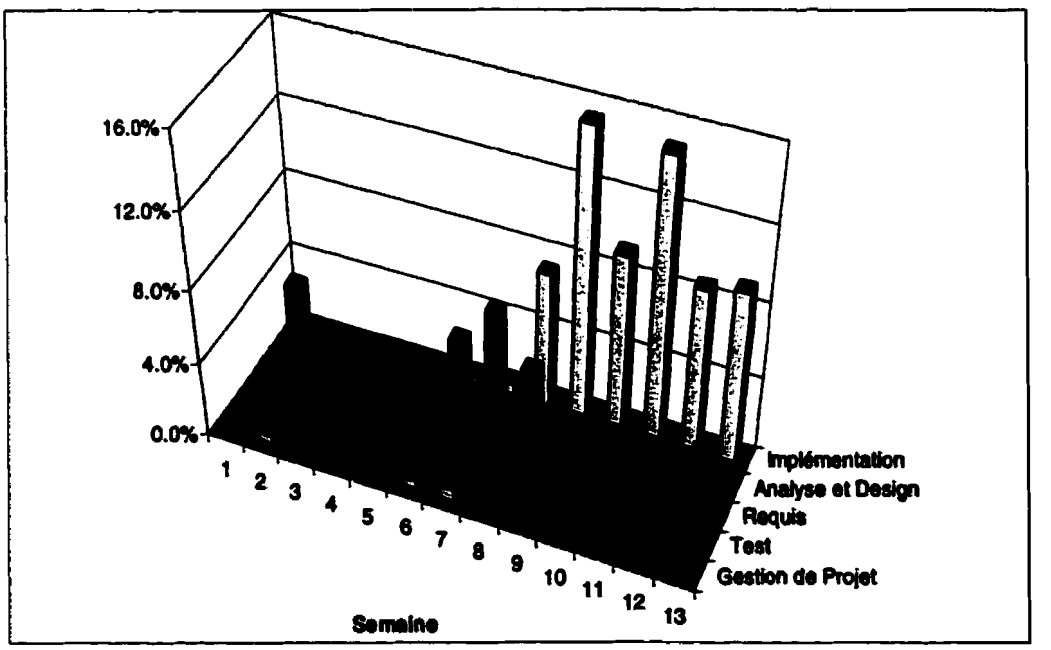


Figure 4.5: Signature du Processus de l'Effort : Équipe 3

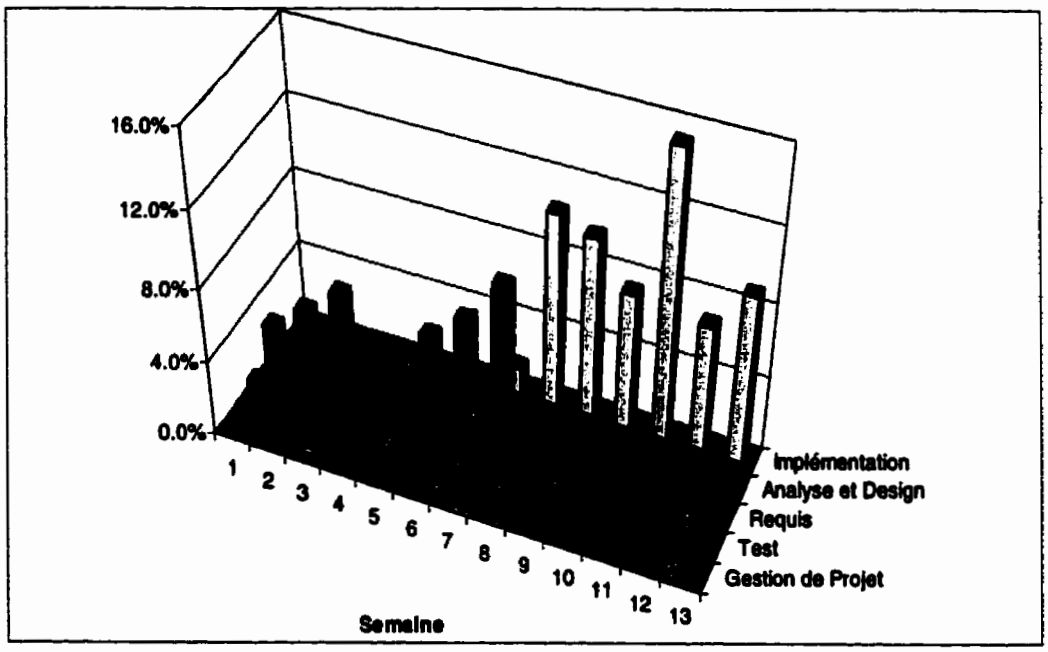


Figure 4.6: Signature du Processus de l'Effort : Équipe 4

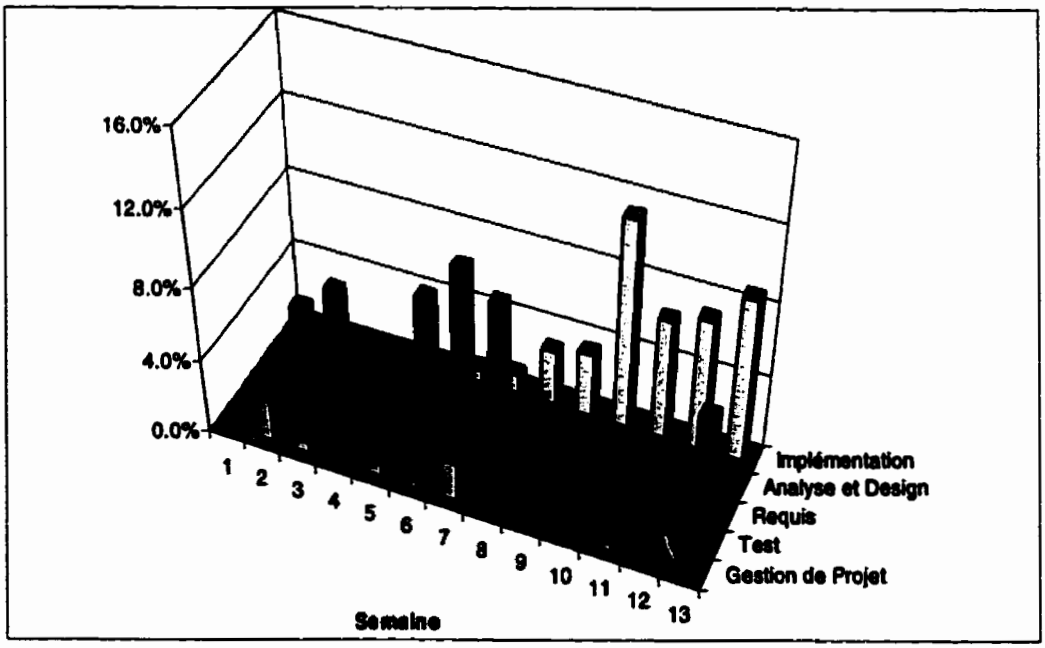


Figure 4.7: Signature du Processus de l'Effort : Équipe 5

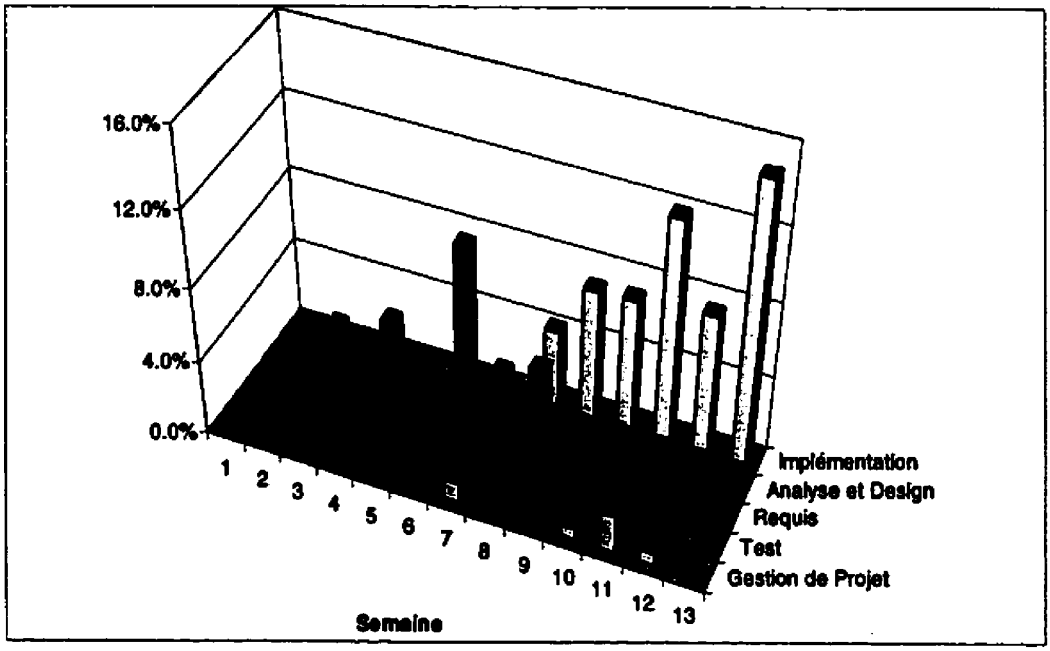


Figure 4.8: Signature du Processus de l'Effort : Équipe 6

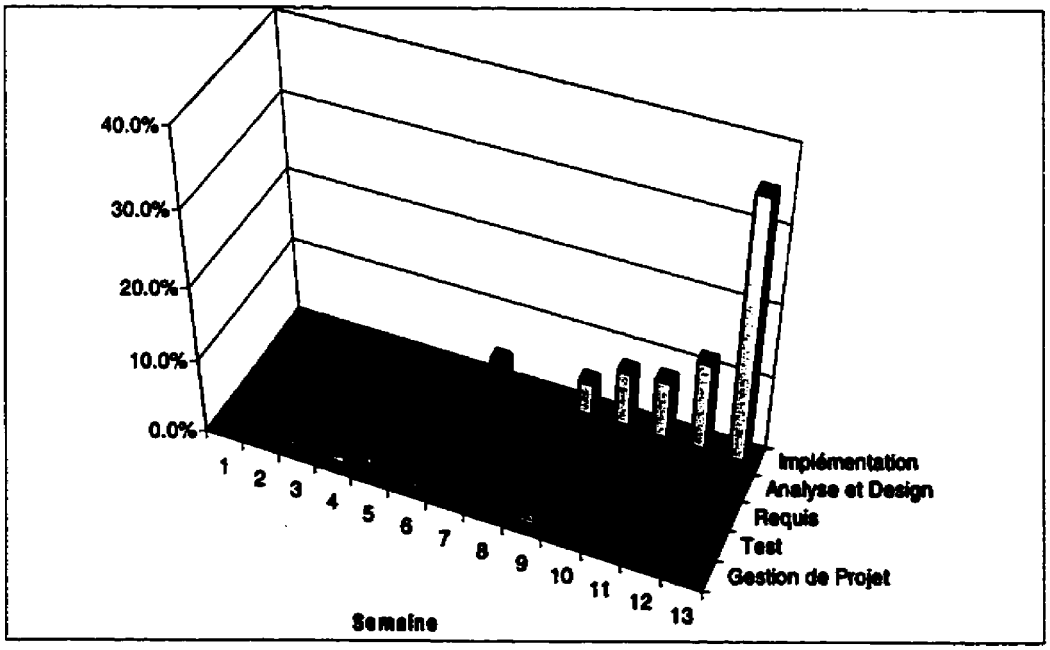


Figure 4.9: Signature du Processus de l'Effort : Équipe 7

D'autres observations possibles à partir de ces graphiques correspondent à la quantité d'effort accordée aux activités de test, de même qu'à la gestion de projet. Ces deux phénomènes ont également été observés par Robillard et Robillard (1998) lors d'une étude comparative entre une équipe d'étudiants et une équipe de professionnels. Ces auteurs expliquent que la diminution d'effort, de la part des étudiants, pour les activités de gestion est probablement reliée à un manque de compréhension de l'activité même. Puisque l'activité de gestion peut être considérée comme une activité de collaboration obligatoire (e.g.: des rencontres formelles planifiées sur une base régulière), Robillard (2000), de même que Robillard et Robillard (1998) notent que ce genre d'activités ne représente que 4% du temps total accordé aux activités de collaboration durant un projet. De plus, ils mentionnent que la nécessité de gestion ne s'avère plus requise lorsque tous les membres de l'équipe connaissent bien leurs tâches à l'intérieur du processus. Quant à l'activité de test, ces auteurs ont également remarqué que cette dernière demeure très faible durant un projet de session. Ces auteurs croient que les étudiants, ayant tendance à coder jusqu'à la fin de leur projet, arrivent à l'échéance du travail avant de pouvoir exécuter les tests. Puisque leur logiciel est considéré comme étant «jetable», les étudiants considèrent peu les avantages que les tests peuvent apporter une fois le logiciel complété.

Dans leur étude, Robillard et Robillard (1998) ont découvert que les étudiants, contrairement aux professionnels, ont souvent tendance à mettre plus d'effort aux activités de codage qu'aux autres activités (e.g.: analyse et design, requis, tests). Par exemple, ces auteurs rapportent que les étudiants ont attribué en moyenne 42% de leur temps au développement, contrairement à seulement 20% pour les professionnels. Ce phénomène s'est clairement reproduit parmi les équipes d'étudiants analysées dans le cadre de ce mémoire. Tel que mentionné lors d'une section précédente, les équipes ont investi plus de 50% de leur temps à l'implémentation. Toutefois, Royce (1998) mentionne que selon son expérience, plus de 65% de l'effort correspond à la phase de construction du cycle de vie du RUP, laquelle consiste principalement aux activités d'implémentation. Quelques explications peuvent sans doute contribuer à comprendre

cette différence marquée entre les résultats de l'étude menée par Robillard et Robillard (1999) et l'intuition de Royce (1998). La première explication, la plus évidente, est que, contrairement à Robillard et Robillard (1998), la distribution des pourcentages de Royce (1998) ne correspond pas aux résultats d'une étude scientifique. Une deuxième explication possible est que ces auteurs se sont basés sur un processus de développement logiciel différent. En effet, le processus utilisé dans l'étude de Robillard et Robillard correspond aux standards du IEEE, contrairement à Royce qui s'est appuyé sur une version du RUP. Ainsi, les activités incluses dans une phase peuvent être différentes d'un processus à un autre. Si la distribution de l'effort s'avère vraiment unique à chaque équipe, on pourrait dire qu'il existe autant de signatures de la distribution de l'effort qu'il existe d'équipes. Donc, il semblerait difficile de généraliser cette distribution à toutes les équipes, encore plus à tous les projets. Puisqu'il existe peu études empiriques sur la distribution de l'effort durant un processus, il s'avère toutefois difficile de supporter cette hypothèse.

4.2.3.2. Calcul des Priorités des Artefacts et Workflows

Puisque nous avons décidé d'évaluer le processus de façon hebdomadaire, il devient nécessaire de calculer, pour chaque semaine, les priorités des éléments faisant partie de la hiérarchie du PQAL () i.e.: les workflows et artefacts/activités. Tel que nous l'avons expliqué à la section , nous avons calculé les priorités de chacun des éléments en prenant les moyennes des équipes. Le Tableau 4.4 illustre les priorités hebdomadaires pour les workflows et le , celles pour les artefacts/activités.

Tableau 4.4: Priorités Hebdomadaires des Workflows.

Semaine	Requis	A&D	Implémentation	Test	PM
2000-01-10	0.71	0.00	0.00	0.27	0.02
2000-01-17	0.86	0.00	0.00	0.00	0.14
2000-01-24	0.72	0.02	0.00	0.23	0.03
2000-01-31	0.33	0.39	0.00	0.18	0.10
2000-02-07	0.11	0.73	0.02	0.08	0.06
2000-02-14	0.00	0.90	0.04	0.02	0.05
2000-02-21	0.03	0.66	0.08	0.15	0.08
2000-02-28	0.00	0.32	0.64	0.01	0.02
2000-03-06	0.00	0.13	0.72	0.06	0.09
2000-03-13	0.00	0.03	0.94	0.01	0.03
2000-03-20	0.00	0.00	0.94	0.02	0.03
2000-03-27	0.00	0.00	0.98	0.00	0.02
2000-04-03	0.00	0.00	0.62	0.38	0.00

Tableau 4.5: Priorités Hebdomadaires des Artefacts/Activités.

Semaine	Detail Use-Case (SS)	Detail UC (UC)	Structure UC Model	Class Design	UC Design	Implement Comp.	Design Test	Execute Test	Develop Project Plan
01-10	0.00	0.65	0.35	0.00	0.00	0.00	1.00	0.00	1.00
01-17	0.10	0.56	0.34	0.00	0.00	0.00	0.00	0.00	1.00
01-24	0.57	0.34	0.09	1.00	0.00	0.00	1.00	0.00	1.00
01-31	0.00	0.79	0.21	0.47	0.53	0.00	0.94	0.06	1.00
02-07	0.00	1.00	0.00	0.54	0.46	1.00	1.00	0.00	1.00
02-14	0.00	0.00	0.00	0.46	0.54	1.00	0.00	1.00	1.00
02-21	0.00	0.52	0.48	0.57	0.43	1.00	0.94	0.06	1.00
02-28	0.00	0.00	0.00	0.62	0.38	1.00	0.00	1.00	1.00
03-06	0.00	0.00	0.00	1.00	0.00	1.00	0.00	1.00	1.00
03-13	0.00	0.00	0.00	1.00	0.00	1.00	1.00	0.00	1.00
03-20	0.00	0.00	0.00	0.00	0.00	1.00	0.00	1.00	1.00
03-27	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00
04-03	0.00	0.00	0.00	0.00	0.00	1.00	0.17	0.83	1.00

4.2.4. Quels seront nos critères d'intensité?

Tel que nous l'avons mentionné durant la discussion du PQAL au Chapitre 2, afin de mesurer l'impact des résultats des métriques sur le but global, il s'avère nécessaire d'établir des niveaux d'intensité pour ces résultats. Une fois ces intensités déterminées, nous pouvons ensuite accorder des priorités à chacune. Cette procédure s'avère

essentielle puisque l'impact des résultats des métriques sur le but global de la hiérarchie dépend directement du poids accordé à l'intensité.

Rappelons que dans notre analyse, nous avons établi une hiérarchie basée sur la distribution d'effort des workflows et des activités/artefacts. La métrique utilisée pour évaluer l'effort de chaque artefact correspond aux jetons enregistrés dans l'outil de jetons utilisé par chaque équipe (voir Chapitre 3). Ces jetons décrivent le temps alloué à chaque artefact par chaque étudiant. Puisque chaque équipe a investi une quantité de temps hebdomadaire différente pour les mêmes artefacts, il s'avère nécessaire de déterminer une plage d'intensités qui accordera un ordre d'importance à ces périodes de temps. En d'autres mots, pour un artefact et une semaine donnés, nous voulons connaître les critères d'intensité ou les plages, qui nous indiqueront si le temps investi par une équipe est idéal, acceptable, pauvre ou inacceptable. Cependant, ceci soulève une question importante. Ainsi, puisqu'un des objectifs de l'évaluation de la distribution de l'effort consiste à comparer les valeurs entre les équipes, il s'avère important d'identifier le type d'échelles de mesure à utiliser. À première vue, les jetons définissent le nombre d'heures passées sur un artefact, ce qui représente une échelle absolue. Toutefois, il est aussi possible de normaliser ces valeurs par le temps global du projet, ce qui correspond une échelle de ratio. Bien que ces deux échelles de mesure soient riches en information (voir Chapitre 2), le type d'information contenue dans chacune s'avère différent. Quelle serait donc la différence entre une comparaison exécutée à partir de valeurs globales basées sur une échelle absolue versus une échelle de ratio ?

Nous avons constaté que les comparaisons basées sur l'échelle absolue ne tiennent pas compte de l'expérience des équipiers. Par exemple, nous savons que l'équipe 2 a investi 325 heures durant le projet contre 1459 heures pour l'équipe 5. Considérant que les équipes sont composées d'étudiants, il semble valable de présumer que le travail de l'équipe 2 s'avère probablement inférieur à celui de l'équipe 5, et ce, avant même d'appliquer le PQAL. En effet, il nous est possible d'émettre une telle supposition

puisque nous présumons que les étudiants ont, en général, à peu près tous le même niveau d'expérience. Toutefois, si ces mêmes valeurs provenaient de deux équipes en milieu industriel, il serait, à ce moment, injustifiable de poser la même hypothèse, puisque nous ignorons le niveau d'expérience des équipiers. Malgré que les valeurs provenant des milieux académique et industriel pourraient être similaires, nous ne pourrions en tirer la même conclusion.

Par conséquent, ceci nous conduit à l'utilisation de l'échelle de ratio. Prenons par exemple la Figure 4.4 et Figure 4.7 qui représentent la distribution hebdomadaire normalisée des workflows pour les équipes 2 et 5 respectivement. Ces figures n'offrent aucune information quant à la durée des workflows. Par conséquent, il serait tout à fait possible que les équipes 2 et 5 possèdent une signature de distribution similaire en dépit d'une différence significative quant au temps investi dans le projet. Donc ce ratio, allant au delà de la quantité même de temps, reflète la distribution de l'effort. Toutefois, il est important de noter que nous perdons de l'information, c'est-à-dire la quantité de temps investie, lorsque nous passons d'une échelle absolue à celle de ratio. Donc, le choix d'échelle dépend essentiellement de ce que l'on veut évaluer. Ainsi, il serait favorable d'utiliser l'échelle absolue pour évaluer la quantité de temps accordée aux artefacts, d'autre part, l'échelle de ratio semble plus avantageuse lorsque nous sommes principalement intéressés à la distribution de l'effort. Mentionnons toutefois qu'il s'avère également possible, à l'aide du PQAL, de combiner les deux types échelles. Considérant que notre intérêt primaire pour ce mémoire consiste à évaluer la distribution de l'effort, nous avons opté pour l'utilisation, comme intensité, de l'échelle de ratio à partir des sommes hebdomadaires normalisées.

Une fois l'échelle de mesure des intensités identifiée, il s'agit ensuite de déterminer les critères d'intensité qui seront utilisés pour comparer les valeurs normalisées des artefacts à une base de référence. Par exemple, quels seront les critères qui, pour un certain artefact, pourront classifier le pourcentage d'effort mesuré pour une semaine donnée,

comme étant un résultat idéal, acceptable ou pauvre? Ayant toujours la même limite, c'est-à-dire le manque d'études scientifiques reliées à la distribution de l'effort durant l'application du processus UPEDU, nous avons décidé de choisir, comme base de référence pour chaque artefact, les valeurs moyennes hebdomadaires de l'effort. En effet, bien qu'il existe plusieurs façons de comparer une valeur mesurée à une valeur idéale, nous avons opté pour un schème utilisant la moyenne (μ) ainsi que les écarts types (σ) des distributions de l'effort pour un artefact et une période de temps donnés. Cette technique permet de déterminer la distance d'une valeur hebdomadaire normalisée d'un artefact quelconque, en écart type, par rapport à la moyenne. Par exemple, si la valeur moyenne d'effort mesuré pour un artefact quelconque correspond à 5% avec un écart type de 2% et que la valeur d'effort mesuré de cet artefact pour l'équipe X est 3%, nous pouvons dire que la distance cet l'artefact est à 1 écart type de la moyenne. Ensuite, nous pouvons établir une plage d'intensité, telle qu'illustrée dans le Tableau 4.6. L'avantage d'utiliser un tel critère d'intensité présume qu'il existe une plage idéale d'effort pour un artefact donné. Si par exemple, une équipe n'investit pas suffisamment d'effort ou investit trop d'effort pour l'artefact en question, l'écart type, dans les deux cas, s'avère significatif. En d'autres mots, investir trop de temps sur un artefact n'est guère mieux que d'en investir insuffisamment.

Tableau 4.6: Exemples de Plages d'intensités Basées sur la distance en Écart type.

Plage Idéale	$(0, 1\,\sigma)$
Plage Acceptable	$(1\,\sigma, 2\,\sigma]$
Plage Minimale	$(2\,\sigma, 3\,\sigma]$
Plage Extrême	$> 3\,\sigma$

Puisque nous désirons mesurer le processus de chaque équipe à toutes les semaines afin de tenir compte de l'évolution temporelle des artefacts, il s'avère nécessaire de calculer la moyenne ainsi que l'écart type de chaque artefact de façon hebdomadaire. Nous avons alors répertorié, dans le Tableau 4.7 et le Tableau 4.8, les valeurs normalisées

hebdomadaires (i.e.: l'effort d'un artefact pour une semaine divisé par la somme totale de l'effort pour la durée du projet) de chacune des équipes. À la suite de cet exercice, nous avons constaté que plusieurs artefacts possédaient des valeurs extrêmes, c'est-à-dire que les valeurs des écarts types dépassaient souvent 75% de la moyenne (e.g.: $\mu = 4$ et $\sigma = 3$), ce qui représente en soit, une grande variation entre les valeurs. Pour remédier à ce problème, nous avons décidé d'éliminer certaines valeurs extrêmes afin d'augmenter l'homogénéité au niveau de la moyenne et de l'écart type. Pour ce faire, nous avons analysé toutes les valeurs hebdomadaires normalisées de chaque artefact en se servant de trois règles d'élimination de valeurs extrêmes, illustrées dans le Tableau 4.9.

Tableau 4.7: Valeurs Normalisées Hebdomadaires pour les Équipes 2, 3 et 4.

Équipe 2									
2000-01-10									
2000-01-17									
2000-01-24							3.1%		
2000-01-31									
2000-02-07				3.8%					
2000-02-14									
2000-02-21							1.0%		
2000-02-28					1.0%				
2000-03-06				0.8%					
2000-03-13									
2000-03-20									
2000-03-27									
2000-04-03									

Équipe 3									
2000-01-10									0.1%
2000-01-17		3.3%	2.6%						0.5%
2000-01-24	2.6%								
2000-01-31									0.2%
2000-02-07							0.3%		
2000-02-14				3.1%		0.3%			0.5%
2000-02-21				2.7%	3.3%	0.7%	1.8%		
2000-02-28				1.9%	1.8%	7.3%			
2000-03-06				0.4%				0.7%	
2000-03-13						9.4%	0.3%		
2000-03-20									
2000-03-27						8.8%			
2000-04-03						9.2%		5.0%	

Équipe 4									
2000-01-10		1.9%	1.1%				1.3%		
2000-01-17		3.8%	0.7%						
2000-01-24	3.5%		1.3%	0.9%					0.4%
2000-01-31									
2000-02-07		1.3%			3.5%				
2000-02-14				3.3%	1.7%				
2000-02-21				5.9%	2.0%	1.5%			
2000-02-28									
2000-03-06									
2000-03-13						7.4%			
2000-03-20									
2000-03-27						6.9%			
2000-04-03						9.0%			

Tableau 4.8: Valeurs Normalisées Hebdomadaires pour les Équipes 5, 6 et 7.

Équipe 5									
2000-01-10		0.29%	0.09%						
2000-01-17	1.24%	3.09%	0.27%						2.34%
2000-01-24		4.07%	0.85%						0.62%
2000-01-31		3.13%	0.36%	0.68%	1.29%				
2000-02-07				3.16%	2.39%		1.08%		0.73%
2000-02-14				1.61%	6.12%	0.73%		0.13%	0.36%
2000-02-21				3.82%	2.74%	1.06%	3.09%	0.40%	2.12%
2000-02-28				1.39%		3.01%		0.27%	
2000-03-06				0.49%		3.45%		0.18%	0.20%
2000-03-13				0.20%		11.23%			
2000-03-20						8.50%		0.35%	0.35%
2000-03-27						7.16%			
2000-04-03						8.80%	2.83%	4.24%	0.18%

Équipe 6									
2000-01-10									
2000-01-17			2.0%						
2000-01-24	3.2%						4.1%		
2000-01-31			0.7%	2.0%	1.7%		2.3%		
2000-02-07						0.3%			
2000-02-14				3.4%	5.5%				0.3%
2000-02-21		0.1%							1.1%
2000-02-28						4.2%			
2000-03-06				1.0%		7.0%		0.3%	
2000-03-13				1.2%		7.1%			0.7%
2000-03-20									
2000-03-27						7.4%			0.7%
2000-04-03								4.4%	

Équipe 7									
2000-01-10									
2000-01-17	0.5%		0.5%						
2000-01-24	3.5%	3.5%							
2000-01-31		0.7%					0.3%	0.2%	1.3%
2000-02-07		0.5%							0.3%
2000-02-14								0.3%	
2000-02-21		0.6%	0.6%	4.3%	4.1%	0.4%			
2000-02-28				1.3%					0.5%
2000-03-06						4.8%			1.7%
2000-03-13						7.8%			0.5%
2000-03-20						7.9%			0.2%
2000-03-27									
2000-04-03									

Tableau 4.9: Règles d'Élimination de Valeurs Extrêmes

Règle 1	Afin d'éliminer des valeurs extrêmes, il est nécessaire que plus de trois valeurs normalisées soient présentes.
Règle 2	Suite à l'élimination de valeurs extrêmes, on doit préserver, au plus, deux valeurs normalisées.
Règle 3	L'élimination des valeurs extrêmes est cessée lorsque l'écart type représente moins de 25% de la moyenne.

Les valeurs extrêmes qui ont été éliminées sont illustrées en gris dans le Tableau 4.7 et le Tableau 4.8. Le Tableau 4.10 et le Tableau 4.11, quant à eux, représentent toutes les valeurs, moyennes et écarts types respectivement, qui seront utilisées pour classifier les valeurs normalisées.

Tableau 4.10: Valeurs des Moyennes Utilisées pour Déterminer les Intensités

2000-01-10		1.11%	0.60%				1.29%		0.11%
2000-01-17	0.87%	3.39%	1.22%						1.45%
2000-01-24	3.21%	3.80%	1.06%	0.55%			3.65%		0.49%
2000-01-31		1.90%	0.52%	1.32%	1.52%		1.26%	0.17%	0.74%
2000-02-07		0.92%		3.49%	2.95%	0.35%	0.69%		0.53%
2000-02-14				2.85%	4.46%	0.53%		0.23%	0.42%
2000-02-21		0.34%	0.62%	4.02%	3.03%	0.89%	1.93%	0.40%	1.63%
2000-02-28				1.50%	1.40%	4.83%		0.27%	0.50%
2000-03-06				0.67%		5.01%		0.39%	0.97%
2000-03-13				0.71%		8.58%	0.33%		0.60%
2000-03-20						7.21%		0.35%	0.26%
2000-03-27						7.55%			0.70%
2000-04-03						9.01%	2.83%	4.52%	0.18%

Tableau 4.11: Valeurs des Écarts Types Utilisés pour Déterminer les Intensités

2000-01-10		1.16%	0.72%						
2000-01-17	0.52%	0.35%	1.03%						1.27%
2000-01-24	0.41%	0.38%	0.33%				0.71%		0.18%
2000-01-31		1.74%	0.20%	0.91%	0.32%		1.43%		0.73%
2000-02-07		0.59%		0.44%	0.78%		0.52%		0.28%
2000-02-14				0.84%	2.40%	0.28%		0.14%	0.11%
2000-02-21		0.35%		1.17%	0.88%	0.48%	1.05%		0.70%
2000-02-28				0.21%	0.50%	2.21%			
2000-03-06				0.28%		1.80%		0.24%	1.06%
2000-03-13				0.72%		1.73%			0.14%
2000-03-20						1.01%			0.13%
2000-03-27						0.84%			
2000-04-03						0.21%		0.38%	

La prochaine étape consiste alors à définir, pour chaque équipe, la distance de chaque valeur hebdomadaire de la moyenne, en terme d'écart type, tel qu'illustrée dans le Tableau 4.12 et le Tableau 4.13. Afin d'appliquer le PQAL, il s'avère maintenant nécessaire d'accorder une priorité ou un poids d'importance à chacune des intensités identifiées ci-haut. Pour ce faire, nous utiliserons la méthode des plages, laquelle sera élaborée à la prochaine section.

Tableau 4.12: Distance en Écart Type pour les Équipes 2, 3 et 4

Équipe 2									
2000-01-10									
2000-01-17									
2000-01-24	6.30	4.42	3.16				0.71		
2000-01-31									
2000-02-07				0.71					
2000-02-14				4.11					
2000-02-21				4.17			0.84		
2000-02-28					0.71	3.28			
2000-03-06				0.41		3.19			
2000-03-13						3.74			
2000-03-20						2.47			
2000-03-27						2.86			
2000-04-03						79.74			

Équipe 3									
2000-01-10									1.00
2000-01-17		0.28	1.38						0.71
2000-01-24	1.43								
2000-01-31									0.71
2000-02-07				4.05			0.71		
2000-02-14				0.26	1.54	0.71			1.15
2000-02-21				1.13	0.34	0.49	0.27		1.31
2000-02-28				0.70	0.71	1.11			
2000-03-06				0.99		5.77		1.08	
2000-03-13						0.48	1.00		
2000-03-20						7.69			
2000-03-27						1.44			
2000-04-03						0.96		1.14	

Équipe 4									
2000-01-10		0.71	0.71				1.00		
2000-01-17		1.11	0.47						
2000-01-24	0.60	6.53	0.71	1.00			4.11		0.71
2000-01-31									
2000-02-07		0.71			0.71				1.58
2000-02-14				0.56	1.15				
2000-02-21		6.99		1.25	1.19	1.22	1.26		
2000-02-28						2.55			
2000-03-06						2.64			
2000-03-13						0.67			
2000-03-20						8.03			
2000-03-27						0.82			
2000-04-03						0.06		5.77	

Tableau 4.13: Distance en Écart Type pour les Équipes 5, 6 et 7.

Équipe 5									
2000-01-10		0.71	0.71						
2000-01-17	0.71	0.83	0.83						0.71
2000-01-24	4.86	0.71	0.71						0.71
2000-01-31		0.71	0.71	0.71	0.71				
2000-02-07				0.71	0.71		0.71		0.71
2000-02-14				1.48	0.69	0.71		0.71	0.56
2000-02-21				0.33	0.32	0.35	1.11	1.00	0.71
2000-02-28				0.45		0.82		1.00	
2000-03-06				0.83		0.87		0.89	0.71
2000-03-13				0.71		1.53			3.72
2000-03-20						0.71		1.00	0.71
2000-03-27						0.47			
2000-04-03						1.04	1.00	0.72	1.00
Équipe 6									
2000-01-10									
2000-01-17		5.16	0.73						
2000-01-24	0.05	7.16					0.71		
2000-01-31			0.71	0.71	0.71		0.71		
2000-02-07					2.21	1.00			
2000-02-14				0.86	0.45	1.58			0.60
2000-02-21		0.71		2.16	2.10		1.58		0.71
2000-02-28				9.53		0.29			
2000-03-06				1.21		1.09		0.19	
2000-03-13				0.71		0.87			0.71
2000-03-20						4.80			13.96
2000-03-27						0.16			1.00
2000-04-03						28.31		0.42	
Équipe 7									
2000-01-10									
2000-01-17	0.71	3.43	0.70						
2000-01-24	0.77	0.71							
2000-01-31		0.71					0.71	1.00	0.71
2000-02-07		0.71							0.71
2000-02-14					1.23			0.71	4.45
2000-02-21		0.71	1.00	0.22	1.17	1.08			
2000-02-28				1.15		1.81			1.00
2000-03-06						0.22			0.71
2000-03-13						0.47			0.71
2000-03-20						0.71			0.71
2000-03-27						5.43			
2000-04-03						127.20		8.49	

4.2.5. Quelles sont les plages et les priorités des critères d'intensité?

Ayant l'intensité de toutes les valeurs hebdomadaires, il nous faut maintenant déterminer les plages afin de pouvoir classifier ces valeurs et leur accorder un poids d'importance. Pour ce faire, nous allons nous baser sur l'information à posteriori que nous possédons, c'est-à-dire, nous connaissons déjà les équipes qui ont réussi avec excellence leur projet. En effet, le professeur et le chargé de laboratoire du cours ont conclu que le meilleur

travail a été accompli par l'équipe 5, suivi de l'équipe 3. En comparant les intensités de ces deux équipes contre ceux des quatre autres équipes, il sera ainsi possible de déterminer les plages en écarts types. En d'autres mots, nous allons subjectivement favoriser ces deux équipes en leur donnant le plus grand poids pour un artefact donné. Pour ce mémoire, nous avons décidé d'établir quatre plages d'écarts types pour chaque artefact et de fixer des poids d'importance pour chacune des plages. Par conséquent, le poids des quatre plages correspondent à: 1.0 (i.e.: pour les distances que l'on considère idéales), 0.75 (i.e.: pour les distances que l'on estime acceptables), 0.5 (i.e.: pour les distances identifiées comme pauvres) et 0.0 (i.e.: pour les distances jugées inacceptables).

Pour débuter, nous avons créé deux groupes agglomérant toutes les distances encourues durant le projet, un pour les équipe 3 et 5 et un autre, pour les équipes 2, 4, 6 et 7. En prenant un artefact à la fois et en comparant ces deux groupes, il a été possible de déterminer les plages de façon subjective. Ces résultats sont représentés dans le Tableau 4.14. Le Tableau 4.15, quant à lui, représente les poids accordés aux intensités de chaque élément hebdomadaire (voir et), et ce, en se basant sur les plages d'intensités. Une fois toutes les valeurs nécessaires à l'application du PQAL identifiées (i.e.: poids d'importance de chaque élément de la hiérarchie), la prochaine étape consiste à regrouper simplement ces valeurs afin d'évaluer la globalité des distributions hebdomadaires de l'effort de chaque équipe.

Tableau 4.14: Plages d'Intensités pour Utilisation du PQAL.

Artefacts/Activités		
Detail UC (SS)	[1.0 – 5.0)	>6.0
Detail UC (UC)	[1.0 - 3.0)	>5.0
Use-Case Model	[1.0 - 1.5)	>4.0
Design Class	[1.0 – 2.0)	>4.1
UC Realizations	[1.0 – 2.0)	>3.0
Comp.	[2.0 – 6.0)	>8.0
Test Plan	[1.0 - 1.2)	>3.0
Test Plan	[1.0 - 1.5)	>3.0
Project Plan	[1.0 - 1.5)	>4.0

Tableau 4.15: Poids Accordés aux Intensités de Chaque Valeur Hebdomadaire

2000-01-10									
Équipe 2									
Équipe 3									
Équipe 4		0.65	0.35						
Équipe 5		0.65	0.35						
Équipe 6									
Équipe 7									
2000-01-17									
Équipe 2									
Équipe 3		0.56	0.25						1.00
Équipe 4		0.42	0.34						
Équipe 5	0.10	0.56	0.34						1.00
Équipe 6			0.34						
Équipe 7	0.10	0.28	0.34						
2000-01-24									
Équipe 2		0.17					1.00		
Équipe 3	0.43								
Équipe 4	0.57		0.09	1.00					1.00
Équipe 5	0.43	0.34	0.09						1.00
Équipe 6	0.57						1.00		
Équipe 7	0.57	0.34							
2000-01-31									
Équipe 2									
Équipe 3									1.00
Équipe 4									
Équipe 5		0.79	0.21	0.47	0.53				
Équipe 6			0.21	0.47	0.53		0.94		
Équipe 7		0.79					0.94	0.06	1.00
2000-02-07									
Équipe 2				0.54					
Équipe 3				0.27			1.00		
Équipe 4		1.00			0.46				0.50
Équipe 5				0.54	0.46		1.00		1.00
Équipe 6					0.23				
Équipe 7		1.00							1.00
2000-02-14									
Équipe 2									
Équipe 3				0.46	0.40	1.00			0.75
Équipe 4				0.46	0.40				
Équipe 5				0.35	0.54	1.00		1.00	1.00
Équipe 6				0.46	0.54	1.00			1.00
Équipe 7					0.40			1.00	
2000-02-21									
Équipe 2							0.94		
Équipe 3				0.43	0.43	1.00	0.94		0.75
Équipe 4				0.43	0.32	1.00	0.47		
Équipe 5				0.57	0.43	1.00	0.70	0.06	1.00
Équipe 6		0.52		0.29	0.21		0.47		1.00
Équipe 7		0.52		0.57	0.32	1.00			
2000-02-28									
Équipe 2					0.38	0.75			
Équipe 3				0.62	0.38	1.00			
Équipe 4						0.75			
Équipe 5				0.62		1.00		1.00	
Équipe 6						1.00			
Équipe 7				0.46		1.00			1.00
2000-03-06									
Équipe 2				1.00		0.75			
Équipe 3				1.00		0.75		0.75	
Équipe 4						0.75			
Équipe 5				1.00		1.00		1.00	1.00
Équipe 6				0.75		1.00		1.00	
Équipe 7						1.00			1.00
2000-03-13									
Équipe 2						0.75			
Équipe 3						1.00			
Équipe 4						1.00			
Équipe 5				1.00		1.00			0.50
Équipe 6				1.00		1.00			1.00
Équipe 7						1.00			1.00

4.3. Calcul et Analyse des Résultats de l'Évaluation de la Globalité des Distributions Hebdomadaires de l'Effort.

La prochaine étape dans l'évaluation du processus consiste à tout simplement regrouper les valeurs (i.e., priorités des workflows, artefacts et intensités) calculées dans les sections précédentes pour ensuite les incorporer dans le PQAL. La

Figure 4.10 illustre un exemple de l'application du PQAL pour la septième semaine (i.e.: 2000/02/21), ici, il est possible de constater toutes les valeurs nécessaires aux calculs des priorités en ce qui concerne l'évaluation de la distribution hebdomadaire de chacune des équipes. Le Tableau 4.16 regroupe les valeurs des priorités hebdomadaires de chaque équipe au niveau de l'évaluation du processus. Tel que mentionné antérieurement, nous sommes intéressés à évaluer la distribution globale (i.e.: du projet en entier) de l'effort en évaluant la distribution de façon hebdomadaire afin de tenir compte de l'aspect temporel.

Tableau 4.16: Valeurs des Priorités Hebdomadaires de la Distribution de l'Effort.

	01-10	01-17	01-24	01-31	02-07	02-14	02-21	02-28	03-06	03-13	03-20	03-27	04-03
Équipe 2	0%	0%	35%	0%	40%	0%	14%	61%	67%	70%	71%	73%	0%
Équipe 3	0%	84%	31%	10%	28%	85%	85%	97%	71%	94%	47%	98%	85%
Équipe 4	71%	66%	53%	0%	47%	78%	65%	48%	54%	94%	0%	98%	62%
Équipe 5	71%	100%	65%	72%	88%	90%	94%	85%	100%	98%	100%	98%	100%
Équipe 6	0%	29%	64%	63%	17%	98%	49%	64%	88%	99%	71%	100%	31%
Équipe 7	0%	62%	65%	54%	17%	38%	69%	81%	81%	96%	98%	73%	0%

La Figure 4.11 représente la distribution graphique de l'évaluation hebdomadaire de l'effort pour chaque équipe. Mentionnons d'abord qu'un poids de 0 lors de la première semaine signifie que l'équipe n'a pas travaillé à ce moment, mais n'a débuté ses efforts qu'en deuxième semaine. Toutefois, un poids de 0 pour les semaines suivantes peut également signifier que les intensités des artefacts correspondent à la plage «inacceptable» (voir section précédente).

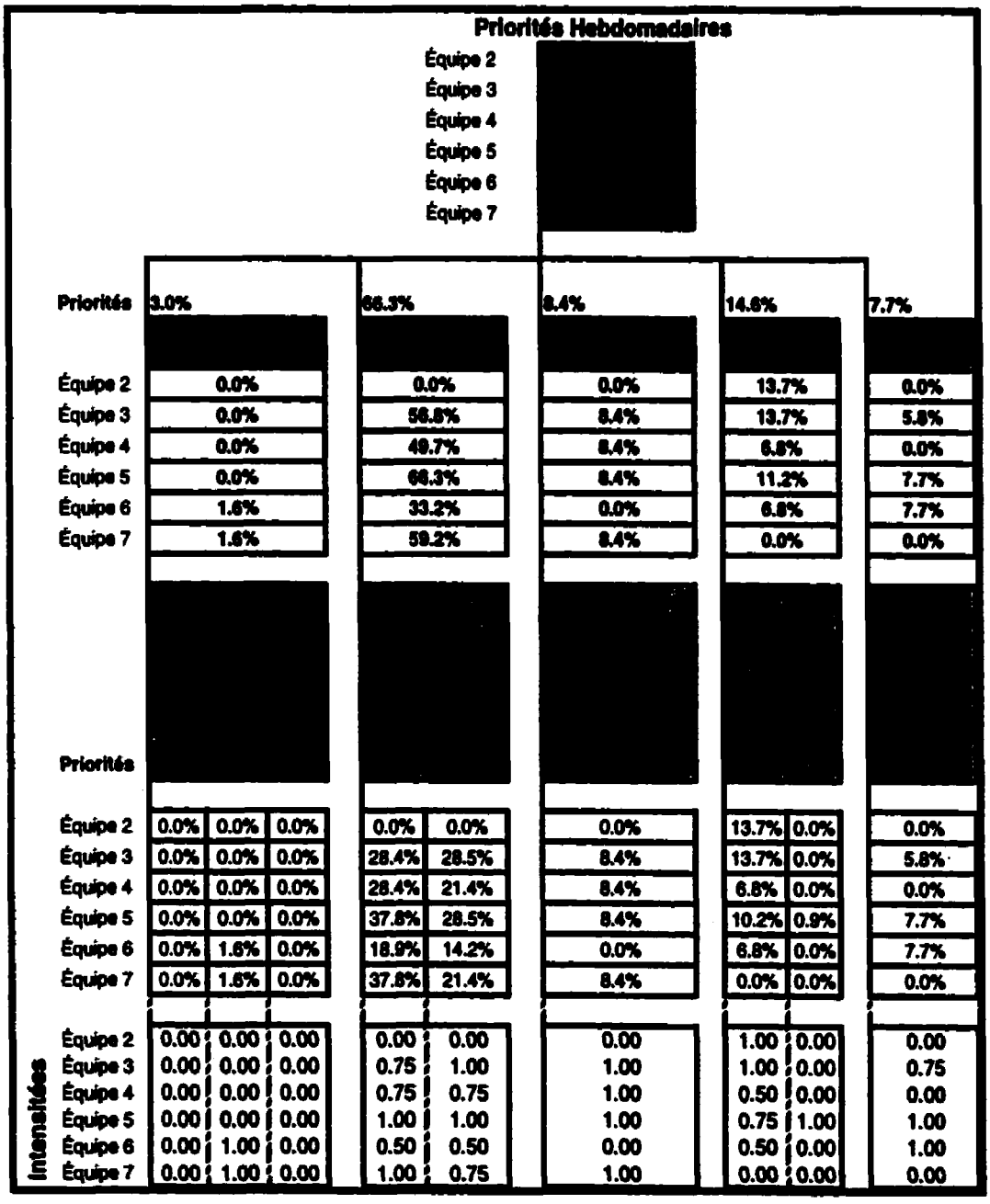


Figure 4.10: Application du PQAL pour la Semaine du 2000/02/21.

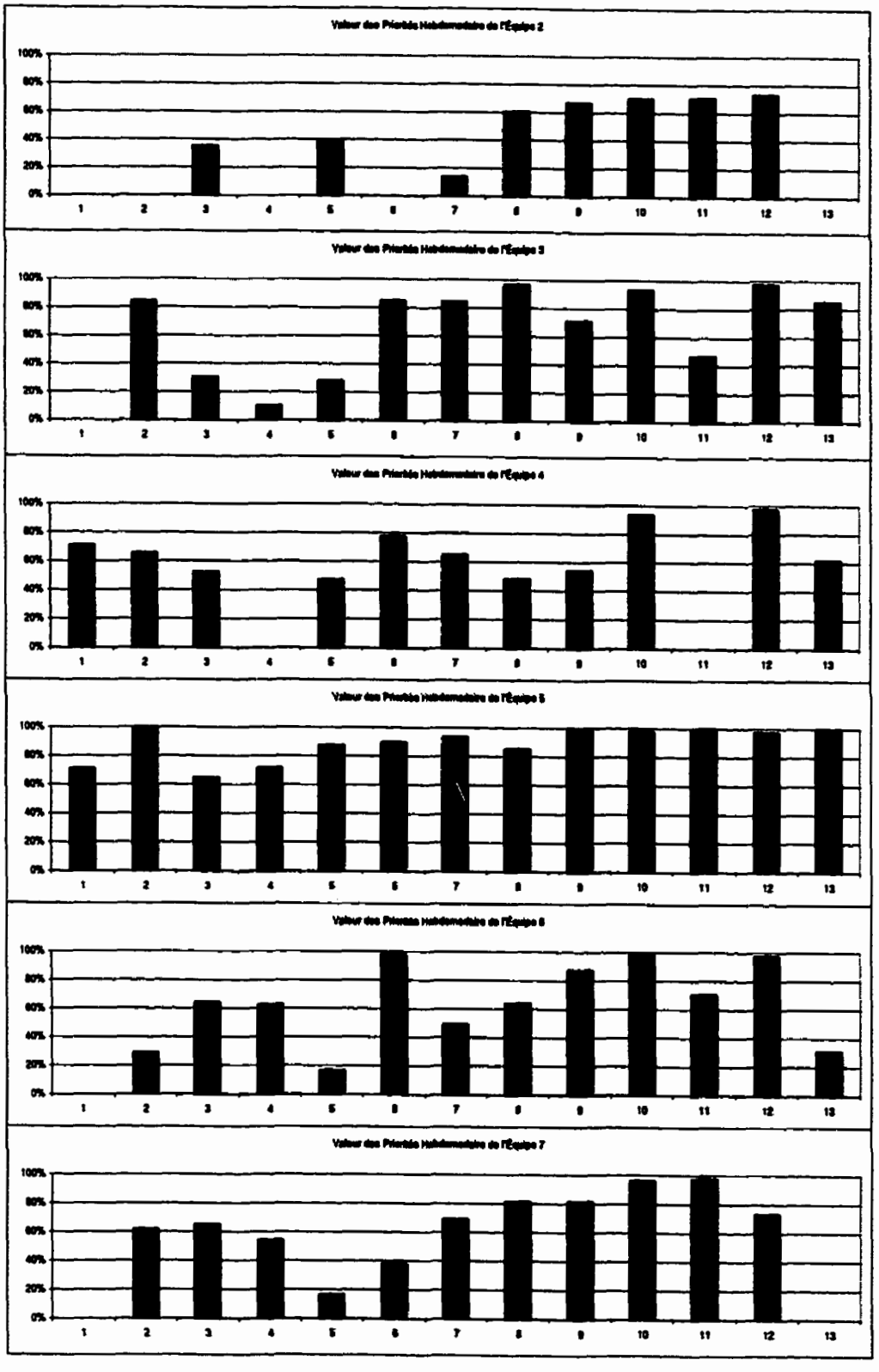


Figure 4.11: Distribution Graphique de l'Évaluation Hebdomadaire de l'Effort.

À la lumière de la Figure 4.11, nous observons que l'évaluation de la distribution de l'effort générée par l'équipe 5 demeure élevée de façon constante tout au long du projet. Toutefois, nous ne pouvons tirer les mêmes conclusions pour les autres équipes. Par exemple, l'équipe 2 semble avoir éprouvé des difficultés à distribuer leur effort entre la première et la septième semaine du projet. Rappelons que cette période correspond à la phase d'inception et d'élaboration. En effet, ce phénomène semble se produire également pour les équipes 3, 6 et 7. De la huitième à la douzième semaine, on note une amélioration quant à l'évaluation du processus, et ce pour la plupart des équipes. Une des raisons pouvant expliquer ce phénomène est que cette période correspond à la phase de construction pour la majorité des équipes, laquelle est dominée par le workflow d'implémentation qui ne contient qu'une seule activité, celle de *Implement Component.* Ceci semble corroborer l'observation soulignée antérieurement, à savoir que les étudiants ont généralement l'habitude d'investir plus de temps durant les activités de développement que pour toutes autres activités. Au cours de la dernière semaine, la dominance du workflow d'implémentation laisse place à celui des tests, qui ne contient que deux activités, celles de *Design Test* et *Execute Test.* Tel que nous pouvons le constater (Tableau 4.7 et Tableau 4.8), peu d'équipes ont réalisé ces tests, expliquant ainsi leur faible évaluation.

En somme, l'application du PQAL nous a permis de constater que la plupart des équipes, mise à part l'équipe 5, semblent présenter des difficultés avec la distribution de l'effort durant les huit premières semaines du projet. Notons que cette période est dominée par les workflows des requis, analyse et design, et tests, lesquels se caractérisent par plusieurs artefacts. Toutefois, on note une amélioration pour la majorité de ces équipes lors de la phase de codage. De plus, tel que nous l'avons mentionné, cette période est dominée par le workflow d'implémentation, représenté par un seul artefact, les «components» (i.e.: le code même). Ceci semble indiquer une diminution dans la sensibilité du PQAL. En effet, les résultats nous portent à croire que plus notre hiérarchie contient d'éléments, c'est-à-

dire plusieurs artefacts pour un même workflow, plus le PQAL s'avère sensible ce qui permettrait une meilleure discrimination entre les équipes.

La prochaine étape consiste à calculer et analyser les valeurs globales de la distribution de l'effort pour le projet entier.

4.4. Calcul et Analyse des Résultats de l'Évaluation de la Distribution Globale de l'Effort.

Rappelons que la valeur globale du projet a été calculée en utilisant le principe du AHP, c'est-à-dire que nous avons aggloméré toutes les valeurs globales hebdomadaires sous une hiérarchie et les priorités ont été distribuées de façon équivalente (i.e.: chaque évaluation hebdomadaire possède une priorité = 1/13, voir section). Le Tableau 4.17 représente la distribution de l'évaluation globale du processus de chaque équipe.

Tableau 4.17: Valeur Globale de l'Évaluation de la Distribution de l'Effort.

[illegible]	[illegible]
2	33%
3	63%
4	57%
5	89%
6	59%
7	57%

À l'aide du PQAL, nous pouvons constater que toutes les équipes, mise à part l'équipe 5, présentent une faible évaluation de la distribution globale de l'effort. Cette observation démontre que le processus UPEDU était, en général, mal compris des étudiants. Nous pouvons également stipuler une gestion inefficace au niveau du temps et des tâches, et ce pour la plupart des équipes. Toutefois, il s'avère surprenant de constater un faible résultat

pour l'équipe 3, considérant sa performance initialement établie par le professeur et chargé de laboratoire.

Maintenant que nous avons évalué la distribution de l'effort, il est nécessaire d'ajouter une seconde métrique afin d'être en mesure d'évaluer le processus global. La prochaine section présentera cette nouvelle métrique concernant l'évaluation des artefacts.

4.5. Métrique Additionnelle

De façon à exploiter la flexibilité du PQAL à agglomérer diverses métriques, nous allons utiliser, en plus de la distribution de l'effort, l'évaluation des artefacts remis par les étudiants au professeur et au chargé de laboratoire. D'après UPEDU, les artefacts correspondent aux résultats finaux des activités, et leur évaluation est représentative de la «qualité» de la productivité de l'effort entrepris à générer les artefacts. En d'autres mots, même si les équipes ont reçu une évaluation élevée quant à la distribution de l'effort durant le projet, ceci n'est pas nécessairement un indicateur quant à l'excellence des artefacts créés. Par exemple, une équipe peut avoir une valeur globale de la distribution de l'effort élevée mais avoir créé des artefacts de pauvre qualité. Donc, nous allons considérer les évaluations du professeur et du chargé de laboratoire comme étant les résultats de la métrique à utiliser dans le PQAL, afin d'évaluer la «qualité» globale des artefacts de chaque équipe.

Ainsi, cette section présente la création d'une nouvelle hiérarchie, laquelle va inclure les deux niveaux suivants: les workflows et les artefacts. Nous discuterons également de la technique utilisée pour déterminer les priorités de chacun de ces éléments ainsi que le modèle d'intensité utilisé. Enfin, à l'aide du PQAL, nous allons démontrer qu'il s'avère possible d'agglomérer deux hiérarchies différentes afin d'évaluer le processus de développement logiciel, celle de l'évaluation globale de la distribution de l'effort et celle de l'évaluation globale des artefacts.

4.5.1. Évaluation Globale des Artefacts

En s'inspirant du PQAL, la Figure 4.12 représente la nouvelle hiérarchie correspondant à l'évaluation globale des artefacts. Mentionnons que le but global de cette hiérarchie consiste en une évaluation globale des artefacts. Le second niveau de cette hiérarchie se caractérise par les différents workflows. Tel que nous pouvons le constater, le troisième niveau comporte les différents artefacts correspondant à chacun des workflows. Rappelons que nous n'avons pas inclus un niveau pour les activités à cause de la relation 1:1 (voir chapitre précédent) entre les artefacts et les activités.

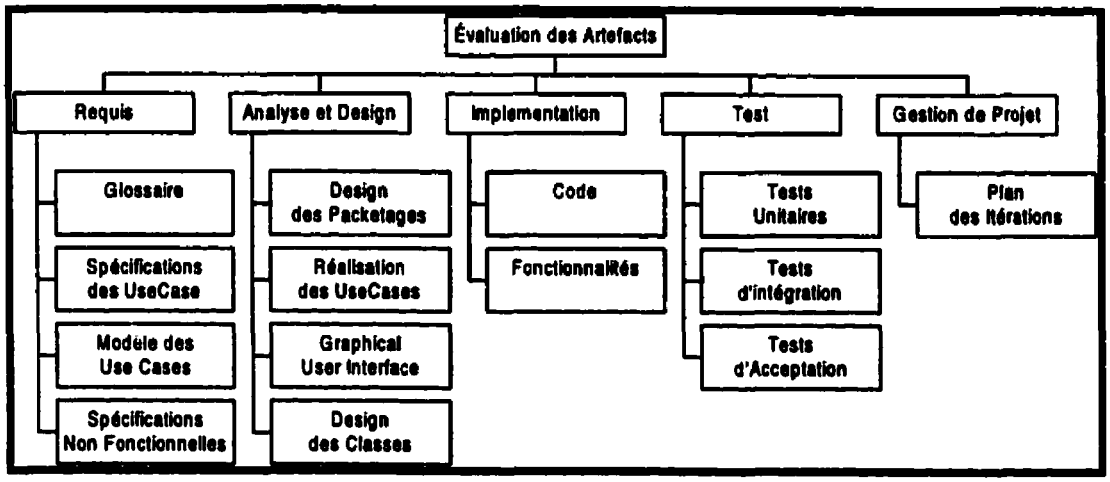


Figure 4.12: Hiérarchie pour l'évaluation globale des Artefacts.

Dans le but d'appliquer notre modèle, il s'avérait alors pertinent d'identifier les priorités des workflows et des artefacts. Pour ce faire, nous avons utilisé les priorités identifiées par le professeur et le chargé de laboratoire à l'aide la méthode AHP. Le chargé de laboratoire a d'abord déterminé les priorités en faisant une comparaison binaire, une fois terminée, les priorités ont été authentifiées, puis ajustées par le professeur. Les deux prochains tableaux démontrent les résultats des priorités pour chacun des éléments de la hiérarchie.

Tableau 4.18: Priorités des Workflows pour l'Évaluation des Artefacts.

WORKFLOWS	PRIORITÉS
Requis	14.6%
Analyse et Design	23.9%
Implémentation	41.9%
Test	12.6%
Gestion de Projet	7.0%

Tableau 4.19: Priorités des Artefacts pour l'Évaluation des Artefacts.

WORFLOWS	ARTEFATCS	PRIORITÉS
Requis	Glossaire	25%
	Spécifications des Use Cases	25%
	Model des Use Cases	25%
	Spécifications Non Fonctionnelles	25%
Analyse et Design	Design des Packetages	25%
	Réalisation des Use Cases	25%
	Graphical User Interface	25%
	Design des Classes	25%
Implémentation	Code	50%
	Fonctionnalités	50%
Test	Tests Unitaires	33.3%
	Tests d'Intégration	33.3%
	Tests d'Acceptation	33.3%
Gestions de Projet	Plan des Itérations	100%

Il semble important de souligner ici que les priorités des éléments de modélisation employées pour évaluer les artefacts ne peuvent être utilisées pour évaluer la distribution de l'effort. En effet, les artefacts ont été remis par les étudiants et évalués par le professeur et chargé de laboratoire seulement en fin de projet, fournissant ainsi une évaluation globale plutôt que temporelle, telle que requise par l'évaluation hebdomadaire de distribution de l'effort. De plus, un artefact qui domine au point vue de l'effort (i.e.: nécessitant un facteur d'heures/personne élevé) ne veut pas nécessairement dire que la priorité de l'artefact produit par cet effort sera le plus important.

Maintenant, nous pouvons identifier la stratégie à utiliser pour déterminer les intensités reliées à l'évaluation faite par le professeur et chargé de laboratoire pour les artefacts. Puisque l'intensité des artefacts reflète le poids qui leur est accordé suite à une mesure, nous avons décidé de nous servir des notes décernées par le professeur et le chargé de laboratoire pour chaque artefact remis par chacune des équipes. Le Tableau 4.20 représente donc les notes accordées aux différentes équipes pour chacun des artefacts remis.

Tableau 4.20: Notes Accordées à Chacun des Artefacts.

ARTEFATCS	ÉQUIPES					
	2	3	4	5	6	7
Glossaire	88%	25%	63%	88%	100%	0%
Spécifications des Use Cases	100%	100%	100%	100%	100%	100%
Model des Use Cases	100%	100%	100%	100%	100%	100%
Spécifications Non Fonctionnelles	63%	100%	100%	75%	100%	100%
Design des Paquetages	50%	100%	88%	100%	75%	0%
Réalisation des Use Cases	50%	100%	100%	88%	100%	75%
Graphical User Interface	100%	88%	100%	100%	0%	50%
Design des Classes	75%	100%	75%	100%	88%	75%
Code	56%	83%	54%	76%	76%	88%
Fonctionnalités	88%	75%	75%	100%	88%	88%
Tests Unitaires	50%	100%	50%	100%	100%	63%
Tests d'Intégration	50%	88%	75%	100%	50%	50%
Tests d'Acceptation	25%	88%	50%	100%	100%	100%
Plan des Itérations	88%	100%	75%	100%	88%	88%

Maintenant que toutes les priorités ont été identifiées, nous pouvons utiliser le PQAL afin d'évaluer la priorité globale des artefacts. Ainsi, la illustre la hiérarchie nécessaire à cette évaluation ainsi que les résultats des priorités globales des artefacts pour chacune des équipes.

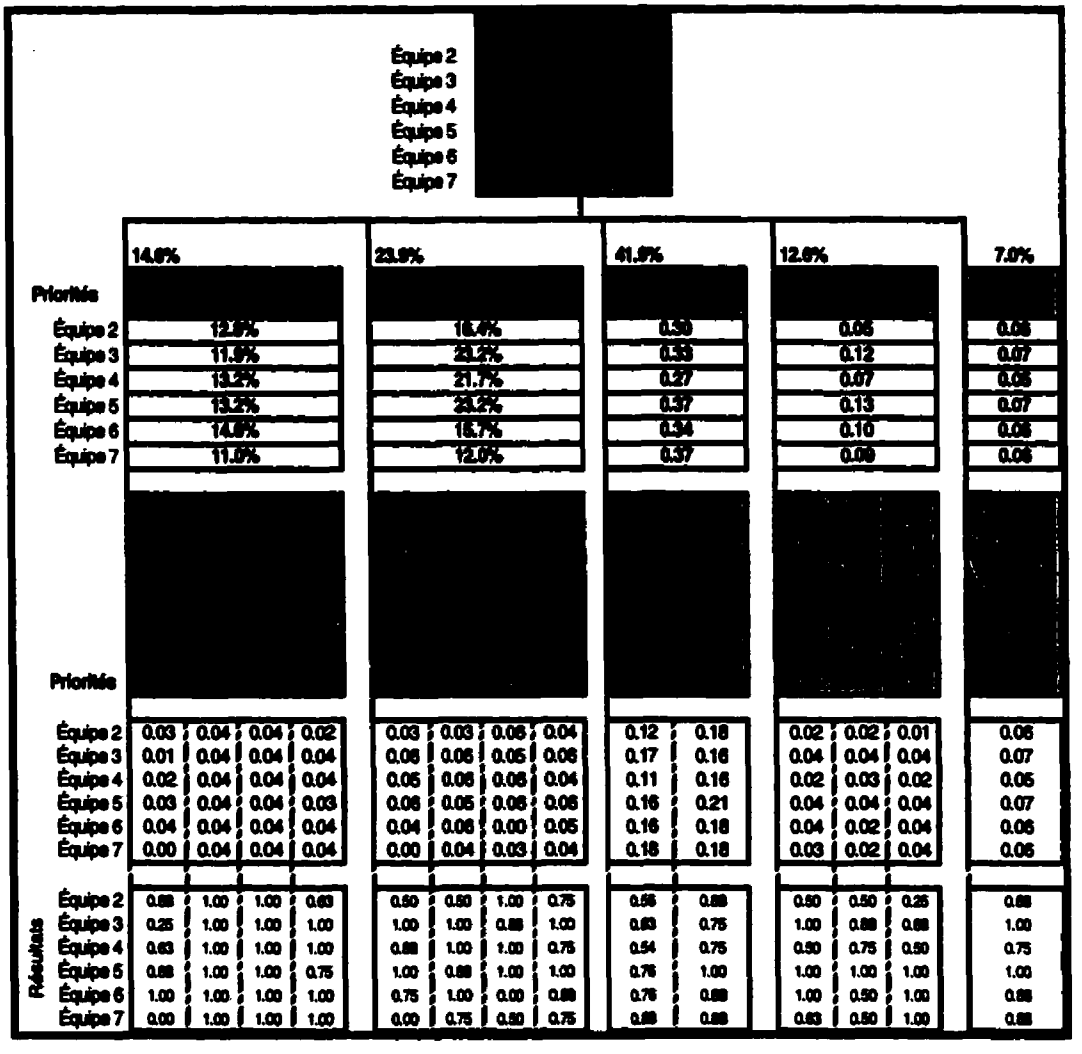


Figure 4.13: Représentation du PQAL à l'évaluation globale des artefacts.

Le PQAL nous a permis de constater, qu'en comparaison, l'évaluation des artefacts s'avère plus élevée que celle de la distribution de l'effort. Mise à part l'équipe 5, l'évaluation des artefacts semble ainsi favoriser considérablement toutes les équipes. Rappelons que ces dernières présentaient de faibles évaluations concernant la distribution de l'effort. Par conséquent, l'application de notre modèle nous incite à croire que les évaluations de ces deux métriques sont indépendantes. En effet, on ne semble pouvoir

prédire l'effort en ne se servant que de la qualité des artefacts générés. La nature indépendante de ces métriques s'avère toutefois avantageuse lorsque nous les combinerons afin d'évaluer la valeur globale du processus de développement logiciel à l'aide du PQAL.

4.6. Calcul et Analyse de l'Évaluation Globale du Processus

Ayant mesuré la valeur globale de la distribution de l'effort et l'évaluation globale des artefacts, nous pouvons re-utiliser le PQAL afin d'évaluer la valeur globale du processus de chaque équipe. La Figure 4.14 présente la hiérarchie nécessaire au calcul nous permettant d'évaluer le processus entier, c'est-à-dire, dans sa globalité.

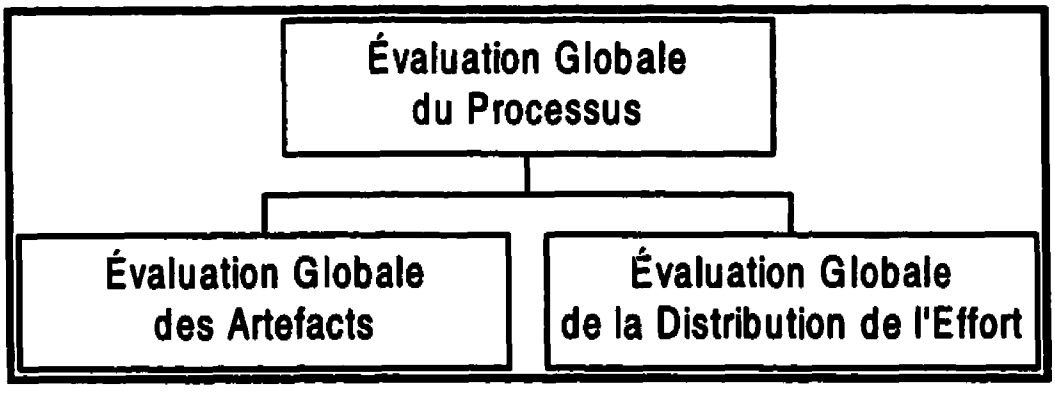


Figure 4.14: Hiérarchie pour l'évaluation globale du processus.

Suite à une discussion avec le professeur, nous avons décidé, comme l'artefact correspond davantage à la représentation du produit final que ne l'est l'effort, d'attribuer un poids de 0.6 à l'artefact et de 0.4 à l'effort. Une fois de plus, les résultats confirment les équipes 5 et 3 comme celles ayant réalisé le meilleur projet et l'équipe 2 comme celle ayant le moins réussi l'objectif fixé par le travail exigé par le professeur (voir Figure .).

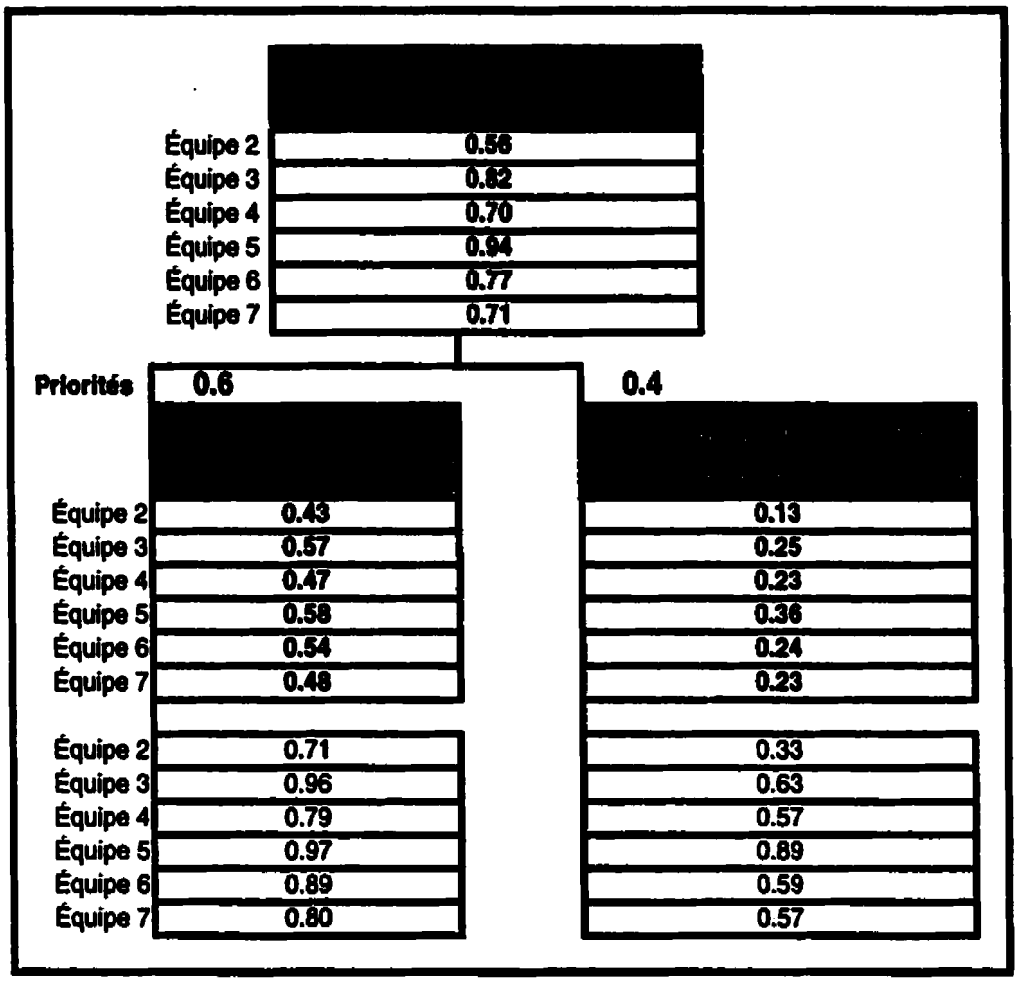


Figure .: Représentation de l'évaluation globale du processus.

Le Tableau 4.21 permettra au lecteur de comparer nos résultats d'évaluation globale du processus de développement logiciel avec les notes finales accordées par le professeur à chacune des équipes. À la lumière de ce tableau, nous pouvons constater une concordance dans le rang attribué par notre modèle et celui assigné par le professeur. Il semble que cette corroboration puisse soutenir la nature prometteuse de l'application de notre modèle.

Tableau 4.21: Comparaison entre les Résultats du PQAL et ceux des Notes Finales.

2	71.5% (6)	55.9% (6)
3	85.5% (2)	82.4% (2)
4	78.8% (5)	69.8% (5)
5	92.5% (1)	93.7% (1)
6	82.7% (3)	77.3% (3)
7	78.9% (4)	70.7% (4)

On peut maintenant se questionner sur les avantages de combiner deux métriques pour évaluer le processus. En d'autres mots, ne serait-il pas suffisant d'évaluer le processus en ne se servant que de la distribution de l'effort ou que de l'évaluation des artefacts? Puisque l'effort correspond davantage au temps, par personne, attribué lors de la réalisation d'un projet, il semble offrir peu d'information quant à la création même des artefacts. D'autre part, la mesure de l'artefact ne fournit aucune information quant à la distribution de l'effort dispensé à l'élaboration et au développement de cet artefact. Il semble donc justifié de considérer ces deux métriques afin d'obtenir un portrait plus détaillé du processus global dans le but de pouvoir fournir une évaluation plus complète. Il va sans dire que le choix des métriques ainsi que des critères utilisés lors de cette application du PQAL ne correspond pas à l'unique solution pour réaliser l'évaluation globale du processus de développement logiciel. Par conséquent, le choix et/ou l'ajout d'une ou de plusieurs métriques additionnelles à l'évaluation du processus s'avérerait sûrement justifiable pour parfaire notre évaluation globale. Toutefois, ceci dépasse l'objectif fixé dans le cadre de cette étude de deuxième cycle.

Nos conclusions, contributions et recommandations feront l'objet du prochain chapitre.

CHAPITRE 5
CONCLUSION

Dans le cadre de ce chapitre, nous présenterons d'abord nos conclusions relatives aux résultats de cette étude. Par la suite, une discussion quant aux contributions possibles sur le plan théorique, industriel et académique suivra. Enfin, nous élaborerons quelques recommandations au niveau de la pratique et de la recherche.

5.1. Conclusion

Il existe une multitude de métriques en génie logiciel, mais l'importance qu'occupe chacune d'elles dans l'évaluation des attributs logiciels demeure toujours incomprise et ce, principalement en raison des lois de la théorie de la mesure. En effet, à notre connaissance, aucun des articles recensés portant sur les métriques en génie logiciel n'offre une solution pour l'agglomération des résultats de métriques. Ainsi, la résolution d'une telle problématique pourrait contribuer au renouvellement des pratiques de l'ingénieur dans le domaine du génie logiciel.

Le but de ce mémoire consistait donc à développer un modèle permettant l'agglomération des résultats de diverses métriques, et ce indépendamment des échelles de mesure utilisées, permettant ainsi la quantification de l'évaluation des attributs de la qualité. Pour ce faire, après avoir exposé la problématique, nous avons décrit le modèle *Analytical Hierarchy Process* (AHP) de Saaty (1996) dans le but de l'intégrer au domaine du génie logiciel, afin de créer notre modèle, que nous avons surnommé le Processus de Quantification des Attributs Logiciel (PQAL). Comme attribut de la qualité, nous avons utilisé le processus de développement logiciel, tel que mentionné dans le modèle de qualité de Boloix et Robillard (1995). Dans le but de pouvoir appliquer notre modèle à l'évaluation d'un processus de développement logiciel en milieu académique, nous avons initialement validé une base de données existante qui à été générée par six équipes d'étudiants travaillant sur le même projet dans le cadre d'un cours de premier cycle,

Atelier de Génie Logiciel. Cette étape a été réalisée en nous servant des critères de vérification et de validation suggérés par Florac et Carleton (1999). Finalement, nous avons procédé à l'application du modèle et à l'évaluation du processus de développement.

Dans le cadre de ce mémoire, nous avons utilisé comme attribut de qualité, le processus de développement logiciel, tel qu'identifié dans le modèle de qualité de Boloix et Robillard (1995a). Rappelons que deux métriques, celle de la distribution de l'effort, représentant un élément qui est absorbé par le processus, et celle de l'évaluation des artefacts, correspondant à un élément produit suite au processus, ont été retenues afin de mesurer le processus de développement logiciel. Malgré que le projet était essentiellement le même pour toutes les équipes d'étudiants, le PQAL nous a permis d'observer que la «signature» graphique de la distribution de l'effort pour chacun des workflows différait pour chacune des équipes. Cette observation indiquait que la distribution de l'effort semblait peu dépendante du projet, mais plutôt des équipiers. À l'aide de notre modèle, nous avons constaté que toutes les équipes, mise à part l'équipe 5, présentaient une faible évaluation de la distribution globale de l'effort. Cette observation pouvait nous laisser supposer une mauvaise compréhension du processus UPEDU et/ou une gestion inefficace au niveau du temps et des tâches, et ce pour la plupart des équipes. Toutefois, il s'avérait surprenant de constater un faible résultat pour l'équipe 3, considérant sa performance initialement établie par le professeur et le chargé de laboratoire. On pouvait donc présumer que l'évaluation qui nous a été initialement offerte par le professeur et le chargé de laboratoire, s'inspirait davantage des artefacts générés par les équipes que de leur effort.

À l'ajout de la seconde métrique, le PQAL nous permettait de constater, qu'en comparaison, l'évaluation des artefacts s'avérait significativement plus élevée que celle de la distribution de l'effort. Mise à part l'équipe 5, l'évaluation des artefacts semblait ainsi favoriser considérablement toutes les équipes. Rappelons que ces dernières

présentaient de faibles évaluations concernant la distribution de l'effort. Par conséquent, l'application de notre modèle nous incitait à croire que les évaluations de ces deux métriques étaient indépendantes l'une de l'autre. En effet, il semblait improbable de prédire l'effort en ne se servant que de la qualité des artefacts générés.

L'étape suivante consistait à agglomérer les résultats des deux métriques afin de pouvoir évaluer le processus dans sa globalité. À la lumière des résultats obtenus à l'aide du PQAL, il nous était possible de noter une concordance entre le rang attribué aux équipes par notre modèle et celui assigné par le professeur. Tel que nous l'avons mentionné au chapitre 3, nous devons être vigilant lors de l'interprétation des résultats suite au PQAL en raison du nombre élevé de corrections subjectives et objectives lors de la vérification et de la validation de la base de données. Toutefois, une telle corroboration nous porte à croire en l'utilité future de notre modèle.

En conclusion, il a été démontré qu'il était possible, par l'entremise du PQAL, d'agglomérer les résultats de plusieurs métriques afin de générer une seule valeur quantitative quant à l'attribut de qualité logicielle visé, et ce, tout en respectant la théorie de la mesure. D'ailleurs, cette conformité est tributaire à la validité de la plupart des tests statistiques. Bien que la théorie de prise de décisions multicritères (PDMC) existe depuis plus de 50 ans dans les domaines mathématique et économique, elle s'avérait, jusqu'à présent, peu utilisée dans notre discipline. De plus, on utilise davantage cette théorie comme modèle de classement et non comme modèle d'évaluation. Le PQAL, au contraire, se veut un modèle d'évaluation s'appuyant sur le modèle AHP. L'application d'une théorie «empruntée» à une discipline connexe peut donc s'avérer une contribution inédite, particulièrement lorsqu'un modèle d'évaluation est créé afin d'opérationnaliser cette théorie.

5.2. Contributions

5.2.1. Contributions au Niveau Théorique et de Recherche

Le modèle développé dans le cadre de ce mémoire, le Processus de Quantification des Attributs Logiciels (PQAL), pourra permettre aux étudiants ainsi qu'aux entreprises de mieux comprendre l'implication ainsi que l'interaction des métriques lors de l'évaluation des attributs logiciels.

L'utilisation du PQAL dans le cadre de recherches ultérieures pourrait contribuer à développer les connaissances quant aux interactions existant entre différentes métriques dans le domaine du génie logiciel. De telles études pourraient éventuellement permettre aux utilisateurs une sélection informée des métriques nécessaires à l'évaluation des attributs de la qualité.

5.2.2. Applications Futures

Dans le cadre de ce mémoire, nous avons évalué la distribution de l'effort de chaque équipe à posteriori, c'est-à-dire que nous avons mesuré l'effort de chaque équipe une fois leur projet terminé. Toutefois, nous croyons qu'il s'avère possible d'utiliser le PQAL ainsi que les données à priori, afin de mesurer l'effort au fil du projet (mesure progressive de l'effort) ou de façon prédictive (mesure prédictive de l'effort), permettant ainsi de constater la direction que prendra l'effort global. Les deux prochaines sections simuleront ces approches.

5.2.2.1. Mesure Progressive de l'Effort

Cette simulation présume que nous ne possédons aucune des données en début de projet. On recevrait plutôt les jetons de chacune des équipes sur une base hebdomadaire. Par exemple, à la fin de la seconde semaine du projet, nous aurions recueilli deux semaines

de jetons. En prenant pour acquis que le poids de chaque semaine demeure équivalent, il nous serait alors possible de mesurer l'effort global pour cette période en utilisant les mesures hebdomadaires de ces deux semaines. Cette mesure progressive de l'effort permettrait un ajustement en cours de projet, puisqu'il fournirait un indicateur de la mesure globale jusqu'au moment présent. Ainsi, une faible mesure globale de l'effort à mi-projet pourrait contribuer à repenser la stratégie employée jusqu'à maintenant de façon à redistribuer l'effort de manière plus efficace.

La Figure 5.1 illustre la distribution de l'effort pour cette simulation pour chacune des équipes de notre base de données. À la lumière de cette figure, nous pouvons observer que l'équipe 5 a maintenu un pourcentage élevé de l'effort tout au long du projet. Une telle observation n'était pas applicable pour les autres équipes. Par exemple, on notait une baisse de l'évaluation de l'effort pour l'équipe 4 à partir de la quatrième semaine du projet. Cette information aurait pu fournir au professeur une aide pédagogique précieuse en cours de trimestre, afin de sensibiliser l'équipe à visualiser les points à améliorer, et par le fait même à réajuster la distribution de leur effort. Ceci s'avère applicable pour les autres équipes ayant une faible évaluation. En présence d'une forte évaluation, le professeur pourrait également utiliser cette approche de façon à encourager l'équipe à maintenir l'effort fournit jusqu'à présent.

Cette simulation nous a également permis de constater qu'une telle approche pourrait accroître la sensibilité du PQAL, quant à l'évaluation du but global, dans un contexte commercial ou industriel. Lors de notre discussion au sujet de l'attribution des priorités des éléments de la hiérarchie (voir Chapitre 4), nous avions signalé la difficulté à justifier cette répartition. En s'inspirant de cette simulation, un chef de projet pourrait ré-ajuster les priorités du PQAL suite à une évaluation. Par exemple, à la quatrième semaine, si le PQAL indiquerait une faible évaluation de l'effort, le chef de projet ainsi que son équipe pourrait considérer que le projet se déroule mieux que ce que laisse croire l'évaluation. Ä ce moment, le chef de projet pourrait utiliser ces informations de manière à ré-ajuster les

priorités afin que celles-ci reflètent davantage la réalité. À l'opposé, si la valeur globale de l'effort s'avèrerait élevée mais que le chef de projet considérait que le projet ne se déroule pas bien, il serait alors possible de ré-ajuster les priorités afin que le PQAL indique, de façon plus précise, l'état actuel du projet. En d'autres mots, il ne semble pas primordial de se baser uniquement sur des données historiques ou empiriques pour évaluer les priorités. Une considération de la situation réelle actuelle peut fortement contribuer à l'évaluation des priorités et au ré-ajustement des stratégies mises en place.

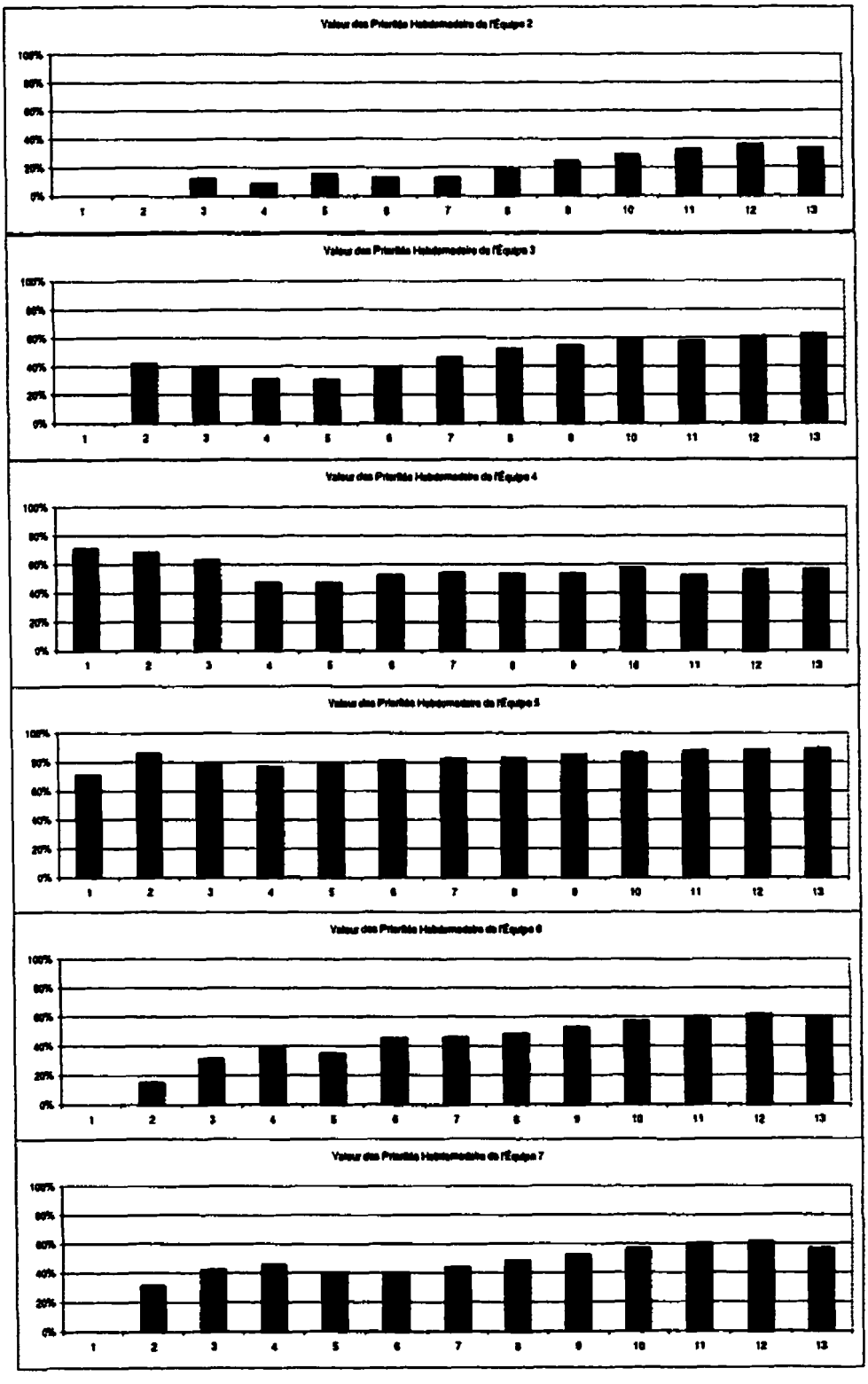


Figure 5.1: Distribution Progressive de l'Effort.

5.2.2.2. Mesure Prédictive de l'Effort

Cette simulation présume également que nous ne sommes pas en possession de toutes les données. Toutefois, contrairement à l'approche précédente, nous désirons connaître immédiatement, quelle sera l'évaluation finale future de l'effort pour le projet, en se basant sur les données recueillies jusqu'au moment présent. Cette simulation suppose que nous accorderions, en début de projet, à chacune des évaluations hebdomadaires de l'effort un poids équivalent de 1 ou 100% (i.e., une valeur normalisée de 1/13 (0.08) pour un projet d'une durée de 13 semaines). Ainsi, toujours en début de projet, le poids global associé au projet en entier correspondrait à 100%. Au fur et à mesure, chacune de ces valeurs serait remplacée par l'évaluation hebdomadaire de l'effort actuel. Par exemple, si l'évaluation de l'effort pour la première semaine correspondait à 1, le poids global de l'effort demeurerait 100%. Toutefois, si l'évaluation s'annonçait inférieure à 1, l'évaluation de cette semaine serait normalisée pour les 13 semaines et ferait, par conséquence, diminuer le poids global de l'effort.

Les jetons de la base de données employée dans le cadre de ce mémoire ont été utilisés afin de démontrer l'effet de cette simulation. La Figure 5.2 illustre les résultats associés à cette simulation. Sauf pour les équipes 4 et 5, nous observerions que l'impact de l'évaluation globale pour la première semaine entraînerait une diminution significative de l'évaluation globale. Remarquons également que cette diminution est presque linéaire pour chacune des équipes. Toutefois, nous pouvons noter que la pente de cette descente varie entre les équipes. Il serait donc possible, et probablement fort utile, à un chef de projet, de calculer la pente des évaluations précédentes et d'extrapoler sur les 13 semaines (ou la durée estimée du projet), afin de prédire l'évaluation éventuelle de la distribution de l'effort global de son équipe. Par exemple, plus la pente s'avère négative, plus l'évaluation globale diminuera et vice-versa. Ainsi, autant que la simulation précédente peut permettre au chef de projet d'évaluer l'effort encouru jusqu'au moment présent, cette

simulation prédictive pourrait lui fournir des informations pertinentes quant à la direction éventuelle, ainsi que le degré de diminution de la distribution de l'effort global.

Une telle simulation semble également prometteuse dans le domaine de l'enseignement. Un professeur pourrait utiliser cette approche afin d'indiquer aux étudiants la direction que prendra l'évaluation globale de l'effort, en s'appuyant sur la distribution présente de leur effort à l'intérieur du projet.

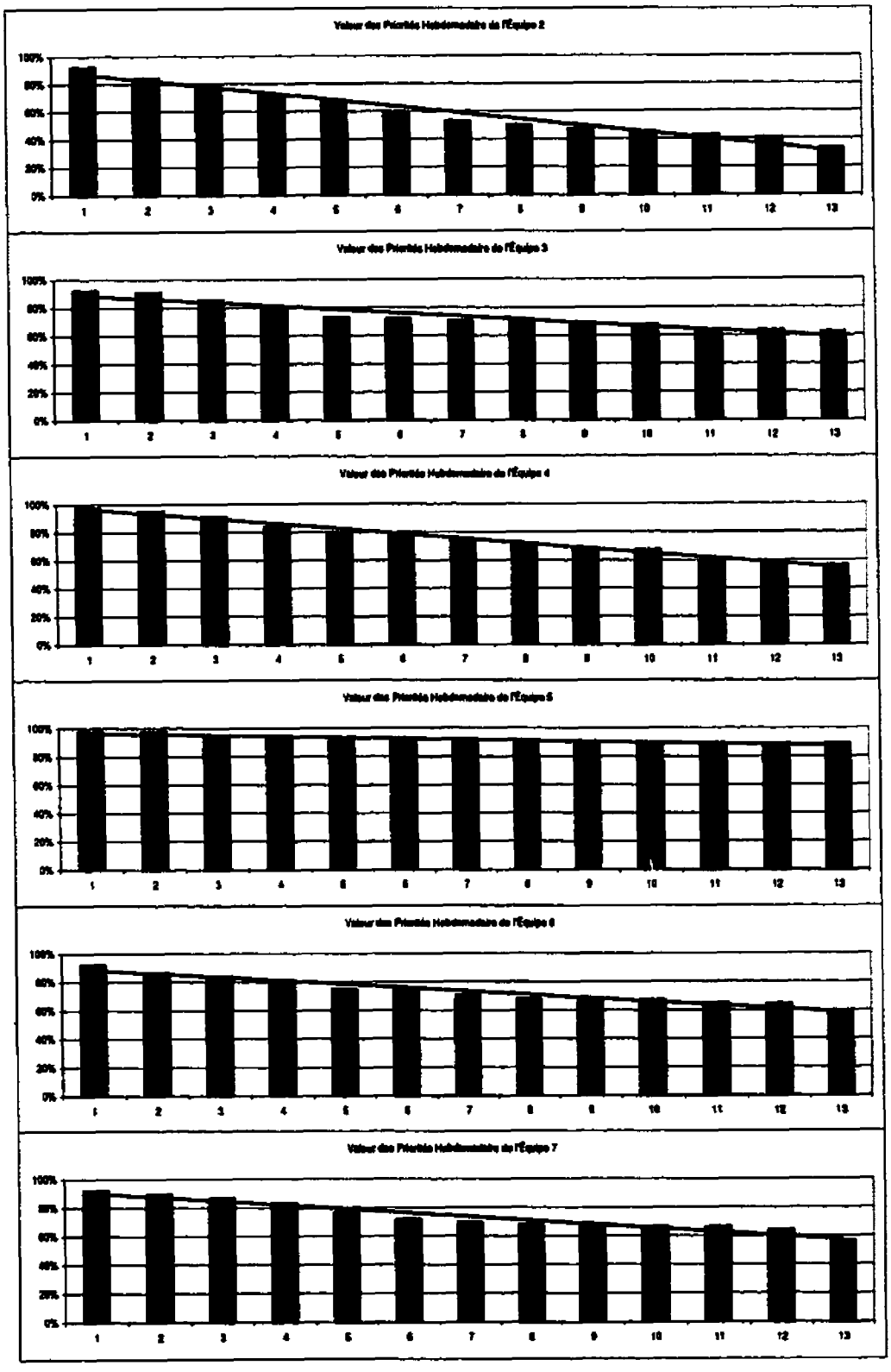


Figure 5.2: Représentation Prédictive de l'Effort.

5.2.3. Contributions dans le Milieu Industriel

Nous croyons que le PQAL pourra faciliter les prises de décisions reliées au choix et au nombre de métriques requises afin de mesurer divers attributs d'un produit logiciel quelconque. Le PQAL pourra ainsi contribuer aux prises de décisions associées aux ressources financières et humaines nécessaires pour promouvoir le but global préalablement visé (i.e., la qualité). D'autre part, une compagnie pourrait davantage justifier les dépenses associées aux métriques ayant des priorités élevées, puisque leur impact sur l'indicateur s'avère significatif. Bref, le PQAL pourra faciliter, dans le cadre d'une entreprise, le choix des métriques afin que le ratio coût/bénéfice soit optimal.

5.3. Recommandations

5.3.1. Recommandations sur le Plan Pratique

Suite aux résultats de cette étude de deuxième cycle, nous recommandons aux utilisateurs d'agglomérer, à l'aide du PQAL, des métriques déjà existantes dans leur milieu. Le PQAL pourra ainsi contribuer à la pratique du génie logiciel en permettant de quantifier les attributs logiciels, tâche qui, jusqu'à maintenant, s'avère problématique. Ainsi, l'utilisation du PQAL dans les entreprises pourrait faciliter l'assurance qualité, la compréhension et le suivi du développement logiciel, lesquelles sont essentielles à la répartition des ressources.

5.3.2. Recommandations sur le Plan Recherche

Nous croyons qu'il serait pertinent de répliquer cette étude en utilisant, cette fois-ci, une base de données provenant du milieu industriel. En plus, d'une corroboration quant à l'application du PQAL, une telle recherche pourrait également faire l'objet de comparaison entre les milieux industriel et académique.

Il serait également important d'entreprendre une étude visant à déterminer le nombre optimal de métriques requises par le PQAL pour l'évaluation d'un attribut de la qualité logicielle. Ces résultats pourraient contribuer à optimiser la mise en application du PQAL.

5.4. Limites de l'Étude

Plusieurs bénéfices financiers semblent associés à la qualité logicielle (e.g., augmentation de la fiabilité entraînant une diminution des coûts de maintenance, identification des priorités facilitant la gestion des ressources, etc.). Bien que le PQAL à lui seul ne peut garantir un logiciel de qualité TOTALE, il pourra toutefois agir à titre d'indicateur, contribuant ainsi à promouvoir, de façon significative, la qualité globale du produit. Toutefois, il n'en demeure pas moins que toutes les méthodes de PDMC possèdent des problèmes qui leur sont intrinsèques. Il s'avère, par conséquent, nécessaire d'être vigilant lors de l'interprétation des données. Les méthodes de PDMC ne peuvent donc être considérées des panacées à tous les problèmes multicritères, ce sont plutôt des outils ou processus visant à faciliter les prises décisions pour l'utilisateur. En effet, on ne doit pas négliger le fait que la décision finale revient toujours au décideur et non au modèle utilisé.

5.5. Mot de la Fin

Jusqu'à présent, les réalisateurs en génie logiciel éprouvent des difficultés à interpréter, de façon globale, les résultats de métriques puisqu'il n'existe aucun processus d'agglomération respectant la théorie de la mesure. Le PQAL, quant à lui, pouvant agglomérer les résultats en respectant la théorie de la mesure, contribuera de façon certaine à l'évaluation holistique du produit logiciel, visant éventuellement à promouvoir une qualité globale. Les bénéfices financiers associés à la qualité logicielle s'avèrent nombreux. Bien que le PQAL à lui seul ne peut garantir un logiciel de qualité, il pourra toutefois agir à titre d'indicateur, contribuant ainsi à promouvoir de façon significative la

qualité globale du produit. Les logiciels étant omniprésents dans notre société actuelle, leur qualité s'avère tributaire à la sécurité de la population. Toute contribution pouvant collaborer à la promotion de la qualité logicielle, telle que celle offerte par le PQAL, s'avère donc pertinente.

REFERENCES

AL-SHEMMERI, T., AL-KLOUB, B. et PEARMAN, A. (1997 March). Model choice in multicriteria decision aid. European Journal of Operational Research, 97, 3, 550-560.

BASILI, V.R. et ROMBACH, H.D. (1988 June). A TAME project: Towards improvement-oriented software environments. IEEE Transactions on Software Engineering, 14, 6, 758-773.

BASILI, V.R. et WEISS, D.M. (1984 November). A methodology for collecting valid software engineering data. IEEE Transactions on Software Engineering, 10, 6, 728-738.

BELTON, V. et GEAR, T. (1983). On a shortcoming of Saaty's method of analytic hierarchies. Omega, 11, 3, 228-230.

BEVINGTON, P.R. et ROBINSON, D.K. (1992). Data reduction and error analysis for the physical sciences (2nd Edition). McGraw-Hill, New York.

BOLOIX, G. et ROBILLARD, P.N. (1995). A software system evaluation framework. Computer, 28, 12, 17-26.

BOUCHER, T. et MacSTRAVIC, E. (1991). Multi-attribute evaluation within a present worth framework and its relation to the Analytic Hierarchy Process. The Engineering Economist, 37, 1-32.

BRADAC, M.G., PERRY, D.E. et VOTTA, L.G. (1994 October). Prototyping a process monitoring experiment. IEEE Transactions on Software Engineering, 20, 10, 774-784.

DASKALANTONAKIS, M.K. (1994). Achieving higher SEI levels. IEEE Software, 11, 4, 17-24.

DRAKE, T. (1996 November). Measuring software quality: A case study. Computer, 29, 11, 78-87.

DUTOIT, A.H. et BRUEGGE, B. (1998). Communication Metrics for Software Development. IEEE Transactions on Software Engineering, 24, 8, 615-628.

DYER, J.S. (1990a). Remarks on the Analytic Hierarchy Process. Management Science, 36, 3, 249-258.

DYER, J.S. (1990b). A clarification on the «Remarks on the Analytic Hierarchy Process». Management Science, 36, 3, 274-275.

DYER, J.S. et RAVINDER, H.V. (1983). Irrelevant alternatives and the Analytic Hierarchy Process. Working Paper, The University of Texas at Austin, USA.

DYER, J.S. et WENDELL, R.E. (1985). A critique of the Analytic Hierarchy Process. Working paper 84/85-4-24, Department of Management. The University of Texas at Austin, USA.

EBERT, C. (1999 February). Technical controlling in software development, International Journal of Project Management, 17, 1, 17-28.

FENTON, N.E. et PFLEEGER, S.L. (1997). Software metrics: A rigorous approach and practical approach. PWS Publishing Company, Boston.

FINNIE, G.R., WITTIG, G.E. et PETKOV, D.I. (1993 August). Prioritizing software development productivity factors using the analytic hierarchy process. Journal of Systems and Software, 22, 2, 129-139.

FLORAC, W.A. et CARLETON, A.D. (1999). Measuring the software process: Statistical process control for software process improvement. Addison-Wesley, Reading, Massachusetts.

FLORAC, W.A. et CARLETON, A.D. (1999). Measuring the software process: Statistical process control for software process improvement. Addison-Wesley, Reading, Massachusetts.

HALL, T. et FENTON, N. (1997 March-April). Implementing effective software metrics programs. IEEE Software, 14, 2, 55-65.

HAUSER, J. et KATZ, G. (1998 October). Metrics: You are what you measure! European Management Journal, 16, 5, 517-528.

HOLLENBACH, C., YOUNG, R., PFLUGRAD, A.L. et SMITH, D. (1997 June). Combining quality and software improvement. Communications of the ACM, 40, 6, 41-45.

HO-WON, J. (1998). Optimizing value and cost in requirements analysis. IEEE Software, 15, 4, 74-78.

IEEE 1061-1992 (1992). IEEE standard for a software quality metrics methodology. ANSI, New York.

IEEE 982.1-1998 (1998). IEEE Standard Dictionary of Measures to Produce Reliable Software. ANSI, New York.

ISO 9126 (1991). Information technology – Software product evaluation – Quality characteristics and guidelines for their use. ANSI, New York.

KAMENETSKY, R.D. (1982). The relationship between the Analytic Hierarchy Process and the additive value function. Decision Science, 13, 702-713.

KARLSSON, J. et RYAN, K. (1997 September - October). Cost-value approach for prioritizing requirements. IEEE Software, 14, 5, 67-74.

KARLSSON, J., WOHLIN, C. et REGNELL B. (1998 February). An evaluation of methods for prioritizing software requirements. Information and Software Technology, 39, 14-15, 939-947.

KITCHENHAM, B., PFLEEGER, S.L. et FENTON, N. (1995). Towards a Framework for Software Measurement Validation. IEEE Transactions on Software Engineering, 21, 12, 929-943.

KRUCHTEN, P. (2000). Introduction au Rational Unified Process. Éditions Eyrolles, Paris.

LEBLANC, D. (1999 Septembre). Notes de cours: Gestion du risque, échéanciers et suivi de projets. Département de Génie Industriel. École Polytechnique de Montréal, Canada.

LEBLANC, D., PASIN, F. et PRIMEAU, M. (1993 Janvier). Construction d'un préordre complet sur les projets avec la méthode AHP modifié. Département de Génie Industriel. École Polytechnique de Montréal, Canada.

LEE, J. et XUE, N.-L. (1999). Analyzing user requirements by use cases: A goal-driven approach. IEEE Software, 16, 4, 92-101.

OMAN, P. ET PFLEEGER, S.L. (1997). Applying software metrics. IEEE Computer Society Press. Los Alamitos, CA.

PERRY, D.E., STAUDENMAYER, N.A. et VOTTA, L.G. (1994). People, organizations, and process improvement. IEEE Software, 11, 4, 36-45.

PFLEEGER, S.L. (1995 November). Maturity, models, and goals: How to build a metrics plan. Journal of Systems and Software, 31, 2, 143-155.

PFLEEGER, S.L., FITZGERALD JR., J.C. et RIPPY, D.A. (1992). Using multiple metrics for analysis of improvement. Quality Software, 1, 27-36.

PFLEEGER, S.L., JEFFERY, R., CURTIS, B. et KITCHENHAM, B. (1997 March – April). Status report on software measurement. IEEE Software, 14, 2, 33-43.

PRESSMAN, R.S. (1997). Software engineering: A practitioner's approach. McGraw-Hill, New York.

REBSTOCK., S. et KAULA, R. (1996). Effectiveness of an analytic hierarchy process in group decision making: A case Study. International Journal of Computer Application in Technology. 9, 2-3, 95-104.

ROBERTS, F.S. (1979). Encyclopedia of mathematics and its applications : Measurement theory. Addison-Wesley, Reading, Massachusetts.

ROBILLARD, P.-N. (2000). Case study analysis of measuring effort in a software engineering process. Proceeding 6th Maghrebian Conference on Computer Sciences.

ROBILLARD, P.-N. et ROBILLARD, M.P. (1998). Improving academic software engineering projects: A comparative study of academic and industry projects. Annals of Software Engineering, 6, p. 343-363.

ROBILLARD, P.-N. et ROBILLARD, M.P. (2000). Types of Collaborative work in software engineering. The Journal of Systems and Software, 53, 3, 219-224.

ROY, B. et BOUYSSOU, D. (1993). Aide multicritère à la décision: Méthodes et cas. Économica, Paris.

ROYCE, W. (1998). Software project management: A unified framework. Addison-Wesley, Reading, Massachusetts.

SAATY T.L. (1972). An eigenvalue allocation model for prioritization and planning. Energy Management and Policy Center, University of Pennsylvania, USA.

SAATY, T.L. (1984). Décider face à la complexité: Une approche analytique multicritère d'aide à la décision. Université Entreprise, Paris, France.

SAATY T.L. (1995). Decision making for leaders : The Analytic Hierarchy Process for decisions in a complex world, Vol. II of the AHP Series. RWS Publication, Pittsburgh, PA.

SAATY T.L. (1996). Multicriteria decision making: The Analytic Hierarchy Process, Vol. I of the AHP Series. RWS Publication, Pittsburgh, PA.

SCHONER, B. et WEDLEY, W.C. (1989). Ambiguous criteria weights in AHP: Consequences and solutions. Decision Sciences, 20, 462-475.

TERVONEN, I. (1996). Support for quality-based design and inspection. IEEE Software, 13, 1, 44-54.

VOOGH, H. (1983). Multicriteria evaluation for urban and regional planning. Pion, London.

ZANAKISA, S.H., SOLOMONB, A., WISHARTA N. et DUBLISH. S. (1998 June). Multi-attribute decision making: A simulation comparison of select methods. European Journal of Operational Research, 107, 3, 507-529.

ZELENY, M. (1982). Multiple Criteria decision making. McGraw-Hill, New York.