ÉCOLE DE TECHNOLOGIE SUPÉRIEURE
UNIVERSITÉ DU QUÉBEC

THÈSE PRÉSENTÉE À
L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE
À L'OBTENTION DU
DOCTORAT EN GÉNIE
Ph.D.

PAR
CHEIKHI, Laila

ÉTUDES EMPIRIQUES DES RELATIONS ENTRE LES MODÈLES DE QUALITÉ DU LOGICIEL D'ISO 9126 EN UTILISANT LE RÉFÉRENTIEL DE DONNÉES D'ISBSG ET LA MÉTHODE TAGUCHI

MONTRÉAL, LE 6 MARS 2008

CETTE THÈSE A ÉTÉ ÉVALUÉE

PAR UN JURY COMPOSÉ DE :

M. Alain Abran, directeur de thèse
Département de génie logiciel et des technologies de l'information à l'École de technologie supérieure

M. Witold Suryn, président du jury
Département de génie logiciel et des technologies de l'information à l'École de technologie supérieure

M. Pierre Bourque, membre du jury
Département de génie logiciel et des technologies de l'information à l'École de technologie supérieure

M. Yann-Gaël Guéhéneuc, examinateur externe
Département d'informatique et recherche opérationnelle à l'Université de Montréal

ELLE A FAIT L'OBJET D'UNE SOUTENANCE DEVANT JURY ET PUBLIC

LE 19 FÉVRIER 2008

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

À la mémoire de mon père

À ma mère

REMERCIEMENTS

J'adresse tout d'abord mes remerciements les plus sincères et ma gratitude la plus profonde au professeur Dr. Alain Abran, mon directeur de thèse, qui m'a fait confiance en me proposant ce travail de recherche. Je tiens à lui exprimer ma profonde reconnaissance pour sa disponibilité, son support continu, ses conseils judicieux, ses directives si enrichissantes et son aide inconditionnelle durant mes études à l'École de Technologie Supérieure. Sans son soutien ce travail n'aurait pas été accompli.

Je tiens à remercier les membres de mon jury : le professeur Witold Suryn pour avoir accepté la présidence du jury, le professeur Pierre Bourque d'avoir bien accepté être un membre de ce jury et le professeur Yann-Gaël Guéhéneuc d'avoir bien voulu être membre externe sur ce jury.

Une dette spéciale de gratitude va à mes très chers parents qui ont fait de moi ce que je suis aujourd'hui. Je les remercie pour leurs conseils, leurs encouragements, leurs sacrifices tout au long de mes études et ma vie en général, et surtout pour avoir semé en moi le goût pour la connaissance et le désir de l'acquérir.

J'adresse mes remerciements à mes frères, à mes sœurs et à tous les membres de ma famille, et qu'ils trouvent ici la reconnaissance de leurs encouragements et leurs soutiens continuels.

Je tiens à remercier tous les membres du laboratoire GÉLOG de l'ÉTS et toutes les personnes qui ont contribué de près ou de loin à la réalisation de ce travail.

Cette thèse est dédiée à tous les membres de ma famille, à mes amies et à mes collègues. Merci à tous.

ÉTUDES EMPIRIQUES DES RELATIONS ENTRE LES MODÈLES DE QUALITÉ DU LOGICIEL D'ISO 9126 EN UTILISANT LE RÉFÉRENTIEL DE DONNÉES D'ISBSG ET LA MÉTHODE TAGUCHI

CHEIKHI, Laila

RÉSUMÉ

Avec l'évolution du domaine du génie logiciel, la gestion de la qualité du logiciel a évolué : elle s'est orientée non seulement vers la production de logiciels qui s'exécutent, mais aussi vers la satisfaction des objectifs pour lesquels ces logiciels sont conçus.

Contrairement au génie industriel, la gestion de la qualité du logiciel ne porte pas sur le traitement des produits physiques (hard) mais des produits logiques (soft), incluant le développement du produit logiciel. En génie logiciel la gestion de la qualité requiert l'élaboration d'un ensemble de critères de qualité que le logiciel doit satisfaire et la proposition de mesures pour évaluer cette qualité, tel que présenté dans ISO 9126. La série ISO 9126 comprend une norme internationale sur la qualité du produit logiciel et trois rapports techniques. La norme ISO 9126-1 définit deux modèles de qualité du produit logiciel. Le premier modèle de « qualité interne » et de « qualité externe » comprend un ensemble de six caractéristiques, subdivisées en un ensemble de 27 sous-caractéristiques pour lesquelles des mesures sont proposées dans les rapports techniques ISO TR 9126-2 et 3. Le deuxième modèle de « qualité en utilisation » comprend un ensemble de quatre caractéristiques et des mesures sont proposées dans le rapport technique ISO TR 9126-4 pour évaluer ces caractéristiques.

Cette série ISO (9126 parties 1 à 4) propose des liens entre ces trois modèles de qualité. Cependant, ces liens, bien que définis par un consensus international d'experts ISO, n'ont pas nécessairement été démontrés objectivement et empiriquement : ces liens ne devraient donc être considérés que comme un ensemble théorique intéressant, et pour lequel des expérimentations rigoureuses sont requises afin d'en démontrer indubitablement la validité.

L'expérimentation nécessite une collecte de données pour réaliser les expériences et faire ressortir les résultats des expériences. Cependant, en absence d'opportunités d'expérimentations en industrie, il est possible d'utiliser des référentiels de données disponibles en génie logiciel : par exemple, le référentiel de données industrielles de l'International Software Benchmarking Standards Group (ISBSG) regroupe un ensemble d'informations sur les différentes phases du cycle de vie du logiciel.

L'approche choisie pour ce projet de recherche est de combiner deux disciplines (génie industriel et génie logiciel) à travers l'utilisation de la méthode Taguchi afin de mener les expérimentations avec le contenu des documents ISO 9126 et en exploitant le

référentiel de données d'ISBSG. La méthode Taguchi de conception de plan d'expériences, développée par le Dr. Genichi Taguchi, combine des pratiques industrielles et statistiques et offre un moyen d'évaluer la qualité.

Le but principal de ce projet de recherche est d'explorer la pertinence des relations entre les modèles de qualité d'ISO 9126 dans la production de logiciels de qualité. Afin de démontrer la pertinence ou non des relations entre ces modèles, les objectifs spécifiques de cette recherche sont de démontrer, par des études empiriques, si les relations prises pour acquises par ISO 9126 sont supportées par des données empiriques. Il s'agit des relations entre :

1. La qualité interne et la qualité externe.
2. La qualité externe et la qualité en utilisation.
3. La qualité interne et la qualité en utilisation.

Pour réaliser ces objectifs, la méthodologie suivante a été suivie :

- vérifier jusqu'à quel point le questionnaire d'ISBSG tient compte des trois types de qualité définis dans la norme ISO 9126-1. Pour cela, nous avons aligné les différentes parties du questionnaire d'ISBSG par rapport à la qualité interne, la qualité externe et la qualité en utilisation de la norme ISO 9126-1;
- identifier les caractéristiques de qualité interne, externe et en utilisation couvertes par le questionnaire d'ISBSG à travers l'identification des données de qualité du questionnaire d'ISBSG et leurs mesures correspondantes dans les rapports techniques ISO TR 9126-2 à 4;
- analyser le référentiel de données d'ISBSG afin de déterminer les données de qualité disponibles pour évaluer les trois types de qualité du produit logiciel d'ISO 9126. Nous avons proposé des mesures propres à ISBSG à base de ces données. Ces mesures seront utiles lors de la conception des plans d'analyses empiriques avec la méthode Taguchi;
- adapter la méthode Taguchi de conception de plans d'expériences, d'ordre industriel, au contexte d'analyse empirique en génie logiciel, laquelle adaptation a porté sur l'étape de paramètres de design de la stratégie hors production (off-line) de contrôle de la qualité de Taguchi;
- établir et analyser les résultats des plans d'analyses empiriques permettant de vérifier les liens entre les trois types de qualité d'ISO 9126-1 en utilisant la méthode Taguchi et en exploitant les données de l'extrait du référentiel d'ISBSG mis à notre disposition pour des fins de recherche.

Les résultats de ces travaux de cette recherche ont permis la vérification des hypothèses des liens de la norme ISO 9126-1 entre les trois types de qualité : la qualité interne affecte la qualité externe qui affecte, à son tour, la qualité en utilisation du produit logiciel est effectivement justifiée dans cette thèse.

Les résultats de ces travaux de recherche pourront être utiles pour les chercheurs, praticiens et industriels dans le domaine de la qualité du logiciel. Les résultats de cette recherche pourront également être utiles pour l'organisation ISBSG et les experts d'ISO 9126 en particulier, et pour la discipline du génie logiciel en général.

Mots-clés : Standard ISO 9126, Modèles de Qualité, Mesures du Logiciel, Qualité du Logiciel, ISBSG, Taguchi, Paramètres de Design, Études Empiriques, ANOVA.

EMPIRICAL STUDIES OF THE RELATIONSHIPS ACROSS THE SOFTWARE QUALITY MODELS OF ISO 9126 THROUGH THE USE OF THE ISBSG REPOSITORY AND THE TAGUCHI METHOD

CHEIKHI, Laila

ABSTRACT

With the evolution of the software engineering discipline, the management of software quality has also evolved: it is directed not only towards the development of software which executes correctly, but also towards the satisfaction of the objectives for which this software is designed.

In software engineering the management of quality requires the development of a set of quality criteria that the software product must meet, including related software measures to evaluate this quality, such as proposed in ISO 9126. The ISO 9126 series includes an international standard on software product quality and three technical reports. The standard ISO 9126-1 defines two models of software product quality. The first model of 'internal quality' and 'external quality' includes a set of six characteristics, subdivided into a set of 27 subcharacteristics, and for which a large inventory of software measures are proposed in the technical reports ISO TR 9126-2 and 3. The second model of 'quality in use' includes a set of four characteristics and a number of measures are also proposed in the technical report ISO TR 9126-4 to evaluate them.

This series of ISO (9126 parts 1 to 4) takes for granted that these three quality models are inter-linked. However, these links, although defined through an international consensus of ISO participants, were not based on documented experimental studies: therefore these links proposed in ISO 9126 should be regarded as a theoretical interesting set of links but for which rigorous experiments are still required to provide documented evidence.

Experimentation requires the collection of data to carry out the experiments and to provide documented evidence. However, in the absence of experimentation opportunities in industry, most of time because of excessive costs, it is possible to use data available in software engineering repositories: for example, the International Software Benchmarking Standards Group (ISBSG) repository of industry data includes information for different phases of the software life cycle.

The approach chosen for this research project is to use of the Taguchi method of experiments to investigate the relationships proposed in ISO 9126 through the analysis of the ISBSG data repository. The Taguchi method for setting up experimental designs, combines industrial and statistical practices and offers a means of evaluating the quality.

The main goal of this research is to explore the relevance of the relationships between the ISO 9126 quality models in the development of the software. The goal of this research is to demonstrate the relevance or not of the relations between these models, while the specific objectives of this research are to demonstrate, through empirical studies, if the relationships taken for granted in ISO 9126 are indeed supported by empirical data: that is, the relationships between:

1. internal quality and external quality;
2. external quality and quality in use;
3. internal quality and quality in use.

The research methodology selected is the following:

- Verification to which extent the ISBSG questionnaire takes into account the three types of quality defined in the ISO 9126-1 standard. For this, the different parts of the ISBSG questionnaire were mapped to the models of internal quality, external quality and quality in use of the ISO 9126-1 standard.
- Identification of internal, external and in use quality characteristics covered by ISBSG through the identification of the quality related data from the ISBSG questionnaire and their corresponding measures in the technical reports ISO TR 9126-2 to 4.
- Analysis of the ISBSG data repository to determine the availability of quality related data to evaluate the three types of quality of the software product of ISO 9126. We have proposed measures specific to ISBSG based on these data. These measures will be useful when designing empirical analysis plans with the Taguchi method.
- Adaptation of the Taguchi method of design of experiments to the context of empirical analysis in software engineering, which adaptation is related to the stage of parameter design of the Taguchi strategy of off-line quality control.
- By using the Taguchi method and by exploiting the ISBSG data repository, design and analysis of the results of the empirical analysis plans to verify the links between the three types of quality of ISO 9126-1.

The results of this research project have confirmed the assumptions of the relationships of the ISO 9126-1 standard between the three types of quality, that is: internal quality affects the external quality which affects, in turn, quality in use of the software product.

The results of this thesis can be useful for researchers, practitioners and software organizations working on software quality. The research results can also be useful to the ISBSG organization and to ISO 9126 experts, as well as software engineering in general.

Keywords: ISO 9126 Standard, Quality Models, Software Measurements, Software Quality, ISBSG, Taguchi, Design Parameter, Empirical Studies, ANOVA.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABRÉVIATIONS, SIGLES ET ACRONYMES

AFP	Adjusted Function Points
AHP	Analytical Hierarchy Process
ANOVA	ANalysis Of Variance
ARFF	Attribute-Relation File Format
ASQ	Software Quality Assurance
Cfsu	COSMIC functional size unit
COCOMO	COnstructive COst Model
COSMIC	COmmon Software Measurement International Consortium
CDL	Cycle de Développement du Logiciel
CVL	Cycle de Vie du Logiciel
CVS	Concurrent Versions System
Dl	Degré de liberté
DQR	Data Quality Rating
F	Variance ratio
FFP	Full Function Points
FSM	Functional Size Measurement
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IFPUG	International Function Point Users Group
IS	International Standard
IT	Information Technology

LOC	Lines Of Code
Mark-II	Mark II Function Point Analysis
MS	MicroSoft
MIS	Management Information Systems
NESMA	NEtherlands Software Metrics Users Association
NFS	Network File System
PSM	Practical Software Measurement
QEST	Quality factor + Economic, Social & Technical dimensions
QoS	Quality of Service
S/B	Signal-Bruit
S/N	Signal-to-Noise
ISBSG	International Software Benchmarking Standards Group
ISO	International Organisation for Standardisation
MC	Mean Square
NASA	National Aeronautics and Space Administration
PC	Percentage of Contribution
PROMISE	PRedictOr Models In Software Engineering
SC	Sum of Squares
SLCP	Software Life Cycle Processes
SQuaRE	Software product Quality Requirements and Evaluation
SI	Système International d'unités
SWEBOK	SoftWare Engineering Body Of Knowledge

TR	Technical Report
UFP	Unadjusted Function Points
UML	Unified Modeling Language
UP	Unified Process
VIM	Vocabulaire International des termes fondamentaux et généraux de Métrologie
Weka	Waikato environment for knowledge analysis

INTRODUCTION

La qualité a toujours suscité l'intérêt aussi bien des producteurs de produits industriels que des développeurs de produits logiciels afin d'assurer la satisfaction de leurs clients et pour affronter la concurrence internationale. En génie logiciel, la qualité du logiciel a été reconnue comme un domaine de connaissance distinct des différents domaines de connaissances du « Guide au corpus des connaissances en génie logiciel » (Guide to the Software Engineering Body of Knowledge – SWEBOK-ISO TR 19759).

Problématique de recherche

Avec l'expansion et la diversité des domaines d'utilisations du logiciel, les organisations sont en quête continuelle de facteurs permettant de produire des logiciels de haute qualité. Un de ces facteurs est de veiller à assurer cette haute qualité durant les différentes phases du cycle de vie du logiciel, tout en prenant les mesures nécessaires à chaque étape. Ces mesures peuvent aider, en particulier, à évaluer la qualité du produit logiciel. D'après Fenton et Pfleeger (1997, p. 337), pour mesurer la qualité, il faut être capable de définir la qualité en termes d'attributs spécifiques du produit logiciel qui intéressent l'utilisateur ; c'est-à-dire, il faut disposer des connaissances nécessaires afin de permettre de spécifier les attributs de qualité sous forme de mesures. Il est aussi mentionné dans le Guide SWEBOK que « With the increasing sophistication of software, questions of quality go beyond whether or not the software works to how well it achieves measurable quality goals » (IEEE, 2004, p. 165). La qualité ne réside donc pas uniquement dans le fait d'avoir un logiciel opérationnel, mais également dans l'évaluation par rapport à des objectifs mesurables de qualité.

Jusqu'à présent, plusieurs mesures, plusieurs modèles de qualité et plusieurs normes ont été proposés afin de contribuer à la production de logiciels de qualité. L'Organisation Internationale de Standardisation (ISO), à travers sa série des documents ISO 9126, a

donné plus de formalisme à l'approche qualité du produit logiciel en normalisant plusieurs concepts spécifiques à la qualité du logiciel en tant que produit. Le premier document de la série ISO 9126 (2001a) est une norme internationale définissant un méta-modèle de qualité du produit logiciel, composé d'un ensemble de caractéristiques et sous-caractéristiques de qualité. Les trois autres documents de la série ISO 9126 (2003a), (2003b) et (2004) sont des rapports techniques proposant des inventaires de mesures pour évaluer quantitativement les caractéristiques et sous-caractéristiques du méta-modèle ISO de la qualité du produit logiciel.

Par ailleurs, le méta-modèle de la norme ISO 9126-1 propose trois modèles de qualité distinguant trois types de qualité : interne, externe et en utilisation. Ce méta-modèle propose aussi des liens entre ces différents types de qualité : la qualité interne influence la qualité externe qui influence, à son tour, la qualité en utilisation du produit logiciel. Par conséquent, la réalisation de la qualité en utilisation dépend, selon la proposition de ces modèles ISO, de la réalisation de la qualité externe qui dépend, à son tour, de la réalisation de la qualité interne du produit logiciel. Toutefois, ces liens, bien que définis par un consensus international d'experts ISO, sont basés principalement sur des opinions d'experts et non pas sur des démonstrations scientifiques transparentes, c'est-à-dire bien documentées tant en termes méthodologiques qu'en termes de traçabilité des résultats. Ainsi, les liens proposés par ISO 9126-1 ne devraient être considérés que comme un ensemble théorique, intéressant certes, mais pour lesquels des expérimentations rigoureuses sont encore requises afin d'en démontrer indubitablement d'une part la validité et, d'autre part, les limites et contraintes opérationnelles.

Il est d'ailleurs à observer qu'en génie logiciel les normes sont souvent fondées principalement sur des opinions d'experts qui se réunissent pour former un groupe de travail ISO. En effet, même si les normes apportent des connaissances importantes à la discipline du génie logiciel dans tous ses domaines de connaissances (incluant la qualité), le manque de support expérimental à certaines de ces normes est un signe

d'immaturité. Cette faiblesse de certaines normes a également été mentionnée par Pfleeger, Fenton et Page (1994, p. 71) :

> « Standards have codified approaches whose effectiveness has not been rigorously and scientifically demonstrated. Rather, we have too often relied on anecdote, "gut feeling," the opinions of experts, or even flawed research, rather than on careful, rigorous software engineering experimentation. »

La norme ISO 9126 est une de ces normes basées principalement sur des opinions d'experts et pour laquelle des expérimentations devraient permettre d'en démontrer, ou d'en infirmer, les hypothèses de base et ainsi à contribuer à la maturation de la discipline du génie logiciel.

Dans cette problématique, il est utile d'explorer par des études expérimentales les modèles de qualité proposés par la norme ISO 9126-1 pour la production de logiciels de qualité. Dans ce projet de recherche, nous nous intéressons à explorer la relation entre la qualité interne, la qualité externe et la qualité en utilisation du produit logiciel.

Pour effectuer une étude expérimentale, il faut tout d'abord la planifier. Il est à noter que dans ce texte, les expressions « expériences » et « expérimentations » seront préférées à l'expression « études expérimentales ». Différentes méthodes de conception de plans d'expériences sont présentées dans la littérature. Pour ce projet de recherche, la méthode Taguchi (2004) de design d'expérimentations est choisie. Cette méthode a été développée par le Dr. Genichi Taguchi au Japon dans les années 1960. Cette méthode d'expérimentation se base sur des techniques statistiques et permet à terme, entre autres, la réduction du coût de production et l'amélioration de la qualité des produits ou des processus. La méthode Taguchi a prouvé son efficacité dans divers domaines du secteur industriel (Peace, 1993; Phadke, 1989; Ross, 1996; Roy, 2001; Taguchi, Chowdhury et Wu, 2004), son utilisation dans le domaine du génie logiciel en est par contre à ses débuts.

De plus, mener une expérience nécessite généralement la collecte de données. En effet, une des contraintes majeures entravant la conduite des expériences en génie logiciel réside dans la difficulté de collecter les données dans un contexte de calibre industriel (et ce, de préférence à une collecte de données dans des projets étudiants dans un contexte académique). De plus, les bases de données valides pour supporter les études empiriques en génie logiciel sont rarement disponibles pour les chercheurs.

Il existe toutefois une exception à cette non disponibilité générale de données en génie logiciel. En effet, l'International Software Benchmarking Standards Group (ISBSG, 2005) dispose d'une base de données que l'on peut accéder à prix modique pour des fins de recherche. En fait, ISBSG possède une base de données de projets de développement de nouveaux logiciels, de redéveloppement ou d'amélioration à des logiciels existants. De plus, les questionnaires de collectes de ces données sont disponibles sur le site Web d'ISBSG. Cette base de données d'ISBSG, de plus de 3000 projets, a été utilisée par plusieurs chercheurs et pour divers objectifs de recherche, à savoir : l'estimation de l'effort, de la productivité, etc. (Abran, Garbajosa et Cheikhi, 2007; Bourque *et al.*, 2007; Déry et Abran, 2005; Dolado *et al.*, 2007). Dans ce projet de recherche, il s'agit d'utiliser cette base de données pour l'évaluation de la qualité du produit logiciel.

Objectifs de la recherche

ISO 9126 propose un ensemble de modèles de qualité avec caractéristiques, sous-caractéristiques et mesures pour évaluer la qualité interne, la qualité externe et la qualité en utilisation du logiciel. Le but principal de ce projet de recherche est d'explorer la pertinence des relations entre les modèles de qualité d'ISO 9126 dans la production de logiciels de qualité.

Afin de démontrer la pertinence ou non des relations entre ces modèles, les objectifs spécifiques de cette recherche sont de démontrer, par des études empiriques, si les

relations prises pour acquises par ISO 9126 sont supportées par des données empiriques. Il s'agit des relations entre :

- la qualité interne et la qualité externe;
- la qualité externe et la qualité en utilisation;
- la qualité interne et la qualité en utilisation.

La méthodologie que nous avons conçue pour atteindre ces objectifs est repartie en trois grandes phases : exploration, préparation et plan d'analyse empirique. La phase exploration consiste en une exploration globale de la série des documents ISO 9126, de la méthode Taguchi de conception de plans d'expériences et des référentiels de données disponibles dans la littérature en génie logiciel. L'exploration détaillée porte sur un ensemble de mises en correspondance entre la série des documents ISO 9126 (modèles de qualité et mesures) et ISBSG (questionnaire de collecte de données et extrait de données d'ISBSG-Release 9 de 2005).

La deuxième phase consiste en la préparation de l'environnement des plans d'analyses empiriques : la sélection des données à exploiter à partir de l'extrait de données d'ISBSG de février 2006 (obtenu suite à notre demande d'accès aux données d'ISBSG spécifiques à ce projet de recherche), la vérification de ces données et la présentation des différentes étapes d'analyse des résultats des plans d'analyses avec la méthode d'analyse du ratio signal-bruit (S/B) de Taguchi.

La troisième phase consiste en l'adaptation de la méthode Taguchi de conception de plans d'expériences au contexte d'analyse empirique pour vérifier les hypothèses des liens entre les trois types de qualité du produit logiciel de la norme ISO 9126-1.

Organisation de la thèse

Le présent document est organisé autour de 10 chapitres suivi d'une conclusion et de six annexes.

Le chapitre un permet de situer notre problématique de recherche dans un contexte plus global de la qualité du produit logiciel. Ce chapitre présente une revue de la littérature sur les principaux points en relation avec la qualité du produit logiciel. Tout au long de ce chapitre un ensemble de questions sont identifiées en relation avec la mesure, le modèle de qualité, l'expérimentation et l'immaturité du génie logiciel.

Le chapitre deux fait le point sur la série ISO 9126 relative à la qualité du produit logiciel par rapport à ses quatre parties : modèles et mesures de qualité interne, externe et en utilisation du produit logiciel. Ce chapitre deux présente un résumé de l'état de l'art des travaux de recherche effectués avec la série ISO 9126 ainsi qu'une liste non exhaustive de ses forces et ses faiblesses.

Le chapitre trois décrit les différents référentiels de données disponibles dans la littérature tout en faisant ressortir les limites et les avantages de leur utilisation pour ce travail de recherche. Il s'agit principalement des bases de données du référentiel d'ingénierie de logiciel « PRedictOr Models In Software Engineering (PROMISE) » et du référentiel de données industrielles d'ISBSG ainsi que de son questionnaire de collecte de données.

Le chapitre quatre introduit la méthode Taguchi de design de plans d'expériences. Ce chapitre présente les concepts de base utilisés par la méthode Taguchi, à savoir : la stratégie de qualité, les tables orthogonales, le ratio signal-bruit, etc. Ce chapitre cite des avantages de l'utilisation de cette méthode pour le logiciel et présente un état de l'art de

travaux de recherche ayant utilisé la méthode Taguchi dans des expérimentations avec des logiciels.

Le chapitre cinq présente les objectifs de cette recherche ainsi que la méthodologie de recherche conçue pour aborder cette recherche. Ce chapitre décrit les principales étapes à réaliser afin d'atteindre nos objectifs de recherche.

Le chapitre six est axé principalement autour d'ISO 9126 (modèles de qualité est mesures) et d'ISBSG (questionnaire de collecte de données et extrait de données d'ISBSG-Release 9 de 2005). Ce chapitre présente en premier lieu une mise en correspondance (de haut niveau et détaillée) des concepts de qualité de la norme ISO 9126-1 et du questionnaire d'ISBSG, puis une analyse de l'extrait de données d'ISBSG-Release 9 de 2005 afin de faire ressortir les champs de données de qualité non disponibles dans cette extrait et, par la suite, en faire une demande d'accès à ces données auprès d'ISBSG. Ce chapitre présente ensuite le résultat de la mise en correspondance (de haut niveau et détaillée) des mesures de qualité des rapports techniques ISO TR 9126-2 à 4 par rapport au questionnaire d'ISBSG, suivi de l'exploration des exemples de mesures d'ISO 9126. Enfin, ce chapitre identifie dans la base de données d'ISBSG les mesures de qualité du produit logiciel jugées utiles pour les expériences.

Le chapitre sept est consacré aux étapes de préparation de l'environnement des plans d'analyses empiriques : il s'agit de recenser les données à exploiter à partir d'un nouvel extrait de données d'ISBSG de février 2006 reçu suite à notre demande d'accès aux données les plus récentes, puis vérifier la qualité et la complétude des données de cet extrait et enfin présenter les différentes étapes de l'analyse des résultats des plans d'analyses avec la méthode d'analyse du ratio signal-bruit de Taguchi.

Les chapitres huit, neuf et dix présentent les plans d'analyses conçus avec la méthode Taguchi afin de vérifier les hypothèses des liens entre les trois types de qualité d'ISO 9126-1 :

- le chapitre huit explore le lien entre la qualité interne et la qualité externe;
- le chapitre neuf explore le lien entre la qualité externe et la qualité en utilisation;
- le chapitre dix explore le lien entre la qualité interne et la qualité en utilisation.

Le chapitre de la conclusion générale dresse le bilan de ce travail de recherche, présente les limites rencontrées et montre l'apport et l'impact de ce travail de recherche sur le domaine de la qualité en génie logiciel. Ce chapitre précise aussi nos principales contributions et fournit des perspectives d'avenir dans ce domaine.

Un ensemble de six annexes clos cette thèse. La première, Annexe I, énumère la liste des publications issues de ce travail de recherche. La deuxième, Annexe II, présente les contenus des modèles de qualité de McCall, Boehm et Dromey. La troisième, Annexe III, présente la structure détaillée de l'extrait de données d'ISBSG-Release 9 de 2005. La quatrième, Annexe IV, présente l'analyse effectuée sur l'extrait de données d'ISBSG-Release 9 de 2005. La cinquième, Annexe V, présente l'ensemble des mesures identifiées à partir des données disponibles dans le questionnaire d'ISBSG. La sixième, Annexe VI, présente le questionnaire préparé dans cette recherche sur la qualité du produit logiciel ainsi que le sommaire des résultats de l'utilisation de ce questionnaire.

CHAPITRE 1

QUALITÉ DU PRODUIT LOGICIEL

1.1 Introduction

> « In spite of the millions of software professionals worldwide and the ubiquitous presence of software in our society, software engineering has only recently reached the status of a legitimate engineering discipline and a recognized profession. » (IEEE, 2004, p. 24)

Pour approfondir notre problématique de recherche, nous proposons, dans ce chapitre, une revue de la littérature sur les différentes définitions (ou aspects) de la qualité proposées par les experts en qualité. Ensuite, nous présentons un axe important de la qualité du logiciel : la mesure. Vu l'importance de ce thème pour la discipline du génie logiciel, nous présentons une discussion sur les mesures du logiciel en trois volets : maturité, expérimentation et modèles de qualité.

1.2 Qualité du logiciel

La quête de la qualité est un but qui ne date pas d'aujourd'hui et avec l'avancement technologique la qualité est devenue un problème d'actualité et un défi à relever dans presque tous les domaines. En génie logiciel comme en génie industriel, assurer la qualité du produit contribue à la survie et au succès des organisations. La poursuite de la qualité est un processus continu dans lequel chaque membre de l'organisation participe avec son expertise dans son achèvement : c'est l'affaire de tout le monde et non pas d'une seule personne.

Les questions que nous nous posons sont : Qu'est ce que la qualité du logiciel ? Comment atteindre cette qualité ?

1.2.1 Qu'est ce que la qualité du logiciel ?

Kitchenham et Pfleeger (1996, p. 21) soulignent que la qualité « is a complex concept. Because it means different things to different people, it is highly context-dependent. Just as there is no one automobile to satisfy everyone's needs, so too there is no universal definition of quality ». En l'absence d'une définition universellement acceptée du concept qualité, la diversité des points de vue rend le terme ambigu, voire difficile à comprendre. Selon Kan (2003), la difficulté de compréhension du terme qualité vient du fait que la qualité représente un concept multidimensionnel (comprenant l'entité, le point de vue sur cette entité et les attributs de qualité de cette entité), se comprend dans plusieurs niveaux d'abstraction et son utilisation diffère d'une personne à l'autre.

Afin de réduire l'ambigüité que présente ce concept, David Garvin a décrit la qualité selon cinq grandes perspectives (Pfleeger et Altee, 2005) :

- la vue transcendantale : la qualité peut être reconnue sans pouvoir la définir;
- la vue utilisateur : la qualité correspond à une réponse aux besoins (i.e., « fitness for purpose »);
- la vue manufacturier : la qualité est la conformité aux spécifications;
- la vue produit : la qualité est liée aux caractéristiques inhérentes du produit;
- la vue valeur : la qualité est dépendante du prix que le client est disposé à payer.

Ainsi, dans la littérature, plusieurs définitions (ou concepts) en relation avec la qualité du logiciel ont été émises mettant en évidence cette diversité de points de vue. Nous en avons choisies quelques unes (Jones, 1996) parmi les plus populaires :

- conformité aux exigences;
- satisfaction de l'utilisateur;
- absence de défauts;
- respect de l'échéancier et du budget;
- propriétés mesurables du logiciel comme la fiabilité, la maintenabilité, etc.

Il apparait clairement de ces différentes définitions sur la qualité un manque de consensus parmi les experts de la qualité sur sa définition ou sa signification. Cette diversité d'opinions peut s'expliquer ainsi : « the perspective we take on quality influences how we define it » (Kitchenham et Pfleeger, 1996, p. 15). Néanmoins, bien que ces définitions soient différentes, elles adressent des aspects ou des pratiques à prendre en considération en vue d'atteindre la qualité du logiciel, à savoir : la conformité aux exigences, l'absence de défauts, la satisfaction des utilisateurs, etc.

D'autre part, tandis que chacune des définitions précédentes se concentre sur un aspect ou une pratique spécifique de la qualité (sens étroit de la qualité), la norme ISO 9126 englobe tous les aspects et perspectives de qualité (sens large de la qualité). La qualité est ainsi définie dans ISO 9126 (2001a, p. 20) comme étant : « The totality of characteristics of an entity that bear on its ability to satisfy stated and implied needs ». Cette définition représente le consensus des experts d'ISO en qualité du produit logiciel.

Dans ce travail de recherche, nous adoptons cette définition d'ISO 9126 pour le contexte de la qualité du produit logiciel. En particulier, l'entité à mesurer (i.e., « entity ») représente le produit logiciel et ce dernier est défini comme étant : « The set of computer programs, procedures, and possibly associated documentation and data. Products include intermediate products, and products intended for users such as developers and maintainers » (ISO, 2001a, p. 21).

1.2.2 Comment atteindre la qualité du logiciel ?

Comment atteindre la qualité du logiciel ? Répondre à ce défi requiert la mise en place d'une méthodologie afin de produire des logiciels de qualité. Cette méthodologie doit prendre en considération différents facteurs influençant la production de logiciel, à savoir : les méthodes, les techniques, les ressources humaines, la mesure, le moyen financier, le domaine d'application, etc.

La réponse à ce défi requiert une ingénierie de la qualité telle que définie par Suryn (2005, p. 6) (et adaptée de la définition même du génie logiciel proposée par l'Institut des ingénieurs électriciens et électroniciens (IEEE)) : « the application of a continuous, systematic, disciplined, quantifiable approach to the development and maintenance of quality of software products and systems; that is, the application of quality engineering to software ».

De cette définition, il en ressort que l'ingénierie de la qualité est une approche continue requérant un suivi et une amélioration permanente, systématique, nécessitant un ensemble de méthodes, de techniques et d'outils dont l'utilisation requiert de la rigueur et de la discipline. C'est aussi une approche quantifiable soulignant d'un côté la nécessité de la mesure comme une partie entière de l'ingénierie de la qualité et, d'un autre côté, son utilité pour contrôler et améliorer la qualité durant le développement et la maintenance du produit logiciel.

1.3 Mesures en génie logiciel

À la fin du 19ème siècle, le physicien Lord Kelvin disait de la mesure :

> « When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind: it may be the beginning of knowledge, but you have scarcely, in your thoughts, advanced to the stage of science. » (Cité par Pressman, 2004, p. 79)

La mesure ne se limite pas à un domaine bien particulier. En effet, on mesure souvent sans se rendre compte de l'acte lui-même même lorsque l'on s'intéresse à une valeur qualitative plutôt qu'uniquement à une représentation numérique. D'après cette citation de Lord Kelvin, la mesure représente une mise en correspondance entre des choses appartenant à un monde empirique (monde réel : « what you are speaking about ») et d'autres choses appartenant au monde mathématique (monde des nombres et symboles :

« express it in numbers »). Ce passage entre deux mondes différents permet d'enrichir les connaissances (« you know something about it »), vu que la représentation mathématique offre un moyen plus rigoureux de manipulation, d'expérimentation et d'évaluation qu'offre une observation directe du monde réel.

Les mesures sont importantes non seulement dans le domaine des sciences exactes (par exemple, physique, chimie, etc.), mais aussi en ingénierie du logiciel. Dans cette section, nous discutons des mesures du logiciel et de la maturité du génie logiciel, des mesures du logiciel et de l'expérimentation, et des mesures du logiciel et de modèles de qualité tout en s'appuyant sur des travaux de recherche dans le domaine.

1.3.1 Mesures du logiciel et maturité du génie logiciel

Avec l'évolution du génie logiciel et de ses techniques, plusieurs mesures ont été proposées dans la littérature afin de répondre aux différents besoins en matière de qualité, donnant naissance en particulier à deux écoles de pensée sur la mesure en génie logiciel.

D'un côté, il y a les mesures qui sont conçues dans le cadre du paradigme structuré ou dans le cadre de l'orienté objet (Abreu et Carapuça, 1994; Briand, Daly et Wuest, 1997; 1999; Briand, Devanbu et Melo, 1997; Chidamber et Kemerer, 1991; 1994; Li et Henry, 1993; Lorenz et Kidd, 1994); il s'agit des mesures qui touchent les propriétés architecturales du design de logiciel dont le couplage, la taille, la cohésion, l'héritage, etc.

D'un autre côté, il existe des mesures qui sont indépendantes de la technique utilisée; il s'agit des mesures proposées par ISO 9126. En effet, ISO TR 9126-2 (2003a) et ISO TR 9126-3 (2003b) proposent un ensemble de mesures pour mesurer de façon quantitative les caractéristiques de qualité interne et externe du produit logiciel dont la capacité

fonctionnelle, la fiabilité, la facilité d'utilisation, le rendement, la maintenabilité et la portabilité. ISO TR 9126-4 (2004) propose un ensemble de mesures pour les caractéristiques de qualité en utilisation du produit logiciel dont l'efficacité, la productivité, la sécurité et la satisfaction.

Ces deux écoles de pensée sont donc différentes dans leurs manières de définir les mesures; toutefois elles convergent vers la qualité du logiciel final et la satisfaction des besoins de l'utilisateur.

D'autre part, même si le domaine de génie logiciel propose un très grand nombre de mesures et un corpus de connaissances de principes de mesures, méthodes de mesures, objectifs de mesures, domaines d'exploitation des résultats des mesures et quelques normes de mesures, ce domaine n'est pas encore complètement mature. En effet, afin de juger qu'un domaine est mature ou non, il faut se baser sur des critères reconnus de comparaison pour pouvoir comparer et juger sa maturité. Un ensemble d'outils est utilisé par les chercheurs pour des fins d'analyse dans le domaine de la mesure. Citons à titre d'exemple d'outils, le modèle de processus de mesure proposé par (Jacquet, Abran et Dupuis, 1997; Jacquet et Abran, 1997) et le Vocabulaire International des termes fondamentaux et généraux de Métrologie (VIM) (ISO, 1993). Le premier s'intéresse à la conception d'une méthode de mesure, son application, ses résultats et l'exploitation des résultats. Le deuxième (VIM) présente les fondements relatifs à la métrologie, dont les grandeurs et les unités, le mesurage, les étalons, les instruments de mesures, les résultats de mesures et les caractéristiques des instruments de mesures.

Des études ont été effectuées dans (Abran, Al-Qutaish et Cuadrado-Gallego, 2006; Abran, Lopez et Habra, 2004; Abran et Sellami, 2004; Abran, Sellami et Suryn, 2003; Al-Qutaish et Abran, 2005; Cheikhi, Abran et Miguel, 2005) pour analyser des mesures du logiciel en utilisant le modèle de processus de mesure proposé par Jacquet & Abran et le VIM. De ces études, il en ressort qu'à date le domaine de la mesure en génie

logiciel présente encore des insuffisances dans les designs des mesures, les instruments de mesures et les étalons.

Par ailleurs, la notion de mesure est déjà stable et mature dans les domaines des sciences exactes. En effet, cette maturité et cette stabilité sont le fruit de plusieurs siècles de recherches, de théories, de pratiques et d'expérimentations. À titre d'exemple, l'unité de mesure du mètre a pris beaucoup de temps afin d'être construite et acceptée dans le système international.

1.3.2 Mesures du logiciel et expérimentation

Pour améliorer et faire évoluer le domaine de la mesure, il faut savoir d'abord l'utilité de la mesure pour le logiciel : pourquoi a-t-on besoin de mesurer le logiciel ? En effet, on ne mesure pas un logiciel pour le plaisir de mesurer, mais pour des objectifs bien définis. Oman et Pfleeger (1997, p. 1-2) ont distingué six objectifs pour mesurer. Ces objectifs sont : mesurer pour la compréhension, mesurer pour l'expérimentation, mesurer pour le contrôle de projet, mesurer pour l'amélioration du processus, mesurer pour l'amélioration du produit et mesurer pour la prédiction. À ces objectifs s'ajoute l'objectif de mesurer pour l'évaluation proposé par Fenton et Pfleeger (1997, p. 13).

Les mesures permettent, entre autres, une amélioration des connaissances en matière de production de logiciel et une meilleure compréhension des facteurs ayant un grand impact sur la qualité du logiciel final. D'après le guide SWEBOK :

> « If they are selected properly, measures can support software quality (among other aspects of the software life cycle processes) in multiple ways. They can help in the management decision-making process. They can find problematic areas and bottlenecks in the software process; and they can help the software engineers assess the quality of their work for SQA purposes and for longer-term process quality improvement. » (IEEE, 2004, p. 165)

Ainsi, les mesures représentent un repère qui nous permet de savoir où somme nous à l'état actuel, d'identifier les défaillances et les forces et enfin d'améliorer pour le futur. De ce fait, disposer de mesures est un avantage pour :

- les gestionnaires lors de la prise de décision afin de réduire le risque de s'embarquer dans des dépenses coûteuses, inutiles ou imprévisibles en évaluant les ressources : humaines, financières, temps, effort et plannings nécessaires;
- les ingénieurs lors du développement et de l'implémentation du logiciel afin d'améliorer les méthodes, les techniques et les processus;
- les utilisateurs afin d'évaluer jusqu'à quel point les besoins sont satisfaits et les objectifs sont atteints.

Par ailleurs, une étude a été menée par Bourque *et al.* (2004) afin d'évaluer les perceptions du rôle de la mesure dans le domaine du génie logiciel à travers l'analyse d'un ensemble de commentaires obtenus sur deux études Delphi et un sondage. D'après ces auteurs, l'absence de commentaires sur l'objectif « mesurer pour l'expérimentation » peut être relié à la faible présence de l'expérimentation en ingénierie du logiciel (Bourque *et al.*, 2004). Une telle remarque laisse entendre qu'en génie logiciel, l'expérimentation n'occupe pas encore une place importante (Wolff, 1999).

Néanmoins, les travaux de (Basili, 1996; Basili, Selby et Hutchens, 1986) permettent de clarifier l'utilité de l'expérimentation en génie logiciel. En effet, Basili, Selby et Hutchens (1986, p. 733-741) ont souligné d'une part que l'expérimentation en génie logiciel « supports advancement of the field through an iterative learning process » et, d'autre part, qu'elle est effectuée « in order to help us better evaluate, predict, understand, control, and improve the software development process and product ». Nous avons donc besoin de l'expérimentation pour évaluer, prédire, comprendre, contrôler et améliorer le processus de développement du logiciel et le produit résultant.

Il est à noter que les objectifs de l'expérimentation sont les mêmes que ceux de la mesure (identifiés au début de cette section), regroupés dans la Figure 1.1. Ainsi, la mesure et l'expérimentation sont des processus complémentaires et le développement de bonnes techniques de mesure est un pré-requis pour la bonne expérimentation (Curtis, 1980).

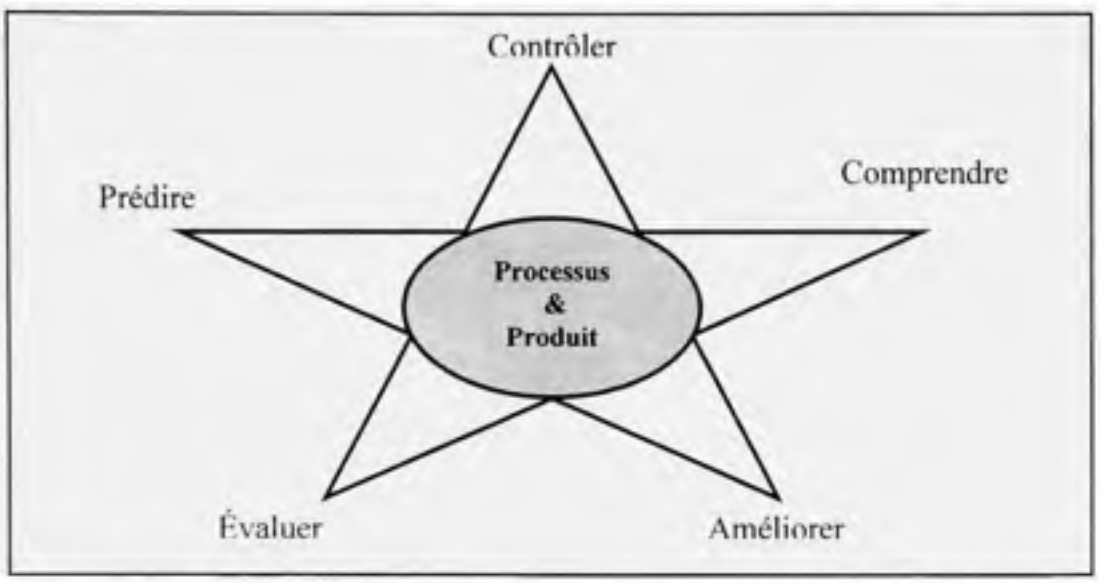

Figure 1.1 ***Les objectifs de mesurer et expérimenter en génie logiciel.***

En génie logiciel, les études expérimentales ne sont devenues plus nombreuses que vers les années 80. Vu l'importance qu'a prise ce type d'études au sein de la communauté des chercheurs et des praticiens en génie logiciel, Basili, Selby et Hutchens (1986, p. 733) ont élaboré un cadre d'expérimentation « to help structure the experimental process and to provide a classification scheme for understanding and evaluating experimental studies ». Ce cadre a été utilisé avec succès dans diverses expériences en génie logiciel dans le secteur industriel, notamment dans (Bourque, Maya et Abran, 1996; Côté *et al.*, 1996; Desharnais *et al.*, 1997) pour différents objectifs, à savoir : caractérisation des logiciels industriels de gestion, implantation de programmes de mesure de la maintenance de logiciel, mesure de taille pour les travaux de maintenance adaptive du produit, etc.

Tableau 1.1

Cadre d'expérimentation de Basili
(1986, p. 734 - Notre traduction)

I. Définition					
Motivation	Objet	But	Perspective	Domaine	Portée
Comprendre Estimer Gérer Améliorer Valider Apprendre Assurer	Produit Processus Modèle Mesure Théorie	Caractériser Évaluer Prédire Motiver	Développeur Gestionnaire Client Utilisateur Chercheur Agent de maintenance	Équipe du logiciel Projet logiciel	Un projet- une équipe Plus qu'un projet- une équipe Un projet- plus qu'une équipe Plus qu'un projet- plus qu'une équipe
II. Planification					
Conception		Critères		Mesures	
Conception expérimentale Méthodes d'analyse statistique		Critères directs Critères indirects		Définition et sélection de mesures Méthodologie de collecte des mesures	
III. Opération					
Préparation		Exécution		Analyses	
Étude pilote		Collection des données Validation des données		Analyse préliminaire des données Analyse formelle des données	
IV. Interprétation					
Contexte d'interprétation		Extrapolation		Impact	
Cadre statistique But de l'étude Domaine de recherche		Représentativité de l'échantillon		Visibilité Réplication Application	

Le Tableau 1.1 montre le résumé du cadre d'expérimentation de Basili. Ce cadre est composé de quatre principales phases : la définition, la planification, l'opération et l'interprétation. Chaque phase est constituée d'un ensemble d'étapes et chaque étape regroupe un ensemble d'activités utiles pour cette étape. Un tel cadre, selon Bourque,

Maya et Abran (1996), permet d'éviter les erreurs et les pièges lors de l'expérimentation, et encourage à bien structurer les expériences.

D'autre part, l'expérimentation dans le secteur industriel constitue un aspect important pour le développement de nouveaux processus de production de produits industriels. Ainsi, les méthodes de conception d'expériences sont importantes pour obtenir une grande robustesse aussi bien dans le processus que dans le résultat issu de son application. Citons à titre d'exemple de méthodes d'expérimentation, la méthode Taguchi de conception de plan d'expériences développée par le Dr. Genichi Taguchi afin d'améliorer la qualité des produits fabriqués. Il est à noter que Taguchi ne fournit pas une définition théorique et des critères de qualité, mais offre un outil d'expérimentation de la qualité du produit. Une description de cette méthode fait l'objet du chapitre 4 de cette thèse.

1.3.3 Mesures du logiciel et modèles de qualité

La compréhension du problème à résoudre représente la principale étape vers sa résolution. Schneidewind (2002) dans son corpus de connaissances pour la mesure de la qualité souligne, entre autres, qu'établir et valider des modèles de qualité représente l'une des fonctions importantes pour l'ingénieur afin de produire un produit logiciel de qualité. En outre, selon Basili (1996, p. 442), « in order to understand the effects of problem solving on the environment, we need to be able to model various product characteristics, such as reliability, portability, efficiency, as well as model various project characteristics such as cost and schedule ». Il est aussi souligné par Fenton et Pfleeger (1997, p. 338) que « because quality is really a composite of many characteristics, the notion of quality is usually captured in a model that depicts the composite characteristics and their relationships ». Ainsi, les modèles de qualité sont importants.

Depuis plusieurs années, des chercheurs ont proposé des modèles de qualité du logiciel avec des caractéristiques ou des attributs pour évaluer la qualité du logiciel, notamment McCall (McCall, Richards et Walters, 1977), Boehm (Boehm *et al.*, 1978) et Dromey (Dromey, 1996) .

Le modèle de qualité présenté par Jim McCall en 1977 est le premier modèle qui a exprimé la qualité en termes de facteurs et de critères. Ce modèle adopte une structure hiérarchique de trois niveaux. Le premier niveau consiste en trois grands axes pour définir la qualité du produit logiciel, à savoir : la révision du produit, le fonctionnement du produit et la transition du produit. Chaque axe est décomposé en un ensemble de facteurs de qualité pour encadrer l'axe soit 11 facteurs en deuxième niveau. Chaque facteur de qualité est décomposé ensuite en un ensemble de critères de qualité soit 23 critères distincts en troisième niveau. Le Tableau II.1 de l'Annexe II présente les trois axes, leurs facteurs de qualité et les critères de qualité correspondants.

Dans le modèle de Boehm, la qualité est exprimée au niveau supérieur par trois caractéristiques : la portabilité, l'utilité et la maintenabilité. Le niveau intermédiaire est composé de sept caractéristiques, lesquelles sont réparties dans le dernier niveau en 15 caractéristiques primitives distinctes (Boehm, Brown et Lipow, 1976). La caractéristique primitive constitue la base à partir de laquelle des mesures peuvent être définies afin de pouvoir la mesurer. Le Tableau II.2 de l'Annexe II montre le contenu du modèle de qualité de Boehm avec ces trois niveaux de caractéristiques.

Le modèle de qualité de Dromey est un modèle basé produit, c'est-à-dire l'évaluation de la qualité diffère d'un produit à l'autre. D'où la nécessité d'un modèle assez général afin qu'il soit applicable à tous les types de systèmes (Dromey, 1995). Ce modèle s'articule autour de quatre grandes propriétés du produit logiciel, à savoir : exactitude, interne, contextuelle et descriptive. Ces propriétés sont réparties ensuite en sept distincts attributs

de qualité. Le Tableau II.3 de l'Annexe II montre le contenu du modèle de qualité de Dromey.

Il est important de noter que chacun de ces modèles individuels est proposé par son auteur et reflète son point de vue sur la qualité : c'est le fruit du travail d'une seule personne. De plus, il est à remarquer que ces modèles utilisent des facteurs de qualité communs, en plus d'autres facteurs spécifiques aux modèles, dans des architectures différentes avec des définitions différentes et des liens différents. Par ailleurs, avoir un modèle unique serait intéressant pour l'ensemble de l'industrie du logiciel. D'une part, un modèle unique permettrait de former une base de connaissances de pratiques, des caractéristiques et des sous-caractéristiques de qualité avec leurs mesures correspondantes. D'autre part, un modèle, obtenu par consensus, servirait à normaliser les bases de connaissances afin d'assurer à la communauté des chercheurs et des praticiens du logiciel des définitions claires, des pratiques compréhensibles et une utilisation adéquate et non confuse du modèle. Une telle préoccupation a donné naissance à la norme internationale ISO 9126-1 pour un modèle de qualité du produit logiciel.

Le choix d'un modèle de qualité de produit logiciel à utiliser représente un défi. Pour ce choix, nous nous sommes basés sur les travaux de recherche de (Boehm *et al.*, 1978; Côté, 2005; Dromey, 1996; Fenton et Pfleeger, 1997; McCall, Richards et Walters, 1977) afin de dresser un tableau comparatif de ces quatre modèles de qualité selon les cinq critères suivants :

- style de construction du modèle (Comment ?) : indique l'approche utilisée dans la construction du modèle de qualité, à savoir : du haut vers le bas (« top to bottom ») et du bas vers le haut (« bottom to top ») de l'IEEE 1061(1998);
- caractéristiques et mesures (Quoi ?) : indique l'absence ou la présence aussi bien des caractéristiques de qualité que des propositions de mesures de qualité de logiciel;

- perspectives (Qui ?) : indique la ou les perspectives prises en considération par le modèle, à savoir : transcendantale, utilisateur, manufacturier, produit et valeur de David Garvin (Kitchenham et Pfleeger, 1996);
- application du modèle (Quand ?) : indique les étapes du cycle de vie du logiciel où le modèle de qualité est applicable;
- type du modèle (Par qui ?) : indique si le modèle est fait par une personne ou obtenu par consensus.

Tableau 1.2

Tableau comparatif des quatre modèles de qualité du produit logiciel

Modèles de qualité / Critères	Modèle de qualité de McCall	Modèle de qualité de Boehm	Modèle de qualité de Dromey	Modèle de qualité d'ISO 9126-1
Style de construction du modèle	Du bas vers le haut	Du bas vers le haut	Du bas vers le haut	Du bas vers le haut Du haut vers le bas
Caractéristiques et mesures de qualité	-Présence de caractéristiques de qualité -Absence de propositions de mesures	-Présence de caractéristiques de qualité -Présence de propositions de mesures	-Présence de caractéristiques de qualité -Absence de propositions de mesures	-Présence de caractéristiques de qualité -Présence de propositions de mesures
Perspectives	Utilisateur	Utilisateur	Produit	Transcendantale Utilisateur Manufacturier Valeur Produit
Application du modèle	Produit final (code exécutable)	Produit final (code exécutable)	Produit final (Code Source)	Toutes les phases du cycle de vie du logiciel
Type du modèle	-Élaboré par une personne -Porte le nom de son concepteur	-Élaboré par une personne -Porte le nom de son concepteur	-Élaboré par une personne -Porte le nom de son concepteur	-Modèle obtenu par consensus -Reconnu par les autorités nationales et internationales

Du Tableau 1.2, il est à remarquer que le modèle de qualité d'ISO 9126, à la différence des autres modèles, est le seul modèle qui a été développé par consensus : c'est-à-dire, il représente le consensus d'un ensemble d'experts des normes dans le domaine de la qualité du produit logiciel, et ce, à l'échelle internationale. En outre, le modèle de qualité d'ISO 9126 tient compte des différentes perspectives de la qualité du produit logiciel, supporte les deux approches du standard IEEE 1061-1998, prend en considération la qualité dès le début du cycle de vie du logiciel, s'intéresse à différents types de qualité et ne se limite pas à la qualité du produit final. Plus encore, ce modèle d'ISO 9126 est présenté sous forme d'un modèle hiérarchique débutant par les facteurs importants de haut niveau de qualité exprimés en caractéristiques, décomposées en deuxième niveau en sous-caractéristiques pour lesquelles des mesures sont proposées pour pouvoir les évaluer. C'est pourquoi nous avons choisi d'utiliser ce modèle de qualité, présenté en détail dans le chapitre 2, dans cette thèse.

CHAPITRE 2

STANDARD ISO 9126

2.1 Introduction

> « Quality is the totality of characteristics of an entity [software product] that bear on its ability to satisfy stated and implied needs. » (ISO, 2001a, p. 20)

Tel que souligné dans le chapitre 1, ISO 9126 est né de la nécessité d'avoir un consensus pour un modèle de qualité du produit logiciel. La première version d'ISO 9126 date de l'année 1991 et porte sur l'évaluation du produit logiciel : cette première version définit uniquement les caractéristiques et sous-caractéristiques de qualité (ISO, 1991). La deuxième version d'ISO 9126 a été publiée entre 2001 et 2004, sous le titre général : Technologies de l'information - qualité du produit logiciel. Actuellement, cette deuxième version est composée de quatre parties :

- ISO IS 9126-1 : modèle de qualité du produit logiciel (2001a);
- ISO TR 9126-2 : mesures de qualité externe (2003a);
- ISO TR 9126-3 : mesures de qualité interne (2003b);
- ISO TR 9126-4 : mesures de qualité en utilisation (2004).

Dans les sections suivantes nous présentons une vue d'ensemble aussi bien des modèles de qualité que des mesures de qualité d'ISO 9126. Par la suite, nous résumons l'état de l'art des travaux de recherche ayant utilisé ISO 9126. Enfin, nous présentons un ensemble de bénéfices et d'insuffisances que présente ISO 9126 dans sa version 2001-2004.

2.2 Modèles de qualité d'ISO 9126-1 : Vue d'ensemble

La norme internationale ISO 9126-1 correspond à un consensus international sur le thème défini, modèle de la qualité d'un produit logiciel, et a été approuvée par un organisme reconnu de normalisation : ISO. Le modèle de qualité de cette norme est décrit comme étant un cadre qui explique la relation entre différentes approches de la qualité (ISO, 2001a) et distingue trois grands types de qualité : interne, externe et en utilisation :

- La qualité interne est définie comme étant « la totalité des caractéristiques du produit logiciel d'un point de vue interne » (ISO, 2001c, p. 6), laquelle peut être évaluée en mesurant les propriétés internes du produit logiciel en dehors de toute exécution.
- La qualité externe est définie comme étant « la totalité des caractéristiques du produit logiciel d'un point de vue externe » (ISO, 2001c, p. 6), laquelle peut être évaluée durant l'exécution du produit logiciel en mesurant ses propriétés externes.
- La qualité en utilisation est définie comme étant « la vue de l'utilisateur de la qualité du produit logiciel lorsqu'il est utilisé dans un environnement et un contexte spécifiques » (ISO, 2001c, p. 6), laquelle correspond à l'utilisation du produit logiciel durant les phases d'opération et de maintenance, et elle n'est pas reliée à ses propriétés intrinsèques.

Ainsi, la qualité du produit logiciel peut être évaluée durant toutes les phases du cycle de vie du logiciel (Figure 2.1) en mesurant statiquement ses attributs internes, ou dynamiquement ses attributs externes, ou lors de l'utilisation dans un contexte et environnement réel.

L'ensemble des types de la qualité d'ISO 9126 (Figure 2.1) est basé sur les hypothèses que la qualité interne influence la qualité externe qui influence, à son tour, la qualité en utilisation. Par conséquent, l'achèvement de la qualité en utilisation dépendrait en quelque sorte de l'achèvement de la qualité externe qui, à son tour, dépendrait de

l'achèvement de la qualité interne du produit logiciel lui-même. Cependant, cette relation n'est qu'une proposition théorique et cette norme ISO ne fait référence à aucune expérimentation pour démontrer la validité de ces hypothèses. Le but de cette thèse est de contribuer à la vérification de ces hypothèses.

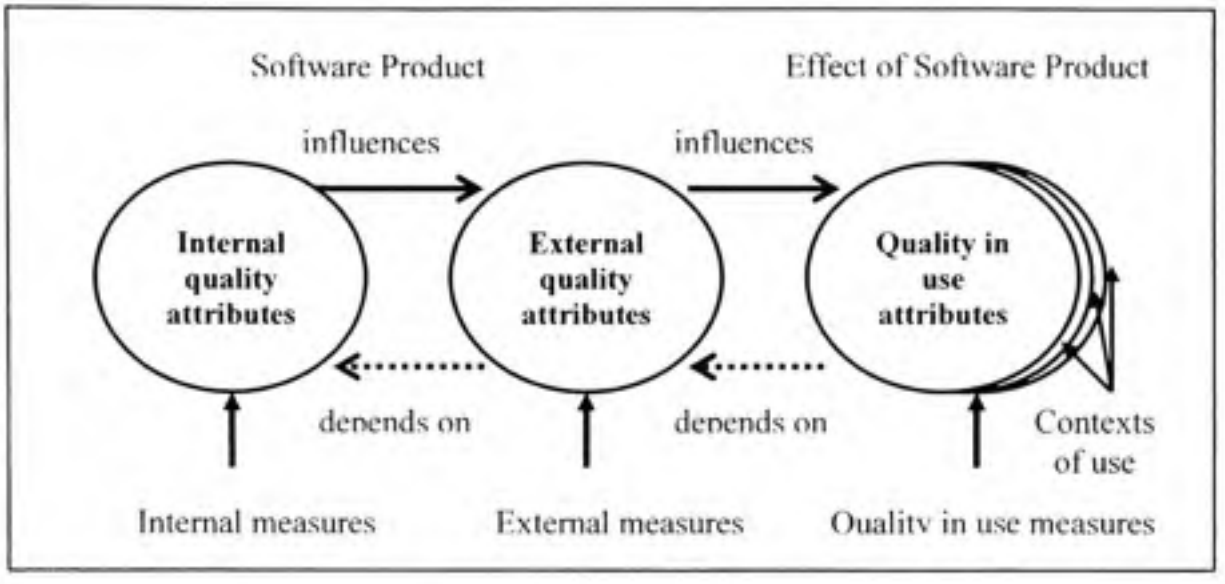

Figure 2.1 ***Qualité dans le cycle de vie.***
(ISO, 2001a)

En génie logiciel, cerner les spécifications et les objectifs du projet avant son développement diminue le risque de refaire le travail, de retarder les plannings et de dépasser le budget. Le même principe se présente en qualité. Évaluer la qualité interne et la qualité externe du logiciel avant de le livrer à l'utilisateur pour qu'il évalue sa qualité en utilisation permet de rectifier les erreurs, de porter les changements nécessaires et de diminuer le risque de s'embarquer dans des dépenses coûteuses ou imprévisibles. L'objectif est d'assurer la qualité du logiciel durant toutes ses phases depuis sa spécification jusqu'à sa livraison à l'utilisateur final. De cette manière, les différents types de qualité suivent le cheminement du projet.

En outre, le modèle de qualité d'ISO 9126 -1 comprend deux modèles de qualité : un premier modèle pour la qualité interne et externe, et un deuxième modèle pour la qualité en utilisation. Ces deux modèles sont décrits en détail dans les deux sections suivantes.

2.2.1 Modèle de qualité interne et externe

La qualité interne et la qualité externe partagent un modèle avec la même structure hiérarchique avec deux niveaux; c'est pourquoi la qualité interne et la qualité externe sont représentées par un seul modèle de qualité (Figure 2.2). Le premier niveau est composé de six caractéristiques, lesquelles sont subdivisées par la suite en 27 sous-caractéristiques.

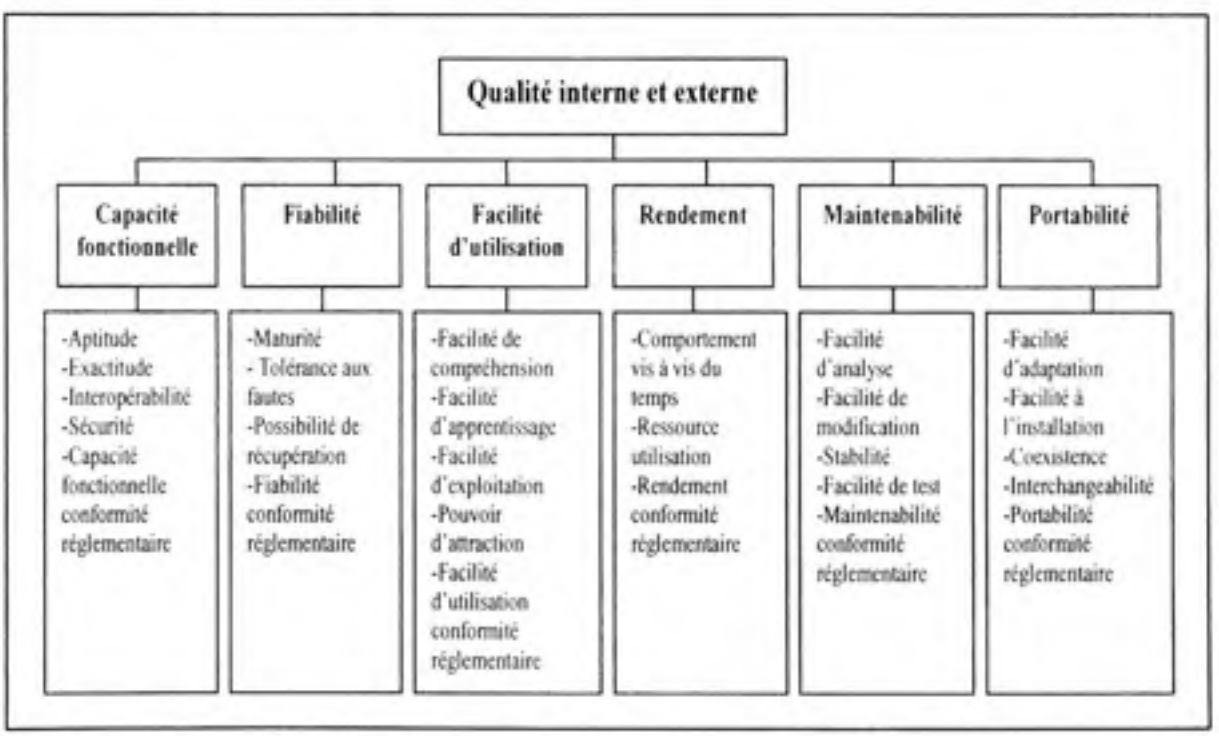

Figure 2.2 ***Modèle de qualité interne et externe d'ISO 9126.***
(ISO, 2001c)

Aussi bien pour la qualité interne que pour la qualité externe, ces caractéristiques et sous-caractéristiques sont définies par ISO 9126-1 de la même manière. Nous présentons ci-après ces définitions comme décrites dans (ISO, 2001c, p. 8-12).

La capacité fonctionnelle correspond à la « capacité du produit logiciel à fournir des fonctions qui répondent à des besoins exprimés et implicites lorsque le logiciel est utilisé dans des conditions spécifiées ». Cette caractéristique est composée de cinq sous-caractéristiques, savoir :

- Aptitude : « Capacité du produit logiciel à fournir un ensemble adéquat de fonctions pour les tâches et les objectifs spécifiés de l'utilisateur ».
- Exactitude : « Capacité du produit logiciel à fournir des résultats ou des effets justes ou convenus avec le degré de précision nécessaire ».
- Interopérabilité : « Capacité du produit logiciel à interagir avec un ou plusieurs systèmes spécifiés ».
- Sécurité : « Capacité du produit logiciel à protéger les informations et les données de sorte que des personnes ou systèmes non autorisés ne puissent pas les lire ni les modifier et que les personnes ou systèmes autorisés y aient accès ».
- Conformité relative à l'aptitude fonctionnelle : « Capacité du produit logiciel à respecter l'application de normes, de conventions, de réglementations de droit ou de prescriptions similaires relatives à l'aptitude fonctionnelle ».

La fiabilité correspond à la « capacité du produit logiciel à maintenir un niveau de service spécifié lorsqu'il est utilisé dans des conditions précises » et elle est composée de quatre sous-caractéristiques présentées comme suit :

- Maturité : « Capacité du produit logiciel à éviter des défaillances dues à des défauts du logiciel ».
- Tolérance aux fautes : « Capacité du produit logiciel à maintenir un niveau de service donné en cas de défaut du logiciel ou de violation de son interface spécifiée ».

- Possibilité de récupération : « Capacité du produit logiciel à rétablir un niveau de service donné et de restaurer les données directement touchées en cas de défaillance ».
- Conformité relative à la fiabilité : « Capacité du produit logiciel à respecter l'application de normes, de conventions ou de réglementations relatives à la fiabilité ».

La facilité d'utilisation correspond à la « capacité du produit logiciel à être compris, connu, utilisé et à plaire à l'utilisateur, dans des conditions spécifiées d'utilisation ». Cette caractéristique est subdivisée en cinq sous-caractéristiques :

- Facilité de compréhension : « Capacité du produit logiciel à permettre à l'utilisateur de savoir si le logiciel est approprié et comment il peut être utilisé pour remplir des tâches particulières dans des conditions d'utilisation données ».
- Facilité d'apprentissage : « Capacité du produit logiciel à permettre à l'utilisateur d'apprendre son application ».
- Facilité d'exploitation : « Capacité du produit logiciel à permettre à l'utilisateur de l'exploiter et de contrôler son exploitation ».
- Pouvoir d'attraction : « Capacité du produit logiciel à attirer l'utilisateur ».
- Conformité relative à la facilité d'utilisation : « Capacité du produit logiciel à respecter l'application de normes, de conventions, de guides de style et de réglementations relatifs à la facilité d'utilisation ».

Le rendement correspond à la « capacité du produit logiciel à fournir des performances appropriées en fonction de la qualité de ressources utilisées, dans des conditions données ». Cette caractéristique comprend trois sous-caractéristiques décrites comme suit :

- Comportement vis-à-vis du temps : « Capacité du produit logiciel à fournir des temps de réponse, de traitement et des débits appropriés lors de l'exécution de sa fonction, dans des conditions données ».

- Utilisation des ressources : « Capacité du produit logiciel à utiliser des quantités et des types de ressources appropriés lorsque le logiciel exécute sa fonction, dans des conditions données ».
- Conformité relative au rendement : « Capacité du produit logiciel à respecter l'application de normes ou de conventions relatives au rendement ».

La maintenabilité est définie par « la capacité du produit logiciel à être modifié. Les modifications peuvent inclure des corrections, des améliorations ou l'adaptation du logiciel aux changements d'environnement, d'exigences et de spécifications fonctionnelles ». Cette caractéristique est composée de cinq sous-caractéristiques, à savoir :

- Facilité d'analyse : « Capacité du produit logiciel à faire l'objet d'un diagnostic des déficiences ou des causes de pannes du logiciel, ou des pièces à modifier ».
- Facilité de modification : « Capacité du produit logiciel à permettre la mise en œuvre d'une modification spécifiée ».
- Stabilité : « Capacité du produit logiciel à éviter les effets inattendus des modifications du logiciel ».
- Facilité de test : « Capacité du produit logiciel à permettre la validation des modifications du logiciel ».
- Conformité relative à la maintenabilité : « Capacité du produit logiciel à respecter l'application de normes ou de conventions relatives à la maintenabilité ».

La portabilité correspond à la « capacité du produit logiciel à être transféré d'un environnement à un autre », décomposée en cinq sous-caractéristiques :

- Facilité d'adaptation : « Capacité du produit logiciel à s'adapter à différents environnements spécifiés sans avoir recours à d'autres actions ou moyens que ceux prévus à cet effet pour le logiciel considéré ».
- Facilité d'installation : « Capacité du produit logiciel à être installé dans un environnement donné ».

- Coexistence : « Capacité du produit logiciel à coexister avec d'autres logiciels indépendants dans un environnement commun et en partageant des ressources ».
- Interchangeabilité : « Capacité du produit logiciel à être utilisé à la place d'un autre produit logiciel donné dans le même but et dans le même environnement ».
- Conformité relative à la portabilité : « Capacité du produit logiciel à respecter l'application de normes ou de conventions relatives à la portabilité ».

Dans l'ensemble, même si la qualité interne et la qualité externe permettent d'étudier la qualité du produit logiciel de différentes facettes, la qualité en utilisation est l'effet combiné de ces deux types de qualité pour l'utilisateur (ISO, 2001a).

2.2.2 Modèle de qualité en utilisation

Le modèle de qualité en utilisation de la norme ISO 9126-1 comprend un seul niveau et il est composé de quatre caractéristiques (Figure 2.3).

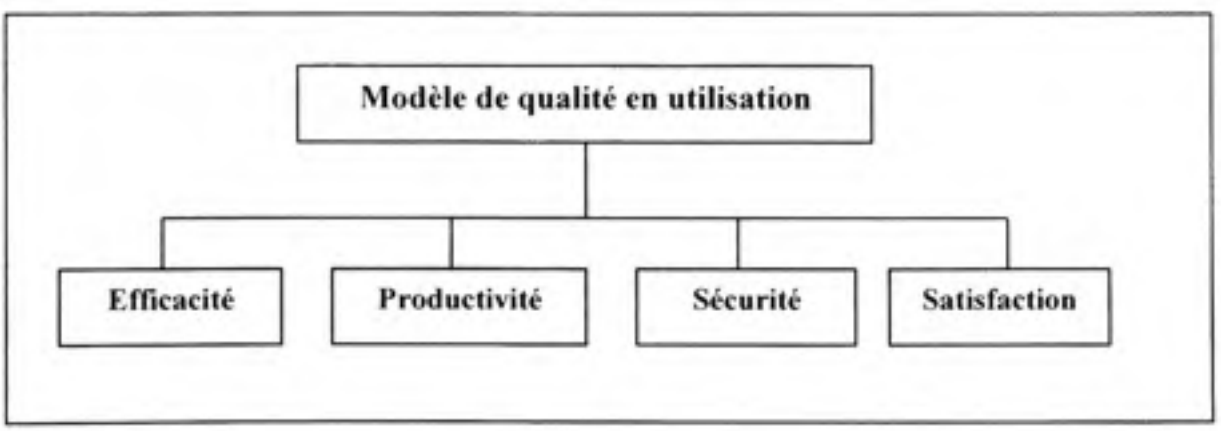

Figure 2.3 ***Modèle de qualité en utilisation d'ISO 9126.***
(ISO, 2001c)

Ces caractéristiques sont définies dans (ISO, 2001c, p. 13) comme suit :

- L'efficacité correspond à la « capacité du produit logiciel à permettre aux utilisateurs d'atteindre des objectifs spécifiés avec exactitude et exhaustivité dans un contexte d'utilisation spécifié ».
- La productivité correspond à la « capacité du produit logiciel à permettre aux utilisateurs d'employer des quantités appropriées de ressources en relation avec l'efficacité accomplie dans un contexte d'utilisation donné ».
- La sécurité correspond à la « capacité du produit logiciel à atteindre des niveaux acceptables de risque de danger pour les personnes, l'activité, le logiciel, la propriété ou l'environnement dans un contexte d'utilisation spécifié ».
- La satisfaction correspond à la « capacité du produit logiciel à satisfaire les utilisateurs dans un contexte d'utilisation spécifié ».

2.3 Mesures d'ISO TR 9126-2 à 4

2.3.1 Vue d'ensemble

Tel que déjà cité à la section 2.2 de ce chapitre, le modèle de qualité interne et de qualité externe est réparti en deux niveaux. Le troisième et dernier niveau de ce modèle hiérarchique correspond aux mesures. Les rapports techniques ISO TR 9126-2 et 3 associent une série de mesures pour chaque sous-caractéristique du modèle de qualité interne et externe. Le rapport technique ISO TR 9126-4 associe un ensemble de mesures à chaque caractéristique du modèle de qualité en utilisation. Ci-dessous les définitions fournies dans ISO 9126-1 pour chaque catégorie de mesures de qualité.

Les mesures de la qualité interne :

> « Internal metrics can be applied to a non executable software product (such as a specification or source code) during designing and coding. When developing a software product the intermediate products should be evaluated using internal metrics which measure intrinsic properties, including those which can be derived from simulated behaviour. » (ISO, 2001a, p. 15)

Les mesures de la qualité externe :

> « External metrics use measures of a software product derived from measures of the behaviour of the system of which it is a part, by testing, operating and observing the executable software or system. » (ISO, 2001a, p. 15)

Les mesures de la qualité en utilisation :

> « Quality in use metrics measure the extent to which a product meets the needs of specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in a specified context of use. This can be only achieved in a realistic system environment. » (ISO, 2001a, p. 15)

Un autre type de mesures interne est cité à titre informatif dans le rapport technique ISO TR 9126-3. Il s'agit des mesures internes pures :

> « Pure Internal metrics are used to measure certain attributes of the software design and code of the software product that will influence the same or all of the overall software characteristics and sub-characteristics. » (ISO, 2003b, p. 56)

En résumé, les mesures de la qualité interne sont appliquées au produit logiciel sans qu'il soit exécuté et les mesures de qualité externe peuvent être utilisées pour mesurer la qualité du produit logiciel lors de son exécution. La qualité en utilisation est mesurée en termes de résultat de l'utilisation du logiciel et non pas de ses propres propriétés.

La norme ISO 9126-1 (2001a, p. 15) recommande que « the internal metrics are used which have as strong a relation as possible with the target external metrics, so that they can be used to predict the values of external metrics ». Cependant, cette norme n'identifie pas les mesures de la qualité interne qui sont en relation avec celles de la qualité externe, puisque « it is generally difficult to design a rigorous theoretical model

which provides a strong relationship between internal and external metrics » (ISO, 2001a, p. 15).

2.3.2 Propriétés des mesures d'ISO TR 9126-2 à 4

Les rapports techniques ISO TR 9126-2 à 4 relatifs aux mesures de qualité de produit logiciel proposent une série de 211 mesures réparties comme suit : les mesures de qualité interne sont au nombre de 70, les mesures de qualité externe sont au nombre de 111 et les mesures de qualité en utilisation sont au nombre de 15. Un autre type de mesures internes, soit les « mesures internes pures » au nombre de 15, sont citées à titre informatif, c'est-à-dire à titre d'indication seulement en annexe d'ISO TR 9126-3. Ces rapports techniques ne prétendent pas offrir un inventaire exhaustif de toutes les mesures, mais uniquement les mesures qui ont été approuvées par ISO et ayant fait l'objet d'un consensus minimum international.

Ces mesures sont toutes décrites selon une structure de documentation identique, en fonction de dix termes, dans la clause 7 des rapports techniques ISO TR 9126-2 à 4 (2003a; 2003b; 2004). Des considérations à prendre en compte lors de l'utilisation de ces mesures sont présentées dans ces rapports techniques ainsi que des exemples de la façon d'appliquer ces mesures durant le cycle de vie du logiciel.

2.4 État de l'art - ISO 9126

Nous avons relevé auparavant que certaines normes ne s'appuient pas sur des expérimentations rigoureuses qui en démontrent la validité des hypothèses sur lesquelles ces normes s'appuient. De plus, Jabir et Moore (1998, p. 155) soulignent la problématique suivante :

> « It is widely posited that practice standards should be based upon observation, recording and consensual validation of implemented 'best practices'. This strategy has resulted, though, in the development of a corpus of standards that are sometimes alleged to be isolated, unconnected and dis-integrated, because each standard performs a local optimization of a single observed practice. »

Une telle problématique vient du fait que les normes en génie logiciel sont issues des travaux d'un groupe d'experts partageant un thème bien défini, mais parfois sans collaboration entre les différents groupes traitant les différents aspects en génie logiciel. Par conséquent, il peut en résulter, parfois, des duplications, des inconsistances, etc.

D'un autre côté, l'apport que peut fournir les autres normes pour une norme bien définie est invisible ou ignoré. En effet, même si une norme peut représenter une bonne pratique en elle-même, sa liaison avec les autres normes constituerait une valeur ajoutée à prendre en considération.

Étant conscients de ce problème, des experts d'ISO procèdent actuellement à la révision et la modification d'ISO 9126 afin de répondre aux changements que connaît le domaine de technologies de l'information. La nouvelle génération des standards de la qualité du produit logiciel, désigné par ISO 25000-SQuaRE (Software Product Quality Requirements and Evaluation) va combiner les deux séries de documents ISO 9126 et ISO 14598 (Suryn, Abran et April, 2003). Plus encore, l'avantage d'utiliser certaines normes ensemble vu leur complémentarité est bien illustré par le travail effectué par Suryn *et al.* (2002) dont l'objectif porte sur comment utiliser TL9000 (Quality Management System Requirements and Measurements Handbooks), ISO 9126 (Software Product Quality) et ISO 14598 (Software Product Evaluation) ensemble pour définir, mesurer, évaluer et finalement réaliser la qualité appropriée du point de vue utilisateur du produit logiciel.

En outre, le modèle de qualité d'ISO 9126 a gagné de l'importance auprès des chercheurs et des praticiens qui l'ont utilisé dans leurs travaux de recherche combiné à d'autres outils, à savoir :

- Bhatti (2005) a pour objectif d'aider à établir des mesures de qualité de logiciel en se basant sur ISO 9126 pour les diagrammes UML (Unified Modeling Language), en particulier les mesures externes de la caractéristique « Capacité fonctionnelle » du modèle de qualité ISO 9126-1.
- Koscianski et Costa (1999) mettent l'emphase sur le problème d'extraction des informations à propos de la qualité à partir des résultats des mesures du produit logiciel. L'objectif est de combiner AHP (Analytical Hierarchy Process) et ISO 9126-1 pour construire un cadre d'évaluation de la qualité, puisqu'ils s'appuient tous les deux sur une décomposition hiérarchique.
- Abran *et al.* (2003b) ont développé un prototype qui fournit un outil pour l'utilisation des modèles de qualité d'ISO 9126 avec la représentation multidimensionnelle de la mesure de la qualité du logiciel QEST (Quality factor + Economic, Social & Technical dimensions).
- Losavio *et al.* (2004) ont utilisé Unified Process (UP) qui utilise la notation d'UML et le modèle de qualité d'ISO 9126-1. Leur approche est basée sur la personnalisation du modèle de qualité du produit logiciel d'ISO 9126-1 à l'architecture du logiciel.

Par ailleurs, les rapports techniques ISO TR 9126-2 à 4 n'ont pas encore atteint le statut de norme internationale mais ont été mis à la disposition de la communauté des chercheurs et des praticiens du logiciel pour les enrichir et les faire progresser. Depuis, diverses analyses ont été effectuées sur cette série de documents ISO TR 9126-2 à 4.

Du point de vue utilité et applicabilité des mesures ISO 9126, Gil et Suryn (2005) ont analysé les mesures de qualité interne d'ISO TR 9126-3 et leur utilité. Des propositions d'améliorations ont été présentées aussi bien pour les mesures que pour le modèle de qualité interne afin d'améliorer l'applicabilité du standard actuel. Une autre analyse a été effectuée par Côté (2005) pour mesurer l'applicabilité des mesures de qualité en utilisation d'ISO TR 9126-4 et des propositions d'améliorations ont été aussi présentées. Pour faciliter la conception des mesures dérivées, Abran *et al.* (2005) ont présenté en annexe de leur article l'ensemble des mesures de base identifiées pour les rapports techniques ISO TR 9126-2 à 4 ainsi que leurs unités de mesure correspondantes.

Selon la perspective de métrologie, Sellami (2005) a analysé aussi bien la structure de documentation des mesures des rapports ISO TR 9126-2 à 4 que quelques mesures par rapport à son cadre de vérification des mesures de logiciels, lequel cadre est basé sur les concepts de métrologie. Suite à cette analyse, un ensemble de forces et de faiblesses ont été présentées. Une autre analyse a été effectuée par Abran, Al-Qutaish et Cuadrado-Gallego (2006) sur les mesures proposées dans le rapport technique ISO TR 9126-4 afin de déterminer jusqu'à quel point ces mesures sont conformes aux critères de métrologie typiques de la mesure classique. Des propositions d'amélioration dans le design et la documentation des mesures proposées ont été identifiées.

Toujours avec cette série des documents ISO 9126, Al-Kilidar, Cox et Kitchenham (2005) ont étudié l'utilisation et l'utilité de ce standard, en particulier son aptitude à mesurer la qualité du design des produits logiciels. Par ailleurs, pour enlever l'ambigüité que pose la caractéristique de qualité « Facilité d'utilisation », Abran *et al.* (2003a) ont mené une analyse de cette caractéristique dans les standards ISO 9126 et ISO 9241 qui s'est achevée par la proposition d'un modèle de qualité amélioré de cette caractéristique. Dans ce cadre, Cheikhi, Abran et Suryn (2006) ont effectué une analyse des mesures de cette caractéristique dans ISO TR 9126-3 et son utilité pour évaluer la qualité en

utilisation du produit logiciel. Suite à cette analyse, des améliorations du modèle de qualité en utilisation ont été proposées.

En résumé, ces travaux de recherche montrent d'une part l'importance et l'intérêt de cette série des documents ISO 9126 et, d'autre part, diverses insuffisances que présentent ISO TR 9126-2 à 4. Ces bénéfices et insuffisances sont présentés dans la section suivante.

2.5 Bénéfices et Insuffisances - ISO 9126

De l'exploration des quatre parties d'ISO 9126 effectuée dans les sections 2.2 et 2.3, et de l'état de l'art présenté dans la section 2.4 de ce chapitre, nous avons relevé un ensemble de bénéfices et d'insuffisances présentés au fil de cette section.

Selon la norme ISO 9126-1, le modèle de qualité d'ISO 9126 (2001a) présente les bénéfices suivants :

- peut être appliqué dans toutes les phases du cycle de vie du logiciel;
- permet la spécification et l'évaluation de la qualité du produit logiciel des différentes perspectives et par les différentes personnes participant au projet logiciel, à savoir : les utilisateurs, les développeurs, les personnes chargées de la maintenance, les réalisateurs, les acquéreurs, les évaluateurs et les responsables de qualité;
- sert, entre autres, à identifier les exigences, les objectifs de conception et les objectifs de test du logiciel;
- peut être utilisé à tous les types de logiciels;
- offre un ensemble de caractéristiques et sous-caractéristiques qui peuvent être combinées afin de spécifier les exigences de qualité du logiciel.

Les propriétés des rapports techniques ISO TR 9126-2 à 4 se résument comme suit (ISO, 2001a) :

- les mesures internes s'appliquent au produit logiciel intermédiaire et portent sur l'aspect statique du produit logiciel. Ces mesures permettent d'évaluer la qualité du produit logiciel dès le début du cycle de développement du logiciel. Théoriquement, ces mesures doivent assurer la réalisation de la qualité externe puisqu'elles devraient être utilisées comme des indicateurs des mesures externes;
- les mesures externes s'appliquent au produit logiciel final et portent sur l'aspect dynamique du logiciel. Ces mesures permettent d'évaluer la qualité du produit logiciel durant les phases de test et d'opération du cycle de développement du logiciel, et s'intéressent à ce que fait le logiciel et non pas à comment il le fait ou comment il est fait;
- les mesures de qualité en utilisation reflètent la qualité du point de vue utilisateur du système contenant le logiciel. Ces mesures prennent en considération l'environnement et le contexte du logiciel et mesurent jusqu'à quel point les objectifs des utilisateurs sont réellement atteints.

Voici maintenant quelques observations sur ces rapports techniques relatifs aux mesures de qualité du produit logiciel. En effet, ces rapports sont récents (ISO TR 9126-2 et 3 datent de l'année 2003 et ISO TR 9126-4 date de l'année 2004) et proposent des listes non exhaustives de mesures, listes qui peuvent être enrichies et applicables pour n'importe quel type de produit logiciel. Toutefois, il est à noter que ces rapports techniques d'ISO 9126 ne contiennent pas des mesures dédiées à l'orienté objet même si les mesures de cette technologie de développement sont nombreuses et datent des années 90 et plus. Par conséquent, les mesures proposées par les rapports manquent d'actualisation et risquent d'être difficilement applicables aux nouvelles technologies.

En outre, les mesures des rapports techniques ISO TR 9126-2 à 4 ne sont pas bien détaillées, en particulier la méthode de collecte des données relatives au calcul de la mesure. Ces mesures sont citées à titre informatif (i.e. l'utilisation de ces mesures n'est pas obligatoire), ont été développées par des discussions des experts et n'ont pas fait

l'objet de vérifications ou d'expérimentations (absence de travaux d'expérimentation utilisant le modèle avec ses caractéristiques et mesures). Par conséquent, des travaux d'expérimentation devraient être faits afin de raffiner ces mesures et de pouvoir les utiliser de façon adéquate de telle sorte qu'elles reflètent adéquatement les caractéristiques mesurées.

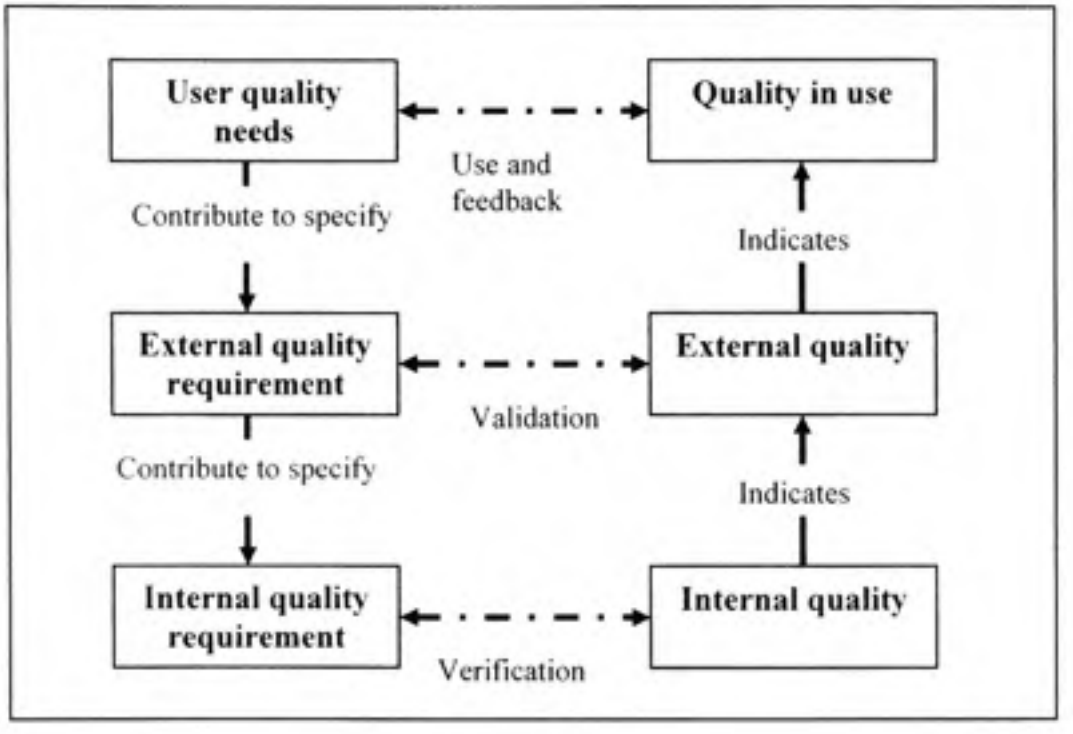

Figure 2.4 ***Qualité dans le cycle de vie du logiciel.***
(ISO, 2001a)

Plus encore, le modèle de qualité et les mesures associées ont été définis essentiellement pour une utilisation durant toutes les phases du cycle de vie du logiciel (Figure 2.4). Les hypothèses soutenant ces définitions ont été prises pour acquises par la norme, de même que pour les liens entre les différents types de qualité (Figure 2.1). De tels liens ne sont pas confirmés explicitement par la norme ISO 9126-1 ni par des travaux de recherche expérimentaux bien documentés.

D'où, la nécessité de l'expérimentation pour vérifier les relations entre ces différents types de qualité, en utilisant le modèle de qualité ISO 9126-1 et les mesures proposées dans les rapports techniques ISO TR 9126-2 à 4. Pour ce faire, un design d'expérimentation robuste s'avère important afin de bien mener les expériences. La méthode Taguchi, faisant l'objet du chapitre 4 est l'un des outils d'expérimentation utilisé avec succès dans le secteur industriel.

Par ailleurs, mener des expérimentations repose nécessairement sur la collecte de données. Tout comme le soulignent Fenton et Pfleeger (1997, p. 16), « Data collection is easier said than done, especially when data must be collected across a diverse set of projects ». Plus encore, selon Kan (2003, p. 118), « Gathering software engineering data can be expensive, especially if it is done as part of a research program ». En effet, non seulement la collecte des données est coûteuse, mais aussi beaucoup de temps est requis avant d'avoir les données nécessaires. En particulier pour les données de qualité du logiciel, Jones (1996, p. 356) souligne que : « Quality data must be measured, and it must be accumulated for quite some time before enough of it is available to be useful ». Pour cette raison, nous avons opté d'explorer, dans le chapitre 3 de cette thèse, les référentiels de données disponibles dans la littérature en génie logiciel.

CHAPITRE 3

RÉFÉRENTIELS DE DONNÉES EN GÉNIE LOGICIEL

3.1 Introduction

> « Collecting data from real world software engineering projects is problematic. Software projects are notoriously difficult to control and corporations are often reluctant to expose their own software development record to public scrutiny. » (Shirabad et Menzies, 2005)

En génie logiciel, l'importance des données pour les études empiriques est reconnue et la problématique de collecter ces données est soulignée par (Shirabad et Menzies, 2005). En fait, il est souhaitable de collecter les données pour chaque étude selon ses objectifs. Cependant, collecter les données pour un objectif bien défini n'est pas toujours possible pour diverses raisons, à savoir : la disponibilité de l'organisme institutionnel ou industriel pour collecter les données, la contrainte du temps, le coût de la collecte, etc.

Face à ces difficultés, incluant la durée et les coûts des expérimentations en génie logiciel, quelques référentiels de données de logiciel ont été mis sur pied ces dernières années afin de les rendre disponibles aux chercheurs. À cette étape-ci de cette thèse, il est maintenant opportun d'explorer ces référentiels de données disponibles aux chercheurs en génie logiciel afin d'en examiner la pertinence comme sources potentielles de données empiriques de recherche. Dans ce chapitre, nous avons choisi d'explorer deux référentiels de données, parmi les plus connus et référencés dans la littérature.

La première section présente le référentiel d'ingénierie du logiciel PROMISE, lequel contient plusieurs ensembles différents de données (datasets). La deuxième section présente la description du référentiel de données d'ISBSG ainsi que du questionnaire de collecte de données d'ISBSG. La troisième section présente une discussion de ces deux

référentiels de données dans le but de répondre à la question : Lequel des deux référentiels de données est le plus approprié pour ce travail de recherche ?

3.2 Référentiel de données PROMISE

Le référentiel PROMISE (PRedictOr Models In Software Engineering) a été initié par Sayyad Shirabad et Tim Menzies, en decembre 2004. Ce référentiel a été créé dans le but de : « encourage repeatable, verifiable, refutable, and/or improvable predictive models of software engineering » (Shirabad et Menzies, 2005).

Pour atteindre ce but, PROMISE met à la disposition du public plusieurs ensembles de données (datasets) pour la construction de modèles prédictifs de logiciel. Ces ensembles de données sont fournis par la communauté du génie logiciel pour servir la recherche et les chercheurs de ce domaine et ils sont accessibles, gratuitement, via le site web (http://promise.site.uottawa.ca/SERepository/datasets-page.html) de l'Université d'Ottawa - École d'ingénierie et de technologie de l'information.

3.2.1 PROMISE : Vue d'ensemble

En 2007, le référentiel PROMISE rassemble 20 ensembles de données (Tableau 3.1). D'une part, ces ensembles de données sont issus de différentes sources. En effet, certains proviennent de travaux de recherche publiés en génie logiciel : par exemple, l'ensemble de données « COCOMO81 » dont les données sont originaires du livre de Boehm (Software Engineering Economics). D'autres proviennent de projets « open source », comme le cas des ensembles de données fournis par la NASA (National Aeronautics and Space Administration), « Nickle », « XFree86 » et « Xorg ».

D'autre part, ces ensembles de données de PROMISE ont été utilisés pour différents objectifs, à savoir : prédiction de défauts de logiciel, estimation de l'effort de

développement du logiciel, traçabilité des exigences, réutilisation et bibliothèque de calcul numérique.

Tableau 3.1

Vue d'ensemble des ensembles de données de PROMISE

Nom de l'ensemble de données	Thème	Année	Source
CM1	Prédiction de défauts	2004	NASA
JM1	Prédiction de défauts	2004	NASA
KC1	Prédiction de défauts	2004	NASA
KC1- CL- DC	Prédiction de défauts	2005	NASA
KC1- CL	Prédiction de défauts	2005	NASA
KC1- CL– T5	Prédiction de défauts	2005	NASA
KC2	Prédiction de défauts	2004	NASA
PC1	Prédiction de défauts	2004	NASA
DATATRIEVE	Prédiction de défauts	2005	Digital Engineering Italy
COCOMO81	Estimation de l'effort	2004	(Boehm, 1981)
COCOMONasa	Estimation de l'effort	2004 2005	NASA
COCOMONasa2	Estimation de l'effort	2006	NASA
Desharnais	Estimation de l'effort	2005	Projets commerciaux canadiens
MODIS	Traçabilité des exigences	2005	NASA
CM1-TR	Traçabilité des exigences	2005	NASA
Reuse	Réutilisation	2004	(Morisio, Ezran et Tully, 2002)
Nickle	Non identifié	2005	Archives CVS de Nickle
XFree 86	Non identifié	2005	Archives CVS de XFree86
Xorg	Non identifié	2005	Archives CVS de Xorg
Qos	Bibliothèque de calcul numérique	2006	(Zhou, Cooper et Yen, 2006)

Par ailleurs, les ensembles de données sont fournis dans le format recommandé par les auteurs de PROMISE; il s'agit d'un fichier de format ARFF (Attribute-Relation File Format). Les données du fichier sont accompagnées de commentaires dans le même fichier afin d'en bien comprendre le contenu. La Figure 3.1 présente un résumé de la structure de ce fichier. Les directives pour remplir ce fichier sont fournies en ligne (PROMISE, 2005).

Figure 3.1 ***Exemple de Fichier ARRF.***

(PROMISE, 2005)

Le fichier de type ARFF est utilisé par le logiciel Weka (Waikato environment for knowledge analysis). C'est un logiciel « open source » offert par l'Université de Waikato et il est utilisé essentiellement dans la fouille de données (Ian et Frank, 2005). Dans le cas où un autre format est utilisé, tel le cas des bases de données « MODIS » et « CM1-TR », les commentaires de description de l'ensemble de données sont fournis dans un fichier séparé du fichier de données.

3.2.2 PROMISE : Vue détaillée

Le Tableau 3.2 présente plus de détails sur les 20 ensembles de données du référentiel PROMISE. La deuxième colonne présente le nombre d'instances dans chaque ensemble de données (i.e., la taille de l'ensemble de données). La troisième colonne présente le nombre d'attributs et la quatrième colonne présente le type de ces attributs; il s'agit des variables indépendantes et de la variable dépendante.

Les ensembles de données « CM1, JM1, KC1, KC2 et PC1 » ont tous le même nombre d'attributs, mais avec des tailles variables. Les attributs sont essentiellement des attributs de code source du logiciel (à savoir : le nombre de lignes de code, les mesures de McCabe et de Halstead) afin de prédire si le code source est défectueux ou non.

Les ensembles de données « KC1-CL-DC, KC1-CL et KC-CL-T5 » ont tous le même nombre d'attributs et la même taille. Les attributs concernent des attributs de code source du logiciel développé en orienté objet; il s'agit des attributs au niveau de la « classe » afin de prédire si cette dernière est défectueuse ou non.

Les ensembles de données « COCOMO81, COCOMONasa et COCOMONasa2 » varient aussi bien dans le nombre d'attributs que dans la taille. Les attributs mesurés concernent COCOMO (COnstructive COst MOdel), les multiplicateurs de l'effort et la taille en lignes de code afin de prédire l'effort de développement du logiciel.

Tableau 3.2

Vue détaillée des ensembles de données du référentiel PROMISE

Nom de l'ensemble de données	Nombre Instances	Nombre Attributs	Type des attributs	
			Indépendant	Dépendant
CM1	498	22	-Mesures de lignes de code -Mesures de McCabe -Mesures d'Halstead	Défauts
JM1	10885	22	-Mesures de lignes de code -Mesures de McCabe -Mesures d'Halstead	Défauts
KC1	2109	22	-Mesures de lignes de code -Mesures de McCabe -Mesures d'Halstead	Défauts
KC1- CL- DC	145	95	-Mesure de niveau classe	Défauts
KC1- CL	145	95	-Mesure de niveau classe	Défauts
KC1- CL– T5	145	95	-Mesure de niveau classe	Défauts
KC2	522	22	-Mesures de lignes de code -Mesures de McCabe -Mesures d'Halstead	Défauts
PC1	1109	22	-Mesures de lignes de code -Mesures de McCabe -Mesures d'Halstead	Défauts
DATATRIEVE	130	9	-Mesures de lignes de code	Fautes
COCOMO81	63	17	-Multiplicateurs d'effort -Mesures de lignes de code	Effort
COCOMONasa	60	17	-Multiplicateurs d'effort -Mesures de lignes de code	Effort
COCOMONasa2	93	24	-Mesures COCOMO-I -Attributs de projet -Mesures de lignes de code	Effort
Desharnais	81	12	-Attributs de projet -Points de fonction	Non identifié
Reuse	24	28	-Attributs de haut niveau -Attributs de bas niveau	Réutilisation
Nickle	2972	10	-Attributs de fichier	Non identifié
XFree 86	175658	10	-Attributs de fichier	Non identifié
Xorg	136435	10	-Attributs de fichier	Non identifié
Qos	272	6	-Attributs de composants	Temps, Espace, Qualité

Les ensembles de données « Nickle, XFree86 et Xorg » ont tous le même nombre d'attributs, diffèrent dans la taille et offrent des données générées à partir des archives CVS (Concurrent Versions System) de projets « open source » Nickel, Xorg et XFree86. Il est à noter que ces ensembles de données n'ont pas été utilisés auparavant et ils doivent être revus afin d'être utilisables dans des modèles de prédiction en génie logiciel.

« Desharnais et Reuse » sont les seuls ensembles de données qui offrent des attributs en relation avec le projet logiciel. En effet, « Desharnais » fournit quelques attributs relatifs à l'effort, l'expérience de l'équipe du projet, l'expérience de la personne chargée du projet, etc. en plus de la taille du logiciel exprimée en points de fonction. « Reuse » n'offre que des attributs de projets, à savoir : le type de projet, l'expérience de l'équipe de développement, le domaine d'application, le processus utilisé, le type du logiciel, l'approche de développement utilisée, etc.

L'ensemble de données « Qos » (Quality of Service) offre un ensemble d'attributs de composants afin de déterminer le temps de réponse, l'espace mémoire utilisé et l'exactitude ou la précision du composant.

Les ensembles de données, format non ARRF, ne sont pas inclus dans le Tableau 3.2; « MODIS » et « CM1-TR ». Ces derniers consistent en des exigences de haut niveau et leurs exigences correspondantes de bas niveau, et les liens entre ces deux types d'exigences.

En résumé, la plupart des ensembles de données du référentiel PROMISE ont été utilisés dans des modèles de prédiction afin de trouver une relation de causalité entre les variables prédictives d'entrées et la variable à prédire de sortie. Certains ensembles de données sont de taille relativement faible (faible nombre d'instances) et ceux qui présentent un grand nombre d'instances ne portent pas sur des caractéristiques du

produit logiciel. En effet, la majorité des ensembles de données possèdent des attributs quantitatifs ou des attributs qualitatifs de projets logiciels et non pas les deux ensembles. Plus encore, les ensembles de données offrent un nombre limité d'attributs, lesquels concernent principalement des mesures de code source du produit logiciel disponibles lors des dernières phases du cycle de développement du produit logiciel.

3.3 Référentiel de données d'ISBSG

L'International Software Benchmarking Standards Group (ISBSG), est une organisation formée en 1994 et elle a pour but tel que indiqué dans la charte originale « to develop the profession of software measurement by establishing a common vocabulary and understanding of terms » (ISBSG, 2006b). Le groupe ISBSG est composé d'un ensemble d'associations nationales de la mesure logicielle, représentant en 2007 13 pays différents. Le référentiel de projets logiciels d'ISBSG (2006c, p. 13) permet de fournir aux :

> « Software development practitioners with industry output standards against which they may compare their aggregated or individual projects, and real data of international software development that can be analyzed to help improve the management of IT resources by both business and government. »

Pour atteindre ce but, cette organisation à but non lucratif offre gratuitement un ensemble de questionnaires de collecte de données sur des projets logiciels, incluant la taille fonctionnelle du logiciel mesurée avec les mesures standards reconnues par ISO. Par la suite, ISBSG rassemble ces données dans un référentiel en Australie et fournit un extrait de ces données aux praticiens et chercheurs dans un fichier MS-Excel (Figure 3.2).

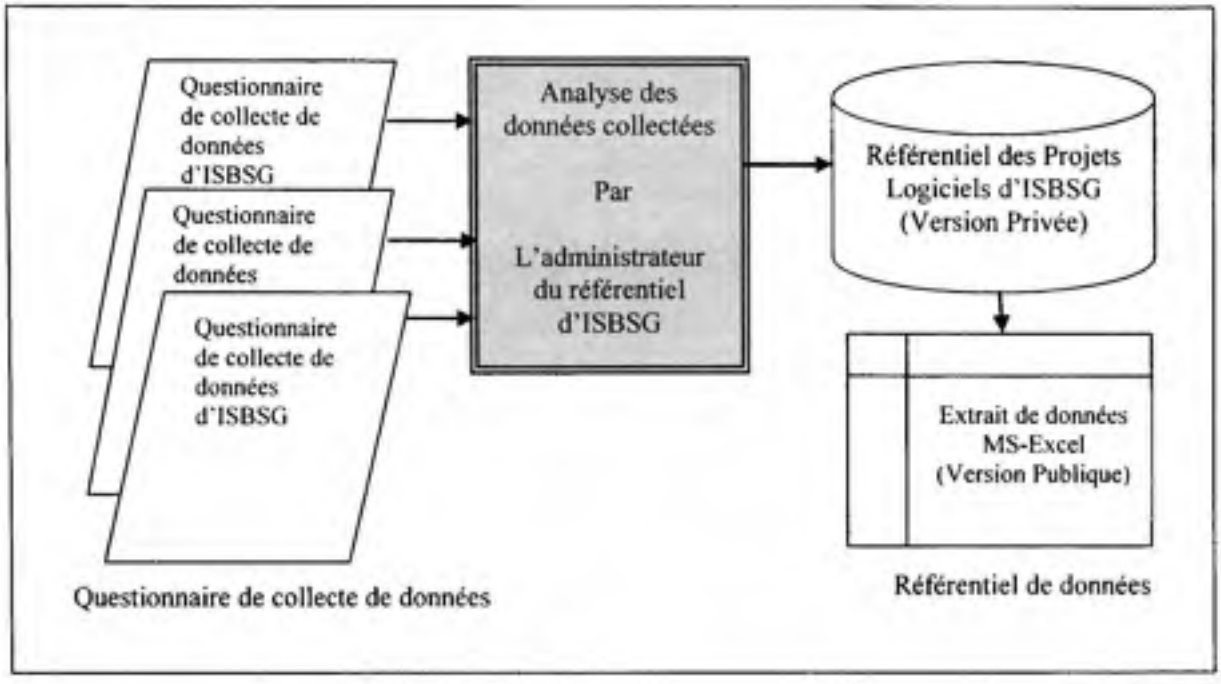

Figure 3.2 ***Gestion du référentiel ISBSG.***

3.3.1 Vue interne d'ISBSG : Questionnaire de collecte de données

En plus de la vue publique visible dans l'extrait de données d'ISBSG (fichier MS-Excel), il y a une vue interne du référentiel de données d'ISBSG qui correspond étroitement au questionnaire de collecte de ces données, avec quelques champs de données ajoutés par le responsable du référentiel d'ISBSG. Le questionnaire de collecte de données est disponible gratuitement sur le site web d'ISBSG (www.isbsg.org) et inclut un grand nombre d'informations quantitatives et qualitatives sur les différentes caractéristiques d'un projet logiciel, à savoir : l'équipe du projet, l'effort par phase de développement, les méthodes et techniques de développement, etc.

En parallèle, ISBSG met à la disposition des utilisateurs un dictionnaire de termes et de mesures qu'elle a définis (ISBSG, 2006b) afin de faciliter la compréhension du

questionnaire, d'assister à la collecte de données des projets dans le référentiel et de normaliser la manière dont les données collectées sont analysées.

Le questionnaire est composé de sept sections décomposées en plusieurs sous-sections. Pour les informations d'une même section qui n'appartiennent à aucune de ces sous-sections, nous les avons regroupées sous le mot « *Information Générale* » en italique (Figure 3. 3).

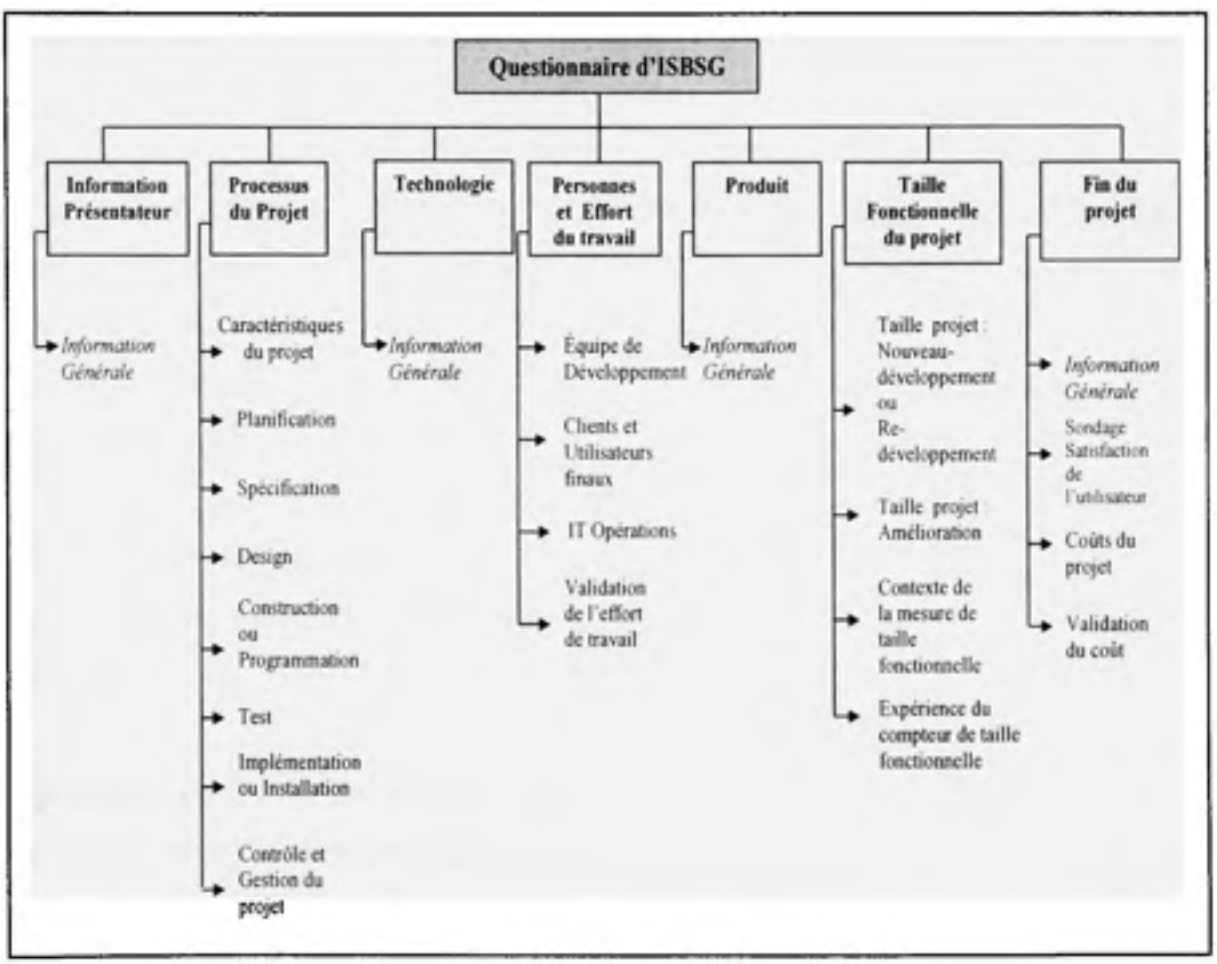

Figure 3.3 ***Structure du questionnaire de collecte de données d'ISBSG.***

Ces différentes sections sont décrites brièvement dans les paragraphes suivants.

Information du présentateur : cette section comprend les informations relatives à l'organisation et à la personne remplissant le questionnaire. Ces informations restent confidentielles pour ISBSG.

Processus du projet : cette section comprend les informations sur la manière dont le projet est effectué. ISBSG fournit des termes bien définis pour décrire le processus du projet, offre une simple structure pour rassembler les données et permet d'effectuer des comparaisons précises parmi les projets. Les informations de cette section sont structurées selon les différentes phases du cycle de vie du logiciel comme spécifié dans le questionnaire d'ISBSG : planification, spécification, design, construction ou programmation, test, implémentation ou installation. En effet, sachant la diversité des sources de données et la variabilité des processus de projets utilisés en industrie, les personnes qui remplissent le formulaire sont appelées à transposer leur processus à celui standardisé par ISBSG. Cette section est d'une grande importance, puisque le processus suivi par le projet logiciel a un impact important aussi bien sur ses performances que sur ses résultats (ISBSG, 2006c).

Technologie : cette section regroupe les informations relatives aux outils utilisés dans le développement du projet. Pour chaque étape du cycle de vie du logiciel, ISBSG offre une liste d'outils.

Personnes et Effort du travail : cette section regroupe les informations sur les différents types de personnel travaillant sur le projet, leurs nombres, leurs rôles, leurs expériences et l'effort qu'ils ont dépensé durant les différentes phases du processus du projet. Par ailleurs, pour des raisons de qualité d'ISBSG, des données doivent être collectées sur les procédures de collecte de l'effort. En fait, le référentiel distingue trois types de personnel : équipe ou organisation de développement, clients et utilisateurs finaux, et IT (Information Technology) opérations.

Produit : cette section regroupe les données sur le produit logiciel lui-même; il s'agit des informations de description, à savoir : le type d'application, la réutilisation, la plateforme de déploiement, etc.

Taille fonctionnelle du projet : cette section regroupe les informations relatives à la taille fonctionnelle du projet, son contexte et l'expertise de la personne ayant effectué la mesure de la taille. Cette section est légèrement différente pour chacune des méthodes de mesure reconnues par ISBSG : Common Software Measurement International Consortium – Full Function Points (COSMIC-FFP), International Function Point Users Group (IFPUG), NEtherlands Software Metrics users Association (NESMA) et Mark II function point analysis (Mark-II). Le questionnaire COSMIC-FFP d'ISBSG inclut, par exemple, des tables pour rassembler des informations quantitatives sur des transferts de données (entrées, sorties, écritures, lectures) pour chaque type de projet : nouveau développement de logiciel ou redéveloppement de logiciel, ou amélioration de logiciel.

Fin du projet : cette dernière section du questionnaire fournit une vue d'ensemble sur le projet une fois achevé, à savoir : la durée du projet, le nombre de défauts détectés durant le premier mois d'utilisation du logiciel, la durée totale du projet, le nombre de lignes de code, la satisfaction de l'utilisateur et le coût du projet incluant la validation de ce coût.

Le questionnaire COSMIC-FFP de collecte de données d'ISBSG, à titre d'exemple, comprend 38 pages et se compose de 137 questions réparties sur ces sept sections de données (Tableau 3.3); certaines de ces questions peuvent contenir un certain nombre de sous-questions. Par conséquent, le nombre d'informations rassemblées par le questionnaire est beaucoup plus grand. Quelques unes de ces informations sont obligatoires tandis que la plupart sont facultatives.

Tableau 3.3

Questionnaire COSMIC d'ISBSG - Groupes de données et Questions

Groupes de données	Nombre de questions	Pourcentage (%)
Information du présentateur	6	4%
Processus du projet	46	34%
Technologie	9	7%
Personne et Effort du travail	23	17%
Produit	6	4%
Taille fonctionnelle COSMIC	30	22%
Fin du projet	17	12%
Total	**137**	**100%**

3.3.2 Vue publique d'ISBSG : Extrait de données d'ISBSG

Comme déjà mentionné, les données collectées par les questionnaires d'ISBSG sont stockées dans un référentiel de données en Australie. En 2005, ce référentiel contenait plus de 3 000 projets (ISBSG, 2005). Cependant, ISBSG ne met à la disposition des praticiens et des chercheurs qu'un sous-ensemble de 100 champs de données, lesquels sont fournis dans un extrait de données à un prix modique à ces utilisateurs. Dans cet extrait de données (fichier MS-Excel) d'ISBSG-Release 9 de 2005, l'information est structurée en 15 catégories, comme illustré par la Figure 3.4.

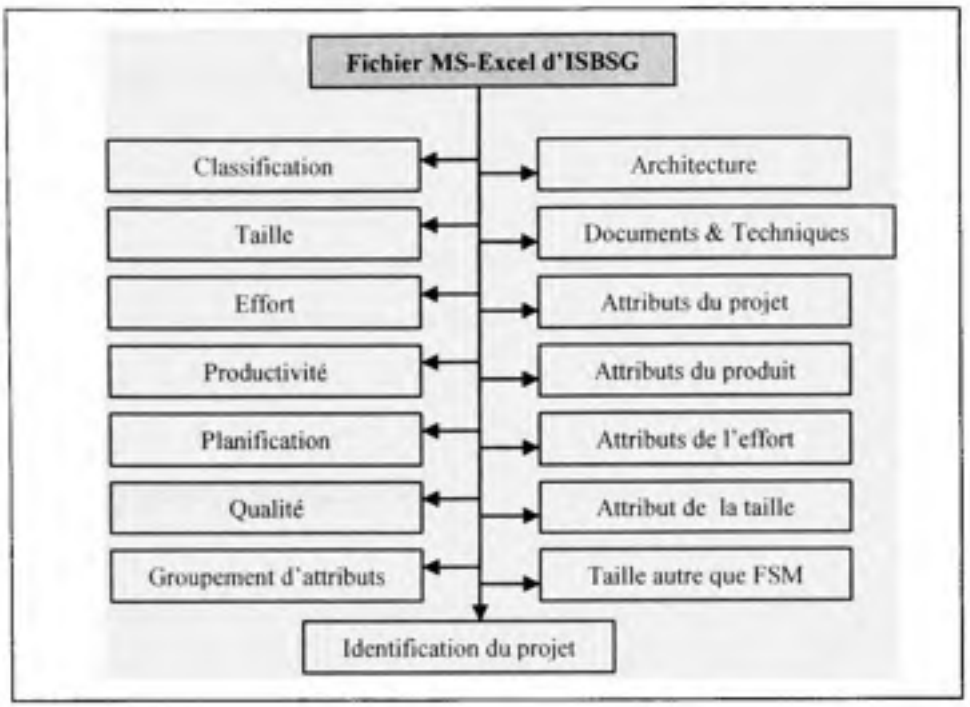

Figure 3.4 ***Structure de l'extrait de données (fichier MS-Excel) d'ISBSG - Release 9 de 2005.***

Cet extrait de données d'ISBSG-Release 9 de 2005 englobe les informations collectées des différents questionnaires d'ISBSG (COSMIC-FFP, NESMA, IFPUG, etc.). Plus encore, certaines informations sont dérivées des informations fournies dans le questionnaire et d'autres proviennent des versions antérieures des questionnaires d'ISBSG. Ci-dessous une description de façon sommaire de chaque catégorie de données. La structure détaillée de cet extrait de données d'ISBSG-Release 9 de 2005 est présentée en Annexe III.

Identification du projet : ISBSG assigne à chaque projet un numéro d'identification pour garder l'anonymat du projet et afin d'assurer la confidentialité des informations collectées, car leur divulgation peut entrainer l'identification de l'organisation qui a

soumis le projet. En fait, l'objectif d'ISBSG est d'offrir des données réelles sur des projets des organisations tout en gardant anonyme les origines de ces données.

Classification : les personnes chargées de la qualité dans ISBSG procèdent à la vérification des données avant leur stockage dans la base de données et, par la suite, assignent un code à chaque projet pour refléter la qualité des données collectées. Il en est de même pour la taille fonctionnelle : les experts de qualité d'ISBSG analysent les données rassemblées et en évaluent leur qualité, essentiellement pour les méthodes de mesure de taille fonctionnelle qui se basent sur les points de fonction avant ajustement.

Taille : cette catégorie regroupe l'ensemble des informations sur la taille du projet, à savoir : l'approche utilisée dans la collecte de la taille (IFPUG, COSMIC-FFP, etc.), la taille fonctionnelle elle-même (Adjusted Function Points (AFP), Unadjusted Function Points (UFP)), etc.

Effort : l'effort total consacré dans le développement du projet, exprimé en heures, est donné de façon globale pour toutes les phases du processus et tous rôles confondus.

Productivité : cette catégorie regroupe les informations sur le taux de la productivité livrée exprimé en heures par point de fonction. Ces informations représentent le ratio entre l'effort et la taille.

Planification : la durée du projet, la portée des activités du projet ainsi que les informations sur l'effort consacré pour chaque phase du processus du projet sont groupées dans cette catégorie d'informations.

Qualité : cette catégorie regroupe essentiellement l'ensemble des informations relatives au nombre de défauts détectés pendant le premier mois d'utilisation du logiciel.

Groupement d'attributs : cette catégorie d'informations regroupe le type de développement, le type d'organisation, le type d'application, le degré de réutilisation, etc.

Architecture : cette catégorie regroupe les informations relatives aux caractéristiques du produit logiciel comme l'architecture (client/serveur), le type de serveur, les rôles du serveur, etc.

Documents et techniques : le référentiel d'ISBSG fournit pour les différentes phases du processus logiciel, du planning jusqu'à l'implémentation, des informations sur les documents produits, les techniques utilisées ou les activités réalisées.

Attributs du projet : cette catégorie regroupe les informations relatives aux caractéristiques du projet comme les différentes plateformes de développement, les multiples langages de programmation, le type de méthodologie utilisée, etc.

Attributs du produit : cette catégorie regroupe l'ensemble des données sur le nombre d'utilisateurs utilisant le système en même temps, le marché ciblé, etc.

Attributs de l'effort : cette catégorie regroupe les informations collectées sur la taille de l'équipe du projet, le pourcentage de l'effort non collecté, etc.

Attributs de la taille : cette catégorie regroupe l'ensemble des informations sur la taille exprimée en points de fonction avant ajustement : les entrées externes, les sorties externes, les fichiers logiques internes, les interfaces externes et les requêtes externes. Cette catégorie regroupe aussi les informations sur le nombre de fonctions ajoutées, modifiées ou supprimées lors de l'amélioration du logiciel.

Taille autre que FSM (Functional Size Measurement) : cette catégorie regroupe des informations sur la taille du logiciel exprimée en lignes de code (LOC) ainsi que le pourcentage de lignes de code qui ne sont pas de type « instructions » du programme.

En résumé, il est important de noter que l'organisation ISBSG regroupe les informations collectées du questionnaire (Figure 3.3) dans une structure différente de l'extrait de données (Figure 3.4) dans le but de faciliter son exploitation par les praticiens et les chercheurs. De plus, dans l'extrait de données d'ISBSG (Release 9 de 2005) (Figure 3.4), il y a une catégorie de données relative à la qualité et qui correspond au nombre de défauts (mineurs, majeurs, extrêmes ou total) : plus précisément, le nombre de défauts reportés dans le premier mois d'utilisation du logiciel (voir Annexe III).

3.4 Sommaire et discussion

Pourquoi a-t-on choisi ces deux référentiels parmi ceux qui existent en génie logiciel ? En effet, plusieurs initiatives de création de référentiels de données publics ont été faites durant cette dernière décennie. Cependant, ces initiatives ont connu une fin prématurée (Cukic, 2005). Ainsi, nous avons choisi d'explorer les deux référentiels de données les plus connus, utilisés et surtout les plus à jour dans la littérature. Ces deux référentiels ont des points communs et divergent dans d'autres : ceci est discuté maintenant dans cette section.

Les deux référentiels de données ISBSG et PROMISE sont nés de la nécessité de créer des données normalisées et de référence permettant aux chercheurs de comparer leurs résultats avec ceux des autres chercheurs. En effet, ces référentiels peuvent aider les chercheurs à mieux comprendre certaines relations de causes-effets en étudiant les variables disponibles dans les référentiels et, par la suite, à faire ressortir lesquelles de ces variables contribuent le plus dans l'achèvement de certains objectifs, à savoir : augmenter la productivité, améliorer la qualité, etc.

ISBSG offre à coûts modiques au public les données collectées des différentes organisations du monde entier, avec différentes méthodologies, techniques et phases de cycle de vie du logiciel sous la forme d'un format standard. D'une part, ces données proviennent des différents secteurs utilisant le logiciel et, d'autre part, elles sont destinées aux divers utilisateurs, aussi bien aux chercheurs qu'aux industriels. Ce référentiel offre des données utiles pour des buts multiples, à savoir : la comparaison de modèles de productivité, les modèles d'estimation de l'effort, etc. De tels modèles peuvent être utilisés par les organisations pour améliorer leurs capacités en termes de planification et de contrôle de projets. En outre, le référentiel d'ISBSG collecte un grand nombre de variables discrètes et de données quantitatives portant sur les différentes caractéristiques du projet logiciel, incluant le processus du projet logiciel avec ses diverses phases de la planification à l'achèvement.

PROMISE offre gratuitement au public 20 ensembles de données, lesquels sont issus des travaux de recherche ou de systèmes « open source » en relation avec le génie logiciel. Le but des auteurs de PROMISE est d'offrir aux chercheurs et praticiens de ce domaine l'opportunité de tester leurs hypothèses et ainsi améliorer leurs pratiques. Ces ensembles de données ont été utilisés pour différents buts, à savoir : la prédiction de défauts de logiciel, l'estimation du coût du logiciel, etc. En outre, ces ensembles de données s'intéressent essentiellement aux attributs collectés lors des dernières phases du cycle de vie du logiciel, en particulier le code source du logiciel et non pas aux différentes étapes du cycle de vie du logiciel, tel le cas du référentiel d'ISBSG.

Pour notre recherche, le référentiel de données d'ISBSG est le plus approprié. Ce choix vient du fait qu'ISBSG collecte, entre autres, des données relatives à la qualité du logiciel qui s'étalent sur tout le cycle de vie du logiciel depuis son initiation jusqu'à son achèvement. De telles données permettront de vérifier les liens entre les trois types de qualité d'ISO 9126.

En outre, le référentiel d'ISBSG offre diverses catégories d'informations sur le projet logiciel, à savoir : la technologie, la qualité, l'effort, le coût, la satisfaction du client, etc. Ces catégories sont liées et une catégorie peut influencer ou être influencée par une autre catégorie. Par exemple, tel que montré dans le modèle intégré d'analyse détaillée du « Practical Software Measurement (PSM) » proposé dans (McGarry *et al.*, 2002), la qualité du produit logiciel influence la satisfaction du client et peut elle-même être influencée de façon directe ou indirecte par les autres catégories dont la technologie, la taille, la performance, les ressources et le planning. Ainsi, divers facteurs peuvent contribuer à un problème bien déterminé. C'est pour cette raison qu'il est important de prendre en considération les différentes catégories d'informations fournies par le référentiel d'ISBSG dans l'identification des facteurs influençant la qualité interne, externe et en utilisation lors de la conception des plans d'analyses empiriques avec la méthode Taguchi de conception de plan d'expériences.

CHAPITRE 4

MÉTHODE TAGUCHI

4.1 Introduction

> « La réussite d'un plan d'expériences ne se mesure pas à la complexité du modèle obtenu, mais à la satisfaction qu'il a générée chez le client. » (Pillet, 1997, p. 292)

Pour résoudre un problème, nous avons souvent tendance à procéder par intuition, par des tâtonnements, par des tests, par des clauses « si » et « sinon », etc. Dans ce cas, nous perdons beaucoup de temps avant d'aboutir à la solution du problème. D'où l'importance de la planification des expériences afin de gagner en temps et moyens. Plus encore, ces différentes manières de procéder sont orientées plus vers la résolution du problème que vers la compréhension des moyens permettant sa résolution.

La méthode Taguchi de conception de plan d'expériences est une méthode orientée vers deux objectifs : comprendre et résoudre le problème étudié. Cette méthode est très utilisée dans le domaine industriel, en particulier dans le domaine de l'automobile, de l'électronique ainsi que dans la production de produits et la conception de processus de production.

Afin d'approfondir nos connaissances sur la méthode Taguchi, nous commençons par présenter brièvement la notion de plan d'expériences suivi d'une vue d'ensemble sur la méthode Taguchi et ses avantages pour le génie logiciel. Ensuite, nous présentons l'état de l'art sur l'utilisation de cette méthode dans les expérimentations avec des logiciels.

4.2 Plan d'expériences

De façon générale, une expérience est une étude qui peut être vue comme un processus de production (Figure 4.1), composé d'intrants, de facteurs qu'on peut contrôler ou gérer, du processus lui-même, des facteurs influençant le processus mais difficilement contrôlables et de la sortie désirée qui montre jusqu'à quel point l'objectif visé par l'expérience est accompli.

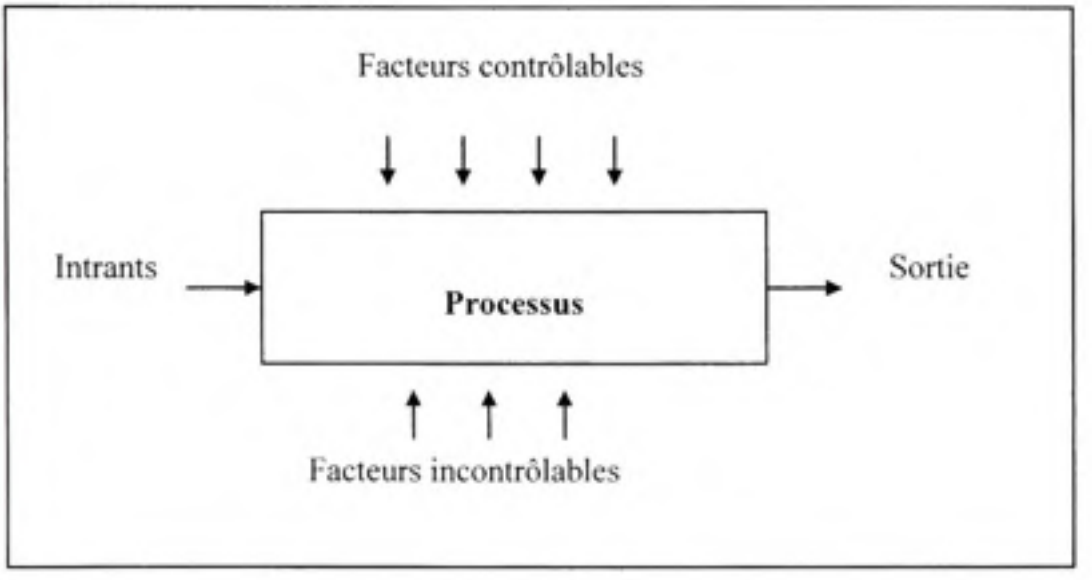

Figure 4.1 ***Processus de production.***

Le plan d'expériences est une technique statistique introduite par Ronald Fisher en Angleterre au début des années 1920 dans le cadre de recherches agronomiques pour étudier l'effet des variables comme la quantité de pluie, d'eau, de rayons solaires, etc. nécessaires pour produire de bonnes récoltes.

D'après Cesarone (2001), les plans d'expériences peuvent être considérés comme des techniques d'optimisation des processus qui possèdent des entrées contrôlables et des

sorties mesurables. Ces plans sont utilisés aussi bien dans les scénarios de production que de conception.

Divers types de plan d'expériences existent et ces types diffèrent dans la manière de traiter les facteurs étudiés, les niveaux associés à ces facteurs et les interactions qui peuvent exister, à savoir :

- plan d'expériences à un seul facteur : un seul facteur varie;
- plan d'expériences factoriel (à n facteurs) : plus efficace que le plan d'expériences à un seul facteur, et dans lequel tous les facteurs sont utilisés de façon simultanée;
- plan d'expériences factoriel complet 2^f : cas spécial de plan d'expériences factoriel, et dans lequel chacun des facteurs (f) possède deux (2) niveaux;
- plan d'expériences factoriel fractionnaire : plan optimal qui tient compte des effets des facteurs et de leurs interactions toute en limitant le nombre des essais.

C'est à cette dernière catégorie qu'appartiennent les plans d'expériences de Taguchi, mais appliqués différemment avec des principes spécifiques.

4.3 Méthode Taguchi de design de plan d'expériences

La méthode Taguchi a été développée par le Dr. Genichi Taguchi, un chercheur à l'« Electronic Control Laboratory » au Japon vers les années 1960. Pierre Souvay (2002) souligne que Taguchi a exploité les plans d'expériences proposés par Ronald Fisher en les améliorant.

D'après Beauchamp (2005, p. 6), « G. Taguchi a été reconnu dans le domaine de la qualité pour sa contribution au niveau de la fonction perte de qualité, de tables orthogonales et des graphes linéaires, et de la robustesse ». Beauchmap résume la philosophie de Taguchi avec les trois points suivants :

- les produits et les procédés doivent être conçus de manière robuste par rapport aux variations externes;
- les plans d'expériences sont des outils permettant d'atteindre cet objectif;
- tendre vers une valeur cible (minimale, nominale, maximale).

Par ailleurs, la méthode Taguchi « s'inscrit naturellement dans le cadre de la maîtrise et de l'amélioration de la qualité telle que définie par les normes ISO 9000 (version 2000) » (Souvay, 2002, p. 5). C'est une méthode d'expérimentation basée sur des techniques industrielles et des techniques statistiques : elle est utilisée non seulement dans la conception de systèmes, mais aussi dans les processus de production. En outre, cette méthode nécessite une compréhension du système de l'étude et des connaissances suffisantes (théoriques ou pratiques) afin de déterminer les facteurs, leurs niveaux et les interactions.

Taguchi offre une vue de la qualité qui prend en considération la relation entre la qualité et le coût. La qualité d'un produit est définie comme « loss imparted by the product to the society from the time product is shipped » (Taguchi, Chowdhury et Wu, 2004, p. 128). En effet, la fonction perte de qualité proposée par Taguchi permet d'analyser économiquement la perte engendrée par une mauvaise qualité. Trois types de fonctions sont disponibles selon la caractéristique de qualité étudiée.

Pour faciliter la compréhension de la méthode Taguchi, des notions sont introduites brièvement dans les sections suivantes, toutes documentées plus en détail dans (Beauchamp, 2005; Peace, 1993; Pillet, 1997; Ross, 1996; Roy, 1990; 2001; Souvay, 2002; Taguchi, Chowdhury et Wu, 2004).

4.3.1 Stratégie de contrôle de qualité

Taguchi partage sa stratégie de contrôle de qualité en deux phases : en production (on-line) et hors production (off-line).

La phase en production (on-line) concerne les méthodes et les techniques utilisées pour contrôler la qualité lors de la production du produit. Le but est de maîtriser les processus de production tout en visant la meilleure qualité avec un coût minimum.

La phase hors production (off-line) a pour objectif d'optimiser la conception du produit pour supporter la qualité en production. Cette phase correspond aux méthodes et techniques à prendre en considération avant que le produit soit manufacturé (c'est-à-dire lors des phases de design, de développement, etc.) et soit disponible aux clients. Taguchi souligne le besoin de prendre en considération le contrôle de la qualité lors de la phase de conception puisque les activités d'inspection et de contrôle de la qualité du produit ne peuvent pas corriger les inconvénients d'une mauvaise conception (Liu, 2004).

Le contrôle de qualité hors production se base sur trois étapes séquentielles, à savoir : le design de système (system design), les paramètres de design (parameter design) et la tolérance de design (tolerance design).

1. Design de système : représente l'étape de conception du système dans laquelle les ingénieurs et les scientifiques développent de nouveau concepts, idées ou méthodes originales pour la production de nouveaux produits ou l'amélioration. L'objectif est de prendre ces nouvelles idées, lesquelles sont théoriques, et de les convertir en quelques choses pratiques dans la deuxième étape.
2. Paramètres de design : l'objectif dans cette étape est d'utiliser les plans d'expériences pour déterminer les paramètres qui permettent de satisfaire les objectifs de la première étape.

3. Tolérance de design : cette phase a pour objectif de trouver le meilleur arrangement entre le coût du produit et sa fonctionnalité souhaitée afin de réduire la variation de la sortie désirée.

Il est à noter que la plupart des ouvrages adressent l'étape de paramètres de design, connue sous le nom de la méthode Taguchi.

4.3.2 Paramètres de design

L'étape de paramètres de design de Taguchi vise à optimiser le système, c'est-à-dire sélectionner les niveaux optimums pour les facteurs contrôlés, pour qu'il soit robuste aux facteurs bruits. Les principales étapes sont décrites brièvement comme suit :

1. Définition de l'objectif : consiste à déterminer la caractéristique de qualité à optimiser.
2. Identification des facteurs et leurs niveaux : consiste à déterminer les facteurs contrôlés, les facteurs bruits, les niveaux de chaque facteur ainsi que les interactions possibles. Taguchi considère que les interactions entre plusieurs facteurs sont toujours négligeables, exception faite pour les interactions entre deux facteurs si elles sont bien identifiées.
3. Choix de la table orthogonale : consiste à sélectionner la table orthogonale appropriée du plan étudié. Cette table représente la matrice d'expériences (essais) à mener. Le choix de la table orthogonale est fait en fonction du nombre de facteurs, de niveaux et d'interactions, s'il y a lieu.
4. Réalisation des expériences : consiste à effectuer les expériences et enregistrer les résultats.
5. Analyse des résultats : consiste à déterminer la condition optimale, c'est-à-dire la configuration optimale des paramètres de design et, par la suite, calculer l'équation de prédiction du plan étudié.

6. Test de confirmation : Taguchi recommande de faire un test de confirmation (i.e., de nouveaux essais différents de ceux déjà effectués) en utilisant la condition optimale afin de confirmer ou non le résultat issu de l'analyse des résultats du plan étudié.

Ci-après quelques définitions dans le vocabulaire utilisé dans l'étape de paramètres de design.

Caractéristique de qualité : c'est un nouveau terme pour signifier le résultat ou la réponse, mais avec une signification plus large qui englobe le sens du résultat désiré, c'est-à-dire la direction prévue du résultat. C'est donc un résultat avec une direction. Trois types de caractéristiques de qualité sont identifiés :

- l'optimum est une valeur minimale (Smaller-the-better);
- l'optimum est une valeur nominale (Nominal-the-best);
- l'optimum est une valeur maximale (Larger-the-better).

Facteur : c'est une variable (continue ou discrète) qui peut influencer le résultat. Deux types de variables ou facteurs se distinguent, à savoir : le facteur contrôlé ou contrôlable qu'on peut gérer ou régler et le facteur bruit qui influence le processus, mais qui est difficilement contrôlable.

Niveaux de facteur : représentent les états ou les valeurs que peut prendre un facteur lors de la réalisation des essais.

Interaction : représente l'effet d'un facteur renforcé par l'effet d'un autre facteur de telle sorte que le résultat obtenu par leur combinaison est différent du résultat s'ils sont pris séparément.

Ratio signal-bruit (S/B) « Signal-to-Noise (S/N) » : Taguchi recommande l'utilisation de ce ratio pour représenter la robustesse. Un produit est robuste si sa qualité est

insensible aux facteurs bruits. Le signal désigne la moyenne de la réponse et le bruit représente la dispersion de la réponse en fonction des facteurs bruits. Trois types de ratio S/B sont disponibles selon la caractéristique de qualité, avec des formules différentes.

Plan produit : c'est une combinaison de deux plans d'expériences : l'un porte sur les facteurs contrôlés et l'autre sur les facteurs bruits. Le plan produit consiste à répéter les essais du premier plan pour chaque configuration du deuxième plan.

Tables orthogonales : elles représentent l'un des points forts de la méthode Taguchi et elles sont destinées à construire des plans d'expériences. Taguchi propose un ensemble de tables orthogonales composées des lignes représentant les différents essais à mener et des colonnes. Ces dernières permettent de représenter les facteurs et les interactions dépendamment du plan étudié. Les cellules contiennent les niveaux des facteurs.

À titre d'exemple, la table orthogonale de Taguchi représentée par $L_4(2^3)$ correspond à quatre essais pour un plan constitué de trois facteurs avec deux niveaux chacun (Tableau 4.1).

Tableau 4.1

Exemple de table orthogonale L_4 de Taguchi

Colonnes / Essai	1	2	3
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

Taguchi utilise les tables orthogonales afin de limiter le nombre d'essais à mener. Comparées au plan factoriel, ces tables représentent une sélection ou un sous-ensemble réduit d'essais. Les tables orthogonales à deux niveaux les plus utilisées sont : $L_4(2^3)$, $L_8(2^7)$, $L_{12}(2^{11})$, $L_{16}(2^{15})$ et $L_{32}(2^{31})$. Les tables orthogonales à trois niveaux les plus utilisées sont : $L_9(3^4)$, $L_{18}(2^1 \times 3^7)$ et $L_{27}(3^{13})$.

Graphes linéaires : Taguchi associe à certaines tables orthogonales un ou plusieurs graphes linéaires. Ces derniers facilitent le positionnement des facteurs et des interactions dans la table orthogonale. Dans un graphe linéaire, chaque point noir représente une colonne dans la table orthogonale. Le trait entre les deux points représente l'interaction entre les facteurs représentés par les deux points à chaque extrémité du trait. Le numéro associé au point ou au trait représente la colonne de la table orthogonale à laquelle est affectée respectivement le facteur ou l'interaction.

Par exemple, un seul graphe linéaire est associé à la table orthogonale L_4 (Tableau 4.1). Ce graphe est illustré par la Figure 4.2.

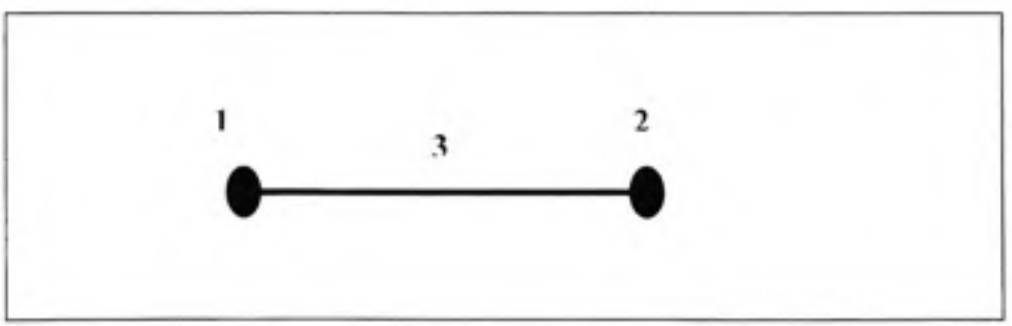

Figure 4.2 ***Graphe linéaire de la table orthogonale*** L_4.

Pour un plan d'étude comportant trois facteurs A, B et C avec deux niveaux chacun, le facteur A est affecté à la colonne 1 de la table orthogonale L_4 (voir Tableau 4.1), le facteur B à la colonne 2 et le facteur C à la colonne 3.

Dans le cas ou le plan d'étude comporte deux facteurs A et B et une interaction (AB) alors le facteur A est affecté à la colonne 1 de la table orthogonale L_4 (voir Tableau 4.1), le facteur B à la colonne 2 et l'interaction (AB) est affectée à la colonne 3.

4.4 Avantages de la méthode Taguchi pour le logiciel

Taguchi dans sa philosophie de qualité n'a pas exclu le produit logiciel. En effet, le logiciel est aussi un produit, mais inclut le développement et non la production. Plus encore, la nature générique de la qualité est applicable aussi bien dans les domaines matures que ceux immatures, le génie logiciel par exemple. Ainsi, l'utilisation de la méthode Taguchi dans le contexte de la qualité du produit logiciel est possible.

Parmi les avantages de l'utilisation de la méthode Taguchi, nous citons à titre d'exemple :

- l'amélioration de la qualité : la stratégie de contrôle de qualité de Taguchi ne se limite pas à la qualité lors de la production, mais aussi à la qualité dès les premières phases du cycle de développement du logiciel. Tout comme le soulignent Sculmeyer et McManus (1999, p. 78-79), « the application of off-line quality control to software development would place development variables under control factors » et « on-line quality control applies to software development when a company has a defined, repeatable process »;
- la réduction des coûts : Taguchi dans sa définition de la qualité met l'emphase sur le coût du produit. L'objectif est d'améliorer la qualité, mais pas à n'importe quel coût et n'importe comment. C'est une méthode peu coûteuse dont la force de son utilisation réside dans son habilité à évaluer plusieurs facteurs de qualité dans un

nombre minimum d'essais (Seshadri, Kannan et Sathish, 2004), et ce, à travers l'utilisation des tables orthogonales et par la suite la réduction de l'effort pour l'exécution des essais;

- l'acquisition des connaissances : tel que reporté par (Basili, Selby et Hutchens, 1986), nous nous intéressons à la résolution du problème et à l'acquisition des connaissances par les expérimentations en génie logiciel. La méthode Taguchi est importante puisqu'elle prend en considération ces deux intérêts de comprendre et de résoudre le problème. De la sorte, cette méthode contribuera à l'amélioration des connaissances et des pratiques en génie logiciel.

4.5 Utilisation de Taguchi dans les expérimentations avec des logiciels

Dans la plupart des livres de plans d'expériences, l'emphase est mis sur l'utilisation de la méthode Taguchi dans le secteur industriel (Beauchamp, 2005; Pillet, 1997; Ross, 1996; Roy, 1990; 2001; Souvay, 2002; Taguchi, Chowdhury et Wu, 2004). Cette méthode a connu beaucoup de succès et a prouvé sa robustesse dans le domaine du génie et de la science; toutefois son utilisation dans le domaine du logiciel en est à ces débuts. En fait, peu de travaux de recherche ont utilisé cette méthode dans des expériences pour différents objectifs (Kanchana et Sarma, 1999; Liu, 2004; Seshadri, Kannan et Sathish, 2004; Tsai, Moskowitz et Lee, 2003) .

L'objectif de Seshardi, Kannan et Sathish (2004) est de déterminer les facteurs cruciaux qui influencent la qualité du produit/processus du logiciel. La caractéristique de sortie considérée représente le dépassement du coût de projet (Cost Overrun). Quatre facteurs (Tableau 4.2) et trois interactions jugées importantes sont séléctionnés. La table orthogonale L_8 (huit essais) est choisie.

De l'analyse des résultats de cette étude, il en ressort que les facteurs de technologie, de complexité de travail et de compétence de gestion de projet influencent la caractéristique

de sortie : dépassement du coût de projet. En outre, les interactions entre les facteurs de technologie et de compétence de gestion de projet avec le facteur niveau de compétence des personnes impliquées dans le travail contribuent dans la variance de ce dernier.

Tableau 4.2

Facteurs et leurs niveaux
(Seshadri, Kannan et Sathish, 2004)

Facteurs	Niveaux	Descriptions
Skill Level of the Associates	1	E1- Novice
	2	E2 – Experienced
Complexity of the work	1	Simple – Effort involved is less than 125 Person/Hours
	2	Average – Effort involved is between 125 - 500 Person/Hours
Technology	1	Legacy – Involving technologies like Mainframe, Unix, C,C++
	2	Cutting Edge - Web logic, Dot Net
Project management skill	1	Low
	2	High

Kanchana et Sarma (1999) veulent optimiser les paramètres de design du logiciel dans le processus de développement du logiciel. L'objectif est de réduire le nombre d'erreurs introduites dans la phase de design par module. Trois facteurs sont utilisés (le couplage, le nombre d'exigences par module et le nombre de la complexité cyclomatique) et leurs niveaux sont choisis. La table orthogonale choisie est L_9 (neuf expériences) et le ratio signal-bruit a été utilisé. De l'analyse des résultats expérimentaux, il en résulte que pour développer un logiciel de qualité, le nombre de la complexité cyclomatique doit être inférieur à cinq, le couplage doit être faible et le nombre de besoins par module est de un besoin par module afin de réduire le nombre d'erreurs.

Rao et Sarda (2003) se sont intéressés à la sous-traitance (outsourcing) de la maintenance de grands systèmes commerciaux et discutent le problème d'estimation de l'effort de maintenance en absence des paramètres (facteurs) clés comme l'effort de développement et l'historique de la maintenance. Un ensemble de 14 facteurs a été utilisé. Le nombre d'expériences a été limité à 36. L'analyse du résultat de l'étude a permis de conclure par un ensemble de points importants. Par exemple, pour un système où la taille exprimée en termes de lignes de code est connue, les paramètres affectant l'effort de maintenance sont : le support aux utilisateurs, le nombre moyen de lignes de code par programme et la complexité du système de fichier utilisé.

Tsai, Moskowitz et Lee (2003) cherchent à trouver une méthode systématique pour la sélection des ressources humaines pour le développement de projets logiciels. L'objectif est de minimiser aussi bien la durée du projet que le coût total du projet. Trois types de ressources humaines sont sélectionnés comme facteurs contrôlables (deux niveaux par facteur) et un plan factoriel complet est appliqué, soit huit essais. Huit types de tâches sont sélectionnés comme facteurs bruits (trois niveaux par facteur) et la table orthogonale L_{81} est choisie. En utilisant le ratio signal-bruit, le résultat de cette étude a montré que la méthode Taguchi est effective et efficace pour la sélection des ressources humaines.

Enfin, l'objectif de Liu (2004) est d'améliorer la performance des systèmes logiciels et de réduire le coût de développement par l'assemblage des services provenant de vendeurs connus ou des services intranet. La cible choisie est le temps de réponse de l'implémentation d'un NFS (Network File System) en Linux. Les facteurs contrôlés sont au nombre de quatre (trois niveaux par facteur) et les facteurs bruits sont au nombre de trois (deux niveaux par facteur). Les tables orthogonales pour ces deux types de facteurs sont : L9 et L4. Le ratio signal-bruit a été utilisé et la configuration optimale des niveaux des facteurs permettant d'optimiser la performance d'un NFS a été déterminée.

En résumé, ces travaux de recherche ont tous utilisé la notion de paramètres de design de Taguchi (section 4.3.2). En outre, ces travaux montrent que cette méthode commence à être utilisée dans le domaine du logiciel pour des objectifs tels que : l'amélioration du processus du logiciel, l'amélioration de la qualité du logiciel, l'évaluation de l'effort de la maintenance, la sélection des ressources humaines pour le développement du projet logiciel, etc.

Dans ces travaux de recherche, à partir des sessions de réflexion (brainstorming), un ensemble de facteurs (contrôlables ou bruits) qui peuvent avoir des effets sur la sortie souhaitée (exemples : effort de maintenance, défauts de conception, dépassement du coût, coût du projet et sa durée, performance, etc.) sont déterminés ainsi que leurs niveaux (deux à trois niveaux par facteur). Certaines de ces expérimentations ne tiennent compte que des facteurs indépendants et leurs effets, alors que d'autres ont choisi les interactions jugées importantes entre les facteurs étudiés et qui peuvent influencer le résultat. La notion de table orthogonale a été utilisée afin de limiter le nombre des essais à mener. Pour chercher la robustesse, le ratio signal-bruit a été utilisé.

Par ailleurs, Akingbehin (2005a; 2005b) a utilisé les fonctions pertes de qualité, dans le cas où *l'optimum est une valeur nominale* et dans le cas *où l'optimum est une valeur minimale*, de Taguchi afin de mesurer quantitativement la qualité. Selon Akingbehin, le produit logiciel peut contribuer à la philosophie de Taguchi « loss imparted to society » de différentes manières, à titre d'exemple :

- le produit logiciel qui est si insatisfaisant qu'il doit être remplacé par un nouveau produit. La perte à la société représente le coût de remplacement du produit logiciel, soit principalement le coût du nouveau produit;
- le produit logiciel qui tombe en panne fréquemment et qui nécessite une réparation à chaque échec. La perte engendrée représente le coût cumulatif de réparation du logiciel.

4.6 Sommaire

Tout d'abord, pourquoi a-t-on choisi la méthode Taguchi de conception de plan d'expériences ? Le développement de méthodes de qualité date de 1920, depuis de multiples méthodes ont vu le jour, à savoir : « Statistical process control », « Design of experiments », « Total quality management », « Taguchi robust design », « Six sigma », etc. Chaque méthode est développée pour un objectif bien défini par son concepteur. Certaines de ces méthodes utilisent des principes des méthodes précédentes en les améliorant : tel est le cas de la méthode Taguchi de conception de plan d'expériences. Le choix de Taguchi est justifié d'une part par les avantages de son utilisation en génie logiciel (section 4.4) et, d'autre part, par la robustesse de cette méthode, c'est-à-dire pour améliorer la conception de systèmes capables de délivrer les fonctionnalités attendues afin qu'ils soient insensibles aux variations causées par les facteurs bruits (Yang et El-Haik, 2003).

Dans ce chapitre, nous avons introduit la notion de plan d'expériences et la méthode Taguchi de conception de plan d'expériences. Les avantages de l'utilisation de cette méthode en génie logiciel résident dans l'amélioration de la qualité du produit logiciel, la réduction du coût et l'acquisition des connaissances. Des travaux de recherche traitant de l'utilisation de cette méthode dans les expérimentations avec des logiciels afin d'améliorer la qualité des processus et des produits logiciels ont été aussi présentés.

Nous avons aussi constaté la diversité des facteurs (quantitatifs et qualitatifs) utilisés pour mener les expériences avec la méthode Taguchi. À titre d'exemple, des mesures ont été utilisées comme facteurs influençant la qualité du logiciel, à savoir : le nombre de la complexité cyclomatique, le couplage, etc. Ces travaux de recherche donnent une idée sur les facteurs importants à prendre en considération lors des études (expérimentales ou empiriques) futures avec des logiciels en utilisant la méthode Taguchi.

CHAPITRE 5

OBJECTIFS ET MÉTHODOLOGIE DE RECHERCHE

5.1 Introduction

Dans les chapitres précédents (1 à 4), nous avons mis l'accent sur les trois points clés de notre approche méthodologique, laquelle approche est précisée maintenant dans ce chapitre. Nous présentons tout d'abord les objectifs de cette recherche suivi d'une vue détaillée de la méthodologie de recherche que nous avons conçue pour rencontrer nos objectifs spécifiques de recherche.

5.2 Objectifs de la recherche

Comme déjà introduit dans l'introduction de cette thèse, la motivation principale dans cette recherche est d'explorer la pertinence des relations entre les modèles de qualité de la norme ISO 9126-1 dans la production de logiciels de qualité, et ce, pour tout le cycle de vie du logiciel.

Afin de démontrer la pertinence ou non des relations entre ces modèles, les objectifs spécifiques de cette recherche sont de démontrer, par des études empiriques, si les relations prises pour acquises par ISO 9126 (Figure 2.1) sont supportées par des données empiriques. Il s'agit des relations entre :

- la qualité interne et la qualité externe;
- la qualité externe et la qualité en utilisation;
- la qualité interne et la qualité en utilisation.

En fait, les besoins en qualité changent au fur et à mesure de l'avancement dans le projet soit par oubli de certaines exigences, ou par des changements des exigences existantes, ou par l'établissement de nouvelles exigences (ISO, 2001a). Par conséquent, les mesures

effectuées au début du cycle de vie du logiciel (de façon statique) pour évaluer sa qualité interne peuvent être insuffisantes et ne pas refléter la qualité du produit logiciel une fois terminé. Cependant, ce n'est que lors des phases d'opérations et de test (de façon dynamique) que les mesures peuvent refléter la qualité externe du produit logiciel. Plus encore, si le logiciel satisfait les contraintes de qualité en utilisation (contexte, environnement, etc.), la qualité pour l'utilisateur final est accomplie. De la sorte, l'évaluation de la qualité interne, de la qualité externe et de la qualité en utilisation du logiciel est importante.

Les questions (i.e., les objectifs spécifiques de recherche) que nous nous posons sont donc les suivantes :

1. Existe-t-il une relation entre la qualité interne et la qualité externe du produit logiciel de telle sorte que la réalisation de la qualité lors des phases de design et de codage affecte la qualité lors de test et d'exécution du logiciel ?
2. Existe-t-il une relation entre la qualité externe et la qualité en utilisation du produit logiciel de telle sorte que la réalisation de la qualité lors des phases d'exécution du logiciel affecte son utilisation une fois livré à l'utilisateur final ?
3. Existe-t-il une relation entre la qualité interne et la qualité en utilisation du produit logiciel de telle sorte que la réalisation de la qualité lors des premières phases du cycle de développement du logiciel affecte son utilisation une fois livré à l'utilisateur final ?

Quelques auteurs ont tenté d'établir des liens entre les différentes caractéristiques (et sous-caractéristiques) des différents types de qualité notamment dans (Côté, 2005; Gil et Suryn, 2005). Cependant, ces liens proposés dans ces travaux ne sont que des suggestions théoriques et manquent de confirmations au moyen d'études expérimentales ou d'études empiriques.

La méthodologie conçue pour répondre à ces questions et, par la suite, la réalisation des objectifs de cette recherche est présentée dans la section suivante.

5.3 Méthodologie de recherche

Pour atteindre la qualité du produit logiciel selon les différentes perspectives et durant tout le cycle de vie du logiciel, il faut disposer d'une boite à outils. Dans une problématique comme la nôtre, cette boite à outils est composée de :

- ISO 9126 : modèles de qualité proposée par la norme ISO 9126-1 et mesures proposées dans les rapports techniques ISO TR 9126-2 à 4;
- méthode Taguchi de conception de plan d'expériences;
- référentiel de données industrielles d'ISBSG et questionnaire de collecte de ces données.

Notre méthodologie de recherche est partagée en trois grandes phases, exploration, préparation et plan d'analyse empirique, détaillées dans les sections suivantes.

5.3.1 Phase 1 : Exploration

La phase d'exploration consiste en deux étapes : la première globale et la deuxième détaillée.

Étape 1 : Exploration globale

L'exploration globale comprend la revue de la littérature de :

- la qualité du produit logiciel (chapitre 1);
- la série des documents ISO 9126 (chapitre 2);
- les référentiels de données disponibles en génie logiciel (chapitre 3);
- la méthode Taguchi de conception de plan d'expériences (chapitre 4).

De cette exploration, il ressort que la qualité est un concept qui signifie différentes choses pour différentes personnes. La norme ISO 9126-1 offre une définition qui englobe trois points de vue et perspectives de qualité du produit logiciel (interne, externe et en utilisation). Les mesures sont importantes pour évaluer la qualité du produit logiciel puisqu'elles permettent d'en apprendre plus sur l'aspect mesuré et l'expérimentation des standards est essentielle pour contribuer à améliorer la maturité de la discipline du génie logiciel.

Le modèle de qualité de la norme ISO 9126-1 est le plus développé parmi les modèles existants dans la littérature en génie logiciel. Ce modèle est composé d'un ensemble de caractéristiques et sous-caractéristiques (le cas de qualité interne et externe) et des mesures sont proposées par les experts d'ISO dans les rapports techniques de la série ISO 9126.

Il est important de noter que la vérification des mesures d'ISO TR 9126-2 à 4 n'entre pas dans le cadre de ce travail de recherche; nous prenons donc pour acquis que :

- les mesures proposées pour chaque caractéristique permettent effectivement d'évaluer la caractéristique correspondante;
- les mesures proposées pour chaque sous-caractéristique permettent effectivement d'évaluer la sous-caractéristique correspondante.

Cependant, puisque ce modèle ISO 9126 ne fournit qu'un aspect théorique de la qualité, nous avons choisi d'utiliser la méthode Taguchi de conception de plan d'expériences qui fournit l'aspect pratique de la qualité. C'est une méthode d'expérimentation qui combine pratiques industrielles et statistiques, et qui a prouvé son efficacité dans divers domaines du secteur industriel. La méthode Taguchi permet, entre autres, la réduction du coût du produit et l'amélioration de la qualité des produits et des processus.

Par ailleurs, une contrainte majeure dans la conduite des expériences réside dans la collecte de données. En l'absence d'opportunités d'expérimentations en industrie pour ce travail de recherche, nous avons choisi d'utiliser le référentiel de données industrielles d'ISBSG parmi ceux disponibles dans la littérature vu qu'il collecte, entre autres, de l'information sur les différentes phases du cycle de vie du logiciel. Nous utiliserons dans un premier temps l'extrait de données d'ISBSG (Release 9 de 2005), de plus de 3 000 projets, disponible à l'École de technologie supérieure de l'Université du Québec pour des fins de recherche, dans ce travail de recherche.

Des travaux de recherche ont été aussi présentés dans les chapitres précédents, principalement pour le standard ISO 9126 et la méthode Taguchi. Le but était de mieux cerner les forces, les faiblesses et les limites rencontrées dans ces travaux et, d'autre part, de déterminer les points importants qui peuvent enrichir, fournir de nouvelles idées et aider à bien concevoir le cadre théorique et méthodologique de nos analyses empiriques.

Tel que mentionné, dans l'introduction du chapitre 4, la réussite d'un plan d'expériences ne se mesure pas à la complexité du modèle obtenu, mais à la satisfaction qu'il a générée chez le client (Pillet, 1997). De façon semblable, nous pouvons dire que la qualité du produit logiciel ne se mesure pas avec sa complexité, mais par sa qualité interne, sa qualité externe et sa qualité en utilisation qui visent la satisfaction des besoins des utilisateurs. La méthode Taguchi peut s'effectuer dans le cadre des caractéristiques et sous-caractéristiques du modèle de qualité interne et externe ou des caractéristiques du modèle de qualité en utilisation de la norme ISO 9126-1. Plus précisément, l'utilisation, entre autres, des diverses mesures proposées dans les rapports techniques ISO TR 9126-2 à 4 et pour lesquelles des données sont disponibles dans le référentiel d'ISBSG comme facteurs affectant la qualité du logiciel.

Étape 2 : Exploration détaillée

L'exploration détaillée est axée autour de la série des documents ISO 9126 (modèles de qualité et mesures) et ISBSG (extrait de données d'ISBSG-Release 9 de 2005 et questionnaire de collecte de données).

Cette étape comprend deux mises en correspondance (chapitre 6) :

1. Mise en correspondance (de haut niveau et détaillée) des types de qualité (interne, externe et en utilisation) de la norme internationale ISO 9126-1 dans le questionnaire de collecte de données d'ISBSG. Le but est de :
   - vérifier si le questionnaire d'ISBSG couvre les différents types de qualité d'ISO 9126-1;
   - recenser les données de qualité collectées par ISBSG et qui sont non disponibles dans l'extrait de données d'ISBSG (Release 9 de 2005) afin de faire la demande d'accès à ces données auprès de l'organisation ISBSG.

2. Mise en correspondance (de haut niveau et détaillée) des rapports techniques ISO TR 9126-2 à 4, relatifs aux mesures de qualité interne, externe et en utilisation du produit logiciel, avec le questionnaire de collecte de données d'ISBSG. Cette partie comprend trois étapes :
   - l'analyse d'ISO TR 9126–2 à 4 et du questionnaire d'ISBSG afin de recenser les caractéristiques (sous-caractéristiques) de qualité d'ISO 9126 qui sont prises en considération par ISBSG;
   - l'exploration des mesures ISO TR 9126-2 à 4 afin de faire ressortir les mesures de base similaires entre ISO 9126 et ISBSG. Ces mesures de base sont analysées ensuite à travers des exemples de mesures d'ISO 9126, essentiellement quant à l'utilisation ou non d'une définition standard de l'attribut mesuré;

- la proposition de mesures propres à ISBSG à partir des données de qualité collectées par ISBSG. Ces mesures seront utiles pour les plans d'analyses empiriques (phase 3).

Ces mises en correspondance seront utiles afin de mieux connaitre les contenus d'ISBSG (extrait de données et questionnaire de collecte de données) et de la série des documents ISO 9126 (modèles de qualité et mesures). D'autre part, ces mises en correspondance permettront de faire ressortir les données importantes à propos de (ou en relation avec) la qualité du produit logiciel et de faire une demande d'accès à ces données (non disponibles dans l'extrait de données d'ISBSG-Release 9 de 2005) à ISBSG. En effet, ISBSG fournit aux chercheurs la possibilité d'accéder aux données, gardées inconnues pour préserver l'anonymat de leurs sources, à condition qu'ils documentent leur thème de recherche et le soumettent à ISBSG pour approbation. Par la suite, ISBSG examine la demande et prépare un extrait de données pour les différentes informations qui ont fait l'objet de la requête. En pratique, ISBSG, selon ses contraintes, décide de fournir toutes les informations dés la première fois ou partiellement.

5.3.2 Phase 2 : Préparation

Cette phase consiste à préparer l'environnement des plans d'analyses empiriques (chapitre 7). Cette préparation porte essentiellement sur le nouvel extrait de données d'ISBSG de février 2006 (reçu suite à la demande d'accès aux données effectuée dans la phase 1) et comprend les étapes suivantes :

1. Le recensement des données à exploiter de l'extrait de données d'ISBSG; il s'agit des facteurs pouvant avoir un impact sur la qualité du produit logiciel. Cette étape comprend :
   - l'établissement d'un questionnaire sur les facteurs potentiels impactant la qualité du produit logiciel;

- la collecte des questionnaires remplis par quelques praticiens expérimentés dans le domaine du logiciel;
- l'analyse des résultats.

2. La préparation des données de l'extrait d'ISBSG afin de sélectionner l'échantillon de projets d'ISBSG approprié pour les plans d'analyses. Cette préparation porte sur :
   - la vérification de la qualité des données (premier niveau de préparation);
   - la vérification de la complétude des données (deuxième niveau de préparation).

3. La présentation détaillée des différentes étapes de l'analyse des résultats des plans d'analyses avec l'analyse du ratio signal-bruit de Taguchi.

5.3.3 Phase 3 : Plan d'analyse empirique

Cette phase de la méthodologie portera essentiellement sur l'adaptation de l'étape de paramétres de design (chapitre 4) de Taguchi au contexte d'analyse empirique. En effet, la méthode Taguchi de conception d'expériences est une méthode qui est généralement utilisée dans la conception de plan d'expériences *a priori* qui mène à la collecte de données, la réalisation des essais, le choix de la condition optimale et enfin l'application de cette dernière dans le processus de production.

Dans notre projet de recherche, nous allons utiliser plutôt la méthode Taguchi *a posteriori* : c'est-à-dire, l'approche sera non pas la collecte des données expérimentales, mais plutôt l'aide à l'analyse des données déjà collectées et, par la suite, la sélection des données pertinentes à chaque plan d'expériences Taguchi établi.

Il est important de noter que cette phase, plan d'analyse empirique, de la méthodologie de recherche consiste à vérifier les hypothèses des liens entre les trois types de qualité du produit logiciel de la norme ISO 9126-1. En effet, dans ce travail de recherche, il n'est

pas possible de tester tous les liens entre les différentes caractéristiques (sous-caractéristiques) des trois modèles de qualité d'ISO 9126-1. Cependant, il est possible à partir des données disponibles dans l'extrait d'ISBSG de février 2006 de déterminer les caractéristiques de qualité, de recenser les facteurs à étudier et de préparer les plans d'analyses empiriques pour chacun des volets suivants :

1. Lien entre la qualité interne et la qualité externe (chapitre 8).
2. Lien entre la qualité externe et la qualité en utilisation (chapitre 9).
3. Lien entre la qualité interne et la qualité en utilisation (chapitre 10).

Pour bien structurer et éviter les omissions qui peuvent non-intentionnellement arriver, le plan d'analyse empirique sera inspiré du cadre d'expérimentation proposé dans (Basili, Selby et Hutchens, 1986). Cette phase consiste en quatre grandes étapes dont la définition, la conception, la réalisation et la conclusion.

Étape 1 : Définition

Cette étape est importante et représente le point de départ; si elle n'est pas bien définie, elle influence les autres étapes qui suivent. Dans cette étape, il faut définir l'objectif du plan d'analyse empirique, lequel objectif dépendra du lien à vérifier entre les trois types de qualité d'ISO 9126-1.

Étape 2 : Conception

Cette étape représente la base de la réussite du plan d'analyse empirique au cours de laquelle il faut documenter chaque sous-étape, noter les obstacles rencontrés et justifier les choix effectués. Cette étape comprend les sous-étapes suivantes :

1. Transformer l'objectif en hypothèses à valider.
2. Déterminer la caractéristique de qualité ou la sortie étudiée.
3. Sélectionner les facteurs qui influencent la caractéristique de qualité étudiée. En fait, deux types de facteurs sont à déterminer : les facteurs contrôlés et, s'il y a lieu, les

facteurs bruits. Ces facteurs peuvent être quantitatifs comme ils peuvent être qualitatifs. Dans cette sous-étape, nous prendrons en considération les résultats du questionnaire sur la qualité du produit logiciel (phase 2) et les mesures propres à ISBSG identifiées dans la phase 1 de cette méthodologie.

4. Choisir les niveaux de chaque facteur sélectionné; au moins deux niveaux par facteur sont recommandés.
5. Déterminer les interactions importantes, si elles existent, qui peuvent avoir un effet sur la sortie étudiée.
6. Choisir la table orthogonale de Taguchi du plan étudié. Le choix de la table prend en considération les sous-étapes (3, 4 et 5) précédentes; il s'agit du nombre de facteurs, de niveaux et d'interactions.
7. Choisir le plan produit de Taguchi si les deux types de facteurs (contrôlés et bruits) sont sélectionnés.
8. Positionner les facteurs dans la table orthogonale de Taguchi pour constituer la matrice d'expériences du plan étudié (i.e., liste des essais à mener).

Pour chaque plan d'analyse conçu, un échantillon de projets d'ISBSG du plan doit être identifié (à partir de l'étape 2 de la phase 2) en fonction des facteurs et de la caractéristique de qualité du plan étudié.

L'étape de conception de plan d'analyse empirique peut se faire en collaboration avec les personnes ayant des compétences et des connaissances (théoriques ou pratiques) dans le domaine pour dresser, s'il y a lieu, le diagramme de causes-effets. Dans cette recherche, ce diagramme sera établi à base des résultats du questionnaire sur la qualité du produit logiciel (étape 1 de la phase 2) et de la disponibilité des données dans l'extrait de données d'ISBSG de février 2006.

Étape 3 : Réalisation

Cette étape porte essentiellement sur l'exécution des essais de la matrice d'expériences et sur l'analyse et l'interprétation des résultats :

1. Exécution : elle consiste à sélectionner à partir de l'échantillon de projets d'ISBSG du plan étudié les projets répondant aux différentes conditions des essais de la matrice d'expériences. Une des bonnes pratiques est d'avoir plusieurs répétitions pour chaque essai.
2. Analyse et interprétation : l'analyse des résultats du plan d'analyse empirique peut se faire par : l'analyse standard (Level average analysis) ou l'analyse du ratio signal-bruit (Signal-to-noise analysis). L'analyse comprend les points suivants :
   - le calcul de la moyenne des réponses/du ratio signal-bruit;
   - le calcul des effets des facteurs;
   - la détermination de la condition optimale;
   - l'analyse de la variance (ANOVA);
   - le calcul de l'équation de prédiction à la condition optimale;
   - le test de confirmation.

   Ces différents points ainsi que les calculs nécessaires sont décrits en détail dans le chapitre 7 de cette thèse.

À la suite de l'analyse, il est possible de confirmer ou de rejeter l'hypothèse formulée dans la deuxième étape par l'interprétation des résultats obtenus. L'interprétation des résultats représente la phase la plus critique et elle doit être soigneusement menée pour éviter des conclusions inadéquates.

Étape 4 : Conclusion

À cette étape, nous avons une idée sur l'influence des facteurs étudiés dans le plan d'analyse empirique sur la caractéristique de qualité fixée au début.

L'ensemble des phases de cette méthodologie et les outils utilisés sont schématisés dans les Figures 5.1 et 5.2.

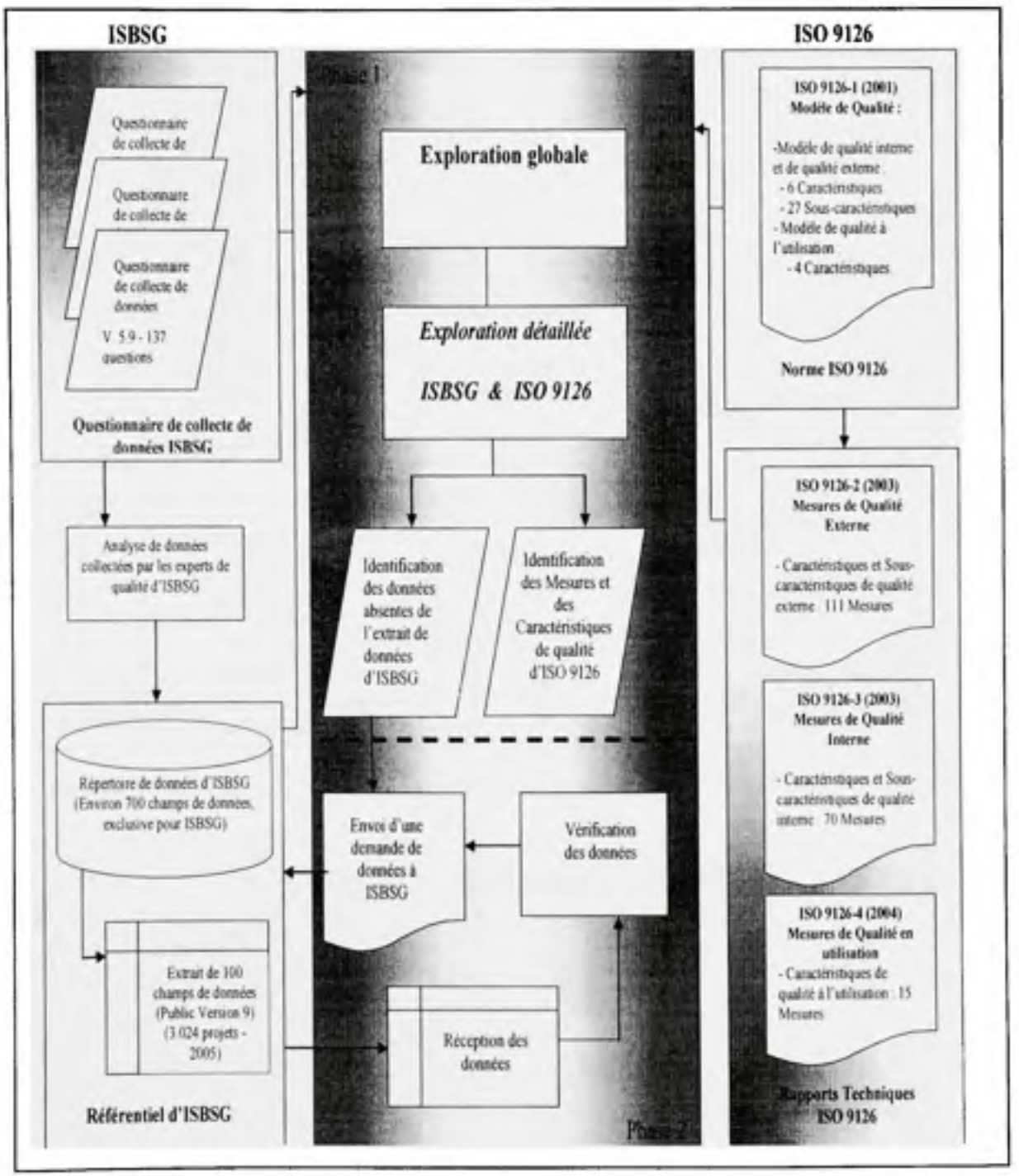

Figure 5.1 ***Méthodologie : phases 1 et 2.***

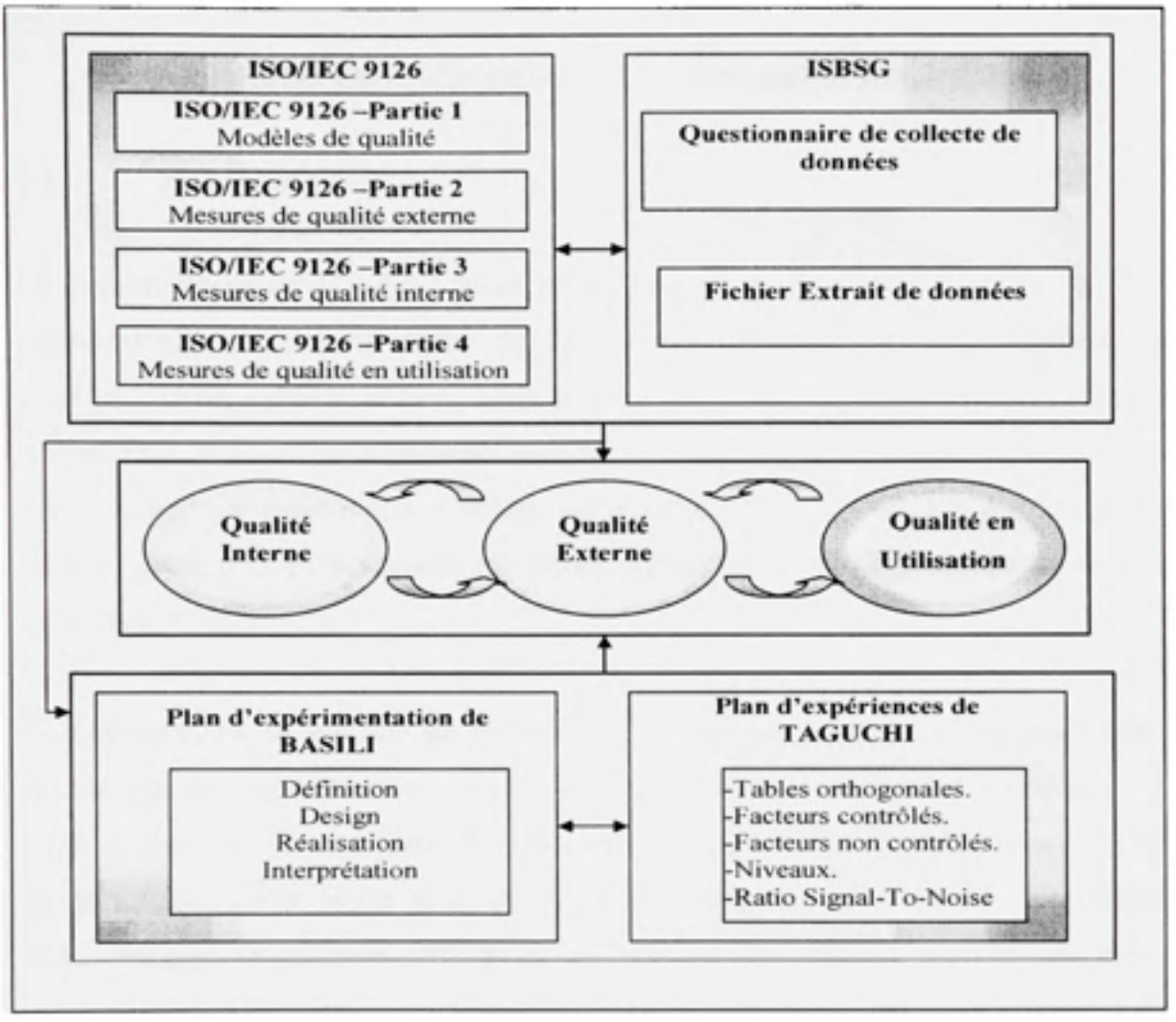

Figure 5.2 ***Méthodologie : phase 3.***

CHAPITRE 6

MISES EN CORRESPONDANCE ENTRE ISO 9126 ET ISBSG

6.1 Introduction

Ce chapitre documente l'étape 2 (exploration détaillée) de la phase 1 (exploration) de la méthodologie de recherche établie dans le chapitre 5. La première section présente la mise en correspondance entre le questionnaire de collecte de données d'ISBSG et la norme ISO 9126-1. La deuxième section présente la mise en correspondance du questionnaire par rapport aux rapports techniques ISO TR 9126-2 à 4 pour la qualité interne, externe et en utilisation du produit logiciel. La dernière section présente le sommaire de ce chapitre.

Le questionnaire de collecte de données d'ISBSG rassemble de l'information liée à l'entité « projet logiciel » avec ses diverses caractéristiques tandis que le standard ISO 9126 se concentre sur une partie de cette entité; il s'agit de la qualité du produit logiciel durant tout son cycle de vie, et ce, de sa spécification jusqu'à sa livraison à l'utilisateur final. Puisque le standard ISO 9126 est pris comme outil d'analyse, les études entreprises dans ce chapitre sont limitées à l'entité « produit logiciel » à l'intérieur des projets logiciels d'ISBSG.

Comme nous avons déjà mentionné dans le chapitre 3, les différents questionnaires de collecte de données d'ISBSG ont la même structure avec une légère différence dans la section « Taille fonctionnelle ». Dans ce travail de recherche nous allons utiliser le questionnaire COSMIC-FFP (à titre d'exemple) de collecte de données d'ISBSG.

6.2 Mise en correspondance : Modèles de qualité d'ISO 9126 et questionnaire d'ISBSG

L'objectif de cette section est de vérifier si les données collectées par le questionnaire d'ISBSG s'alignent avec les différents types de qualité de la norme ISO 9126-1. Pour ce faire, nous effectuons tout d'abord une mise en correspondance de haut niveau, suivi d'une mise en correspondance détaillée. Un sommaire de la mise en correspondance est présenté ensuite, suivi du complément de données à demander à ISBSG. Nous terminons cette section avec une discussion.

6.2.1 Mise en correspondance de haut niveau

Le résultat de notre analyse de l'alignement des différentes phases du questionnaire d'ISBSG par rapport aux modèles de qualité d'ISO 9126-1 est présenté dans le Tableau 6.1.

Tableau 6.1

Alignement du questionnaire d'ISBSG aux modèles d'ISO 9126

ISBSG	**ISO 9126**		
	Qualité Interne	**Qualité Externe**	**Qualité en Utilisation**
Questionnaire de collecte de données d'ISBSG	• Spécification • Design • Construction ou Programmation (Revue/ Inspection)	• Construction ou Programmation (Test unitaire) • Test • Implémentation ou Installation • Fin du projet ▪ Information générale	• Fin du projet ▪ Sondage satisfaction de l'utilisateur

Il est important de noter, en particulier, l'existence d'une couverture globale des différents types de qualité dans le questionnaire de collecte de données d'ISBSG.

La mise en correspondance détaillée des informations relatives à la qualité du questionnaire d'ISBSG avec chacun des trois modèles de qualité d'ISO 9126 est présentée dans les sous-sections suivantes.

6.2.2 Mise en correspondance détaillée

6.2.2.1 Qualité interne

Le modèle de qualité interne d'ISO 9126 spécifie que la qualité interne du produit logiciel peut être évaluée durant les phases de spécification, de design et de codage, et porte sur l'aspect statique du produit logiciel, c'est-à-dire sans exécution de ce dernier. Le questionnaire d'ISBSG collecte des informations durant les phases de planification, de spécification, de design et de construction (ou programmation) du projet logiciel (Tableau 6.2).

La phase de planification « covers both high level project planning and preliminary requirements analysis. It focuses on objectives, stakeholders, risks, budgets and schedules » (ISBSG, 2006c, p. 8). L'équipe de développement :

- « works with the customer to identify and document what functionality, what interfaces, what quality is required » (ISBSG, 2006c, p. 10) : la phase de specification;
- « creates a general, high-level design of the software structure, then possibly a detailed design » (ISBSG, 2006c, p. 11) : la phase de design;
- « performs the programming and unit testing that produces new software or changes existing software » (ISBSG, 2006c, p. 12) : au cours de la phase de construction ou de programmation.

Tableau 6.2

Qualité interne : Liste des informations collectées par ISBSG

(ISBSG, 2006c)

Planning	Specification	Design	Build or Programming
Project objectives	Documents or other work products produced during this phase	Documents or other work products produced during this phase	Source code or other code produced or modified during this phase
Documents or other work products produced during this phase	Size of any functional model created during the specification activity	Techniques used to design the project	Activities which occurred during the project (unit testing, software or system integration, etc.)
Initial measure of the project's functional size made in project planning	Techniques used during the specification of the project	**-Number of defects recorded during design** **-Resolution/Rework effort**	**-Number of defects (minor, major and extreme or total) recorded and resolved during this activity** **- Resolution/Rework effort**
-Estimate of total project effort made in project planning -Methods used to estimate the project effort	**-Number of defects recorded in the documents and other work products of this phase** **-Resolution/rework effort**	**Number of changes raised during design**	**Number of changes raised during build**
-Estimated project completion date set in project planning -Methods used to set the project completion	Functional size measured after the specification activity	Functional size measured after completion of the design	Duration of the build activity
-Estimate of total project cost made in project planning -Methods used to estimate the project cost	Duration of the specification activity	Duration of the design activity	
Size of any preliminary functional model created during this phase			
Duration of project planning			

Des informations collectées pour ces différentes phases, certaines peuvent être utilisées pour l'évaluation de la qualité du produit logiciel du point de vue de la qualité interne : à titre d'exemple, le nombre de défauts détectés et le nombre de requêtes de changements effectuées durant ces différentes phases du cycle de vie du logiciel (en caractères gras dans le Tableau 6.2).

6.2.2.2 Qualité externe

Le modèle de qualité externe d'ISO 9126 spécifie que la qualité externe du produit logiciel peut être évaluée durant les phases de test et d'opération du logiciel, et porte sur l'aspect dynamique du produit logiciel, c'est-à-dire lors de son exécution. Le Tableau 6.3 montre les informations collectées par ISBSG pour la phase de test, la phase d'implémentation ou d'installation et la phase de fin du projet du cycle de vie du logiciel.

La phase de test est définie comme suit dans (ISBSG, 2006c, p. 13) : « planning and performing the various levels of testing on the software by people who may be independent of the developers ». La phase d'implémentation ou d'installation est définie ainsi dans (ISBSG, 2006c, p. 14) : « preparing for the installation of the software by/for customer or end user personnel, then working with them for installation, user documentation and training ». La phase de fin du projet représente la phase du début d'opération du logiciel par l'utilisateur final.

Des diverses données collectées durant les phases de test, d'implémentation ou d'installation et de fin du projet, certaines d'entre elles (comme le nombre de défauts et le nombre de requêtes de changements) peuvent être utiles dans l'évaluation de la qualité externe du produit logiciel.

Tableau 6.3

Qualité externe : Liste des informations collectées par ISBSG

(ISBSG, 2006c)

Test	Implementation or Installation	Project Completion
Documents or other work products produced during test planning or performance	Documents or work products produced during the preparation for, or performance of, the implementation activity	Total duration of the project
Activities occurring during software testing	Number of distinct releases/versions of the software delivered to the customer during the project	Total inactivity on the project: duration
-Number of defects (minor, major, extreme or total) recorded during this activity -Resolution/rework effort	Activities occurring during software implementation	**-Number of defects recorded during the first month of the software's operation (minor, major, extreme or total)**
Number of changes raised during testing	**-Number of defects (minor, major, extreme or total) recorded during this activity -Resolution/rework effort**	-Lines of code generated by this project. -Percentage of these lines of code that are not program statements
Duration of the testing activity	**Number of changes raised during implementation**	
	Functional size measured after completion of implementation	
	Duration of the implementation activity	

6.2.2.3 Qualité en utilisation

Le modèle de qualité ISO 9126 spécifie que la qualité en utilisation représente la qualité du point de vue utilisateur, laquelle qualité est évaluée lors de l'utilisation du logiciel dans un environnement particulier. Le Tableau 6.4 regroupe des informations de la dernière section du questionnaire d'ISBSG : Fin du projet.

L'information collectée une fois le projet terminé et en utilisant un sondage sur la satisfaction de l'utilisateur représente un moyen d'évaluation de la qualité en utilisation du produit logiciel : à titre d'exemple, la satisfaction de l'utilisateur vis-à-vis du produit logiciel est évaluée en tenant compte des réponses aux questions formulées dans le sondage (Tableau 6.4).

Tableau 6.4

Qualité en utilisation : Liste des informations collectées par ISBSG (ISBSG, 2006c)

Project completion - User Satisfaction Survey
Did the project meet stated objectives?
Did the software meet business requirements?
Quality expectations for the software? **Quality expectations for user documentation?**
Ease-of-use requirements for the software? **Was sufficient training or explanation given?**
Schedule for planning & specification? **Schedule for design, build, test & implement?**

6.2.3 Sommaire de la mise en correspondance

En résumé, il est à remarquer d'une part que le questionnaire d'ISBSG collecte des informations pour toutes les phases du cycle de vie du logiciel et, d'autre part, que plusieurs questions (et réponses) du questionnaire d'ISBSG peuvent fournir de l'information pour les trois types de qualité du produit logiciel d'ISO 9126 (qualité interne, externe et en utilisation). Plus spécifiquement (Tableau 6.5) :

- la section « Processus du projet » peut fournir de l'information pour deux des trois types de qualité : interne et externe;
- la section « Fin du projet » peut fournir de l'information pour deux des trois types de qualité : externe et en utilisation.

Tableau 6.5

Les types de qualité d'ISO 9126 dans le questionnaire d'ISBSG

Questionnaire d'ISBSG-Sections / Types de qualité — Sous-sections		**Qualité Interne**	**Qualité Externe**	**Qualité en Utilisation**
Information du présentateur	*Information générale*			
Processus du projet	Caractéristiques du projet			
	Planification			
	Spécification	X		
	Design	X		
	Construction ou programmation (Revue/Inspection - Test unitaire)	X	X	
	Test		X	
	Implémentation ou installation		X	
	Contrôle et gestion du projet			
Technologie	*Information générale*			
Personnes et Effort du travail	Équipe de développement			
	Clients et utilisateurs finaux			
	IT Opérations			
	Validation de l'effort du travail			
Produit	*Information générale*			
Taille fonctionnelle du projet	Taille du logiciel : nouveau développent ou redéveloppent	X	X	
	Taille du logiciel : amélioration		X	
	Contexte de la mesure de taille fonctionnelle			
	Expérience du compteur de taille fonctionnelle			
Fin du projet	*Information générale*		X	
	Sondage satisfaction de l'utilisateur			X
	Coûts du projet			
	Validation du coût			

Les autres sections, « Technologie », « Personne et Effort du travail » et « Produit » ne peuvent fournir de l'information pour aucun des trois modèles de qualité d'ISO 9126 et

les informations de la section « Taille fonctionnelle du logiciel » sont utiles pour des objectifs de normalisation lors du calcul des ratios en relation avec la qualité.

6.2.4 Accès à des données de recherche

Le laboratoire de recherche en génie logiciel (GÉLOG) de l'ÉTS a acheté en 2005 une copie de l'extrait des données du référentiel d'ISBSG, soit la version « Release 9 » de 2005; il faut rappeler que ces extraits de données ne contiennent que le sous-ensemble de données qu'ISBSG rend disponible à l'industrie. En effet, ISBSG ne rend jamais disponible l'ensemble de ses données, en particulier pour préserver la confidentialité de l'information; même pour des fins de recherche ISBSG ne rend jamais disponible la totalité de ses données mais uniquement le sous-ensemble pertinent aux objectifs de recherche spécifiques à chaque chercheur.

Dans cette section, le questionnaire de collecte de données d'ISBSG est utilisé pour analyser si l'extrait de donnée d'ISBSG (Release 9 de 2005) disponible à l'ÉTS contient les données appropriées pour l'utilisation des modèles ISO de la qualité du logiciel. Il est à souligner que dans cette section, l'analyse porte maintenant non plus sur les questions, mais plutôt sur les réponses disponibles dans l'extrait de données d'ISBSG, soit les « données » elles-mêmes.

D'une part, de la mise en correspondance effectuée entre le questionnaire COSMIC-FFP de collecte de données d'ISBSG et les modèles de qualité d'ISO 9126-1 (section 6.2.2), nous avons rassemblé les différentes données directement reliées à la qualité. Le Tableau 6.6 présente ces données de qualité parmi celles collectées dans la catégorie « Processus du projet » et la catégorie « Fin du projet ». Dans ce tableau, la première colonne correspond à la catégorie dans le questionnaire de collecte de données d'ISBSG, la deuxième à ses sous-catégories et la troisième aux données collectées. La colonne de

droite (dernière colonne) liste les numéros des questions correspondantes dans le questionnaire d'ISBSG.

Tableau 6.6

Données d'ISBSG relatives à la qualité
(ISBSG, 2006c)

Categories	Subcategories	Data Collected	ISBSG Questionnaire
Project Process	Specification	-Number of defects recorded in the documents and other work products of this phase -Resolution/rework effort	(Question: 24)
	Design	-Number of defects recorded during design -Resolution/Rework effort	(Question: 29)
		Number of change requests made during design	(Question: 30)
	Build or Programming	-Number of defects (minor, major, extreme or total) recorded and resolved during this activity - Resolution/Rework effort	(Question:35)
		Number of change requests made during build	(Question:36)
	Test	-Number of defects (minor, major, extreme, or total) recorded during this activity -Resolution/rework effort	(Question:40)
		Number of change requests made during testing	(Question:41)
	Implementation or Installation	-Number of defects (minor, major and extreme or total) recorded during this activity -Resolution/rework effort	(Question:46)
		Number of change requests made during implementation	(Question:47)
Project Completion	*General Information*	-Number of defects recorded during the first month of the software's operation (minor, major, extreme or total)	(Question:126)
		-The lines of code generated by this project. - The percentage of these lines of code that are not program statements	(Question:127)
	User Satisfaction Survey	-Did the project meet stated objectives? -Did the software meet business requirements? -Quality expectations for the software? -Quality expectations for the user documentation? -Ease-of-use requirements for the software? -Was sufficient training or explanation given? -Schedule for planning & specification? -Schedule for design, build, test & implement?	(Question:128)

Du Tableau 6.6, pour la catégorie « Processus du projet », il est à observer que :

- L'information relative au « nombre de défauts enregistrés » est présente dans la plupart des phases (question nos. 24, 29, 35, 40 et 46). Plus encore, pour chacune des phases, construction ou programmation, test, implémentation ou installation, le nombre de défauts est catégorisé selon les trois niveaux suivants (ISBSG, 2006c, p. 12) :
  - « Minor defect: does not make the software unusable in any way »;
  - « Major defect: causes part of the software to become unusable »;
  - « Extreme defect: failure causing the software to become totally unusable ».

- L'information relative au « nombre de requêtes de changements effectuées » est aussi collectée pour la plupart des phases du processus du projet (question nos. 30, 36, 41 et 47); c'est-à-dire de la phase de design à la phase d'implémentation ou installation.

Pour la catégorie « Fin du projet », il est à constater que l'information collectée pour les défauts (question no. 126) est classifiée avec les mêmes niveaux de sévérité notés auparavant, et qu'ils portent sur la période du premier mois d'opération du logiciel par ses utilisateurs. En outre, la sous-catégorie relative au « sondage de la satisfaction de l'utilisateur » (question no. 128) collecte de l'information à propos du niveau de satisfaction telle qu'elle est perçue par les utilisateurs finaux.

D'autre part, l'extrait de données d'ISBSG (Release 9 de 2005), introduit dans le chapitre 3, représente un sous-ensemble de données collectées par ISBSG rendu public aux chercheurs pour des fins de recherche. Il est important de rappeler que cet extrait contient les champs de données de qualité relatives au nombre de défauts détectés durant le premier mois d'opération du logiciel, lesquels champs de données correspondent à ceux de la section « Qualité » de la structure détaillée de l'extrait de données d'ISBSG (voir Annexe III).

Toutefois, la disponibilité de ces champs de données dans l'extrait d'ISBSG (Release 9 de 2005) n'implique pas automatiquement que les organisations ont été à même de fournir toutes ces données; en pratique, il y a beaucoup de champs de données qui sont vides dans cet extrait de données. L'Annexe IV présente une analyse de la disponibilité des données pour les champs de données de la section « Qualité ». De plus, dans l'Annexe IV, des exemples d'utilisation de ces données sont illustrés, avec par exemple des analyses de la distribution des défauts par les types de défauts (mineur, majeur, extrême et total) ainsi que des analyses de densité de défauts.

La préparation de données pour ces analyses à l'Annexe IV a indiqué que : alors que le référentiel d'ISBSG (et l'extrait de données) contient des données pour environ 3 024 projets, l'industrie avait fourni des données concernant la qualité pour seulement 361 d'entre eux : c'est-à-dire que seulement 12% de ces organisations fournissant des données à ISBSG possèdent des informations sur des défauts de projet logiciel détectés pendant le premier mois de son opération suivant l'achèvement du projet.

Cet extrait de données d'ISBSG ne contient cependant pas l'information sur les données au sujet des défauts sur les autres phases du cycle de vie du logiciel, de sa spécification jusqu'à son achèvement. Ainsi, l'extrait de données d'ISBSG (Release 9 de 2005) ne contient qu'une partie de l'information sur les défauts (question no 126) alors que les autres informations ne sont pas disponibles dans cet extrait (Tableau 6.6) :

- nombre de défauts détectés durant les phases : spécification, design, programmation, test et implémentation;
- nombre de requêtes de changements effectuées durant les phases : design, programmation, test et implémentation;
- sondage sur la satisfaction des utilisateurs.

De telles informations sont importantes pour poursuivre l'objectif de cette thèse : la vérification des liens entre la qualité interne, la qualité externe et la qualité en utilisation.

Donc, pour atteindre cet objectif et pour effectuer nos plans d'analyses empiriques, il a fallu soumettre auprès de l'organisation ISBSG une demande d'accès aux données détaillées requises pour notre recherche.

6.2.5 Discussion

Cette section a étudié jusqu'à quel point le référentiel d'ISBSG peut être utile pour la qualité du produit logiciel sur la base d'ISO 9126. Plus spécifiquement, nous avons étudié la disponibilité ainsi que la correspondance des données de qualité dans le questionnaire d'ISBSG en utilisant la série ISO 9126 sur la qualité du produit logiciel prise comme base de référence pour cette analyse.

La mise en correspondance du questionnaire d'ISBSG aux trois modèles d'ISO 9126 de la qualité du logiciel (interne, externe et en utilisation) a été réalisée en utilisant le questionnaire COSMIC-FFP (à titre d'exemple) de collecte de données d'ISBSG. L'analyse des résultats de cette mise en correspondance a indiqué que, en théorie, le référentiel d'ISBSG contient plusieurs champs nécessaires pour les trois types des modèles d'ISO de la qualité du logiciel. Cependant, dans la pratique, ISBSG ne fournit pas le contenu total de son référentiel : seulement un sous-ensemble est rendu disponible à un coût raisonnable à l'industrie et aux chercheurs dans un extrait de données (Release 9 de 2005). L'analyse de cet extrait a indiqué par la suite qu'un nombre minime de données relatives à la qualité est offert automatiquement par ISBSG dans cet extrait et que cet extrait ne comprend qu'une partie des types de qualité d'ISO 9126.

Par conséquent, cette disponibilité limitée de données de qualité pour les différentes phases du cycle de vie du logiciel nous mène à conclure que l'extrait de données d'ISBSG (Release 9 de 2005) dans son état actuel ne peut être efficacement utilisé pour l'étalonnage (benchmarking) des caractéristiques de qualité du produit logiciel sur la base d'ISO 9126.

Néanmoins, comme il existe des données additionnelles sur la qualité (Tableau 6.6) dans le référentiel d'ISBSG, les chercheurs peuvent soumettre à ISBSG une demande spécifique d'accès à ces données, moyennement des frais supplémentaires. Soumettre une telle demande exige du demandeur d'avoir beaucoup de connaissances à propos des données d'ISBSG ainsi qu'une très bonne compréhension de la façon dont ces données s'alignent aux différents types de qualité d'ISO 9126.

ISBSG a indiqué que parmi les facteurs considérés par le conseil d'ISBSG pour l'accès à des champs supplémentaires de données, la conservation de l'anonymat des sources de données est le facteur critique; dans la pratique, ISBSG vérifie si la demande d'accès ne requière pas de l'information sur des champs avec des points de repère qui permettraient de retracer des informations pertinentes à une seule organisation.

ISBSG fournit à des chercheurs la possibilité d'accéder à ces données non disponibles au public à condition que les chercheurs soumettent la documentation de leur protocole de recherche. Après analyse, ISBSG prépare un nouvel extrait de données pour les champs requis pour une question spécifique de recherche. Il est à noter que l'organisation ISBSG ne fournit jamais un supplément de données pour des plans de recherche non documentés.

L'ensemble des mises en correspondance (de haut niveau et détaillée) documentées dans cette section est bien entendu nécessaire pour notre recherche. Ces résultats peuvent également être utiles pour les publics suivants :

1. Pour l'industrie du logiciel en général : les organisations peuvent comparer leurs données de qualité de logiciel par rapport au référentiel international d'ISBSG.
2. Pour les chercheurs : pour la préparation de leurs plans de recherche détaillés afin de demander à ISBSG les informations concernant la qualité du logiciel. Par exemple : pour la vérification des relations entre les trois types de la qualité de la norme ISO 9126-1.

3. Pour l'industrie du logiciel : pour analyser et mettre en application des modèles ISO de la qualité du logiciel. À titre d'exemple, les Tableaux 6.1 à 6.5 fournissent des conseils.
4. Pour l'organisation ISBSG elle-même : ISBSG pourrait utiliser nos résultats d'analyse pour améliorer l'alignement de leurs standards de collecte de données par rapport aux modèles ISO de la qualité du logiciel.

Cette section a été limitée à la mise en correspondance d'ISO 9126-1 et du questionnaire d'ISBSG. La section suivante porte sur l'alignement des informations relatives à la qualité de logiciel d'ISBSG par rapport au plus de 200 mesures proposées dans les rapports techniques 2 à 4 de la série d'ISO 9126.

6.3 Mise en correspondance : ISO TR 9126-2 à 4 et le questionnaire ISBSG

Dans cette section, nous effectuons une analyse du questionnaire d'ISBSG par rapport aux rapports techniques ISO TR 9126-2 à 4. L'objectif de cette section est de répondre à la question suivante : quelles sont les caractéristiques et sous-caractéristiques des modèles de qualité (interne, externe et en utilisation) d'ISO qui sont prises en considération par le questionnaire de collecte de données d'ISBSG ?

En outre, l'intérêt est d'explorer les mesures proposées dans la série des documents ISO TR 9126-2 à 4 afin de déterminer les mesures de qualité interne, de qualité externe et de qualité en utilisation qui sont couvertes par le questionnaire d'ISBSG et pour lesquelles des données seraient disponibles dans leur base de données. L'identification de la disponibilité de telles mesures et données correspondantes est nécessaire pour la conception de nos plans d'analyses empiriques en se basant sur la méthode Taguchi.

Pour ce faire, nous procédons à la mise en correspondance entre le questionnaire d'ISBSG et les rapports techniques ISO TR 9126-2 à 4 dans la section 6.3.1, à

l'exploration des mesures d'ISO TR 9126-2 à 4 dans la section 6.3.2 et à la présentation des mesures de qualité du produit logiciel propres à ISBSG dans la dernière section.

6.3.1 Mise en correspondance d'ISO TR 9126-2 à 4 et du questionnaire ISBSG

Dans cette section nous procédons tout d'abord à la mise en correspondance de haut niveau entre le questionnaire d'ISBSG et les rapports techniques ISO TR 9126-2 à 4. Par la suite, nous effectuons une mise en correspondance détaillée de chaque rapport technique d'ISO 9126 avec le questionnaire d'ISBSG.

6.3.1.1 Mise en correspondance de haut niveau

La série ISO 9126 ainsi que l'organisation ISBSG offrent deux moyens différents de collecte d'informations (mesures pour ISO 9126 et champs de données pour ISBSG) relatives à la qualité du produit logiciel.

La documentation du questionnaire de collecte de données (ISBSG, 2006c) suit la structure habituelle des questionnaires : en premier lieu une description sommaire et générale de la « phase du cycle de vie du logiciel » ainsi que son objectif, suivi d'une liste de questions relatives à la phase. Chaque question est structurée de la façon suivante :

- un « énoncé de la question » exprimant l'objectif ou l'information souhaitée;
- un choix parmi une ou plusieurs « réponses » à partir desquelles des mesures peuvent être déduites;
- une « note » soit de description ou d'explication ou comportant des remarques sur l'intérêt de l'information faisant l'objet de la question.

Comme déjà mentionné dans le chapitre 2, les mesures d'ISO 9126 sont toutes décrites selon une structure de documentation identique (en fonction de dix termes) dans la clause 7 des rapports techniques d'ISO 9126 (2003a; 2003b; 2004), à savoir :

- « nom de la mesure (métrique) »;
- « objectif de la mesure »;
- « méthode d'application de la mesure »;
- « mesure, formules et comptages des éléments de données »;
- « interprétation de la valeur mesurée »;
- « types d'échelle de la mesure »;
- « types de mesures »;
- « référence dans ISO 12207 SLCP (Software Life Cycle Processes) »;
- « entrée au mesurage »;
- « public cible ».

De l'analyse de haut niveau de la structure des rapports techniques ISO TR 9126-2 à 4 et de la structure du questionnaire d'ISBSG, nous n'avons identifié aucune correspondance directe entre ces deux structures. Toutefois, des points communs ont été observés, à savoir :

- l'« énoncé de la question » qui traduit l'objectif ou l'information souhaitée dans le questionnaire d'ISBSG peut correspondre à « l'objectif de la mesure » dans ISO 9126 exprimé lui-même sous forme de question;
- les « réponses » relatives à certaines questions d'ISBSG permettent d'en déduire des mesures de base. De telles réponses peuvent correspondre, même partiellement, au terme « mesure, formules et comptage des éléments de données » d'ISO 9126 qui utilise des mesures dérivées (deux ou plusieurs mesures de base);
- les questions d'ISBSG sont groupées par « phase de cycle de vie du logiciel », à savoir : planification, spécification, design, programmation, test, installation et fin du projet. Ce cycle de vie du logiciel d'ISBSG peut correspondre au terme « Référence dans ISO 12207 SLCP (Software Life Cycle Processes) » d'ISO 9126. En effet,

même si ces deux cycles de vie sont différents, une mise en correspondance est possible.

6.3.1.2 Mise en correspondance détaillée

La mise en correspondance détaillée du questionnaire d'ISBSG selon les rapports ISO TR 9126-2 à 4 est effectuée essentiellement en fonction de l'objectif de la mesure d'ISO et de son correspondant, soit la question d'ISBSG. Le choix de l'objectif de mesure comme critère d'analyse réside dans le fait que dans ISO 9126, un objectif de mesure peut correspondre à différents noms de mesures et différentes formules de mesures.

Les Tableaux 6.7, 6.8 et 6.9 résument les résultats de l'analyse détaillée effectuée. Seules les caractéristiques et/ou sous-caractéristiques d'ISO 9126 ayant des données disponibles dans ISBSG pour les évaluer sont présentées.

Le résultat de la mise en correspondance des données du questionnaire d'ISBSG par rapport à ISO TR 9126-3, relatif aux mesures de qualité interne, est présenté dans le Tableau 6.7. Il est à noter que certaines données provenant du questionnaire d'ISBSG permettent d'évaluer les caractéristiques de qualité relatives à la capacité fonctionnelle, la fiabilité et la maintenabilité du modèle de qualité interne d'ISO 9126, et ce, respectivement à travers les sous-caractéristiques de qualité suivantes : l'aptitude, la maturité et la facilité de modification.

Tableau 6.7

Mise en correspondance d'ISO TR 9126-3 et du questionnaire d'ISBSG

ISO TR 9126-3 Mesures de qualité interne (ISO, 2003b)			**ISBSG Questionnaire COSMIC (ISBSG, 2006c)**
Characteristics	**Sub-characteristics**	**Purpose of the measure**	**Question number (Qn)**
Functionality	Suitability	How stable is the functional specification during the development life cycle?	What was the number of specification changes raised during: -Design (Qn:30) -Build (Qn:36) -Test (Qn:41) -Implementation (Qn:47)
Reliability	Maturity	How many faults were detected in the reviewed product?	What was the number of defects recorded during Specification (Qn:24)
			What was the number of defects recorded during Design (Qn:29)
			What was the number of defects recorded during Build (code review) (Qn:35) (Note 1)
Maintainability	Changeability	(Note 2)	Resolution /Rework effort for Specification defects (Qn:24)
			Resolution /Rework effort for Design defects (Qn:29)
			Resolution /Rework effort for Build defects (code review) (Qn:35) (Note 3)

Voici quelques observations associées à quelques cases des Tableaux 6.7 et 6.8 :

- Note 1 : l'information concernant le « nombre de défauts détectés durant la phase de programmation » d'ISBSG regroupe aussi bien les défauts collectés par revue ou inspection du code (qualité interne) que les défauts collectés en exécutant le logiciel lors du test unitaire (qualité externe). De plus, le questionnaire de collecte de données (ISBSG, 2006c), pour la phase de programmation, indique le nombre de

défauts enregistrés et résolus. Cependant, dans le glossaire des termes du questionnaire (ISBSG, 2006b), il est mentionné que seulement les défauts détectés sont collectés. Ainsi, l'information collectée ne concerne que les défauts détectés.

- Note 2 : dans ISO TR 9126-3 (2003b, p. 26), les mesures de qualité interne relatives à la sous-caractéristique « Facilité de modification » indiquent : « a set of attributes for predicting the maintainer's or user's spent effort when trying to implement a specified modification in the software product ». Cependant, ISO TR 9126-3 ne fournit aucune mesure de l'effort dépensé pour effectuer la modification.
- Note3 : la même remarque que la « Note1 » pour l'information « Resolution/Rework effort » pour la phase de programmation d'ISBSG.

La mise en correspondance des données du questionnaire d'ISBSG par rapport à ISO TR 9126-2, relatif aux mesures de qualité externe, est présentée dans le Tableau 6.8. Il est à remarquer que le questionnaire d'ISBSG dispose de données utiles pour évaluer les caractéristiques de qualité relatives à la capacité fonctionnelle, la fiabilité et la maintenabilité du modèle de qualité externe d'ISO 9126, respectivement pour les sous-caractéristiques de qualité suivantes : l'aptitude, la maturité et la facilité de modification.

Tableau 6.8

Mise en correspondance d'ISO TR 9126-2 et du questionnaire d'ISBSG

ISO TR 9126-2 Mesures de qualité externe (ISO, 2003a)			**ISBSG Questionnaire COSMIC (ISBSG, 2006c)**
Characteristics	**Sub-characteristics**	**Purpose of the measure**	**Question number (Qn)**
Functionality	Suitability	How stable is the functional specification after entering operation?	Enhancement software size (Qn:104)
			Size of new or added functionality (Qn:101)
			Size of changed functionality (Qn:102)
			Size of deleted functionality (Qn:103)
Reliability	Maturity	How many faults were detected during defined trial period	What was the number of defects recorded during Build (unit testing) (Qn:35) (Note1)
			What was the number of defects recorded during Test (Qn:40)
			What was the number of defects recorded during Implementation (Qn:46)
			What was the number of defects recorded during the first month of the software's operation (Qn:126)
Maintainability	Changeability	Can the maintainer easily change the software to resolve problem?	Resolution /Rework effort for Build (unit testing) defects (Qn:35) (Note 3)
			Resolution /Rework effort for Test defects (Qn:40)
			Resolution /Rework effort for Implementation (Qn:46)

Le résultat de la mise en correspondance des données du questionnaire d'ISBSG par rapport à ISO TR 9126-4, relatif aux mesures de qualité en utilisation, est présenté dans le Tableau 6.9. Il apparaît clairement que les données provenant du sondage du

questionnaire d'ISBSG permettent d'évaluer la caractéristique satisfaction du modèle de qualité en utilisation d'ISO 9126.

Tableau 6.9

Mise en correspondance d'ISO TR 9126-4 et du questionnaire d'ISBSG

ISO TR 9126-4 Mesures de qualité en utilisation (ISO, 2004)		**ISBSG Questionnaire COSMIC (ISBSG, 2006c)**
Characteristics	**Purpose of the measure**	**Question number (Qn)**
Satisfaction	How satisfied is the user?	User Satisfaction survey (Qn:128): -Did the project meet stated objectives? -Did the project meet business requirements? -Quality expectations for the software? -Quality expectations for user documentation? -Ease-of-use requirements for the software? -Was sufficient training or explanation given? -Schedule for planning & specification? -Schedule for design, build, test & implement?
	How satisfied is the user with specific software features?	

6.3.1.3 Synthèse

De la mise en correspondance détaillée du questionnaire d'ISBSG par rapport à ISO TR 9126-2 à 4, il en ressort que l'organisation ISBSG, à travers les données de qualité collectées par le questionnaire, offre le moyen d'évaluer quelques caractéristiques et/ou sous-caractéristiques des rapports techniques d'ISO TR 9126-2 à 4. La Figure 6.1 regroupe ces caractéristiques et ces sous-caractéristiques.

Les autres caractéristiques et sous-caractéristiques des modèles de qualité d'ISO 9126 (Figures 2.2 et 2.3 du chapitre 2) ne sont mentionnées ni explicitement ni implicitement dans le questionnaire d'ISBSG.

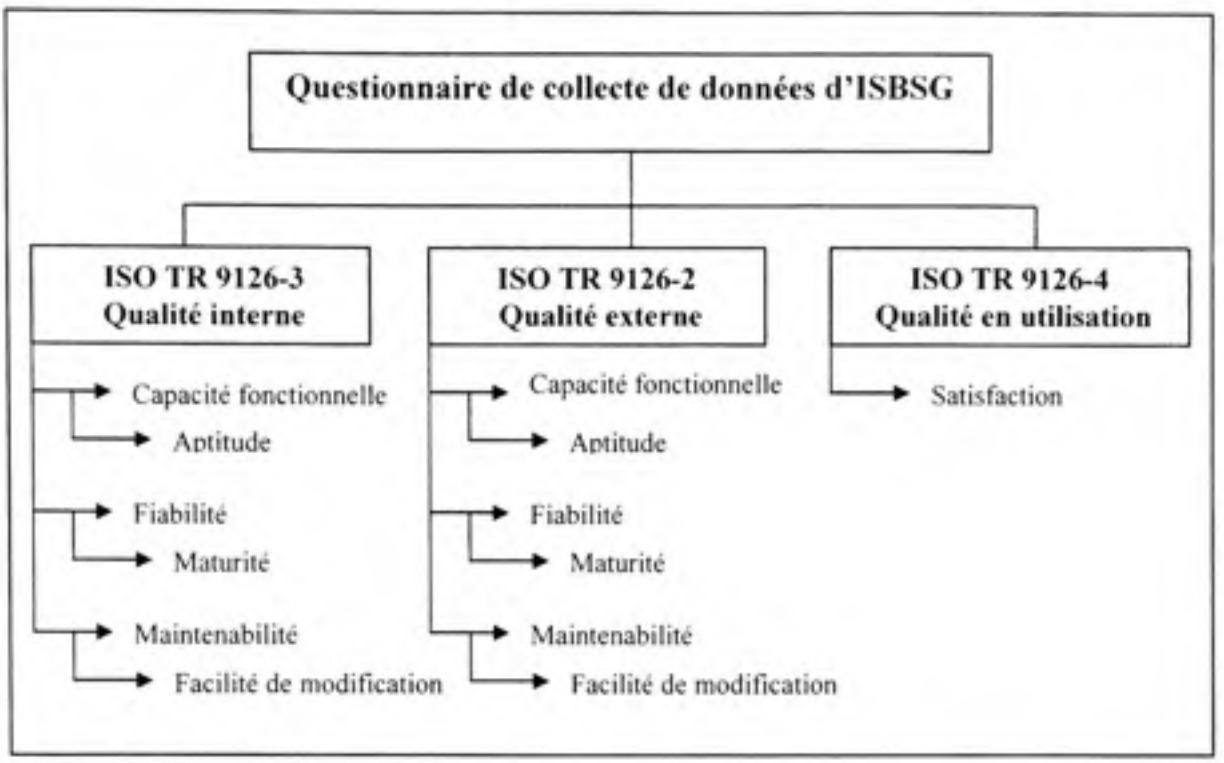

Figure 6.1 *Récapitulatif de l'analyse du questionnaire d'ISBSG selon ISO TR 9126-2 à 4.*

6.3.2 Exploration des mesures d'ISO TR 9126-2 à 4

Les travaux de recherche menés sur les rapports techniques d'ISO 9126, présentés dans la section 2.4 du chapitre 2, montrent que les mesures proposées par ISO ne sont pas assez robustes et présentent des insuffisances. De façon générale, ce problème ne se pose pas seulement pour les mesures proposées dans ces rapports, mais aussi pour plusieurs mesures proposées par les chercheurs et praticiens en génie logiciel. Pour notre étude des mesures ISO 9126, l'objectif est d'identifier et d'analyser les mesures de qualité interne, de qualité externe et de qualité en utilisation que l'on peut utiliser en exploitant les données collectées par le questionnaire d'ISBSG.

Durant notre exploration des mesures proposées dans ISO TR 9126-2 à 4, nous avons remarqué que ces mesures sont de deux types : mesures de base et, dans la plupart des cas, mesures dérivées. Une mesure de base porte principalement sur un seul attribut tandis qu'une mesure dérivée est une combinaison de deux ou plusieurs mesures de base. Pour ISBSG, nous avons observé que les informations du questionnaire ne portent que sur des mesures de base. Ainsi, nous avons déterminé, de façon globale, certaines ressemblances entre des mesures d'ISO 9126 et des données d'ISBSG. Le Tableau 6.10 présente ces ressemblances : elles correspondent essentiellement aux mesures de base.

Tableau 6.10

Ressemblances entre ISO TR 9126-2 à 4 et questionnaire d'ISBSG

Mesures de base d'ISO 9126 (ISO, 2003a; 2003b; 2004)	**Mesures de base d'ISBSG (ISBSG, 2006c)**
Number of functions (added, modified, deleted, implemented, etc.)	Number of functions (added, deleted, changed)
Product size	Software functional size
Number of detected faults	Number of defects recorded
Work time spent to change	Resolution/Rework effort
Number of changes	Number of specification changes
Response to a question	Response to a question (In the user satisfaction survey, there are 8 questions)

Au fil de cette section, nous allons analyser chacune des mesures de base d'ISO et son correspondant d'ISBSG du Tableau 6.10 à travers des exemples de mesures proposées dans ISO TR 9126-2 à 4. Cette analyse est effectuée du point de vue de l'utilisation ou non d'une définition standard à propos de l'attribut mesuré. En effet, un terme bien défini enlève l'ambiguïté aussi bien dans sa signification que dans son utilisation par les différents utilisateurs dans différents contextes. Une telle analyse contribuera à l'identification des faiblesses que présentent respectivement les deux documents. Ces

faiblesses représentent une opportunité d'amélioration des mesures proposées par ISO 9126 ainsi que des données collectées par ISBSG.

Mesure de base : Nombre de fonctions

Le premier exemple correspond à la mesure de qualité interne et de qualité externe « Functional specification stability (volatility) » proposée pour la sous-caractéristique « Aptitude » de la caractéristique « Capacité fonctionnelle » dans les deux rapports d'ISO TR 9126-2 et 3. Cette mesure est exprimée par la même formule « X= 1-A/B » et mesure la stabilité des spécifications fonctionnelles, où :

- « A » représente « Number of functions changed during development life cycle phases » dans ISO TR 9126-3 et « Number of functions changed after entering operation starting from entering operation » dans ISO TR 9126-2;
- « B » correspond au « Number of functions described in requirement specifications » dans les deux rapports techniques d'ISO TR 9126-2 et 3.

Ces mesures de base (A et B) d'ISO 9126 se basent toutes les deux sur le comptage de nombre de fonctions (implémentées dans le logiciel et décrites dans les spécifications des exigences) sans toutefois fournir à quoi correspond l'attribut mesuré : fonction. Ainsi, l'utilisation de ces mesures peut engendrer différentes interprétations parmi les mesureurs, les concepteurs et les développeurs du logiciel au sein d'une même organisation et parmi les organisations. L'absence d'une définition standard affaiblit l'efficacité du « benchmarking ». Un tel problème ne se présente pas dans l'organisation ISBSG, puisque cette dernière utilise des standards quant à la définition du terme fonction et comment le mesurer; il s'agit des points de fonction définis, par exemple dans des standards ISO dont ISO 19761.

Mesures de base : Défaut/Faute et Taille

Le deuxième exemple correspond à la mesure de qualité externe « Fault Density » proposée dans ISO TR 9126-2 pour la sous-caractéristique « Maturité » de la caractéristique « Fiabilité ». Cette mesure dérivée est composée de deux mesures de base et elle est exprimée par la formule suivante : X = A1 / B, avec :

- « A1 » désigne « Number of detected faults »;
- « B » correspond au « Product size ».

La première mesure de base (A1) concerne l'attribut défauts « faults ». Dans ISO TR 9126-2 à 4 (2003a; 2003b; 2004, p. 31), concernant « Number of detected fault type », il est cité que : « The measurement counts the detected faults during reviewing, testing, correcting, operating or maintaining. Severity levels may be used to categorize them to take into account the impact of the fault ». Par ce texte, ISO 9126 spécifie que les types de défauts sont en relation avec la phase où ils sont détectés sans toutefois définir ce qu'est un défaut « fault » ni fournir les niveaux de sévérité. De l'autre côté, ISBSG utilise un autre mot pour designer le défaut; il s'agit de « defect » et énumère trois niveaux de sévérité (mineur, majeur et extrême) en fonction de l'impact du « defect » sur le fonctionnement du logiciel. Il est à noter que ni ISO, dans ces rapports techniques ISO TR 9126-2 à 4, ni ISBSG, dans son questionnaire de collecte de données, ne fournissent une définition claire sur l'attribut mesuré « defect/fault » ni ne font référence à une définition standard[1]. Néanmoins, ISO 9126 et ISBSG utilisent des termes différents pour exprimer le même concept mais de différentes façons donnant ainsi lieu à une certaine confusion.

[1] Par Exemple : IEEE 610.12, Standard Glossary of Software Engineering Terminology.

La deuxième mesure de base (B) correspond à la taille du produit logiciel. Selon ISO TR 9126-2 à 4, Annexe C, divers types de mesures de taille sont énumérés dont « functional size type ». Pour ce type, ISO 9126 fait référence à ISO 14143-Part1 concernant les définitions des concepts requis pour l'application de la méthode de mesure de taille fonctionnelle, mais ne spécifie pas de mesures standards de taille fonctionnelle. Par contre, ISBSG collecte la taille fonctionnelle du produit logiciel en utilisant des méthodes de mesures de taille fonctionnelle reconnues par ISO et ayant des unités de base, à savoir COSMIC-FFP, IFPUG, NESMA et Mark-II. Le questionnaire COSMIC-FFP de collecte de données d'ISBSG utilise le standard ISO 19761[2] (2003c) et son unité de mesure le Cfsu (COSMIC functional size unit).

Mesures de base : Temps/Effort et Nombre de changements

Le troisième exemple correspond à la mesure de qualité externe « Modification complexity » proposée dans ISO TR 9126-2 pour la sous-caractéristique « Facilité de modification » de la caractéristique « Maintenabilité ». Cette mesure dérivée est exprimée par la formule suivante : T = Sum (A/B) / N, avec :

- « A » correspond au « Work time spent to change »;
- « B » correspond au « Size of software change »;
- « N » correspond au « Number of changes ».

La première mesure de base (A) mesure le « temps » de travail nécessaire pour effectuer le changement. Cet attribut est reconnu par le système international d'unités (SI) et son unité de base est standard et internationalement reconnue; la « seconde ». Quant à

[2] The British Computer Society has recognized in July 2006 COSMIC-FFP (ISO 19761) size measure as one of its "Technology Award Medallist" in the "Services" category (www.gelog.etsmtl.ca).

ISBSG, elle utilise un multiple de cette unité de mesure qui est l'heure pour mesurer l'effort de résolution/refaite du travail dépensé dans chaque phase du processus de développement d'ISBSG. Nous remarquons donc qu'ISO 9126 et ISBSG utilisent deux mots différents pour exprimer le même attribut.

La deuxième mesure de base (B), taille du logiciel, est déjà traitée dans l'exemple précédent. Cependant, la note relative à cette mesure « A size of software change may be changed executable statements of program code, number of changed items of requirements specification, or changed pages of document etc. » (ISO, 2003a, p. 64) montre l'ambigüité que présente la dernière mesure de base (N) : le nombre de changements. Ainsi, un changement peut correspondre, entre autres, à un changement dans le code, à un changement dans les exigences de spécifications. Ce dernier type de changements « specification changes » est collecté par l'organisation ISBSG à travers son questionnaire. Là aussi, l'absence d'une description standard de l'attribut « changement », aussi bien dans ISO 9126 que dans ISBSG, engendre diverses interprétations de cette information au sein d'une même organisation et parmi les organisations. Ainsi, il est plausible que cette variété d'interprétations puisse conduire à des applications différentes de la méthode de mesure et éventuellement à des résultats différents et difficiles à comparer à moins d'en connaitre les facteurs de convertibilité entre ces mesures.

Mesure de base : Réponse à une question du sondage

Le dernier exemple correspond à la mesure de qualité en utilisation « Satisfaction questionnaire » proposée dans ISO TR 9126-4 pour la caractéristique « Satisfaction ». Cette mesure utilise la mesure de base « Response to a question », c'est-à-dire la réponse

à une question du questionnaire de satisfaction[3]. Une telle mesure est aussi utilisée par le questionnaire d'ISBSG à travers son sondage sur la satisfaction de l'utilisateur en utilisant une échelle bien déterminée : « échelle Likert ». En effet, divers types de sondages sont utilisés par les organisations qui doivent transposer les résultats de leurs sondages effectués dans celui d'ISBSG.

De façon générale, en présence de définitions confuses et ambigües, et en l'absence de définitions claires, ayant obtenu des consensus sur les concepts ou les attributs mesurés, des applications différentes de la même mesure sont évidentes. Cette diversité d'application peut conduire par la suite à des interprétations inadéquates ou erronées du résultat de la mesure. Dans de telles circonstances, personne ne peut assurer que le résultat de la mesure répond aux critères de répétabilité et de reproductibilité, diminuant ainsi la crédibilité des analyses comparatives, autrement dit, la fiabilité du « benchmarking » aussi bien au sein de la même organisation que parmi les organisations.

6.3.2.1 Synthèse

Il est à noter que les mesures de base d'ISBSG, identifiées à partir du questionnaire, représentent un petit ensemble de celles utilisées dans la conception des mesures dérivées proposées dans ISO TR 9126-2 à 4. Il est donc utile d'utiliser ces dernières dans nos analyses empiriques. Toutefois, pour pouvoir les utiliser il est nécessaire de disposer des autres mesures de base ISO. En l'absence de telles mesures de base, les mesures dérivées d'ISO 9126, telles quelles, ne sont donc pas utilisables.

---

[3] Selon ISO TR 9126-4, Annex F, « questionnaires to measure satisfaction and associated attitudes are commonly built using Likert and semantic differential scales ».

D'un autre côté, durant notre analyse des mesures de base similaires entre ISO 9126 et ISBSG, nous avons remarqué, de façon générale, un manque de clarté dans la définition de l'attribut mesuré aussi bien dans ISO 9126 que dans ISBSG. Le Tableau 6.11 présente un récapitulatif de l'exploration des mesures de base quant à l'utilisation ou non d'un standard.

Tableau 6.11

Récapitulatif de l'analyse des mesures de base similaires (ISO 9126 et ISBSG)

ISO TR 9126-2 à 4		**Questionnaire d'ISBSG**	
Mesures de base	**Standard**	**Mesures de base**	**Standard**
Nombre de fonctions (ajoutées, modifiées, supprimées, etc.)	Non spécifié	Nombre de fonctions (ajoutées, modifiées, supprimées, nouvelles)	COSMIC-FFP (Cfsu)
Taille du projet	Non spécifié	Taille fonctionnelle du logiciel	COSMIC-FFP (Cfsu)
Nombre de fautes détectées	Non spécifié	Nombre de défauts reportés	Non spécifié
Temps de travail dépensé pour changer	Seconde	Effort de résolution/refaite	Heure
Nombre de changements	Non spécifié	Nombre de changements de spécifications	Non spécifié
Réponse à une question	Non spécifié	Réponse à une question	Non spécifié

Il est à noter aussi, du Tableau 6.11, que les mesures de base d'ISO TR 9126-2 à 4 ne possèdent pas des définitions standardisées à propos des attributs qu'elles mesurent. La seule exception concerne la mesure de base « temps », laquelle mesure est reconnue dans le SI, exprimée par l'unité « seconde » et représentée par le symbole « s ». Pour les mesures de base identifiées à partir du questionnaire de collecte de données d'ISBSG, il s'avère que trois des mesures de base possèdent une description standard de l'attribut mesuré dont le temps, la taille du logiciel et la fonction. Ces deux dernières mesures de

base utilisent le même standard ISO 19761 COSMIC-FFP[4] dont l'unité de mesure de base est « 1 COSMIC functional size unit » et représentée par le symbole « Cfsu ».

À partir de l'analyse effectuée, nous suggérons aussi bien aux experts d'ISO 9126 qu'à l'organisation ISBSG de :

1. Identifier clairement les mesures de base aussi bien dans ISO 9126 que dans ISBSG, car l'obtention de la mesure dérivée repose principalement sur l'ensemble des mesures de base qui la compose.
2. Fournir une définition claire de l'attribut mesuré et dans le cas échéant décrire de façon consistante sa méthode de mesure afin d'éviter les différentes interprétations de la même mesure, lesquelles interprétations peuvent conduire potentiellement à des applications différentes de la même mesure.
3. Utiliser des standards du génie logiciel et d'autres domaines de la science qui offrent des définitions standards à propos des termes et des concepts relatifs au domaine de la mesure, par exemple l'IEEE 910.12, l'ISO 15939 et le VIM.

6.3.3 Mesures propres à ISBSG

Rappelons que notre objectif est d'utiliser les mesures de qualité de logiciel dans les plans d'analyses empiriques conçus à base de la méthode Taguchi. Pour ce faire, il faut déterminer, à partir des données de qualité offertes par le questionnaire d'ISBSG, des mesures permettant l'évaluation quantitative de la qualité des produits logiciels. Au fil de cette section, nous présentons tout d'abord la structure de documentation de ces

[4] Les autres standards de point de fonction sont aussi utilisables dont IFPUG et son unité de mesure UFPs, Mark II et son unité de mesure UFPs et NESMA et son unité de mesure UFPs.

mesures suivie de la liste des mesures en s'inspirant de celles des rapports ISO TR 9126-2 à 4 avec une adaptation au contexte d'ISBSG.

6.3.3.1 Structure de documentation des mesures

Étant consciente du problème d'ambigüité que posent les différentes terminologies[5] utilisées dans la littérature du génie logiciel, la communauté des experts ISO a commencé à incorporer le vocabulaire de métrologie dans ses standards, comme dans le cas d'ISO 14143, d'ISO 15939, d'ISO 19761 et d'ISO 25000.

La structure de documentation des mesures que nous proposons, et qui est partiellement arrimée à celle d'ISO 9126, utilise le VIM (ISO, 1993). En effet, seules les deux catégories « Grandeurs et unités » et « Étalon » du VIM sont liées au design de méthode de mesure (Sellami, 2005). Nous allons utiliser certains termes et définitions définis dans la norme ISO 15939 (2001b), lesquels termes se basent sur le VIM.

Cette structure de documentation est composée de six termes, à savoir :

1. Nom de la mesure : nom de la mesure de qualité interne ou de qualité externe ou de qualité en utilisation du produit logiciel.
2. Objectif de la mesure : l'objectif de la mesure est exprimé par une question.
3. Assignation numérique de la mesure : deux types de mesures se distinguent :
    a. Mesure de base : une mesure de base est « une mesure définie en termes d'un attribut et la méthode de sa quantification. Une mesure de base est

[5] Par exemple, l'utilisation du mot « métrique » par la série ISO TR 9126-2 à 4 provoque une ambiguïté aussi bien dans sa signification que dans son utilisation. Dans la prochaine version d'ISO, le mot « métrique » est remplacé par le mot « mesure » qui est utilisé dans toutes les disciplines des sciences.

fonctionnellement indépendante des autres mesures » (ISO, 2001b, p. 3). Cette définition se base sur le concept « grandeur de base » du VIM.

b. Mesure dérivée : une mesure dérivée représente « une mesure définie comme une fonction de deux ou plusieurs valeurs des mesures de base » (ISO, 2001b, p. 3). Cette définition se base sur le concept « grandeur dérivée » du VIM.

4. Interprétation de la valeur mesurée : une valeur correspond au « numerical or categorical result assigned to a base measure, derived measure, or indicator » (ISO, 2001b, p. 7). L'interprétation correspond aux préférences, à savoir : par exemple, une valeur de la mesure grande est la meilleure ou une valeur de la mesure petite est la meilleure.
5. Unité de mesure : une unité de mesure est définie par une « grandeur particulière, définie et adoptée par convention, à laquelle on compare les autres grandeurs de même nature pour les exprimer quantitativement par rapport à cette grandeur » (ISO, 1993, p. 13). À chaque type de mesure est associé un type d'unité :
    a. Unité (de mesure) de base : elle correspond à une « unité de mesure d'une grandeur de base dans un système donné de grandeurs » (ISO, 1993, p. 15). La grandeur de base correspond à la mesure de base.
    b. Unité (de mesure) dérivée : elle correspond à une « unité de mesure d'une grandeur dérivée dans un système donné de grandeurs » (ISO, 1993, p. 15). La grandeur dérivée correspond à la mesure dérivée.

6. Méthode de mesure : une méthode de mesure correspond à une « succession logique des opérations, décrites d'une manière générique, mises en œuvre lors de la quantification d'un attribut par rapport à une échelle spécifique » (ISO, 2001b, p. 4). Cette définition se base sur le concept « méthode de mesure » du VIM. Si la mesure est dérivée, alors il faut fournir la méthode de mesure pour toutes ses mesures de base.

6.3.3.2 Liste des mesures propres à ISBSG

Les mesures déterminées à partir des données de qualité offertes par ISBSG pour évaluer quantitativement la qualité du produit logiciel, essentiellement pour les caractéristiques et sous-caractéristiques de la Figure 6.1, sont regroupées dans le Tableau 6.12. La description détaillée de ces mesures de qualité interne, de qualité externe et de qualité en utilisation est présentée respectivement dans la partie V.I, la partie V.II et la partie V.III de l'Annexe V.

Tableau 6.12

Liste des mesures propres à ISBSG

Caractéristiques / sous-caractéristiques	**Mesures de qualité interne**	**Mesures de qualité externe**	**Mesures de qualité en utilisation**
Capacité fonctionnelle/ Aptitude	-Densité de changements de spécifications	- Densité de changements de fonctionnalités	
Fiabilité/ Maturité	-Densité de défauts	-Densité de défauts	
Maintenabilité / Facilité de modification	-Effort de résolution/refaite des défauts	-Effort de résolution/refaite des défauts	
Satisfaction			- Degré de satisfaction des utilisateurs

En résumé, ces mesures déterminées à partir des données de qualité offertes par ISBSG peuvent être utiles :

1. À l'industrie du logiciel et aux chercheurs recherchant des mesures de qualité du produit logiciel propres à ISBSG pour des besoins de comparaisons de la qualité de leurs projets.
2. À l'organisation ISBSG afin de prendre connaissance de l'importance des données collectées pour la qualité du logiciel. Ainsi, étudier comment améliorer le questionnaire de collecte de données afin qu'il couvre les autres caractéristiques et sous-caractéristiques des trois modèles de qualité d'ISO 9126.
3. Aux chercheurs afin d'utiliser ces mesures dans leurs travaux de recherche. Par exemple : dans la conception de plans d'analyses empiriques avec la méthode Taguchi.

6.4 Sommaire

D'une part, ISO 9126 propose un ensemble de modèles de qualité pour évaluer la qualité de trois points de vue (interne, externe et en utilisation) et un ensemble de mesures dans des rapports techniques : ISO TR 9126-2 à 4. Cependant, ISO 9126 n'a pas mis sur pied la collecte de données de qualité selon ses propres modèles, et il n'existe pas actuellement de référentiels de données organisées selon ISO 9126. En outre, il n'y a pas encore de référentiel de données publiques de disponible réclamant l'alignement avec ISO 9126.

D'autre part, ISBSG offre au public un référentiel de données de projets, incluant un ensemble de variables à propos de la qualité du logiciel. Cependant, il n'était pas connu jusqu'à maintenant si le référentiel de données d'ISBSG pouvait être utilisé ou non pour collecter et analyser des données à propos de la qualité du logiciel en conformité avec les modèles de qualité proposés par ISO 9126.

Ce chapitre a porté sur un ensemble de mises en correspondance entre ISO 9126 et ISBSG. Il est important de noter que l'organisation ISBSG met l'accent sur l'entité

« projet logiciel » alors qu'ISO 9126 s'intéresse à l'entité « produit logiciel ». Nous avons donc orienté nos analyses selon l'entité produit logiciel au sein de l'entité projet logiciel d'ISBSG, en tenant compte en particulier du cycle de vie du produit logiciel de sa planification jusqu'à son achèvement.

De la mise en correspondance (de haut niveau et détaillée) du questionnaire d'ISBSG avec les trois types de qualité d'ISO 9126 (interne, externe et en utilisation), il en ressort que :

- la section « Processus du projet » peut être alignée à deux des trois types : interne et externe;
- la section « Fin du projet » peut être alignée à deux des trois types : externe et en utilisation;
- les autres sections, « Technologie », « Personne et Effort du travail » et « Produit » ne peuvent fournir de l'information pour aucun des trois modèles de qualité d'ISO 9126 et les informations de la section « Taille fonctionnelle du logiciel » sont utiles pour des objectifs de normalisation lors du calcul des ratios en relation avec la qualité.

En outre, de cette mise en correspondance, il est à remarquer que le questionnaire d'ISBSG collecte des informations concernant la qualité du produit logiciel pour toutes les phases du cycle de vie du logiciel. Dans la pratique, ISBSG ne rend pas disponible toutes ses données, pour des raisons de confidentialité. Cependant, ISBSG rend disponible à l'industrie et aux chercheurs un extrait (i.e. un sous–ensemble) de données, à un prix modéré.

L'extrait de données d'ISBSG (Release 9 de 2005) a été analysé ensuite afin d'avoir une idée sur l'information disponible et quelle information pourrait être utile pour implémenter les modèles d'ISO 9126 de la qualité de logiciel. Dans cet extrait d'ISBSG, il y a en effet une section « Qualité » avec des champs de données relatifs au nombre de

défauts (mineur, majeur, extrême ou total) détectés pendant le premier mois d'opération du logiciel. Ainsi, l'extrait de données d'ISBSG (Release 9 de 2005) fournit directement quelques données de qualité pour seulement une des trois types de qualité d'ISO 9126.

Comme cet extrait de données d'ISBSG (Release 9 de 2005) ne contenait donc que quelques informations utiles pour l'évaluation de la qualité du produit logiciel, nous avons par la suite planifié de soumettre à ISBSG, en utilisant le formulaire approprié, une requête de recherche. L'objectif de cette requête etait d'obtenir un complément de données auprès d'ISBSG afin de disposer de données supplémentaires relatives à la qualité interne, la qualité externe et la qualité en utilisation.

De la mise en correspondance (de haut niveau et détaillée) du questionnaire de collecte de données d'ISBSG par rapport à la série des documents ISO TR 9126-2 à 4, il en ressort que dans ISBSG il y a des données de qualité disponibles que pour évaluer les caractéristiques de qualité suivantes :

- la capacité fonctionnelle, la fiabilité et la maintenabilité respectivement via les sous-caractéristiques suivantes : l'aptitude, la maturité et la facilité de modification pour la qualité interne et la qualité externe du produit logiciel;
- la satisfaction pour la qualité en utilisation du produit logiciel.

Les autres caractéristiques et sous-caractéristiques des modèles de qualité d'ISO 9126 (Figures 2.2 et 2.3 du chapitre 2) ne sont mentionnées ni explicitement ni implicitement dans le questionnaire d'ISBSG.

De notre exploration des propositions de mesures des rapports techniques d'ISO 9126, il en ressort que la plupart des mesures proposées dans ISO TR 9126-2 à 4 sont des mesures dérivées composées de deux ou plusieurs mesures de base. D'autre part, il est à remarquer que les informations du questionnaire d'ISBSG ne portent que sur des mesures de base.

De l'analyse des mesures de base similaires entre ISO 9126 et ISBSG, il a été constaté que rares sont les attributs mesurés ayant un standard de défini aussi bien dans ISO 9126 que dans ISBSG.

Enfin, pour utiliser les données de qualité disponibles dans le référentiel d'ISBSG dans nos plans d'analyses empiriques, nous avons identifié des mesures propres à ISBSG permettant d'évaluer la qualité du produit logiciel pour les caractéristiques de la Figure 6.1. Ces mesures utilisent les mesures de base d'ISBSG déterminées avec de l'information offerte par le questionnaire d'ISBSG. La structure de documentation de ces mesures ainsi que leurs descriptions ont été construites à partir de celles des rapports techniques ISO TR 9126-2 à 4, tout en utilisant le standard ISO 15939 et le VIM dans la définition des six termes de cette structure.

En général la principale contrainte dans ce chapitre, lors de la mise en correspondance entre ISO 9126 et ISBSG, a résidé dans la difficulté de trouver des critères de comparaison. Cette difficulté émane des particularités de la série ISO 9126 et du questionnaire d'ISBSG. Les principales différences sont résumées dans le Tableau 6.13.

Tableau 6.13

Différences entre ISO 9126 et ISBSG

Critères	ISBSG	ISO 9126
Cycle de vie du logiciel	Suit, d'une certaine manière, la nomenclature générale du guide SWEBOK	Se base sur la structure du standard ISO 12207 : Processus du cycle de vie du logiciel
Sources des informations (mesures ou données) collectées	Les organisations qui remplissent le questionnaire de collecte de données peuvent, ou non, disposer d'un programme de mesure utilisant les mesures ISO TR 9126-2 à 4	Les mesures d'ISO TR 9126-2 à 4 se basent sur des informations très spécifiques. De telles informations nécessitent que les organisations disposent d'un programme de mesure qui s'étale sur toutes les phases du cycle de vie du logiciel pour pouvoir les collecter
Caractéristiques des informations (mesures ou données) collectées	Les mesures sont proposées par des praticiens. Ces mesures sont collectées pratiquement et reflètent des données réelles et vérifiées	Les mesures sont proposées par des experts de façon théorique. Ces mesures n'ont pas nécessairement été utilisées ni vérifiées dans la pratique avant d'être adoptées par ISO

CHAPITRE 7

ENVIRONNEMENT DU PLAN D'ANALYSE EMPIRIQUE

7.1 Introduction

> « When a man does not know what harbor he is making for, no wind is the right wind. » (Summers, 2006, p. 68)

Ce chapitre traite la phase 2 de la méthodologie de recherche établie dans le chapitre 5 de cette thèse. Ce chapitre présente la préparation de l'environnent des plans d'analyses empiriques permettant de vérifier les hypothèses des liens entre les trois types de qualité de la norme ISO 9126-1 (Figure 2.1).

Suite à la demande d'accès aux données requises pour cette recherche que nous avons effectué auprès de l'organisation ISBSG dans l'étape 2 (exploration détaillée, laquelle étape est traitée dans le chapitre 6) de la phase 1 (exploration) de notre méthodologie de recherche, nous avons reçu un nouvel extrait de données (ISBSG, 2006a) auquel nous référons par extrait d'ISBSG de février 2006. Cet extrait englobe un ensemble de 3 855 projets avec 166 champs de données. La préparation porte essentiellement sur ce nouvel extrait de données d'ISBSG et comprend :

- le recensement des données à exploiter de l'extrait de données d'ISBSG, c'est-à-dire les facteurs pouvant avoir un impact sur la qualité du produit logiciel;
- la préparation des données porte sur la vérification de la qualité et de la complétude des données d'ISBSG de l'extrait de février 2006;
- la présentation détaillée des différentes étapes d'analyse des résultats des plans d'analyses avec la méthode d'analyse du ratio signal-bruit de Taguchi.

7.2 Données exploitées

Dans le cas du processus d'expérimentation normale, cas idéal, la phase de collecte de données et la phase de validation de ces données suivent la phase de planification de l'expérience. De plus, les informations à collecter sont définies de façon aussi exacte que possible et avec le moins d'ambigüité possible. Dans le cas où les données sont déjà collectées pour d'autres fins d'études, le choix d'informations utiles et la validation de ces informations collectées sont beaucoup plus importants.

En termes généraux, la validation de données consiste à découvrir si on dispose des bonnes données pour notre objectif (Maxwell, 2002). L'objectif dans ce travail de recherche correspond à la qualité du produit logiciel. De l'extrait de données d'ISBSG de février 2006, nous avons sélectionné un ensemble de facteurs pouvant avoir un impact sur la qualité du produit logiciel, regroupés dans les 12 catégories (le détail de chaque catégorie est présenté en Annexe VI) suivantes :

1. Taille du projet.
2. Effort dépensé dans le développement du projet.
3. Type de développement.
4. Personnalisation.
5. Architecture du projet.
6. Technologie utilisée.
7. Changements de spécifications.
8. Défauts détectés.
9. Effort de résolution/refaite pour les défauts détectés.
10. Expérience de l'équipe du projet.
11. Versions du projet.
12. Taille fonctionnelle des améliorations.

Puisqu'il est mentionné dans la littérature que la qualité du produit logiciel dépend d'autres facteurs qui ont servi à développer le logiciel, ces facteurs sont aussi pris en considération. Ainsi, la liste des 12 catégories comprend aussi bien les facteurs de qualité que les autres facteurs qui peuvent indirectement influencer la qualité du produit logiciel; à titre d'exemple, la technologie utilisée, l'architecture du projet, etc.

Cette liste sera utile dans le choix des facteurs à utiliser lors de la conception des plans d'analyses avec Taguchi : c'est-à-dire, au choix d'une part des caractéristiques de qualité (attribut de sortie) à évaluer et, d'autre part, à la sélection des facteurs pouvant avoir un effet sur ces caractéristiques de qualité.

7.2.1 Questionnaire sur la qualité du produit logiciel

Dans le but d'avoir une idée sur l'influence de ces 12 catégories de facteurs sur la qualité externe et la qualité en utilisation du produit logiciel, nous avons établi un questionnaire. Ce dernier comprend un ensemble de facteurs de ces catégories et deux caractéristiques de qualité choisies à titre d'exemple : le nombre de défauts détectés lors de l'exécution du logiciel, considéré comme un attribut de qualité externe et le degré de satisfaction des utilisateurs, considéré comme un attribut de qualité en utilisation.

Globalement, le questionnaire sur la qualité du produit logiciel comporte une introduction de l'objectif du questionnaire, une section portant sur des questions sur le profil personnel de la personne qui va remplir le questionnaire suivi d'une section relative aux instructions sur les modalités de réponses aux questions. La dernière section comprend la liste des 12 catégories avec leurs facteurs correspondants. La description détaillée du questionnaire est présentée en Annexe VI.

Pour chaque facteur, il faut remplir deux parties : l'une correspond au nombre de défauts détectés lors de l'exécution du logiciel et l'autre au degré de satisfaction des utilisateurs.

L'objectif est d'indiquer pour chaque facteur s'il a un impact sur ces deux caractéristiques de qualité, à savoir : « Oui », « Non » et « Ne sait pas ». Lorsque la réponse est « Oui », il faut indiquer le degré de cet impact : « Grand » ou « Faible ».

Par la suite, nous avons donné ce questionnaire à quatre sujets (des praticiens avec de l'expérience de quelques années en industrie), le but est de faire une séance de « brainstorming » afin qu'ils remplissent le questionnaire et fournissent leurs commentaires. Ces sujets sont des informaticiens de l'industrie avec différents profils : le premier sujet est un membre d'équipe de développement chargé de la spécification, de la conception, de la programmation, des tests, de l'implémentation et de la maintenance du logiciel. Ce sujet possède huit ans d'expériences dans le domaine du logiciel. Le deuxième et le troisième sont des gestionnaires, chacun est responsable de la gestion de la qualité du logiciel avec diverses expériences dans le domaine du logiciel, à savoir : 31 ans et 3 ans. Le quatrième sujet est un chef de projet chargé de la planification et la conception du logiciel, assurant la tâche de gestion de projets logiciels et ayant 17 ans d'expériences dans le domaine du logiciel.

Dans un premier temps, nous avons rassemblé les questionnaires complétés par les quatre personnes en fonction des impacts. Nous nous sommes aperçus qu'il y a des facteurs pour lesquels il n'y avait pas eu de réponses. Ensuite, nous avons rassemblé les facteurs et leurs impacts respectifs pour chaque caractéristique de qualité. Les résultats de l'analyse des questionnaires dûment remplis par les quatre sujets sont présentés dans les Tableaux VI.1 et VI.2 en Annexe VI. La première colonne présente les différentes catégories d'impacts et la deuxième colonne les facteurs correspondants. Les catégories d'impacts sont présentées en ordre décroissant de l'influence des facteurs, par exemple :

- « Impact : 4G » (G pour designer le degré de l'impact « Grand ») signifie que les quatre sujets ont qualifié l'impact du facteur comme grand;

- « Impact : 2G, 1F » (G pour « Grand » et F pour « Faible » degré d'impact) signifie que parmi les quatre sujets, deux ont qualifié l'impact du facteur comme « Grand », un comme « Faible » et un n'a pas fourni d'information (vide ou « Ne sait pas »).

De façon générale, de l'analyse des résultats du questionnaire, il en ressort que les différentes catégories identifiées au début de cette section, en particulier les différents facteurs de ces catégories présentent des impacts avec des degrés différents sur :

- la qualité externe du produit logiciel (exprimée par le nombre de défauts détectés lors de l'exécution du logiciel);
- la qualité en utilisation du produit logiciel (exprimée par le degré de satisfaction des utilisateurs).

Ces résultats seront utiles dans le choix et la sélection des facteurs à mettre en œuvre lors de la conception des plans d'analyses empiriques avec Taguchi.

7.3 Préparation des données : Extrait d'ISBSG de février 2006

7.3.1 Premier niveau de préparation

Ce premier niveau de préparation porte sur la vérification de la qualité des données de l'extrait d'ISBSG de février 2006 (ISBSG, 2006a). En effet, les gestionnaires d'ISBSG procèdent à l'analyse de la qualité des données collectées pour chaque projet et enregistrent leur jugement dans un champ de classification (Data Quality Rating (DQR)). Les projets sont qualifiés selon le degré d'intégrité des données collectées de « A » (bonne intégrité) jusqu'à « D » (mauvaise intégrité) (voir Annexe IV). La répartition de ces projets est illustrée dans le Tableau 7.1 :

- les projets de type C et D (donc de mauvaise qualité) représentent 252 projets de l'extrait d'ISBSG de février 2006, soit un pourcentage de 7%;

- les projets de type A et B (donc de bonne qualité) constituent un pourcentage de 93% de l'ensemble des projets de l'extrait d'ISBSG de février 2006, soit 3 603 projets.

Tableau 7.1

Répartition des projets de l'extrait d'ISBSG de février 2006 par DQR

Type de projets	Nombre de projets	Pourcentage
A	**806**	**21%**
B	**2797**	**72%**
C	143	4%
D	109	3%
Total	**3855**	**100%**

En prenant en considération comme critère de sélection la bonne intégrité des données collectées et dans le but de réduire le risque d'utiliser des données ne possédant pas une bonne qualité, nous avons limité l'échantillon de travail à la catégorie de projets de types A et B; soit 3 603 projets. Nous référons à cet échantillon de travail par « BD-ISBSG-AB-Février-2006 », lequel nécessitera un autre niveau de préparation de données par la suite, et ce, pour chaque plan d'analyse conçu avec la méthode Taguchi.

7.3.2 Deuxième niveau de préparation

Le deuxième niveau de préparation est nécessaire puisque dans le questionnaire de collecte des données d'ISBSG les informations relatives aux différents facteurs des 12 catégories identifiées dans la section 7.2 et du sondage de la satisfaction des utilisateurs ne sont pas obligatoires. Par conséquent, il peut y avoir des projets qui n'ont pas de données dans ces champs d'informations : il s'agit donc de données manquantes.

La présence de données manquantes[6] dans les données collectées est inévitable, car toutes les variables peuvent ne pas avoir été collectées pour différentes raisons. Parmi ces raisons, nous citons : le manque de temps, l'incompréhension des questions, le manque de connaissances et la confidentialité des informations à divulguer (Strike, Emam et Madhavji, 2001).

Il est important de noter que dans la base de données d'ISBSG de février 2006, les données manquantes sont présentes avec des degrés différents selon les champs de données. Une étape supplémentaire de préparation est donc inévitable en fonction des paramètres utilisés dans chaque plan d'analyse, à savoir : les facteurs et la caractéristique de qualité du plan. Cette préparation porte essentiellement sur la sélection des échantillons appropriés de projets d'ISBSG avec lesquels nous pouvons mener nos analyses des plans conçus avec la méthode Taguchi.

Les étapes de préparation de l'extrait d'ISBSG de février 2006 sont résumées dans la Figure 7.1.

[6] En effet, « while one should strive to minimise missing values, in practice their existence is usually unavoidable. Missing values are not unique to software cost estimation, but is a problem that concerns empirical scientists in other disciplines. » (Strike, Emam et Madhavji, 2001, p.890)

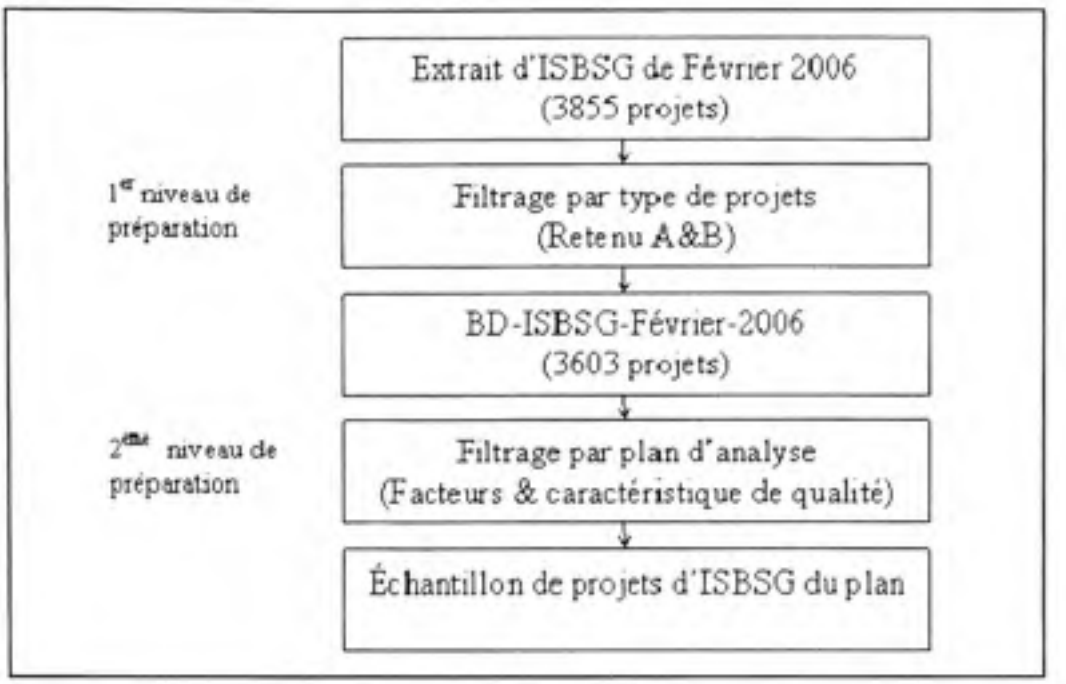

Figure 7.1 ***Étapes de préparation des données d'ISBSG.***

7.4 Analyse du ratio signal-bruit

Dr. Génichi Taguchi propose deux manières pour analyser les plans d'expériences conçus avec la méthode Taguchi, à savoir : l'analyse standard (Level average analysis) et l'analyse de ratio signal-bruit (Signal-to-noise analysis). Peace (1993, p. 273) souligne la différence entre ces deux approches d'analyses par le passage suivant :

> « Within level average analysis, there may or may not be more than one piece of data for each experimental run. If two or more repetitions are produced for each experimental run, then an average response is calculated for each row, and the analysis is based on these averages. Therefore, conclusions are based only on the effect on the mean results for the quality characteristic of interest. With signal-to-noise ratio (S/N) analysis, the calculations take into consideration both the mean and the variation from one result to the next. Therefore, we can think of signal-to-noise ratio analysis as being two-dimensional as opposed to regular analysis being only one-dimensional. »

En outre, selon Roy (1990, p. 143), « To analyze the results of experiments involving multiple runs, use of the S/N ratio over standard analysis (use average of results) is preferred ». Il est donc préférable dans le cas où les plans d'analyses possèdent plusieurs réponses pour le même essai de la matrice d'expériences, comme c'est le cas dans les différents plans d'analyses des chapitres 8, 9 et 10, d'utiliser le ratio signal-bruit. Ce ratio est représenté aussi par la lettre grecque η.

Les étapes à suivre lors de la conception du plan d'analyse empirique sont déjà citées dans la phase 3 de notre méthodologie de recherche (chapitre 5). Nous présentons maintenant de façon plus détaillée certains points de cette phase, essentiellement :

- choix de la table orthogonale du plan étudié;
- analyse et interprétation.

Pour illustrer ces différents points, nous prenons à titre d'exemple un plan « *Pexemple* » constitué de trois facteurs : A, B et C. Chaque facteur possède deux niveaux représentés par 1 et 2. La caractéristique de qualité du plan « *Pexemple* » est représentée par QC.

7.4.1 Choix de la table orthogonale du plan étudié

Comme déjà cité dans la phase 3 (étape 2, sous-étape 6) de notre méthodologie de recherche (chapitre 5), le choix de la table orthogonale prend en considération le nombre de facteurs, leurs niveaux et les interactions, s'il y a lieu. La notion de degré de liberté permet de déterminer laquelle des tables orthogonales de Taguchi est à choisir pour le plan étudié.

7.4.1.1 Degré de liberté de la table orthogonale de Taguchi

Chaque table orthogonale proposée par Taguchi possède son degré de liberté qui lui est propre. Le calcul du degré de liberté (Dl) d'une table orthogonale de Taguchi se fait de la manière suivante (Beauchamp, 2005; Peace, 1993) :

$$\begin{aligned} &\textbf{\textit{Degré de liberté}} = \textit{Nombre de colonnes} \times \textit{Degré de liberté par colonne} \\ &\textbf{\textit{Degré par colonne}} = \textit{Nombre de niveaux par colonne -1} \end{aligned} \qquad (7.1)$$

Il est à noter que pour chaque table orthogonale (simple et non composée) proposée par Taguchi, toutes les colonnes ont le même nombre de niveaux. Des exemples de degrés de liberté de tables orthogonales sont présentés comme suit :

- L4 (2^3) : trois colonnes avec deux niveaux chacun, trois degrés de liberté;
- L8 (2^7) : sept colonnes avec deux niveaux chacun, sept degrés de liberté;
- L9 (3^4) : quatre colonnes avec trois niveaux chacun, huit degrés de liberté.

7.4.1.2 Degré de liberté du plan étudié

Le degré de liberté du plan étudié est calculé de la manière suivante (Peace, 1993) :

$$\begin{aligned} &\textbf{\textit{Degré de liberté du plan étudié}} = \\ &\qquad \textit{$\sum$des degrés de liberté de chaque facteur} \\ &\qquad + \\ &\qquad \textit{$\sum$des degrés de liberté de chaque interaction} \\ &\textbf{\textit{Degré de liberté d'un facteur}} = \textit{Nombre de niveaux du facteur -1} \\ &\textbf{\textit{Degré de liberté d'une interaction (AB)}} = \\ &\qquad \textit{Nombre de niveaux du facteur A - 1} \\ &\qquad \times \\ &\qquad \textit{Nombre de niveaux du facteur B - 1} \end{aligned} \qquad (7.2)$$

L'interaction entre les facteurs A et B est représentée par (AB).

Le choix de la table orthogonale correspondante au plan étudié est fait selon la condition suivante : il faut que le nombre de degrés de liberté de l'étude soit toujours inférieur à celui de la table orthogonale sélectionnée (Beauchamp, 2005). Si le nombre de degrés de liberté de l'étude correspond à une des tables orthogonales de Taguchi, alors celle-ci est choisie. Ainsi, la table choisie constitue la matrice d'expériences (essais) à effectuer pour le plan étudié; il s'agit de la liste des essais et les combinaisons de niveaux des facteurs du plan et des interactions entre facteurs s'il y a lieu.

Pour le plan « *Pexemple* », il n'y a que trois facteurs (A, B et C) avec deux niveaux chacun et sans aucune interaction. Le degré de liberté du plan étudié est égal à trois, lequel nombre correspond au degré de liberté de la table orthogonale de Taguchi L4, soit quatre essais. Cette table est donc la table à choisir pour constituer la matrice d'expériences du plan étudié (Tableau 7.2).

Tableau 7.2

Matrice d'expériences du plan étudié

Liste des essais	**A**	**B**	**C**
1	1	1	1
2	1	2	2
3	2	1	2
4	2	2	1

7.4.2 Analyse et Interprétation

Les différentes étapes de l'analyse du ratio signal-bruit, énumérées dans la phase 3 (étape 3, sous-étape 2) de notre méthodologie de recherche (chapitre 5), sont :

- calcul du ratio signal-bruit (S/B);
- calcul des effets des facteurs;
- détermination de la condition optimale;
- analyse de la variance (ANOVA);
- calcul de l'équation de prédiction;
- test de confirmation.

7.4.2.1 Calcul du ratio signal-bruit

Dans le calcul du ratio signal-bruit, trois équations sont proposées par Taguchi selon le type de la caractéristique de qualité. Nous nous intéressons essentiellement aux deux cas suivants :

- l'optimum est une valeur minimale (Smaller-the-better);
- l'optimum est une valeur maximale (Larger-the-better).

À titre illustratif, supposons que chaque essai de la matrice d'expériences du plan « *Pexemple* » possède trois réponses. La colonne caractéristique de qualité (QC) représente les réponses du plan étudié (Tableau 7.3). Par exemple, les valeurs Y11, Y12 et Y13 représentent les trois réponses pour l'essai numéro1 avec la condition suivante : facteur A au niveau1, facteur B au niveau 1 et facteur C au niveau 1.

Tableau 7.3

Liste des essais de la matrice d'expériences du plan étudié

Liste des essais	A	B	C	QC
1	1	1	1	Y11, Y12, Y13
2	1	2	2	Y21, Y22, Y23
3	2	1	2	Y31, Y32, Y33
4	2	2	1	Y41, Y42, Y43

Cas 1 : l'optimum est une valeur minimale (Smaller-the-better)

Dans le cas où la caractéristique de qualité correspond à *l'optimum est une valeur minimale*, l'équation du calcul du ratio signal-bruit (S/B) proposée par Taguchi est la suivante :

$$S / B = -10 \log \left[\frac{1}{n} \sum_{j=1}^{n} Y_{ij}^{2} \right] \tag{7.3}$$

avec :

« i » : numéro de l'essai;

« n » : nombre de réponses dans l'essai (i);

« Yij » : réponses de l'essai (i), j = 1…n.

Par exemple, le ratio S/B de l'essai numéro1 du Tableau 7.3 est calculé comme suit :

$$S / B1 = -10 \log \left[\frac{1}{3} \times \left(Y_{11}^{2} + Y_{12}^{2} + Y_{13}^{2} \right) \right] \tag{7.4}$$

Le calcul des ratios S/B des autres essais se fait de la même manière.

Cas 2 : l'optimum est une valeur Maximale (Larger-the-better)

Dans le cas où la caractéristique de qualité correspond à *l'optimum est une valeur maximale*, l'équation du calcul du ratio signal-bruit (S/B) proposée par Taguchi est la suivante :

$$S / B = -10 \log \left[\frac{1}{n} \sum_{j=1}^{n} \frac{1}{Y_{ij}^{2}} \right] \tag{7.5}$$

avec :

« i » : numéro de l'essai;

« n » : nombre de réponses dans l'essai (i);

« Yij » : réponses de l'essai (i), j = 1…n.

Par exemple, le ratio S/B de l'essai numéro1 du Tableau 7.3 est calculé comme suit :

$$S / B1 = -10 \log \left[\frac{1}{3} \times \left(\frac{1}{Y_{11}^{2}} + \frac{1}{Y_{12}^{2}} + \frac{1}{Y_{13}^{2}} \right) \right] \tag{7.6}$$

Le calcul des ratios S/B des autres essais se fait de la même manière.

En général, quelque soit le cas, le calcul du ratio signal-bruit est effectué pour chaque essai de la matrice d'expériences en fonction des réponses de l'essai. Une fois effectué, une colonne est ajoutée au Tableau 7.3 avec les ratios S/B des différents essais.

Tableau 7.4

Calcul des ratios S/B du plan étudié

Liste des essais	A	B	C	QC	Ratio S/B
1	1	1	1	Y11, Y12, Y13	S/B1
2	1	2	2	Y21, Y22, Y23	S/B2
3	2	1	2	Y31, Y32, Y33	S/B3
4	2	2	1	Y41, Y42, Y43	S/B4

Les ratios signal-bruit ainsi calculés (Tableau 7.4) sont utiles dans le calcul de la moyenne des ratios signal-bruit pour chaque niveau de chaque facteur du plan étudié et, par la suite, le calcul de l'effet du facteur.

7.4.2.2 Calcul des effets des facteurs

L'effet d'un facteur (ou interaction) est égal « to the difference between the average S/N for each level (two levels) or the difference between the highest average S/N and the lowest average S/N (more than two levels) » (Peace, 1993, p. 277).

La moyenne des ratios S/B (average S/N) pour un niveau d'un facteur (ou d'une interaction) correspond à la somme des valeurs des ratios S/B pour le même niveau du facteur (ou de l'interaction) divisée par le nombre de ratios pour le niveau du facteur (ou de l'interaction) en question.

Pour le plan étudié « *Pexemple* », il n'y a que deux niveaux pour chaque facteur. Alors l'effet du facteur est obtenu en calculant la différence de la moyenne des ratios S/B des deux niveaux du facteur. Pour chaque facteur (colonne) du Tableau 7.4, la moyenne des ratios S/B à chaque niveau du facteur est obtenue en additionnant les valeurs des ratios

S/B de tous les essais (lignes) au niveau du facteur considéré, puis en divisant par le nombre de valeurs additionnées. À titre illustratif, pour le facteur A :

- La moyenne des ratios S/B au niveau 1 est égale au ratio S/B de l'essai numéro1, soit S/B1, plus le ratio S/B de l'essai numéro2, soit S/B2, divisé par le nombre de ratios additionnés, soit 2. Cette moyenne est dénotée par :

$$\bar{A}_1 = ((S/B1) + (S/B2)) / 2 \tag{7.7}$$

- De la même manière se fait le calcul de la moyenne des ratios S/B au niveau 2 du facteur A, dénoté par :

$$\bar{A}_2 = ((S/B3) + (S/B4)) / 2 \tag{7.8}$$

Ainsi, l'effet du facteur A est égal à la valeur absolue de la différence entre ces deux moyennes : $\bar{A}_1$ et $\bar{A}_2$.

$$\text{L'effet du facteur A} = \left| \bar{A}_1 - \bar{A}_2 \right| \tag{7.9}$$

Le calcul de l'effet des autres facteurs du plan étudié se fait de la même manière. Le Tableau 7.5 regroupe le calcul des effets des facteurs du plan étudié.

Tableau 7.5

Calcul des effets des facteurs du plan étudié

Facteurs	A	B	C
Moyenne des ratios S/B au niveau 1	$\overline{A}_1$	$\overline{B}_1$	$\overline{C}_1$
Moyenne des ratios S/B au niveau 2	$\overline{A}_2$	$\overline{B}_2$	$\overline{C}_2$
Effet du facteur (Delta)	$\lvert \overline{A}_1 - \overline{A}_2 \rvert$	$\lvert \overline{B}_1 - \overline{B}_2 \rvert$	$\lvert \overline{C}_1 - \overline{C}_2 \rvert$

À partir de ce tableau, il est possible de déterminer le degré de l'effet de chaque facteur du plan étudié sur sa caractéristique de qualité : c'est-à-dire, identifier les facteurs ayant un grand effet, de ceux moyen et ceux faible. Plus la valeur du delta est grande, plus l'effet du facteur est important.

En outre, la représentation graphique des moyennes des ratios S/B pour tous les niveaux des facteurs permet aussi de déterminer graphiquement les facteurs ayant un grand effet de ceux moyen et ceux faible en comparant « the steepness of the slopes » (Peace, 1993, p. 285).

7.4.2.3 Détermination de la condition optimale

La condition optimale représente les niveaux optimums des facteurs du plan étudié. Dans l'analyse du ratio S/B, quelque soit la caractéristique de qualité; *l'optimum est une valeur minimale* ou *l'optimum est une valeur maximale,* la tendance est de maximiser le résultat. Ainsi, le niveau optimum d'un facteur représente le niveau du facteur qui possède la plus grande valeur de la moyenne des ratios S/B.

Par exemple, en consultant le Tableau 7.5, le niveau du facteur (A) à choisir est celui qui possède la plus grande valeur entre $\overline{A}_1$ et $\overline{A}_2$. Le même raisonnement est à suivre pour les autres facteurs du plan étudié.

Dans la représentation graphique des moyennes des ratios S/B du plan étudié, le niveau optimum pour chaque facteur représente le niveau ayant le plus haut point dans le graphe.

7.4.2.4 Analyse de la variance

L'analyse de la variance (ANOVA) est une technique statistique initialement développée par le statisticien R. A. Fisher dans les années 1920 et 1930. Habituellement, l'analyse de la variance est appliquée aux résultats de l'expérience afin de déterminer le pourcentage de contribution de chaque facteur (Roy, 1990). Ce pourcentage est utilisé pour évaluer l'importance relative de chaque facteur (Yang et El-Haik, 2003).

Tableau 7.6

Analyse de la variance

Facteurs/ source de variation	Degré de liberté (Dl)	Somme des carrés (SC)	Carré moyen =Variance (MC)	Contribution en (%) (PC)	Ratio de variance (F)
A	Dl_A	SC_A	$MC_A = SC_A / Dl_A$	SC_A / SC_T	MC_A / MC_E
B	Dl_B	SC_B	$MC_B = SC_B / Dl_B$	SC_B / SC_T	MC_B / MC_E
C	Dl_C	SC_C	$MC_C = SC_C / Dl_C$	SC_C / SC_T	MC_B / MC_E
Erreur	Dl_E	SC_E	$MC_E = SC_E / Dl_E$		
Total	Dl_T	SC_T	$MC_T = SC_T / Dl_T$		
Erreur estimée	Dl_{EE}	SC_{EE}	$MC_{EE} = SC_{EE} / Dl_{EE}$		

L'interprétation du tableau de l'analyse de la variance (Tableau 7.6) est comme suit :

- colonne relative au pourcentage de contribution : selon Phadke (1989, p. 58), « The larger the contribution of a particular factor to the total sum of squares, the larger the ability is of that factor to influence S/N ». Ainsi, plus le pourcentage de contribution du facteur est grand, plus grande est son influence sur le résultat;
- colonne relative au ratio de variance F : selon Phadke (1989, p. 58), « A value of F less than one means the factor effect is smaller than the error of the additive model. A value of F larger than two means the factor is not quit small, whereas larger than four means the factor effect is quite large ». Ainsi, plus la valeur de F du facteur est grande, plus l'effet du facteur est grand.

Dans le cas où le nombre de degrés de liberté de l'erreur est égal à zéro et la somme des carrés de l'erreur est égale à zéro, le calcul de la variance de l'erreur (MC_E) et du ratio de variance (F) pour chaque facteur n'est pas possible. Cependant, une estimation de la variance de l'erreur peut être obtenue par le principe de groupement (pooling) des sommes des carrés correspondants aux facteurs ayant le plus faible carré moyen (Phadke, 1989). Cette estimation de la variance de l'erreur est notée par MC_{EE} (dernière ligne du Tableau 7.6). Par la suite, le calcul de l'estimation de la variance de l'erreur et de la valeur de F est possible. Pour le calcul de F, il suffit d'utiliser l'estimation de la variance de l'erreur à la place de l'erreur dans la formule de la dernière colonne du Tableau 7.6.

Par ailleurs, dans le calcul de l'ANOVA, nous allons utiliser l'outil fourni par Prakash R. Apte, professeur en « Reliability Engineering and Electrical Engineering at Indian Institute of Technology at Mumbai » au contexte de nos plans d'analyses. Mr Apte a mis sur son site web (http://www.ee.iitb.ac.in/~apte/) un ensemble de fichiers MS-Excel « Taguchi Excel Template » pour différentes tables orthogonales de Taguchi. Ces fichiers sont disponibles gratuitement au public et une assistance est aussi possible par courriel. Chaque fichier permet d'introduire les données du plan étudié et d'avoir le

tableau de l'ANOVA (Tableau 7.6). Les différents calculs de l'ANOVA sont documentés en détail dans le livre « Quality Engineering Using Robust Design » du docteur Madhav S.Phadke[7].

Puisque nous allons utiliser la table orthogonale L4 de Taguchi, nous avons obtenu par correspondance avec le professeur Apte le ficher MS-Excel qui permettrait de faire les calculs de l'ANOVA pour cette table orthogonale.

7.4.2.5 Calcul de l'équation de prédiction

Une fois la condition optimale déterminée, il faut calculer l'équation de prédiction, désigné par $\hat{\eta}$. Comme le souligne Peace (1993, p. 290), cette équation correspond à « an estimate of the predicted signal-to-noise ratio based on the selected levels of the strong effect ». L'équation de prédiction du ratio signal-bruit prévu à la condition optimale se calcule donc à base des niveaux optimums des facteurs les plus influents.

Il est important de noter que les facteurs utilisés dans l'estimation de la variance de l'erreur (i.e., facteurs avec une faible influence) ne sont pas inclus dans l'équation de prédiction du ratio signal-bruit à la condition optimale (Phadke, 1989; Roy, 1990).

L'équation de prédiction du ratio signal-bruit est obtenue par l'ajout de toutes les contributions des facteurs (influents) aux niveaux optimums à la moyenne globale des ratios S/B du plan étudié :

---

[7] Dr. Phadke pioneered the application and development of the Taguchi Method / Robust Design method in USA and is a recipient of the Taguchi Award, 1985. He has worked closely with Dr. Genichi Taguchi since 1980, and they have co-authored several articles, advancing the field of robust design (http://www.isixsigma.com/library/content/c020311a.asp).

- la contribution du facteur au niveau optimum représente « the amount of improvement obtainable by setting the factor to the desired level » (Roy, 2001, p. 131). Cette contribution représente la différence entre la moyenne des ratios S/B du niveau du facteur optimum et la moyenne globale des ratios S/B du plan étudié;
- la moyenne globale des ratios S/B du plan étudié est égale à la somme des ratios S/B des essais de la matrice d'expériences divisée par le nombre de ratios S/B.

À titre illustratif, si le facteur A du plan « *Pexemple* » est le seul facteur retenu comme facteur influent et son niveau préféré est le niveau 1, l'équation de prédiction à la condition optimale est exprimée comme suit :

$$\boxed{\hat{\eta} = \overline{T} + \left(\overline{A_1} - \overline{T}\right)} \qquad (7.10)$$

avec :

$\overline{T}$: la moyenne globale des ratios S/B du plan « *Pexemple* »;

$\overline{T}$ = (SB1+ SB2+ SB3+ SB4)/4 (voir Tableau 7.4);

$\overline{A_1}$: la moyenne des ratios S/B du facteur A au niveau 1 (voir Tableau 7.5);

$(\overline{A_1} - \overline{T})$: la contribution du facteur A au niveau1 (niveau optimum).

7.4.2.6 Test de confirmation

Selon Peace (1993, p. 238), « the confirmation run is a set of units run together under the optimal conditions determined from the analysis. These conditions include the best or preferred settings for mild and weak influences as well as for strong effects ». Ainsi, le test de confirmation consiste à réaliser un ensemble d'expériences avec la condition optimale déterminée dans la section 7.4.2.3, puis calculer le ratio S/B du test de confirmation. Le résultat obtenu doit être comparé à celui prédit par l'équation $\hat{\eta}$, et « if

the actual result is close to the predicted value, the recommended settings can be implemented » (Peace, 1993, p. 290).

Pour pouvoir juger si le ratio S/B du test de confirmation est près ou loin de la valeur prédite du ratio S/B, il faut déterminer l'intervalle de confiance de l'erreur de prédiction et, par la suite, déterminer l'intervalle de confiance de la valeur prédite du ratio S/B. Ces deux intervalles sont calculés de la manière suivante (Phadke, 2004) :

- l'intervalle de confiance à 95 % de l'erreur de prédiction est exprimé par :

$$\pm 2\sqrt{\left(\frac{1}{n_0}+\frac{1}{n_r}\right)\sigma_e^2} \tag{7.11}$$

$$\frac{1}{n_0}=\frac{1}{n}+\sum_{i=1}^{fopt}\left(\frac{1}{n_i}-\frac{1}{n}\right) \tag{7.12}$$

avec :

« σ_e^2 » : variance de l'erreur;

« n » : nombre d'essais (de lignes) dans la matrice d'expériences;

« fopt » : nombre de facteurs utilisés dans l'équation de prédiction;

« ni » : nombre de fois que le niveau optimum du facteur est répété dans la matrice d'expériences (seuls les facteurs inclus dans l'équation de prédiction);

« nr » : nombre d'expériences de vérification.

- l'intervalle de confiance du résultat du ratio S/B est exprimé par :

$$\hat{\eta} \pm 2\sqrt{\left(\frac{1}{n_0}+\frac{1}{n_r}\right)\sigma_e^2} \tag{7.13}$$

avec :

« $\hat{\eta}$ » : représente l'équation de prédiction (section 7.4.2.5).

7.5 Sommaire

Dans ce chapitre, nous avons préparé de façon globale l'environnement des plans d'analyses empiriques. Cette préparation comprend les données exploitées à partir de l'extrait de données d'ISBSG de février 2006 et les étapes de préparation des échantillons appropriés de projets d'ISBSG des plans d'analyses. Enfin, nous avons présenté en détail les différentes étapes à suivre lors de l'analyse des résultats des plans d'expériences avec la méthode d'analyse du ratio signal-bruit de Taguchi ainsi que les calculs nécessaires.

Il est important de souligner que puisque chaque plan d'analyse fait intervenir un ensemble de facteurs et une caractéristique de qualité, le manque de données limite d'une part le choix des facteurs et des niveaux correspondants et, d'autre part, le choix de la caractéristique de qualité à prendre en considération dans un plan d'analyse : ceci limite l'utilisation des tables orthogonales de Taguchi permettant d'étudier plusieurs facteurs et plusieurs niveaux.

En outre, chaque essai de la table orthogonale (matrice d'expériences) est une combinaison particulière des niveaux de facteurs : le manque de données limite la taille de l'échantillon de projets d'ISBSG du plan étudié et, par conséquent, réduit le nombre de répétitions possibles pour les essais de la matrice d'expériences.

Les chapitres 8, 9, et 10 présentent les plans d'analyses conçus avec la méthode Taguchi pour vérifier les hypothèses des liens entre les trois types de qualité de la norme ISO 9126-1.

CHAPITRE 8

VÉRIFICATION DU LIEN ENTRE LA QUALITÉ INTERNE ET LA QUALITÉ EXTERNE

8.1 Introduction

Dans ce chapitre, nous traitons le premier volet de la phase 3 de la méthodologie de recherche établie dans le chapitre 5 de cette thèse traitant l'objectif 1 de cette recherche, soit le lien entre la qualité interne et la qualité externe du produit logiciel de la Figure 2.1 de la norme ISO 9126-1.

Nous présentons dans un premier temps l'objectif de ce chapitre suivi de l'hypothèse à vérifier, de la caractéristique de qualité à évaluer et de la liste des facteurs à étudier dans ce chapitre. Par la suite, nous présentons trois plans d'analyses conçus avec la méthode Taguchi, lesquels plans portent sur des facteurs principaux sans aucune interaction.

Dans ce chapitre, nous utilisons l'analyse du ratio signal-bruit (S/B) de Taguchi dans le cas où l'objectif est de minimiser la caractéristique de qualité : il s'agit du cas où *l'optimum est une valeur minimale*.

8.2 Objectif

Assurer la qualité du logiciel dès les premières phases du développement du logiciel est un facteur important pour réaliser sa qualité une fois en exécution ou en opération. Kitchenham et Pfleeger (1996, p. 14) soulignent que :

> « Assessing quality by measuring internal properties is attractive because it offers an objective and context-independent view of quality. However, more research is needed to confirm that internal quality assures external quality and to determine which aspects of internal quality affect the product's use. »

L'objectif dans ce chapitre consiste à explorer la relation entre la qualité interne et la qualité externe à travers l'étude des facteurs de qualité interne du produit logiciel pouvant potentiellement avoir un impact ou une influence sur sa qualité externe. Cette exploration est réalisée à travers des plans d'analyses empiriques dont les données exploitées sont fournies par l'organisation ISBSG dans l'extrait de février 2006.

Rappelons que :

- la qualité interne est définie comme étant : « the totality of characteristics of the software product from an internal view » (ISO, 2001a, p. 5). Les mesures de qualité interne « can be applied to a non executable software product (such as a specification or source code) during designing and coding » (ISO, 2001a, p. 15);
- la qualité externe est définie comme étant : « the quality when the software is executed, which is typically measured and evaluated while testing in a simulated environment with simulated data using external metrics » (ISO, 2001a, p. 5).

8.3 Hypothèse

L'hypothèse principale à vérifier dans ce chapitre, et qui correspond à l'objectif 1 de cette recherche, est énoncée comme suit : la qualité interne influence la qualité externe du produit logiciel.

Il est à noter qu'il n'est pas possible de vérifier tous les liens qui peuvent exister entre les différentes caractéristiques (sous-caractéristiques) du modèle de qualité interne et celles du modèle de qualité externe de la norme ISO 9126-1. Il est possible cependant dans cette recherche d'explorer les facteurs de qualité interne disponibles dans l'extrait

de données d'ISBSG de février 2006 et d'examiner leurs influences sur la qualité externe du produit logiciel.

Cette hypothèse est explorée à travers la vérification de l'influence des différents facteurs disponibles de qualité interne sur la caractéristique de qualité externe étudiée. Les sous-hypothèses, relatives aux plans, présentées et vérifiées dans les sections 8.6, 8.7 et 8.8 de ce chapitre ont pour but de vérifier la pertinence ou non de l'hypothèse principale.

8.4 Caractéristique de qualité

En génie logiciel, comme dans les autres disciplines, il est souhaitable de produire un logiciel de qualité tout en minimisant les défauts. À la différence des défauts du produit industriel, ceux du produit logiciel sont invisibles et ils sont souvent détectés durant les phases de tests ou d'opération du logiciel. Plus encore, certains défauts peuvent rester invisibles tant qu'il n'y a pas de circonstances permettant de les activer et d'autres défauts peuvent apparaitre durant la maintenance du logiciel.

Galin (2004) classifie les causes des erreurs du logiciel selon les différentes étapes du processus de développement, à savoir : les erreurs dans la définition des exigences, dans les procédures, dans le code, dans la documentation, etc. De façon générale, durant le cycle de vie du logiciel, deux catégories de défauts se distinguent : les défauts détectés lors de l'exécution du logiciel et ceux détectés lorsqu'il n'est pas exécuté. Selon Kan (2003, p. 119), « with regard to defect data, testing defects are generally more reliable than inspection defects ». En effet, durant la phase de test ou d'opération, les défauts se manifestent essentiellement quand le test ou l'exécution d'une fonction subit un échec ou son résultat est invalide. Les défauts détectés en dehors de toute exécution du logiciel reposent principalement sur le jugement des concepteurs et des développeurs du logiciel et portent généralement sur les erreurs de spécifications de logiciel.

Puisque l'intérêt dans ce chapitre porte sur la qualité externe du logiciel, la caractéristique de qualité, attribut de qualité externe, à examiner est la densité du nombre de défauts détectés lors de l'exécution du logiciel. Il est à noter que cet attribut de qualité permet d'évaluer la sous-caractéristique de qualité « Maturité » de la caractéristique « Fiabilité » du modèle de qualité externe de la norme ISO 9126-1 (Figure 2.2). En outre, les mesures externes de maturité du produit logiciel permettent de mesurer « such attributes as the software freedom of failures caused by faults existing in the software itself » (ISO, 2003a, p. 15). Ainsi, la densité du nombre de défauts (ratio du nombre de défauts et de la taille du logiciel) que présente le produit logiciel, entre autres, exprime son degré de maturité et, par la suite, de fiabilité; plus la densité du nombre de défauts est grande, plus la maturité est faible et plus la densité du nombre de défauts est petite, plus la maturité est grande.

Lequel des champs à choisir ?

En ce qui concerne cette caractéristique de qualité, lors de la conception des plans d'analyses, différents champs de données relatifs au nombre de défauts détectés (i.e. nombre total de défauts tous types confondus) lors de l'exécution du logiciel sont disponibles dans l'extrait d'ISBSG de février 2006, à savoir lors de :

- la phase de test;
- la phase d'implémentation;
- la phase de fin du projet.

La phase de codage est exclue étant donné que les champs relatifs aux défauts collectés durant cette phase regroupent aussi bien les défauts d'inspections que les défauts de tests unitaires (voir note 1 de la section 6.3.1.2 du chapitre 6).

Ainsi, faute de données disponibles dans l'extrait d'ISBSG de février 2006, la caractéristique de sortie, attribut externe de qualité, considérée dans ce chapitre est la

densité du nombre de défauts détectés durant la phase de test. Le but est de minimiser la caractéristique de qualité : la densité du nombre de défauts détectés de la phase de test.

8.5 Liste des facteurs

8.5.1 Diagramme de causes-effets

Nombreux sont les facteurs qui peuvent être considérés comme source de causes dans la génération des défauts de logiciel. Selon Paulk (2005, p. 25) :

> « In a survey of project managers, Schneberger identifies eight factors affecting software failures: 1) requirements change, addition, and definition; 2) programmer / team member experience and turnover; 3) design changes, scope, and complexity; 4) coding and testing phase problems; 5) new technology, languages, and tools; 6) ongoing experience; 7) upper management influence, bidding and time constraints; and 8) lack of data available to use in metrics and models. »

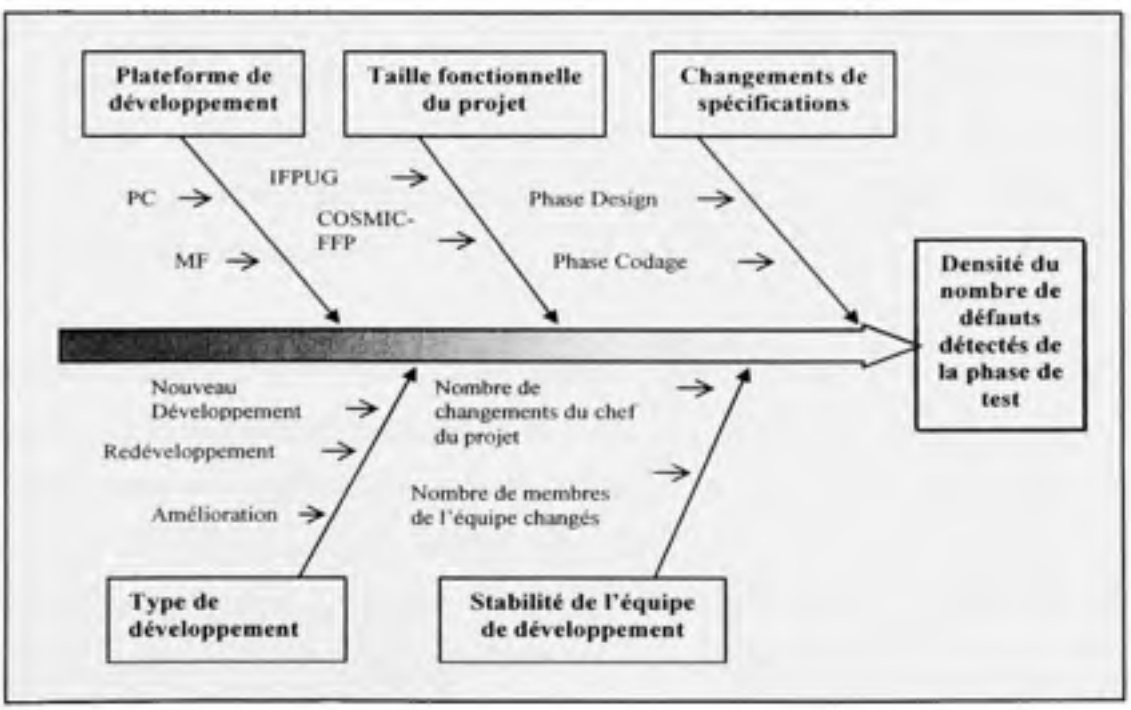

Figure 8.1 ***Diagramme de causes-effets.***

La Figure 8.1 regroupe un sous-ensemble de facteurs pouvant avoir un impact sur la caractéristique de qualité considérée : la densité du nombre de défauts détectés lors de la phase de test. La description de ces facteurs est présentée dans la section suivante. Il est important de noter que le choix des facteurs représentés dans le diagramme de causes-effets est fait en fonction de l'analyse des réponses du questionnaire sur la qualité du produit logiciel (voir Annexe VI, Tableau VI.1) et dans la limite des données disponibles dans l'extrait d'ISBSG de février 2006.

8.5.2 Description des facteurs

8.5.2.1 Taille du projet

Les projets de petites tailles « less than 100 function points, zero-defect levels are possible in terms of the numbers of delivered defects » (Jones, 1996, p. 256), alors que les projets de grandes tailles « larger than 1,000 function points, and especially for software systems larger than 10,000 function points, zero-defect levels are theoretically possible, but in fact, the author has never seen it accomplished in more than 30 years » (Jones, 1996, p. 256). De ce fait, le facteur taille du projet peut avoir un effet sur la qualité du projet : plus le projet est grand plus se présente le risque d'avoir des fautes et d'erreurs lors de son exécution.

Par ailleurs, les différents types de taille utilisés dans ce chapitre sont reconnus comme standards de mesures par l'organisation internationale de standardisation ISO : il s'agit principalement des méthodes de mesures de taille fonctionnelle COSMIC-FFP (ISO, 2003c) et IFPUG-UFP (ISO, 2003d). En outre, pour IFPUG-UFP, les projets à sélectionner représentent ceux dont la taille fonctionnelle mesurée est classifiée comme fiable par les responsables de qualité d'ISBSG; il s'agit des projets ayant le code « UFP Rating = A: The unadjusted FP was assessed as being sound with nothing being identified which might affect its integrity » (ISBSG, 2006a).

Ces deux méthodes de mesures de taille fonctionnelle lorsqu'elles sont prises indépendamment, nous nous retrouvons avec un nombre insuffisant de projets pour mener les expérimentations. Toutefois, comme le soulignent Desharnais, Abran et Cuadrado (2006), les études portant sur la convertibilité entre ces deux types de mesures ont indiqué que la convertibilité peut être simple avec une très bonne corrélation pour la plupart des projets de type système d'information de gestion (MIS). Ainsi, partant de l'hypothèse qu'il y a une convertibilité de près de 1 pour COSMIC-FFP et IFPUG-UFP (Desharnais, Abran et Cuadrado, 2006), nous avons opté pour l'utilisation des deux types de mesure de taille fonctionnelle ensemble afin d'avoir un échantillon de données assez important pour la conception de plans d'expériences avec Taguchi.

8.5.2.2 Changements de spécifications

Les spécifications de logiciel représentent la base de toute conception et leurs changements peuvent se manifester durant toutes les phases du cycle de développement du logiciel. Le changement de spécifications, entre autres, représente un des facteurs principaux dans la génération des défauts.

Selon Pressman (2004, p. 14), « it is true that software requirements change, but the impact of change varies with the time at which it is introduced ». En effet, les changements des exigences du logiciel peuvent être effectués pour diverses raisons, mais ce privilège est fait au dépend de sa qualité. Les changements des exigences effectués au début du cycle de développement du logiciel peuvent ne pas influencer la qualité du produit final. Cependant, plus la demande de changements est faite tardivement dans le cycle plus la modification est coûteuse et les conséquences peuvent être importantes, notamment lors des phases de design, de codage et de test. Plus encore, ce qui est dangereux ce sont les changements lorsque le logiciel est en phase d'opération.

Pour des raisons de disponibilité de données d'ISBSG, nous avons choisi d'étudier séparément les facteurs relatifs au :

- nombre de changements de spécifications de la phase de codage;
- nombre de changements de spécifications de la phase de design.

La densité du nombre de changements de spécifications lors de la phase de codage (ou design) représente le rapport du nombre de changements de spécifications lors de la phase de codage (ou design) par rapport à la taille du projet (section 8.5.2.1).

8.5.2.3 Type de développement

Trois types de développement sont disponibles dans le référentiel d'ISBSG et qui sont décrits dans le glossaire de termes (ISBSG, 2006b) comme suit :

- nouveau développement : une analyse complète du domaine d'application est effectuée, suivi de toutes les étapes du cycle de vie du développement du logiciel (planification/faisabilité, analyse, design, construction et implémentation);
- redéveloppement : concerne une application existante. Les exigences fonctionnelles de l'application sont connues et peuvent nécessiter un changement minime ou pas de changement. Le redéveloppement peut impliquer un changement du matériel ou de la plateforme du logiciel;
- amélioration : correspond aux changements effectués à une application existante par ajout de nouvelles fonctionnalités, suppression ou changement de fonctionnalités existantes.

8.5.2.4 Plateforme de développement

Divers types de plateformes de développement sont utilisés par les projets logiciels d'ISBSG, à savoir : PC (Personnel Computer), MF (Mainframe), MR (Mid Range) et Multiplateforme.

Ces différentes plateformes diffèrent dans la capacité de traitement, le nombre d'utilisateurs (monoposte ou multiposte), etc. Pour des raisons de disponibilité de données d'ISBSG, nous avons choisi pour ce facteur d'étudier les deux premières catégories de plateformes de développement : PC et MF.

8.5.2.5 Stabilité de l'équipe de développement

Le changement des membres de l'équipe peut arriver pour diverses raisons, à savoir : démission, renvoi, etc. La fréquence ou le nombre de changements des membres de l'équipe de développement durant le projet détermine le niveau de stabilité de cette équipe. Il est important de noter que la stabilité de l'équipe peut influencer la conduite de tout le projet et, par la suite, la qualité du produit logiciel durant toutes ses phases de développement.

Selon ISBSG, la stabilité de l'équipe de développement durant le projet est mesurée en fonction du :

- nombre des membres de l'équipe remplacés subitement au cours du projet;
- nombre de changements du chef de projet.

8.6 Plan d'analyse – Cas 1

Ce cas de plan d'analyse explore l'influence de la densité du nombre de changements de spécifications de la phase de codage, du type de développement et de la plateforme de développement sur la densité du nombre de défauts détectés de la phase de test.

8.6.1 Échantillon de projets d'ISBSG du plan

Les projets à sélectionner, à partir de l'échantillon de projets d'ISBSG « BD-ISBSG-AB-Février-2006 » (section 7.3), représentent les projets ayant tous des informations disponibles (exclure les projets n'ayant fourni aucune information « champ vide » ou

avec des expressions comme « Not known », « Some », etc.) pour les trois facteurs de ce plan ainsi que pour sa caractéristique de qualité, à savoir :

- la densité du nombre de changements de spécifications de la phase de codage;
- le type de développement;
- la plateforme de développement;
- la densité du nombre de défauts détectés de la phase de test.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG pour ce plan ne contient que 16 projets.

8.6.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 8.1

Tableau 8.1

Liste des facteurs et leurs niveaux (Cas 1)

Facteurs		Niveaux	Description
Densité du nombre de changements de spécifications - phase codage (DCS-C)	A	1	DCS-C < 0.005
		2	DCS-C ≥ 0.005
Type de développement du logiciel (T-Dev)	B	1	T-Dev = R/A (Redéveloppement /Amélioration)
		2	T-Dev = N (Nouveau développement)
Plateforme de développement (PF-Dev)	C	1	PF-Dev = MF (Main Frame)
		2	PF-Dev = PC (Personnel Computer)

Deux niveaux sont choisis pour chaque facteur de ce plan :

- pour le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur de ce facteur inférieure et supérieure à la médiane[8] de ce facteur dans l'échantillon de projets d'ISBSG de ce plan;
- pour les facteurs relatifs au type de développement (T-Dev) et à la plateforme de développement (PF-Dev), les niveaux 1 et 2 représentent les catégories de ces deux facteurs.

8.6.3 Choix de la table orthogonale du plan

Ce cas de plan comprend trois facteurs sans aucune interaction et chaque facteur possède deux niveaux. Ainsi, le degré de liberté du plan étudié est égal à trois degrés de liberté et la table orthogonale à choisir est L4 (voir section 7.4.1), c'est-à-dire quatre essais. Le Tableau 8.2 montre la matrice d'expériences du plan étudié.

Tableau 8.2

Matrice d'expériences du plan (Cas 1)

Liste des Essais	**Liste des Facteurs du Plan**		
	Densité du nombre de changements de spécifications - phase codage (DCS-C)	**Type de développement (T-Dev)**	**Plateforme de développement (PF-Dev)**
1	< 0.005	R/A	MF
2	< 0.005	N	PC
3	≥ 0.005	R/A	PC
4	≥ 0.005	N	MF

[8] Le choix de la médiane du facteur vient du fait qu'elle partage l'échantillon de données en deux niveaux : les données inférieures à la médiane et celles supérieures. En outre, sa valeur n'est pas affectée par les valeurs extrêmes de l'échantillon de données.

8.6.4 Réalisation

8.6.4.1 Exécution

L'exécution consiste à sélectionner les projets qui répondent aux essais de la matrice d'expériences (Tableau 8.2). Le Tableau 8.3 montre le détail des huit projets sélectionnés (des 16 projets) de l'échantillon de projets d'ISBSG de ce plan et qui répondent aux conditions des essais de la matrice d'expériences de ce plan.

Tableau 8.3

Caractéristiques des projets d'ISBSG (Cas 1)

ID projet	Taille du projet	Nombre de changements de spécifications - phase codage	Densité du nombre de changements de spécifications - phase codage	Type de développement	Plateforme de développement	Densité du nombre défauts - phase test
12922	71	0	0.000	Amélioration	MF	0.113
28338	474	0	0.000	Amélioration	MF	0.101
19730	795	1	0.001	Nouveau développement	PC	0.013
11788	838	0	0.000	Nouveau développement	PC	0.000
27373	74	1	0.014	Amélioration	PC	1.095
17866	61	1	0.016	Amélioration	PC	0.082
31664	916	17	0.019	Nouveau développement	MF	0.159
23918	34	1	0.029	Nouveau développement	MF	0.029

Les autres projets n'ont pas été utilisés pour la simple raison que certains ne correspondent pas aux conditions des essais du Tableau 8.2. Pour les projets qui répondent aux conditions, nous sélectionnons pour chaque essai le nombre de projets qui répondent à l'essai. Le nombre maximal commun pour les quatre essais de l'étude détermine le nombre de répétitions. Par exemple, pour certains essais de ce plan nous

avons trouvé trois projets et pour d'autres nous n'avons trouvé que deux projets, alors le nombre de répétitions est limité à deux : c'est-à-dire, deux projets pour chaque essai de la matrice d'expériences.

Le résultat de l'affectation des projets du Tableau 8.3 aux essais identifiés dans le Tableau 8.2 est présenté dans le Tableau 8.4.

Tableau 8.4

Affectation des projets aux essais (Cas 1)

Liste des Essais	**Liste des Facteurs du Plan**			**Densité du nombre défauts - phase test**	
	Densité du nombre de changements de spécifications - phase codage (DCS-C)	Type de développement (T-Dev)	Plateforme de développement (PF-Dev)	Y1	Y2
1	< 0.005	R/A	MF	0.113	0.101
2	< 0.005	N	PC	0.013	0.000
3	≥ 0.005	R/A	PC	1.095	0.082
4	≥ 0.005	N	MF	0.159	0.029

Pour l'essai numéro1 du Tableau 8.2, les projets correspondants sont les deux premiers projets (12922 et 28338) du Tableau 8.3. Les réponses Y1 et Y2 (deux dernières colonnes du Tableau 8.4) représentent leurs valeurs respectives de la densité du nombre de défauts de la phase de test (dernière colonne du Tableau 8.3), soit 0.113 et 0.101.

Les projets suivants (19730 et 11788) correspondent au deuxième essai du Tableau 8.2 avec leurs réponses Y1 = 0.013 et Y2 = 0.000.

Le même scénario est à suivre pour l'affectation des projets aux essais 3 et 4 de la matrice d'expériences de ce plan.

8.6.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Puisque l'objectif, entre autres, dans un cycle de développement de logiciel est de minimiser la densité du nombre de défauts détectés lors de la phase de test, alors l'équation à utiliser dans le calcul du ratio signal-bruit de ce plan et celle proposée par Taguchi dans le cas où *l'optimum est une valeur minimale*; il s'agit de la formule (7.3). Par exemple, du Tableau 8.4, le calcul du ratio signal-bruit pour le premier essai est fait de la manière suivante :

$$\boxed{S/B = -10 \log \left[\frac{1}{2} \left((Y1)^2 + (Y2)^2 \right) \right]} \tag{8.1}$$

$$\boxed{S/B = -10 \log \left[\frac{1}{2} \left((0.113)^2 + (0.101)^2 \right) \right]} \tag{8.2}$$

$$\boxed{S/B = 19.40} \tag{8.3}$$

Le calcul des ratios S/B des autres essais (lignes) se fait de la même façon. Le résultat du calcul des ratios S/B est présenté dans le Tableau 8.5.

Tableau 8.5

Calcul des ratios S/B (Cas 1)

Liste des Essais	Liste des Facteurs du Plan			Densité du nombre défauts -phase test		Ratio Signal -bruit
	Densité du nombre de changements de spécifications-phase codage (DCS-C)	Type de développement (T-Dev)	Plateforme de développement (PF-Dev)	Y1	Y2	S/B
1	< 0.005	R/A	MF	0.113	0.101	19.40
2	< 0.005	N	PC	0.013	0.000	40.73
3	≥ 0.005	R/A	PC	1.095	0.082	2.20
4	≥ 0.005	N	MF	0.159	0.029	18.84

2) Calcul des effets des facteurs

Dans ce cas de plan, chaque facteur possède deux niveaux. Ainsi, l'effet de chaque facteur est obtenu en calculant la différence entre la moyenne des ratios S/B au niveau 1 et la moyenne des ratios S/B au niveau 2 de chaque facteur (section 7.4.2.2). Le Tableau 8.6 présente les moyennes des ratios S/B pour les deux niveaux des facteurs du plan et les effets de ces facteurs.

Tableau 8.6

Calcul des effets des facteurs (Cas 1)

	Densité du nombre de changements de spécifications - phase codage (DCS-C)	Type de développement (T-Dev)	Plateforme de développement (PF-Dev)
Moyenne des ratios S/B au niveau 1	30.07	10.80	19.12
Moyenne des ratios S/B au niveau 2	10.52	29.79	21.46
Effet du facteur (Delta)	19.55	18.99	2.35

La représentation graphique des niveaux des facteurs du plan et leurs valeurs de la moyenne des ratios S/B (Tableau 8.6) est présentée dans la Figure 8.2. La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 8.5 : (20.29).

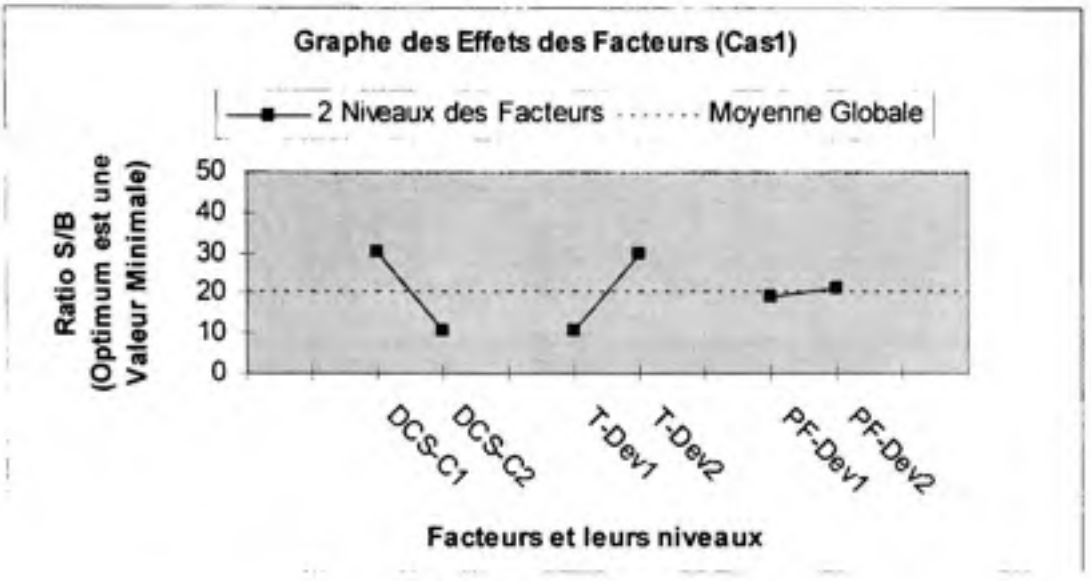

Figure 8.2 ***Graphe des effets des facteurs (Cas 1).***

Du Tableau 8.6 et de la Figure 8.2, il apparait que les facteurs de ce plan présentent des effets différents sur la densité du nombre de défauts détectés lors de la phase de test :

- la plus importante influence correspond au facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C), avec un effet de 19.55;
- l'influence de moindre importance est celle du facteur relatif au type de développement (T-Dev), avec un effet de 18.99;
- la faible influence est celle du facteur relatif à la plateforme de développement (PF-Dev), avec un effet de 2.35.

3) Détermination de la condition optimale

La condition optimale inclut les niveaux des facteurs ayant la plus grande valeur de la moyenne des ratios S/B. Ainsi, du Tableau 8.6 (calcul des effets des facteurs) et de la Figure 8.2 (graphe des effets des facteurs), les niveaux optimums pour les facteurs de ce plan sont les suivants :

- DCS-C au niveau 1 : DCS-C < 0.005;
- PF-Dev au niveau 2 : PF-Dev = PC;
- T-Dev au niveau 2 : T-Dev = N.

La condition optimale est donc : la densité du nombre de changements de spécifications de la phase de codage est inférieure à 0.005, la plateforme de développement correspond à PC et le type de développement est un nouveau développement.

4) Analyse de la variance

Le Tableau 8.7 présente le calcul de l'analyse de la variance pour ce cas de plan.

Tableau 8.7

Analyse de la variance (Cas 1)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DCS-C	1	382.05	382.05	51	69
T-Dev	1	360.53	360.53	48	66
PF-Dev	1	5.50	5.50	1	
Erreur	0	0			
Total	3	748.08		100	
Erreur estimée	1	5.50	5.50		

De l'analyse de la variance (Tableau 8.7), colonne pourcentage de contribution, il est à noter que (voir Figure 8.3) :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C) présente le grand pourcentage de contribution avec 51% à la variabilité de la densité du nombre de défauts détectés lors de la phase de test;
- le facteur relatif au type de développement (T-Dev) a une contribution moins grande avec 48%;
- le facteur relatif à la plateforme de développement (PF-Dev) ne contribue que de 1% : faible contribution.

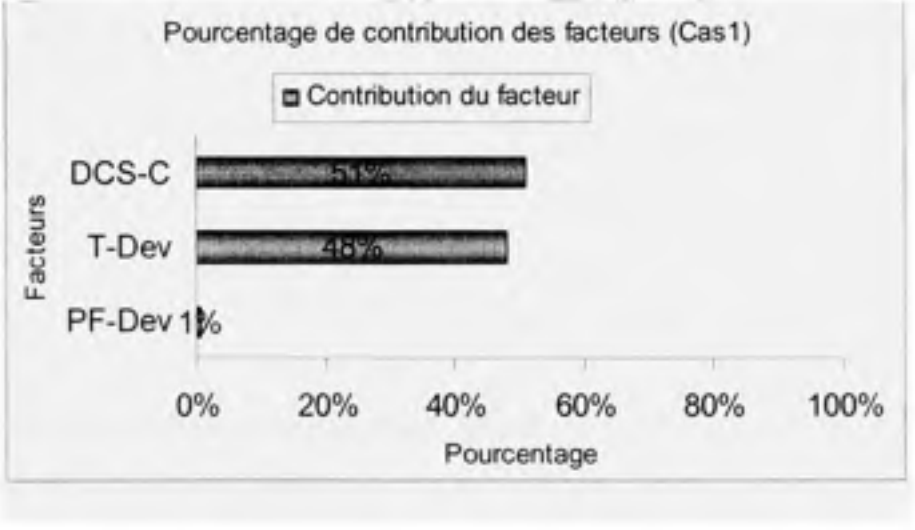

Figure 8.3 ***Pourcentage de contribution des facteurs (Cas 1).***

En outre, de l'analyse de la colonne du ratio de variance F, il est à noter que :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C) présente un effet dominant avec la valeur de F égale à 69;
- le facteur relatif au type de développement (T-Dev) a aussi un effet dominant avec la valeur de F = 66, mais moins important que le facteur (DCS-C).

Le facteur relatif à la plateforme de développement (PF-Dev) a été utilisé (pooling) dans l'estimation de la variance de l'erreur.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 8.7), seuls les facteurs (DCS-C) et (T-Dev) sont retenus comme facteurs ayant des effets importants. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 2. Ainsi, l'équation de prédiction du ratio signal-bruit selon la condition optimale pour ce cas de plan est calculée en fonction seulement de ces deux facteurs.

Le résultat prévu à la condition optimale (section 7.4.2.5) est :

$$\hat{\eta} = \overline{T} + \left(\overline{DCS\ -C_1} - \overline{T}\right) + \left(\overline{T - Dev_2} - \overline{T}\right) \tag{8.4}$$

$$\hat{\eta} = 39.56 \text{ avec } \overline{T} = 20.29 \tag{8.5}$$

Le facteur (PF-Dev) n'est pas inclus dans l'équation de prédiction puisqu'il a été utilisé pour estimer la variance de l'erreur.

6) Test de confirmation

Le test de confirmation consiste à effectuer un ensemble d'expériences selon la condition optimale déterminée (dans l'étape 3 de la section 8.6.4.2) pour ce plan et calculer le ratio signal-bruit du test de confirmation. Pour ce cas de plan, vu l'insuffisance de projets disponibles dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale (un seul projet), le test de confirmation ne peut être effectué.

8.6.5 Synthèse

Dans ce plan, il résulte de l'analyse du ratio signal-bruit que la densité du nombre de changements de spécifications lors de la phase de codage, comme facteur de qualité interne du produit logiciel, est le plus influent des trois facteurs suivi du facteur type de développement, puis du facteur plateforme de développement sur la caractéristique de qualité de ce plan. Cette caractéristique de qualité correspond à la densité du nombre de défauts détectés lors de la phase de test du produit logiciel, considéré comme attribut de qualité externe du produit logiciel.

Ainsi, l'hypothèse est confirmée : la qualité interne du produit (mesurée dans ce plan par la variable « Densité du nombre de changements de spécifications de la phase de codage ») influence la qualité externe du produit logiciel.

8.7 Plan d'analyse – Cas 2

Ce plan permet d'étudier l'influence de la densité du nombre de changements de spécifications de la phase de codage, de la taille du projet et de la stabilité de l'équipe de développement sur la densité du nombre de défauts détectés de la phase de test.

8.7.1 Échantillon de projets d'ISBSG du plan

Dans ce cas de plan, les projets à sélectionner doivent tous avoir des données disponibles (exclure les projets n'ayant fourni aucune information « champ vide » ou contiennent des expressions : « Not known », « Some », etc.) non seulement pour les trois facteurs de ce plan, mais aussi pour sa caractéristique de sortie, à savoir :

- la densité du nombre de changements de spécifications de la phase de codage;
- la taille du projet;
- la stabilité de l'équipe de développement;
- la densité du nombre de défauts détectés de la phase de test.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG (obtenu à partir de l'échantillon de données d'ISBSG « BD-ISBSG-AB-Février-2006 », section 7.3) pour ce plan ne contient que 27 projets.

8.7.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 8.8.

Tableau 8.8

Liste des facteurs et leurs niveaux (Cas 2)

Facteurs		Niveaux	Description
Densité du nombre de changements de spécifications – phase codage (DCS-C)	A	1	DCS-C ≤ 0.005
		2	DCS-C > 0.005
Taille du projet (Taille)	B	1	Taille ≤ 184
		2	Taille > 184
Stabilité de l'équipe de développement (Stabilité)	C	1	Stabilité = S (Stable)
		2	Stabilité = IS (Instable)

Deux niveaux sont choisis pour chaque facteur de ce plan :

- pour le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur de ce facteur inférieure et supérieure à la médiane de ce facteur dans l'échantillon de projets d'ISBSG de ce plan. De même que pour le facteur relatif à la taille du projet (Taille);
- pour le facteur relatif à la stabilité de l'équipe de développement (Stabilité), nous admettons qu'une équipe est :

- stable (S), si elle n'a pas subi de changements ni de ses membres ni du chef du projet (niveau 1), c'est-à-dire la somme des nombres de changements aussi bien des membres que du chef du projet est inférieure à un;
- instable (IS), si elle a subi au moins un changement que ce soit de ses membres ou du chef du projet (niveau 2), c'est-à-dire la somme des nombres de changements aussi bien des membres que du chef du projet est supérieure ou égale à un.

Il est à noter que la médiane du facteur (Stabilité) est égale à un dans l'échantillon de projets d'ISBSG de ce plan.

8.7.3 Choix de la table orthogonale du plan

Ce cas de plan d'analyse comprend trois facteurs avec deux niveaux chacun et sans aucune interaction. Ainsi, le degré de liberté du plan étudié est égal à trois degrés de liberté (voir section 7.4.1) et la table orthogonale à choisir est L4. Le Tableau 8.9 montre la matrice d'expériences du plan étudié.

Tableau 8.9

Matrice d'expériences du plan (Cas 2)

Liste des Essais	**Liste des Facteurs du Plan**		
	Densité du nombre de changements de spécifications - phase codage (DCS-C)	**Taille du projet (Taille)**	**Stabilité de l'équipe (Stabilité)**
1	≤ 0.005	≤ 184	S
2	≤ 0.005	> 184	IS
3	> 0.005	≤ 184	IS
4	> 0.005	> 184	S

8.7.4 Réalisation

8.7.4.1 Exécution

Le Tableau 8.10 montre le détail des huit projets choisis (des 27 projets) de l'échantillon de projets d'ISBSG de ce plan répondant aux conditions des essais du Tableau 8.9. Les autres projets n'ont pas été utilisés pour les mêmes contraintes de la section 8.6.4.1.

Tableau 8.10

Caractéristiques des projets d'ISBSG (Cas 2)

ID projet	Taille du projet	Nombre de changements de spécifications – phase codage	Densité du nombre changements de spécifications – phase codage	Changement des membres de l'équipe et chef de projet	Densité du nombre défauts - phase test
18204	126	0	0.000	0	0.000
30333	125	0	0.000	0	0.008
19730	795	1	0.001	1	0.013
28161	200	0	0.000	1	0.060
31237	146	1	0.007	1	0.116
29311	177	4	0.023	1	0.169
12598	367	3	0.008	0	0.082
22168	848	53	0.063	0	0.035

À chaque essai de la matrice d'expériences de ce plan ne correspond que deux projets d'ISBSG. Ainsi, le même scénario (voir section 8.6.4.1) est à suivre pour ce cas de plan lors de l'affectation des projets du Tableau 8.10 aux essais du Tableau 8.9. Le résultat de cette affectation est présenté dans le Tableau 8.11.

Tableau 8.11

Affectation des projets aux essais (Cas 2)

Liste des Essais	Liste des Facteurs du Plan			Densité du nombre défauts - phase test	
	Densité du nombre de changements de spécifications - phase codage (DCS-C)	Taille du projet (Taille)	Stabilité de l'équipe (Stabilité)	Y1	Y2
1	≤ 0.005	≤ 184	S	0.000	0.008
2	≤ 0.005	> 184	IS	0.013	0.060
3	> 0.005	≤ 184	IS	0.116	0.169
4	> 0.005	> 184	S	0.082	0.035

8.7.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Le calcul du ratio signal-bruit, effectué avec la formule (7.3), pour chaque essai du Tableau 8.11 est présenté dans le Tableau 8.12.

Tableau 8.12

Calcul des ratios S/B (Cas 2)

Liste des Essais	Liste des Facteurs du Plan			Densité du nombre défauts -phase test		Ratio signal-bruit
	Densité du nombre de changements de spécifications - phase codage (DCS-C)	Taille du projet (Taille)	Stabilité de l'équipe (Stabilité)	Y1	Y2	S/B
1	≤ 0.005	≤ 184	S	0.000	0.008	44.95
2	≤ 0.005	> 184	IS	0.013	0.060	27.25
3	> 0.005	≤ 184	IS	0.116	0.169	16.78
4	> 0.005	> 184	S	0.082	0.035	24.01

2) Calcul des effets des facteurs

Le calcul des effets (voir section 7.4.2.2) des trois facteurs étudiés dans ce cas de plan est présenté dans le Tableau 8.13.

Tableau 8.13

Calcul des effets des facteurs (Cas 2)

	Densité du nombre de changements de spécifications - phase codage (DCS-C)	**Taille du projet (Taille)**	**Stabilité de l'équipe (Stabilité)**
Moyenne des ratios S/B au niveau 1	36.10	30.86	34.48
Moyenne des ratios S/B au niveau 2	20.39	25.63	22.01
Effet du facteur (Delta)	15.17	5.23	12.47

La Figure 8.4 représente graphiquement les deux niveaux de chaque facteur du plan et leurs valeurs correspondantes de la moyenne des ratios S/B (Tableau 8.13).

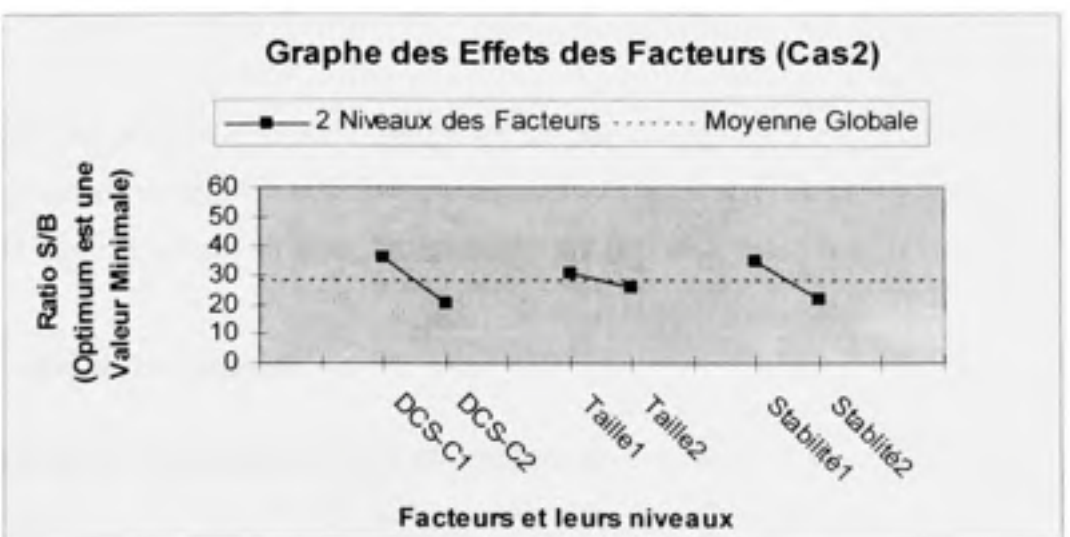

Figure 8.4 ***Graphe des effets des facteurs (Cas 2).***

La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 8.12 : (28.24).

Du Tableau 8.13 et de la Figure 8.4, il est à remarquer que les facteurs de ce plan ont des effets différents sur la densité du nombre de défauts détectés lors de la phase de test :

- une influence importante du facteur relatif à la densité du nombre de changements de spécifications de la phase codage (DCS-C) avec un effet de 15.71;
- une influence de moindre importance du facteur relatif à la stabilité de l'équipe de développement (Stabilité) avec un effet de 12.47;
- une faible influence du facteur relatif à la taille du projet (Taille) avec un effet de 5.23.

3) Détermination de la condition optimale

Du Tableau 8.13 et de la Figure 8.4, les niveaux optimums pour les facteurs de ce plan sont les suivants :

- DCS-C au niveau 1 : DCS-C $\leq$ 0.005;
- Taille au niveau 1 : Taille $\leq$ 184;
- Stabilité au niveau 1 : Stabilité = S.

La condition optimale est donc : la densité du nombre de changements de spécifications de la phase de codage est inférieure ou égale à 0.005, la taille du projet est inférieure ou égale à 184 et l'équipe de développement est stable.

4) Analyse de la variance

Le Tableau 8.14 présente le calcul de l'analyse de la variance pour ce cas de plan.

Tableau 8.14

Analyse de la variance (Cas 2)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DCS-C	1	246.70	246.70	58	9
Taille	1	27.40	27.40	6	
Stabilité	1	155.40	155.40	36	6
Erreur	0	0			
Total	3	429.50		100	
Erreur estimée	1	27.40	27.40		

De l'analyse de la variance (Tableau 8.14), en ce qui concerne le pourcentage de contribution, il apparait que (Figure 8.5) :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C) présente la plus grande contribution avec 58% à la variabilité de la densité du nombre de défauts détectés lors de la phase de test;
- le facteur relatif à la stabilité de l'équipe de développement (Stabilité) présente une contribution moins importante avec 36%;
- le facteur relatif à la taille du projet (Taille) présente la plus faible contribution avec 6%.

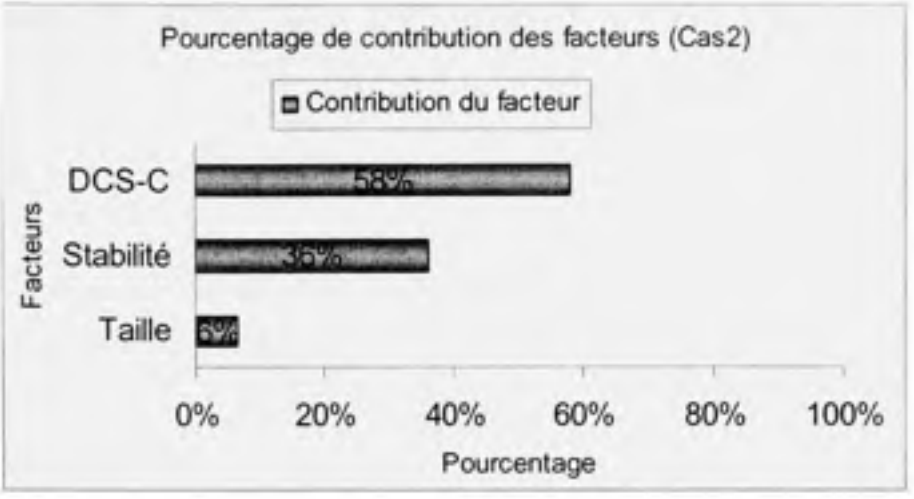

Figure 8.5 ***Pourcentage de contribution des facteurs (Cas 2).***

D'un autre côté, de l'analyse de la colonne relative au ratio de variance F, il est à remarquer que :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage (DCS-C) a un effet dominant avec la valeur du ratio F égale à 9;
- le facteur relatif à la stabilité de l'équipe de développement (Stabilité) a un effet aussi dominant puisque la valeur du ratio F est égale à 6.

Le facteur relatif à la taille du projet (Taille) est utilisé pour constituer (pooling) la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 8.14), seuls les facteurs (DCS-C) et (Stabilité) sont retenus comme facteurs les plus influents. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 1. Ainsi, l'équation de prédiction à la condition optimale de ce plan est calculée seulement en fonction de ces deux facteurs.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{DCS\ -\ C_1} - \overline{T}\right) + \left(\overline{Stabilité\ _1} - \overline{T}\right) \quad (8.6)$$

$$\hat{\eta} = 42.33 \text{ avec } \overline{T} = 28.24 \quad (8.7)$$

Le facteur (Taille) étant utilisé dans l'estimation de la variance de l'erreur, il n'est pas inclus dans l'équation de prédiction.

6) Test de confirmation

Deux projets existent dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale (déterminée dans l'étape 3 de la section 8.7.4.2), regroupés dans le Tableau 8.15.

Tableau 8.15

Projets d'ISBSG - Test de confirmation (Cas 2)

ID projet	Taille du projet	Nombre de changements de spécifications – phase codage	Densité du nombre changements de spécifications – phase codage	Changement des membres de l'équipe et chef de projet	Densité du nombre défauts - phase test
12922	71	0	0.000	0	0.113
13403	184	1	0.005	0	0.054

Le ratio signal-bruit pour le test de confirmation calculé avec la formule (7.3) est : S/B = (21.06).

Par ailleurs, le nombre d'essais de la matrice d'expériences est quatre, le nombre de fois que le niveau 1 du facteur (DCS-C) est répété dans la matrice d'expériences est deux, le nombre de fois que le niveau 1 du facteur (Stabilité) est répété dans la matrice d'expériences est deux, le nombre des expériences de vérification est deux et la variance de l'erreur est égale à 27.40. Ainsi, en utilisant les formules (7.11) et (7.13) :

- l'intervalle de confiance à 95 % de l'erreur de prédiction est : (± 11.71);
- l'intervalle de confiance du résultat du ratio S/B est : [30.63, 54.04].

Le ratio S/B du test de confirmation n'existe pas dans l'intervalle de confiance du résultat du ratio S/B. Ce résultat peut être expliqué par le manque de données qui a peut-être influencé le choix des facteurs et de la caractéristique de qualité de ce cas de plan.

8.7.5 Synthèse

Dans ce plan, il résulte de l'analyse du ratio signal-bruit que la densité du nombre de changements de spécifications de la phase de codage, comme facteur de qualité interne du produit logiciel, est le plus influent des trois facteurs suivi de la stabilité de l'équipe de développement, puis de la taille du projet sur la caractéristique de qualité choisie. Cette caractéristique de qualité correspond à la densité du nombre de défauts détectés lors de la phase de test, considéré comme attribut de qualité externe du produit logiciel.

Ainsi, l'hypothèse est confirmée : la qualité interne du produit (mesurée dans ce plan par la variable « Densité du nombre de changements de spécifications de la phase de codage ») influence la qualité externe du produit logiciel.

8.8 Plan d'analyse – Cas 3

Dans ce cas de plan, nous traitons de l'influence de la densité du nombre de changements de spécifications de la phase de design, du type de développement et de la taille du projet sur la densité du nombre de défauts détectés lors de la phase de test.

8.8.1 Échantillon de projets d'ISBSG du plan

À partir de l'échantillon de données d'ISBSG « BD-ISBSG-AB-Février-2006 » (section 7.3), les projets à sélectionner doivent tous avoir des informations disponibles pour les trois facteurs de ce plan ainsi que pour sa caractéristique de qualité, à savoir :

- la densité du nombre de changements de spécifications de la phase de design;
- le type de développement;
- la taille du projet;
- la densité du nombre de défauts détectés de la phase de test.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG pour ce plan ne contient que 22 projets.

8.8.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 8.16.

Tableau 8.16

Liste des facteurs et leurs niveaux (Cas 3)

Facteurs		Niveaux	Description
Densité du nombre de changements de spécifications – phase design (DCS-D)	A	1	DCS-D < 0.005
		2	DCS-D ≥ 0.005
Type de développement du logiciel (T-Dev)	B	1	T-Dev= R/A (Redéveloppement /Amélioration)
		2	T-Dev= N (Nouveau développement)
Taille du projet (Taille)	C	1	Taille ≤ 200
		2	Taille > 200

Deux niveaux sont choisis pour chaque facteur :

- pour les facteurs relatifs à la taille du projet (Taille) et à la densité du nombre de changements de spécifications de la phase de design (DCS-D), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur du facteur inférieure et supérieure à la valeur voisinage[9] de la médiane du facteur dans l'échantillon de projets d'ISBSG de ce plan;
- pour le facteur relatif au type de développement (T-Dev), les niveaux 1 et 2 représentent les catégories de ce facteur.

8.8.3 Choix de la table orthogonale du plan

Ce plan d'analyse comprend trois facteurs avec deux niveaux chacun. Ainsi, le degré de liberté de ce plan est égal à trois degrés de liberté (section 7.4.1) et la table orthogonale à choisir est L4. La matrice d'expériences de ce plan est présentée dans le Tableau 8.17.

Tableau 8.17

Matrice d'expériences du plan (Cas 3)

Liste des Essais	**Liste des Facteurs du Plan**		
	Densité du nombre de changements de spécifications - phase design (DCS-D)	**Type de développement (T-Dev)**	**Taille du projet (Taille)**
1	< 0.005	R/A	≤ 200
2	< 0.005	N	> 200
3	≥ 0.005	R/A	> 200
4	≥ 0.005	N	≤ 200

[9] Dans le cas où nous ne trouvons pas de projets pour les essais de la matrice d'expériences avec la médiane du facteur, nous procédons au choix de la valeur voisinage de la médiane.

8.8.4 Réalisation

8.8.4.1 Exécution

Le Tableau 8.18 montre le détail des huit projets choisis (des 22 projets) de l'échantillon de projets d'ISBSG de ce plan et qui répondent aux conditions des essais identifiés dans le Tableau 8.17. Les autres projets n'ont pas été utilisés pour les mêmes contraintes du premier plan (Cas 1).

Tableau 8.18

Caractéristiques des projets d'ISBSG (Cas 3)

ID projet	Taille du projet	Nombre de changements de spécifications - phase design	Densité du nombre changements de spécifications - phase design	Type de développement	Densité du nombre défauts - phase test
12596	28	0	0.000	Amélioration	0.000
30333	125	0	0.000	Amélioration	0.008
11788	838	0	0.000	Nouveau Développement	0.000
19730	795	1	0.001	Nouveau Développement	0.013
22168	848	20	0.024	Amélioration	0.035
29644	490	3	0.006	Amélioration	0.049
28020	186	1	0.005	Nouveau Développement	0.000
23918	34	5	0.147	Nouveau Développement	0.029

Pour ce cas de plan aussi, il n'y a que deux projets pour chaque essai. Ainsi, l'affectation des projets du Tableau 8.18 aux essais du Tableau 8.17 est effectuée de la même manière que le premier plan (section 8.6.4.1). Le Tableau 8.19 montre le résultat de cette affectation.

Tableau 8.19

Affectation des projets aux essais (Cas 3)

Liste des Essais	Liste des Facteurs du Plan			Densité du nombre défauts -phase test	
	Densité du nombre de changements de spécifications - phase design (DCS-D)	Type de développement (T-Dev)	Taille du projet (Taille)	Y1	Y2
1	< 0.005	R/A	≤ 200	0.000	0.008
2	< 0.005	N	> 200	0.013	0.000
3	≥ 0.005	R/A	> 200	0.049	0.035
4	≥ 0.005	N	≤ 200	0.029	0.000

8.8.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Le calcul du ratio signal-bruit, effectué avec la formule (7.3), pour chaque essai du Tableau 8.19 est présenté dans le Tableau 8.20.

Tableau 8.20

Calcul des ratios S/B (Cas 3)

Liste des Essais	Liste des Facteurs du Plan			Densité du nombre défauts -phase test		Ratio signal-bruit
	Densité du nombre de changements de spécifications - phase design (DCS-D)	Type de développement (T-Dev)	Taille du projet (Taille)	Y1	Y2	S/B
1	< 0.005	R/A	≤ 200	0.000	0.008	44.95
2	< 0.005	N	> 200	0.013	0.000	40.73
3	≥ 0.005	R/A	> 200	0.049	0.035	27.42
4	≥ 0.005	N	≤ 200	0.029	0.000	33.76

2) Calcul des effets des facteurs

Ce plan étudié comprend trois facteurs et chaque facteur possède deux niveaux. Le calcul des effets (voir section 7.4.2.2) des trois facteurs de ce plan est présenté dans le Tableau 8.21.

Tableau 8.21

Calcul des effets des facteurs (Cas 3)

	Densité du nombre de changements de spécifications - phase design (DCS-D)	**Type de développement (T-Dev)**	**Taille du projet (Taille)**
Moyenne des ratios S/B au niveau 1	42.84	36.18	39.36
Moyenne des ratios S/B au niveau 2	30.59	37.25	34.07
Effet du facteur (Delta)	12.25	1.06	5.28

La Figure 8.6 présente graphiquement les deux niveaux de chaque facteur du plan et leurs valeurs correspondantes de la moyenne des ratios S/B (Tableau 8.21). La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 8.20 : (36.71).

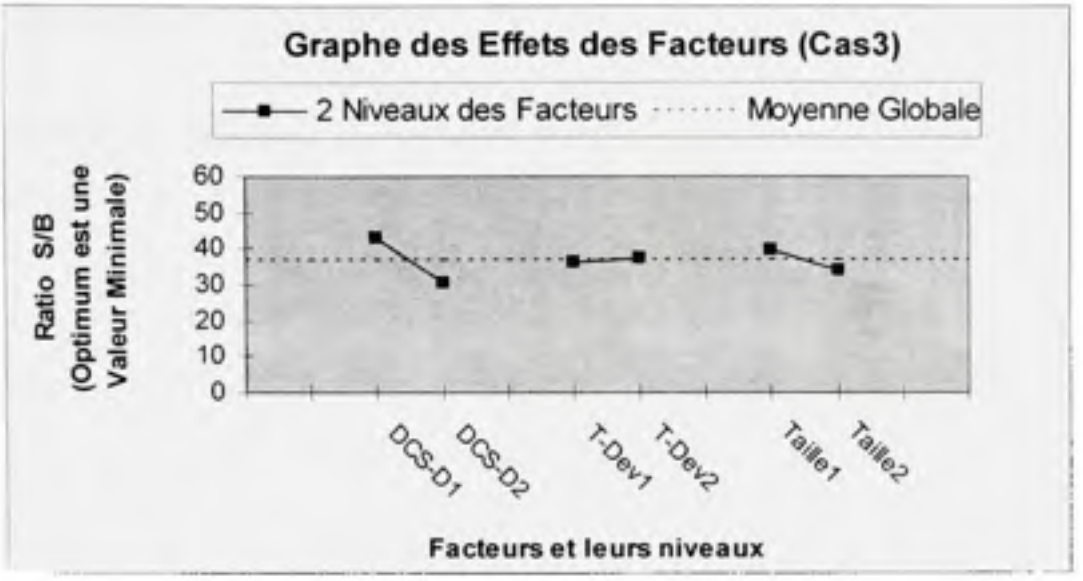

Figure 8.6 ***Graphe des effets des facteurs (Cas 3).***

Du Tableau 8.21 et de la Figure 8.6, il est à remarquer que les facteurs de ce plan présentent des effets différents sur la densité du nombre de défauts détectés lors de la phase de test du produit logiciel :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de design (DCS-D) présente le plus important effet, soit 12.25;
- le facteur relatif à la taille du projet (Taille) vient en deuxième position d'influence avec un effet moins important, soit 5.28;
- le facteur relatif au type de développement (T-Dev) vient en troisième position d'influence avec un faible effet, soit 1.06.

3) Détermination de la condition optimale

Du graphe des effets des facteurs (Figure 8.6) et du Tableau 8.21, les niveaux optimums pour l'ensemble des facteurs de ce plan sont les suivants :

- DCS-D au niveau 1 : DCS-D < 0.005;
- Taille au niveau 1 : Taille $\leq$ 200;

- T-Dev au niveau 2 : T-Dev = N.

La condition optimale est donc : la densité du nombre de changements de spécifications de la phase de design est inférieure à 0.005, la taille du projet est inférieure ou égale à 200 et le type de développement est un nouveau développement.

4) Analyse de la variance

Le Tableau 8.22 présente le calcul de l'analyse de la variance pour ce cas de plan.

Tableau 8.22

Analyse de la variance (Cas 3)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DCS-D	1	150.08	150.08	84	132
T-Dev	1	1.13	1.13	1	
Taille	1	27.90	27.90	15	25
Erreur	0	0			
Total	3	179.11		100	
Erreur estimée	1	1.13	1.13		

De la colonne contribution de l'analyse de la variance, il est à noter que (Figure 8.7) :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de design (DCS-D) contribue pour pratiquement 84% à la variabilité de la densité du nombre de défauts détectés lors de la phase de test du produit logiciel;
- le facteur relatif à la taille du projet (Taille) vient en deuxième position avec une contribution de 15%;
- le facteur relatif au type de développement (T-Dev) ne contribue que de 1% : contribution négligeable.

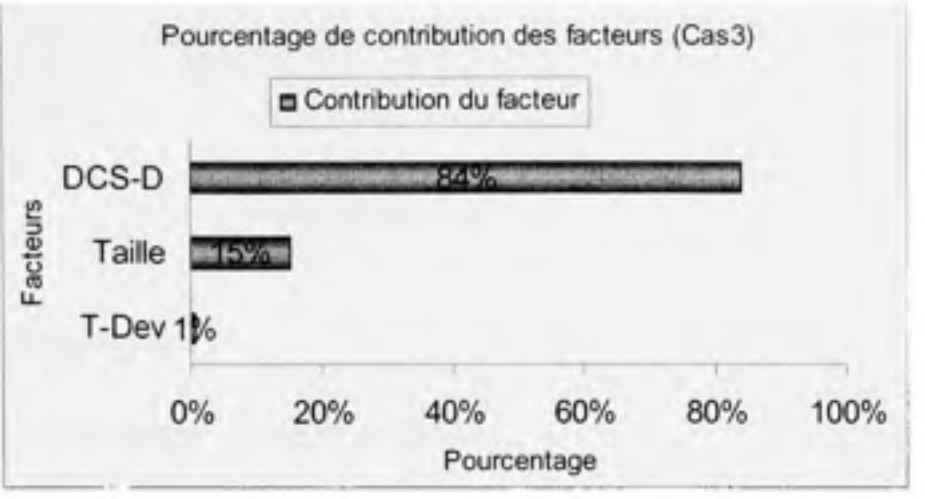

Figure 8.7 ***Pourcentage de contributions des facteurs (Cas 3).***

D'un autre côté, de l'analyse de la colonne relative au ratio de variance F, il est à noter que :

- le facteur relatif à la densité du nombre de changements de spécifications de la phase de design (DCS-D) a un effet dominant avec la valeur de F égale à 132;
- le facteur relatif à la taille du projet (Taille) a un effet moins dominant que le facteur (DCS-D) avec la valeur de F = 25.

Le facteur relatif au type de développement (T-Dev) a été utilisé (pooling) pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 8.22), seuls les facteurs (DCS-D) et (Taille) sont retenus comme facteurs ayant des effets importants. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 1. Ainsi, l'équation de prédiction à la condition optimale pour ce plan est effectuée seulement en fonction de ces deux facteurs. Le facteur (T-Dev) a été utilisé pour estimer la variance de l'erreur.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{DCS_{1}} - \overline{T}\right) + \left(\overline{Taille_{1}} - \overline{T}\right) \tag{8.8}$$

$$\hat{\eta} = 45.48 \text{ avec } \overline{T} = 36.71 \tag{8.9}$$

6) Test de confirmation

Le test de confirmation consiste à effectuer un ensemble d'expériences selon la condition optimale déterminée (dans l'étape 3 de la section 8.8.4.2) pour ce plan et calculer le ratio signal-bruit du test de confirmation. Pour ce cas de plan, vu l'insuffisance de projets disponibles dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale (un seul projet), le test de confirmation ne peut être effectué.

8.8.5 Synthèse

Dans ce plan, il résulte de l'analyse du résultat de ce plan que la densité du nombre de changements de spécifications de la phase de design, comme facteur de qualité interne du produit logiciel, est le plus influent des trois facteurs suivi de la taille du projet, puis du type de développement sur la caractéristique de qualité choisie. Cette caractéristique de qualité correspond à la densité du nombre de défauts détectés lors de la phase de test du produit logiciel, considéré comme attribut de qualité externe du produit logiciel.

Ainsi, l'hypothèse est confirmée : la qualité interne du produit (mesurée dans ce plan par la variable « Densité du nombre de changements de spécifications de la phase de design ») influence la qualité externe du produit logiciel.

8.9 Remarques

Il est à remarquer que la solution optimale déterminée pour chaque plan d'analyse conçu avec la méthode Taguchi donne les niveaux préférables pour chaque facteur du plan. Ces niveaux sont à prendre en considération lors de la réalisation d'un nouveau produit logiciel afin d'assurer une bonne qualité vis-à-vis de la caractéristique de sortie souhaitée. Cependant, implémenter la solution optimale n'est pas toujours facile vu la variabilité des contraintes du projet logiciel. La solution optimale donne une valeur informative aux développeurs de logiciels sur les pratiques à prendre en considération et celles à éviter lors d'un nouveau projet logiciel.

Par ailleurs, à la différence du génie logiciel, la solution optimale est importante dans les processus de production de produits industriels, c'est-à-dire qu'avant de commencer le processus de production d'un produit, il faut initialiser les facteurs selon la condition optimale afin d'assurer l'obtention du résultat souhaité. Cependant, ce processus ne peut s'appliquer lors du développement du produit logiciel pour plusieurs raisons, à savoir : la nature des intrants, du processus lui-même et de la sortie. En effet, le génie logiciel inclut un processus de développement qui produit un produit « soft » et qui dépend du domaine d'application du projet, des contraintes du projet, etc., alors que le domaine industriel inclut un processus de production qui produit un produit physique « hard » et qui est répétable selon les mêmes conditions.

8.10 Conclusion

Dans ce chapitre, nous avons traité le premier volet de la phase 3 de la méthodologie de recherche établie dans le chapitre 5 de cette thèse : vérifier l'hypothèse du lien entre la qualité interne du produit logiciel et sa qualité externe de la Figure 2.1 de la norme ISO 9126-1. Pour ce faire, nous avons présenté trois plans d'analyses conçus avec la méthode Taguchi tout en exploitant les données de l'extrait d'ISBSG de février 2006, mis à notre disposition pour des fins de recherche.

Comme l'objectif des plans de ce chapitre est de minimiser la caractéristique de qualité : le nombre de défauts détectés lors de la phase de test du logiciel, l'analyse des résultats de ces plans a été effectuée avec la méthode d'analyse du ratio signal-bruit de Taguchi dans le cas où *l'optimum est une valeur minimale*.

Des résultats des deux premiers plans, il en ressort que le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage a un grand impact sur la densité du nombre de défauts détectés lors de la phase de test du produit logiciel. Du résultat du troisième plan, il en ressort que le facteur relatif à la densité du nombre de changements de spécifications de la phase de design présente aussi un grand impact sur la densité du nombre de défauts détectés de la phase de test du produit logiciel. En outre, il en résulte aussi dans ce chapitre que le niveau le plus préférable est d'avoir une faible densité du nombre de changements de spécifications lors de la phase de codage/design afin de minimiser la densité du nombre de défauts détectés lors de la phase de test du produit logiciel.

Ainsi, l'hypothèse principale établie dans ce chapitre est confirmée puisque :

- la densité du nombre de changements de spécifications lors de la phase de codage (design) représente une mesure de qualité interne du produit logiciel. Cette mesure permet d'évaluer la sous-caractéristique de qualité « Aptitude » de la caractéristique de qualité « Capacité fonctionnelle » du modèle de qualité interne d'ISO 9126-1;
- la densité du nombre de défauts détectés lors de la phase de test du logiciel représente une mesure de qualité externe du produit logiciel. Cette mesure permet d'évaluer la sous-caractéristique de qualité « Maturité » de la caractéristique de qualité « Fiabilité » du modèle de qualité externe d'ISO 9126-1.

CHAPITRE 9

VÉRIFICATION DU LIEN ENTRE LA QUALITÉ EXTERNE ET LA QUALITÉ EN UTILISATION

9.1 Introduction

Dans ce chapitre, nous traitons le deuxième volet de la phase 3 de la méthodologie de recherche (chapitre 5). Ce volet traite de l'objectif 2 de cette recherche, soit le lien entre la qualité externe et la qualité en utilisation du produit logiciel de la Figure 2.1 de la norme ISO 9126-1.

Dans un premier temps, nous présentons l'objectif et l'hypothèse principale à vérifier, suivi de la caractéristique de qualité et de la liste des facteurs à étudier. Par la suite, nous présentons les trois plans d'analyses empiriques conçus avec la méthode Taguchi. Ces plans portent sur des facteurs principaux sans aucune interaction.

Dans ce chapitre, nous utilisons la méthode d'analyse du ratio signal-bruit de Taguchi dans le cas où l'objectif est de maximiser la caractéristique de qualité; il s'agit de l'analyse du ratio signal-bruit où *l'optimum est une valeur maximale*.

9.2 Objectif

La qualité du logiciel final est l'un des buts importants aussi bien de l'équipe de développement que de l'utilisateur final. D'après ISO 9126-1 (2001a, p. 4), « The goal is not necessarily to achieve perfect quality, but the necessary and sufficient quality for each specified context of use when the product is delivered and actually used by users ».

L'objectif dans ce chapitre consiste à explorer la relation entre la qualité externe et la qualité en utilisation du produit logiciel. L'intérêt est de déterminer les facteurs de

qualité externe du produit logiciel ayant un impact sur sa qualité en utilisation à travers des plans d'analyses empiriques, et ce, en exploitant les données de l'extrait d'ISBSG dans sa version de février 2006.

Rappelons que :

- la qualité externe représente « The quality when the software is executed, which is typically measured and evaluated while testing in a simulated environment with simulated data using external metrics » (ISO, 2001a, p. 5);
- la qualité en utilisation représente « the user's view of the quality of an environment containing software, and is measured from the results of using the software in the environment, rather than properties of the software itself » (ISO, 2001a, p. 12).

9.3 Hypothèse

L'hypothèse principale à vérifier dans ce chapitre, et qui correspond à l'objectif 2 de cette recherche, est énoncée comme suit : la qualité externe influence la qualité en utilisation du produit logiciel.

Dans l'optique de l'hypothèse formulée, il n'est pas possible de cerner ou s'attendre à trouver ou déterminer tous les liens entre la qualité externe et la qualité en utilisation. Toutefois, il est possible dans cette recherche d'explorer les facteurs de qualité externe du logiciel disponibles dans l'extrait d'ISBSG de février 2006 et de vérifier s'ils influencent la qualité en utilisation du logiciel.

Cette hypothèse est explorée à travers la vérification de l'influence des différents facteurs disponibles de qualité externe sur la caractéristique de qualité en utilisation considérée. Les sous-hypothèses établies et vérifiées dans les plans d'analyses des sections 9.6, 9.7 et 9.8 de ce chapitre visent la vérification de la pertinence ou non de l'hypothèse principale.

9.4 Caractéristique de qualité

La satisfaction de l'utilisateur retient l'attention de la communauté des chercheurs et des praticiens du logiciel. En effet, selon Jones (1996, p. 347), « Quality, reliability, and user satisfaction are turning out to be the driving force of competition for high-technology products ». En outre, d'après Kan (2003, p. 375), « Customer satisfaction is the ultimate validation of quality. Product quality and customer satisfaction together form the total meaning of quality ».

Les sondages sur la satisfaction des utilisateurs sont généralement utilisés pour mesurer le degré de satisfaction ou d'insatisfaction des utilisateurs vis-à-vis du projet livré; ces sondages permettent ensuite de déterminer l'écart entre la qualité réalisée et les attentes des utilisateurs. Le résultat du sondage constitue d'une part une base d'aide à l'évaluation des attentes de qualité, des forces et des faiblesses du produit logiciel et, d'autre part, un déterminant d'améliorations à entreprendre à court terme ou à long terme, c'est-à-dire lors d'une nouvelle version du produit logiciel (redéveloppement ou amélioration) ou lors d'un nouveau développement.

Par ailleurs, la qualité externe et la qualité en utilisation peuvent être considérées respectivement comme la vérification et la validation du produit logiciel, autrement dit : lors de l'évaluation de la qualité externe l'objectif est placé sur l'exécution correctement du logiciel et sur la réalisation des exigences préalablement définies, alors que lors de l'évaluation de la qualité en utilisation l'important, entre autres, c'est la satisfaction de ces exigences dans le but de satisfaire l'utilisateur final.

La caractéristique de sortie considérée, comme attribut de qualité en utilisation, est le degré de satisfaction des utilisateurs. Le but est de maximiser la caractéristique de qualité, c'est-à-dire maximiser le degré de satisfaction des utilisateurs.

La caractéristique de qualité relative au degré de satisfaction des utilisateurs, considérée comme attribut de qualité en utilisation, est obtenue à partir du sondage de la satisfaction de l'utilisateur collecté lors de la phase de fin du projet du questionnaire d'ISBSG. Il est à noter que cet attribut de qualité permet d'évaluer la caractéristique de qualité « Satisfaction » du modèle de qualité en utilisation de la norme ISO 9126-1 (Figure 2.3). En outre, les mesures de satisfaction du produit logiciel permettent d'évaluer « the user's attitudes towards the use of the product in a specified context of use » (ISO, 2004, p. 11). De ce fait, l'attitude de l'utilisateur vis-à-vis du logiciel utilisé exprime son degré de satisfaction : en effet, un produit logiciel qui ne satisfait pas le pourquoi de son développement est plausible de ne pas satisfaire les attentes de ses utilisateurs finaux.

Comment déterminer le degré de satisfaction des utilisateurs ?

Le sondage relatif à la satisfaction de l'utilisateur dans le questionnaire d'ISBSG comprend huit questions. L'ensemble des champs de données, associés à ces questions, dans l'extrait de données d'ISBSG de février 2006 est présenté dans le Tableau 9.1.

Tableau 9.1

Sondage relatif à la satisfaction de l'utilisateur

Extrait d'ISBSG de février 2006 (ISBSG, 2006a)	**Questionnaire d'ISBSG (ISBSG, 2006c)**
Meet stated objectives	Did the project meet stated objectives?
Meet business requirement	Did the software meet business requirements?
Quality of functionality	Quality expectations for the software?
Quality of documentation	Quality expectations for the user documentation?
Ease of use	Ease-of-use requirements for the software?
Training given	Was sufficient training or explanation given?
Speed of defining solution	Schedule for planning & specification?
Speed of providing solution	Schedule for design, build, test & implement?

Il est à remarquer qu'il y a une légère différence dans les appellations des champs de données dans l'extrait par rapport à leurs questions correspondantes dans le questionnaire de collecte de données d'ISBSG (Tableau 9.1). Dans ce chapitre et les chapitres qui suivent, nous allons faire référence à ces huit champs de données par ceux de l'extrait d'ISBSG (première colonne).

De l'analyse de l'échelle (ordinale) utilisée pour les réponses aux questions du sondage, il en ressort que les réponses suivent une échelle sémantique et numérique ordonnée en fonction du degré de satisfaction pour chaque question : du moins satisfait (1) au plus satisfait (4) (ISBSG, 2006a) :

- 1-Met to limited extent, or not at all/1- poorly met, or not at all;
- 2-Mostly met/2- Largely met;
- 3-Fully met;
- 4-Exceeded expectations.

Ce type d'échelle est connu sous le nom de l'échelle Likert. La somme des réponses de plusieurs variables de type échelle Likert est admissible (Jung, Kim et Chung, 2004).

Ainsi, pour déterminer le degré de satisfaction des utilisateurs par projet par rapport à toutes les questions du sondage, il faut faire la somme des réponses (i.e. le numéro associé à l'échelle) pour les huit items du sondage par projet. À titre d'exemple, si un projet (*P1*) possède les réponses suivantes aux items du sondage :

- « Meet stated objectives: 1- Met to limited extent, or not at all »;
- « Meet business requirement: 2- Mostly met »;
- « Quality of functionality: 3- Fully met »;
- « Quality of documentation: 3- Fully met »;
- « Ease of use: 3- Fully met »;
- « Training given: 4- Exceeded expectations »;
- « Speed of defining solution: 4- Exceeded expectations »;

- « Speed of providing solution: 1- Met to limited extent, or not at all ».

Alors, le degré de satisfaction des utilisateurs par rapport à tous les items du sondage pour le projet (*P1*) est égal à 21 (=1+2+3+3+3+4+4+1) sur un total maximum potentiel de 8x4 = 32.

Cependant, puisque tous les champs n'ont pas nécessairement de données disponibles dans la base de données d'ISBSG (beaucoup de champs sans réponse), l'évaluation de la satisfaction des utilisateurs se fera par rapport aux items du sondage (Tableau 9.1) avec le plus d'informations. Par exemple, si la caractéristique de qualité choisie pour le projet (*P1*) correspond au degré de satisfaction des utilisateurs par rapport aux deux premiers items du sondage (Tableau 9.1). Les réponses à ces deux items sont comme suit :

- « Meet stated objectives: 1- Met to limited extent, or not at all »;
- « Meet business requirement: 2- Mostly met ».

Alors, le degré de satisfaction des utilisateurs par rapport à ces deux items du sondage pour le projet (*P1*) représente la somme des réponses (i.e. le numéro de l'échelle) pour ces deux items, soit 3 (=1+2) sur un total maximum potentiel de 2x4 = 8.

9.5 Liste des facteurs

9.5.1 Diagramme de causes-effets

Le diagramme de causes-effets de la Figure 9.1 rassemble des facteurs qui peuvent être comme source d'influence sur la caractéristique de qualité : degré de satisfaction des utilisateurs. La description de ces facteurs est présentée dans la section suivante.

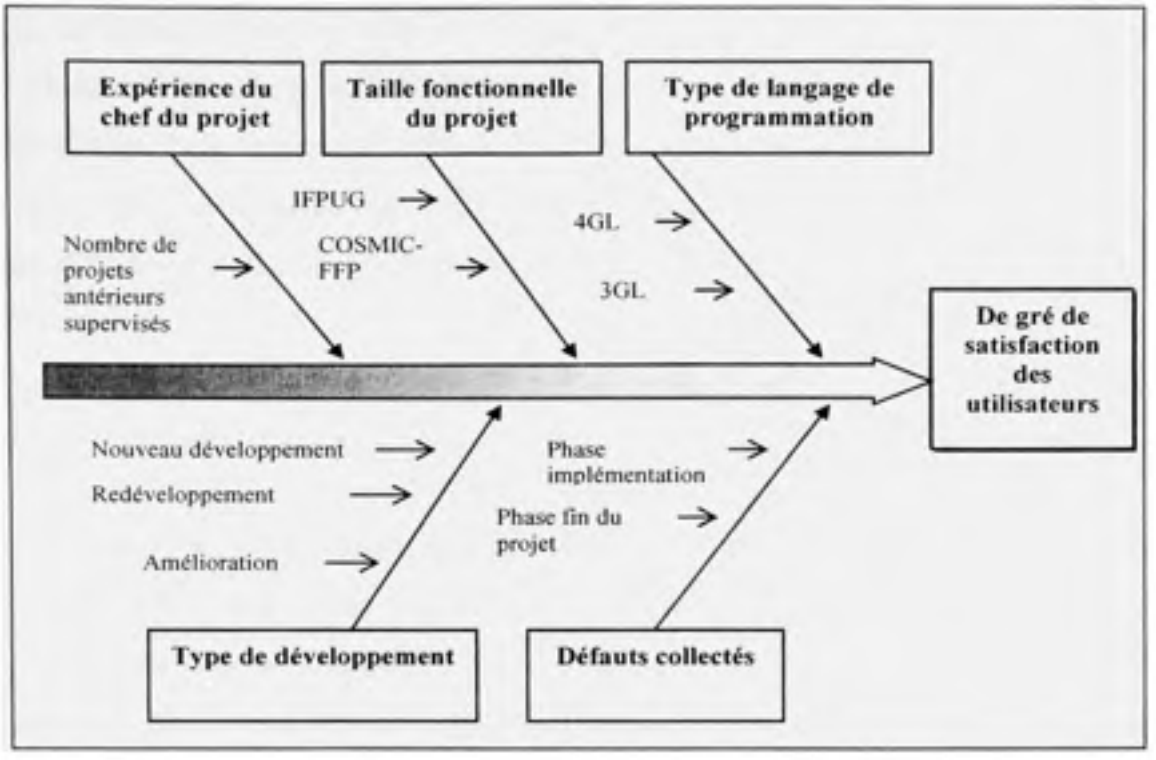

Figure 9.1 ***Diagramme de causes-effets.***

Il est important de noter que le choix des facteurs du diagramme de causes-effets est fait en fonction de l'analyse des réponses du questionnaire sur la qualité du produit logiciel (voir Annexe VI, Tableau VI.2) et dans la limite des données disponibles dans l'extrait d'ISBSG de février 2006.

9.5.2 Description des facteurs

9.5.2.1 Taille du projet

Pour ce facteur, nous allons utiliser la taille fonctionnelle du projet, laquelle taille est exprimée en points de fonction et est mesurée avec l'une des deux méthodes de mesure de taille fonctionnelle : IFPUG-UFP ou COSMIC-FFP. Cependant, compte tenu du nombre insuffisant de projets pour mener les expérimentations et partant du fait que le

ratio de convertibilité est près de 1 entre ces deux mesures (Desharnais, Abran et Cuadrado, 2006), nous avons opté pour l'utilisation de ces deux types de mesure de taille fonctionnelle ensemble (voir section 8.5.2.1). Nous allons donc utiliser les projets dont la taille est mesurée avec COSMIC-FFP et les projets dont la taille est mesurée avec IFPUG-UFP et qui sont classifiés comme fiables par les responsables de qualité d'ISBSG.

9.5.2.2 Défauts collectés

Les défauts relatifs à la qualité externe correspondent aux défauts qui apparaissent lors de l'exécution du logiciel en deux moments; avant et après sa livraison au client durant les phases de tests et d'opération du logiciel. Comparativement aux défauts collectés avant la livraison, ceux collectés une fois le logiciel est chez le client affectent la qualité du produit logiciel du point de vue du client. C'est pourquoi il est préférable de réduire ce type de défauts afin de maximiser la satisfaction de l'utilisateur final. En outre, d'après Kan (2003, p. 97), « to improve customer satisfaction, one has to reduce defects and overall problems and, in addition, manage factors of broader scope ». Ainsi, la réduction du nombre de défauts représente, entre autres, un des moyens pour améliorer la satisfaction de l'utilisateur du logiciel.

D'après Jones (1996), même si les données de satisfaction et les données des défauts sont collectées à différents intervalles par différentes équipes et pour différents objectifs, il est extrêmement utile de combiner ces deux mesures au moins une fois par an. Jones (1996, p. 394) a aussi identifié les différents cas suivants :

- High level defects & High level user satisfaction: Zone of urgent repairs;
- High level defects & Low level user satisfaction: Zone of urgent replacement;
- Low level defects & High level user satisfaction: Zone of excellent applications;
- Low level defects & Low level user satisfaction: Zone of functional enhancement.

Selon ISO 9126 (2004, p. 11), « satisfaction is influenced by the user's perception of properties of the software product (such as those measured by external metrics) and by the user's perception of the efficiency, productivity and safety in use ».

Pour ce facteur, nous étudions indépendamment :

- le nombre de défauts détectés (i.e. nombre total de défauts tous types confondus) lors de la phase d'implémentation du logiciel d'ISBSG, laquelle phase inclut l'interaction avec les utilisateurs finaux durant l'installation, la documentation et la formation;
- le nombre de défauts détectés (i.e. nombre total de défauts tous types confondus) lors du premier mois d'opération du logiciel de la phase de fin du projet d'ISBSG.

La densité du nombre de défauts détectés lors du premier mois d'opération du logiciel de la phase de fin du projet d'ISBSG (d'implémentation) représente le rapport du nombre de défauts détectés lors du premier mois d'opération du logiciel de la phase de fin du projet (d'implémentation) par rapport à la taille du projet (section 9.5.2.1).

9.5.2.3 Expérience du chef du projet

La réussite d'un projet repose, entre autres, sur l'équipe du projet et sur son expertise. Il est admis que l'expertise du chef du projet contribue à l'efficacité du travail, au respect du planning et des échéances, à l'amélioration de la qualité du projet durant toutes les phases du cycle de développement du logiciel et, par la suite, à la production de logiciels de qualité.

Selon ISBSG, l'expérience du chef du projet représente un facteur important dans la performance du projet et elle est mesurée en fonction du nombre de projets antérieurs (IT et non IT) gérés par le chef du projet responsable de la majorité du projet.

9.5.2.4 Type de langage de programmation

Dans l'extrait d'ISBSG, le facteur type de langage de programmation comprend les deux types suivants :

- langage de troisième génération (3GL) : est utilisé dans le développement de logiciel en général, à savoir : JAVA, etc.;
- langage de quatrième génération (4GL) : est une évolution du 3GL, spécialisé dans des domaines bien particuliers comme les bases de données et caractérisé par la facilité d'utilisation.

Ces deux types de langages sont utilisés dans l'industrie dépendamment des besoins, des ressources et du domaine d'application.

9.5.2.5 Type de développement

Un projet peut consister en un nouveau développement du logiciel ou un redéveloppement ou une amélioration d'un logiciel déjà existant. Ces différents types sont décrits dans la section 8.5.2.3 du chapitre 8.

9.6 Plan d'analyse – Cas 1

Ce plan analyse l'influence de la densité du nombre de défauts détectés durant la phase d'implémentation, du type de langage de programmation et de l'expérience du chef du projet sur le degré de satisfaction des utilisateurs.

Pour ce cas de plan d'analyse, le degré de satisfaction des utilisateurs inclut seulement six items parmi ceux du sondage du questionnaire d'ISBSG (voir Tableau 9.1), à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;

- la facilité d'utilisation;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

9.6.1 Échantillon de projets d'ISBSG du plan

Pour des raisons de qualité de données, l'échantillon de travail a été limité aux projets de type A et B, noté par « BD-ISBSG-AB-Février-2006 » (voir section 7.3). À partir de cet échantillon, les projets à sélectionner doivent tous avoir des informations disponibles (exclure les projets contenant des champs vides ou des expressions comme : « Not applicable », etc.) aussi bien pour les facteurs de ce plan que pour sa caractéristique de qualité, à savoir :

- la densité du nombre de défauts détectés durant la phase d'implémentation;
- le type de langage de programmation;
- l'expérience du chef du projet;
- le degré de satisfaction des utilisateurs.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG pour ce plan ne contient que 10 projets.

9.6.2 Liste des facteurs et leurs niveaux

La liste des facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 9. 2.

Tableau 9.2

Liste des facteurs et leurs niveaux (Cas 1)

Facteurs		Niveaux	Description
Densité du nombre de défauts détectés - phase d'implémentation du logiciel (DDéfauts-Imp)	A	1	DDéfauts-Imp $\leq$ 0.006
		2	DDéfauts-Imp > 0.006
Type de langage de programmation (T-Lang)	B	1	T-Lang = 3GL
		2	T-Lang = 4GL
Expérience du chef du projet (Exp-CP)	C	1	Exp-CP = 1 à 9
		2	Exp-CP > 9

Deux niveaux sont choisis pour chaque facteur de ce plan :

- pour le facteur relatif à la densité du nombre de défauts détectés durant la phase d'implémentation (DDéfauts-Imp), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur de ce facteur inférieure et supérieure à la médiane de ce facteur dans l'échantillon de projets d'ISBSG de ce plan;
- pour le facteur relatif à l'expérience du chef du projet (Exp-CP), les niveaux 1 et 2 représentent une valeur voisinage de la médiane de ce facteur;
- pour le facteur relatif au type de langage de programmation (T-Lang), les niveaux 1 et 2 représentent les catégories de ce facteur.

9.6.3 Choix de la table orthogonale du plan

La table orthogonale à utiliser pour ce plan dépend du degré de liberté du plan étudié, lequel degré est calculé en fonction du nombre de facteurs de ce plan, soit trois et le nombre de niveaux des facteurs, soit deux niveaux par facteur. Ainsi, dans ce cas de plan, le degré de liberté est égal à trois degrés de liberté (voir section 7.4.1) et la table orthogonale à choisir est : L4 (quatre essais). Le Tableau 9.3 montre la matrice d'expériences de ce plan.

Tableau 9.3

Matrice d'expériences du plan (Cas 1)

Liste des Essais	Liste des Facteurs du Plan		
	Densité du nombre de défauts détectés - phase implémentation (DDéfauts-Imp)	**Type de langage (T-Lang)**	**Expérience du chef du projet (Exp-CP)**
1	≤ 0.006	3GL	1 à 9
2	≤ 0.006	4GL	> 9
3	> 0.006	3GL	> 9
4	> 0.006	4GL	1 à 9

9.6.4 Réalisation

9.6.4.1 Exécution

L'exécution de la matrice d'expériences consiste à sélectionner les projets correspondants aux conditions des quatre essais du Tableau 9.3.

Le Tableau 9.4 montre le détail des huit projets choisis (des 10 projets) de l'échantillon de projets d'ISBSG de ce plan répondant aux conditions des essais de la matrice d'expériences de ce plan.

Les autres projets n'ont pas été utilisés pour les raisons suivantes :

- pour certains projets, les données des facteurs de ce plan ne correspondent pas aux conditions des essais de la matrice d'expériences (Tableau 9.3);
- pour les projets satisfaisant les conditions, nous sélectionnons pour chaque essai le nombre de projets qui répondent à l'essai. Le nombre maximal commun pour les quatre essais de l'étude détermine le nombre de répétitions, c'est-à-dire le nombre de projets pour chaque essai de l'étude.

Tableau 9.4

Caractéristiques des projets d'ISBSG (Cas 1)

ID projet	Taille du projet	Nombre de défauts détectés- phase implémentation	Densité du nombre de défauts - phase implémentation	Type de langage	Expérience du chef projet (nombre de projets gérés)	Degré de satisfaction des utilisateurs
31237	146	0	0.000	3GL	3	21
29311	177	0	0.000	3GL	6	18
28020	186	0	0.000	4GL	10	16
19730	795	0	0.000	4GL	20	16
28553	788	139	0.176	3GL	12	14
27560	199	38	0.191	3GL	12	14
26093	655	8	0.012	4GL	8	13
10687	55	6	0.109	4GL	1	13

Le résultat de l'affectation des projets du Tableau 9.4 aux essais du Tableau 9.3 est présenté dans le Tableau 9.5.

Tableau 9.5

Affectation des projets aux essais (Cas 1)

Liste des Essais	Liste des Facteurs du Plan			Degré satisfaction des utilisateurs	
	Densité du nombre de défauts détectés - phase implémentation (DDéfauts-Imp)	Type de langage (T-Lang)	Expérience du chef du projet (Exp-CP)	Y1	Y2
1	≤ 0.006	3GL	1 à 9	21	18
2	≤ 0.006	4GL	> 9	16	16
3	> 0.006	3GL	> 9	14	14
4	> 0.006	4GL	1 à 9	13	13

Les deux premiers projets du Tableau 9.4 sont affectés au premier essai du Tableau 9.3 puisqu'ils répondent aux conditions de cet essai. Les valeurs Y1 et Y2 représentent les valeurs de la caractéristique de qualité des deux projets.

Le même scénario est à suivre pour l'affectation des autres projets aux autres essais (2 à 4) de la matrice d'expériences de ce plan.

9.6.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Puisque l'objectif, entres autres, une fois le logiciel est livré, est de maximiser le degré de satisfaction des utilisateurs, alors l'équation du ratio signal-bruit à utiliser pour ce plan est celle proposée par Taguchi dans le cas *où l'optimum est une valeur maximale*; il s'agit de la formule (7.5).

Par exemple, du Tableau 9.5, le calcul du ratio signal-bruit pour le premier essai est fait de la manière suivante :

$$\boxed{S/N = -10\log\left[\frac{1}{2}\left(\left(\frac{1}{Y1^2}\right)+\left(\frac{1}{Y2^2}\right)\right)\right]} \qquad (9.1)$$

$$\boxed{S/N = -10\log\left[\frac{1}{2}\left(\frac{1}{21^2}+\frac{1}{18^2}\right)\right]} \qquad (9.2)$$

$$\boxed{S/N = 25.72} \qquad (9.3)$$

Le calcul des ratios S/B des autres essais (lignes) se fait de la même façon. Le résultat est présenté dans le Tableau 9.6.

Tableau 9.6

Calcul des ratios S/B (Cas 1)

Liste des Essais	Liste des Facteurs du Plan			Degré de satisfaction des utilisateurs		Ratio signal-Bruit
	Densité du nombre de défauts détectés - phase implémentation (DDéfauts-Imp)	Type de langage (T-Lang)	Expérience du chef du projet (Exp-CP)	Y1	Y2	S/B
1	≤ 0.006	3GL	1 à 9	21	18	25.72
2	≤ 0.006	4GL	> 9	16	16	24.08
3	> 0.006	3GL	> 9	14	14	22.92
4	> 0.006	4GL	1 à 9	13	13	22.28

2) Calcul des effets des facteurs

Ce cas de plan comprend trois facteurs avec deux niveaux chacun. Ainsi, pour calculer l'effet d'un facteur, il faut tout d'abord calculer la moyenne des ratios S/B pour chaque niveau du facteur. Par la suite, l'effet du facteur représente la différence entre les deux moyennes des ratios S/B (niveau 1 et niveau 2). Le calcul des effets des facteurs de ce plan est présenté dans le Tableau 9.7.

Tableau 9.7

Calcul des effets des facteurs (Cas 1)

	Densité du nombre de défauts détectés - phase implémentation (DDéfauts-Imp)	Type de langage (T-Lang)	Expérience du chef du projet (Exp-CP)
Moyenne des ratios S/B au niveau 1	24.90	24.32	24.00
Moyenne des ratios S/B au niveau 2	22.60	23.18	23.50
Effet du facteur (Delta)	2.30	1.14	0.50

La Figure 9.2 présente graphiquement les moyennes des ratios S/B des deux niveaux des facteurs de ce plan (Tableau 9.7). La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 9.6 : (23.75).

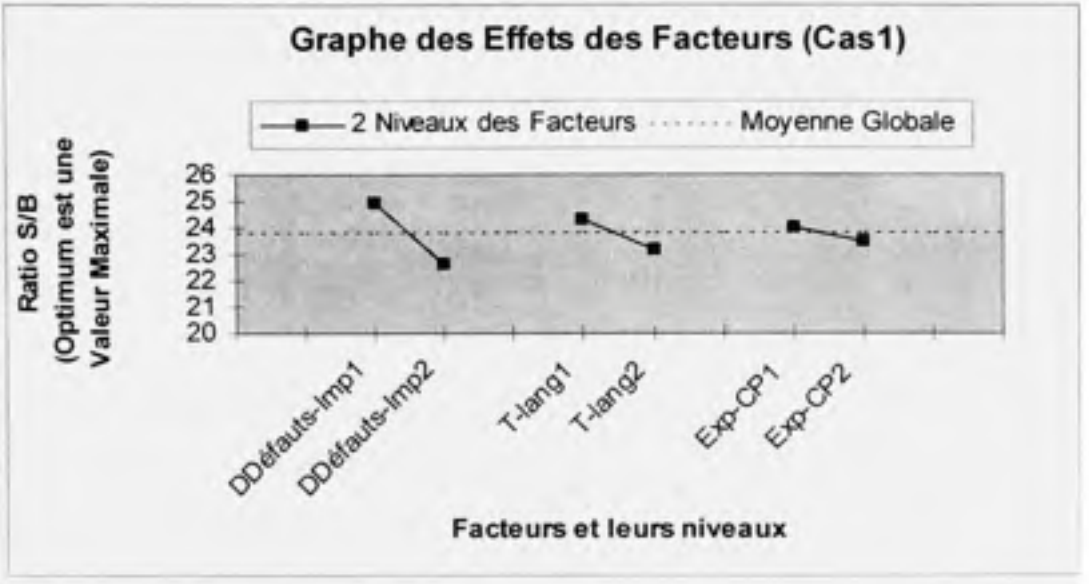

Figure 9.2 ***Graphe des effets des facteurs (Cas 1).***

Du Tableau 9.7 et de la Figure 9.2, il apparait que les facteurs étudiés dans ce cas de plan présentent des effets différents sur le degré de satisfaction des utilisateurs, à savoir :

- le facteur relatif à la densité du nombre de défauts détectés lors de la phase d'implémentation du logiciel (DDéfauts-Imp) présente le plus grand effet, soit 2.30;
- le facteur relatif au type de langage de programmation (T-Lang) vient en deuxième position avec un effet moins important, soit 1.14;
- le facteur relatif à l'expérience du chef du projet (Exp-CP) vient en dernier lieu avec un faible effet, soit 0.50.

3) Détermination de la condition optimale

La condition optimale représente les niveaux optimums de tous les facteurs étudiés dans ce cas de plan. Le niveau du facteur ayant la moyenne des ratios S/B la plus grande est choisi comme niveau optimum du facteur. Du graphe des effets des facteurs (Figure 9.2) et du Tableau 9.7, les niveaux optimums pour l'ensemble des facteurs de ce plan sont les suivants :

- DDéfauts-Imp au niveau 1 : DDéfauts-Imp ≤ 0.006;
- T-Lang au niveau 1 : T-Lang = 3GL;
- Exp-CP au niveau 1 : Exp-CP = 1 à 9.

La condition optimale est : la densité du nombre de défauts collectés lors de phase d'implémentation du logiciel est inférieure ou égale à 0.006, le langage de programmation est du type troisième génération (3GL) et l'expérience du chef du projet (exprimée en nombre de projets antérieurs gérés) varie entre 1 et 9.

4) Analyse de la variance

L'analyse de la variance pour ce cas de plan est présentée dans le Tableau 9.8.

Tableau 9.8

Analyse de la variance (Cas 1)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DDéfauts-Imp	1	5.30	5.30	77	21
T-Lang	1	1.31	1.31	19	5
Exp-CP	1	0.25	0.25	4	
Erreur	0	0			
Total	3	6.85		100	
Erreur estimée	1	0.25	0.25		

Du Tableau 9.8, de la colonne relative à la contribution, il est à noter que (Figure 9.3) :

- le facteur relatif à la densité du nombre de défauts détectés durant la phase d'implémentation (DDéfauts-Imp) présente la plus grande contribution avec 77% à la variabilité du degré de satisfaction des utilisateurs;
- le facteur relatif au type de langage de programmation (T-Lang) présente une contribution moins importante avec 19%;
- le facteur relatif à l'expérience du chef du projet (Exp-CP) ne contribue que de 4% : faible contribution.

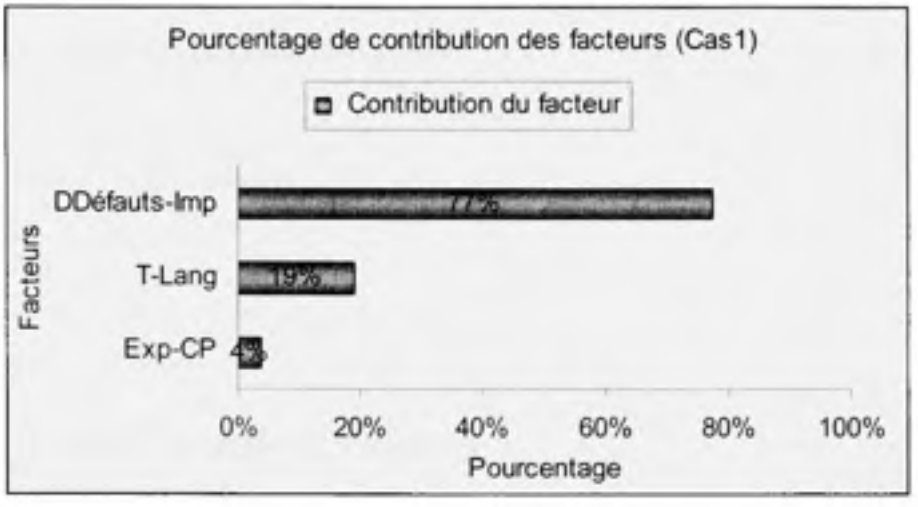

Figure 9.3 ***Pourcentage de contribution des facteurs (Cas 1).***

D'un autre côté, de l'analyse de la colonne du ratio de variance F, il est à constater que :

- le facteur relatif à la densité du nombre de défauts détectés durant la phase d'implémentation (DDéfauts-Imp) a un effet dominant avec F = 21;
- le facteur relatif au type de langage de programmation (T-Lang) a aussi un effet dominant avec F = 5, mais inférieur à celui du facteur (DDéfauts-Imp).

Le facteur relatif à l'expérience du chef du projet (Exp-CP) est utilisé (pooling) pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 9.8), seuls les facteurs (DDéfauts-Imp) et (T-Lang) sont retenus comme facteurs les plus influents. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 1. Ainsi, l'équation de prédiction à la condition optimale de ce plan est calculée en fonction de ces deux facteurs.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{DDéfauts \ - \text{Im}\ p_1} - \overline{T}\right) + \left(\overline{T - Lang_1} - \overline{T}\right) \tag{9.4}$$

$$\hat{\eta} = 25.47 \text{ avec } \overline{T} = 23.75 \tag{9.5}$$

Le facteur (Exp-CP) est exclu de l'équation de prédiction puisqu'il a été utilisé pour estimer la variance de l'erreur.

6) Test de confirmation

Le test de confirmation consiste à effectuer un ensemble d'expériences selon la condition optimale déterminée (dans l'étape 3 de la section 9.6.4.2) pour ce plan et calculer le ratio signal-bruit du test de confirmation. Pour ce cas de plan, vu la non disponibilité de projets dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale, le test de confirmation ne peut être effectué.

9.6.5 Synthèse

Dans ce cas de plan, de l'analyse du ratio signal-bruit, il en résulte que la densité du nombre de défauts collectés lors de la phase d'implémentation du logiciel, comme facteur de qualité externe du produit logiciel, est le plus influent des trois facteurs suivi du type de langage de programmation, puis de l'expérience du chef du projet sur la caractéristique de qualité de ce plan. Cette caractéristique de qualité correspond au degré de satisfaction des utilisateurs, considéré comme attribut de qualité en utilisation du produit logiciel.

Le degré de satisfaction des utilisateurs est exprimée dans le cas de ce plan en fonction de :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;
- la facilité d'utilisation;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

Ainsi, l'hypothèse est confirmée : la qualité externe du produit logiciel (mesurée dans ce plan par la variable « Densité du nombre de défauts détectés lors de la phase d'implémentation du logiciel ») influence sa qualité en utilisation.

9.7 Plan d'analyse – Cas 2

Dans ce plan, nous traitons de l'influence de la densité du nombre de défauts détectés après la fin du projet (essentiellement durant le premier mois d'opération du logiciel), du type de langage de programmation et de la taille du projet sur le degré de satisfaction des utilisateurs.

Pour ce cas de plan, le degré de satisfaction des utilisateurs prend en considération tous les items du sondage d'ISBSG (voir Tableau 9.1), à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;
- la qualité de la documentation;
- la facilité d'utilisation;
- la formation fournie;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

9.7.1 Échantillon de projets d'ISBSG du plan

À partir de l'échantillon « BD-ISBSG-AB-Février-2006 » (section 7.3), les projets à sélectionner doivent tous avoir des informations disponibles (pas de champs vides ni d'expressions comme : « Not applicable », etc.) aussi bien pour les facteurs de ce plan que pour sa caractéristique de qualité, à savoir :

- la densité du nombre de défauts détectés après la fin du projet (durant le premier mois d'opération du logiciel);
- le type de langage de programmation;
- la taille du projet;
- le degré de satisfaction des utilisateurs.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG pour ce plan ne contient que 17 projets.

9.7.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 9.9.

Tableau 9.9

Liste des facteurs est leurs niveaux (Cas 2)

Facteurs		Niveaux	Description
Densité du nombre de défauts détectés durant le premier mois d'opération du logiciel - fin du projet (DDéfauts-FP)	A	1	DDéfauts-FP ≤ 0.009
		2	DDéfauts-FP > 0.009
Type de langage de programmation (T-Lang)	B	1	T-Lang = 3GL
		2	T-Lang = 4GL
Taille du projet (Taille)	C	1	Taille < 186
		2	Taille ≥ 186

Deux niveaux sont choisis pour chaque facteur de ce plan :

- pour les facteurs relatifs à la densité du nombre de défauts détectés de la phase de fin du projet (DDéfauts-FP) et à la taille du projet (Taille), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur du facteur inférieure et supérieure à la médiane du facteur dans l'échantillon de projets d'ISBSG de ce plan;
- pour le facteur relatif au type de langage de programmation (T-Lang), les niveaux 1 et 2 représentent les catégories de ce facteur.

9.7.3 Choix de la table orthogonale du plan

Ce plan inclut trois facteurs avec deux niveaux chacun et sans aucune interaction. Ainsi, le degré de liberté du plan étudié est égal à trois degrés de liberté (voir section 7.4.1). La table orthogonale à choisir pour constituer la matrice d'expériences de ce cas de plan est : L4, c'est-à-dire quatre essais (Tableau 9.10).

Tableau 9.10

Matrice d'expériences (Cas 2)

Liste des Essais	Liste des Facteurs du Plan		
	Densité du nombre de défauts détectés : phase fin du projet (DDéfauts-FP)	**Type de langage (T-Lang)**	**Taille du projet (Taille)**
1	≤ 0.009	3GL	< 186
2	≤ 0.009	4GL	≥186
3	> 0.009	3GL	≥186
4	> 0.009	4GL	<186

9.7.4 Réalisation

9.7.4.1 Exécution

Le Tableau 9.11 montre le détail des huit projets répondant aux conditions des quatre essais de la matrice d'expériences, lesquels projets sont sélectionnés parmi les 17 projets de l'échantillon de projets d'ISBSG de ce plan. Les autres projets ne sont pas sélectionnés à cause des contraintes précitées dans la section 9.6.4.1.

Tableau 9.11

Caractéristiques des projets d'ISBSG (Cas 2)

ID projet	Taille du projet	Nombre de défauts détectés - phase fin de projet	Densité du nombre de défauts - phase fin du projet	Type de langage	Degré de satisfaction des utilisateurs
23826	118	0	0.000	3GL	23
31237	146	1	0.007	3GL	27
16873	273	0	0.000	4GL	25
31360	244	0	0.000	4GL	19
25620	1127	24	0.021	3GL	19
27560	199	5	0.025	3GL	19
19166	35	1	0.029	4GL	11
10687	55	4	0.073	4GL	16

Dans ce cas de plan, il n'y a que deux projets différents satisfaisant les conditions de chaque essai de la matrice d'expériences. Ainsi, l'affectation est faite de la même façon que pour le premier plan (section 9.6.4.1). Le résultat de l'affectation des projets du Tableau 9.11 aux essais du Tableau 9.10 est présenté dans le Tableau 9.12.

Tableau 9.12

Affectation des projets aux essais (Cas 2)

Liste des Essais	**Liste des Facteurs du Plan**			**Degré satisfaction des utilisateurs**	
	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	Type de langage (T-Lang)	Taille du projet (Taille)	Y1	Y2
1	≤ 0.009	3GL	<186	23	27
2	≤ 0.009	4GL	≥186	25	19
3	> 0.009	3GL	≥186	19	19
4	> 0.009	4GL	<186	11	16

9.7.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Le calcul du ratio signal-bruit (en utilisant la formule (7.5)) pour chaque essai du Tableau 9.12 est présenté dans le Tableau 9.13.

Tableau 9.13

Calcul des ratios S/B (Cas 2)

Liste des Essais	Liste des Facteurs du Plan			Degré de satisfaction des utilisateurs		Ratio signal-bruit
	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	Type de langage (T-Lang)	Taille du projet (Taille)	Y1	Y2	S/B
1	≤ 0.009	3GL	<186	23	27	27.88
2	≤ 0.009	4GL	≥186	25	19	26.61
3	> 0.009	3GL	≥186	19	19	25.58
4	> 0.009	4GL	<186	11	16	22.16

2) Calcul des effets des facteurs

Le résultat du calcul des moyennes des ratios S/B pour les deux niveaux de chaque facteur de ce plan et du calcul des effets des facteurs est présenté dans le Tableau 9.14.

Tableau 9.14

Calcul des effets des facteurs (Cas 2)

	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	Type de langage (T-Lang)	Taille du projet (Taille)
Moyenne des ratios S/B au niveau 1	27.24	26.73	25.02
Moyenne des ratios S/B au niveau 2	23.87	24.38	26.09
Effet du facteur (Delta)	3.37	2.35	1.07

La Figure 9.4 présente graphiquement les moyennes des ratios S/B pour les deux niveaux des facteurs de ce plan. La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 9.13 : (25.55).

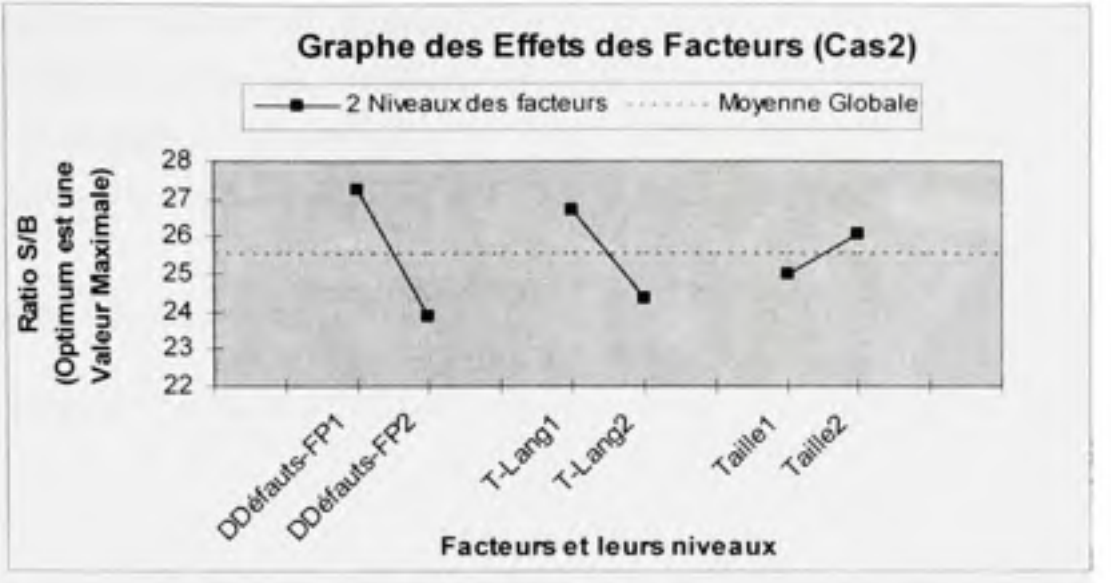

Figure 9.4 ***Graphe des effets des facteurs (Cas 2).***

Du Tableau 9.14 et de la Figure 9.4, il est à constater que les facteurs de ce plan présentent des effets différents sur le degré de satisfaction des utilisateurs, à savoir :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP) est le plus influent avec un effet de 3.37;
- le facteur relatif au type de langage de programmation (T-Lang) présente une influence moins importante avec un effet de 2.35;
- le facteur relatif à la taille du projet (Taille) a une faible influence avec un effet de 1.07.

3) Détermination de la condition optimale

Du graphe des effets des facteurs (Figure 9.4) et du Tableau 9.14, les niveaux optimums pour les trois facteurs de ce plan sont les suivants :

- DDéfauts-FP au niveau 1 : DDéfauts-FP ≤ 0.009;
- T-Lang au niveau 1 : T-Lang = 3GL;
- Taille au niveau 2 : Taille ≥ 186.

La condition optimale est : la densité du nombre de défauts collectés après la fin du projet (lors du premier mois d'opération du logiciel) est inférieure ou égale à 0.009, le type de langage de programmation est dans la catégorie des langages de troisième génération (3GL) et la taille du projet est supérieure ou égale à 186 points de fonction.

4) Analyse de la variance

Le Tableau 9.15 présente le calcul de l'analyse de la variance pour ce cas de plan.

Tableau 9.15

Analyse de la variance (Cas 2)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DDéfauts-FP	1	11.39	11.39	63	10
T-Lang	1	5.49	5.49	31	5
Taille	1	1.15	1.15	6	
Erreur	0	0			
Total	3	18.03		100	
Erreur estimée	1	1.15	1.15		

De l'analyse de la variance (Tableau 9.15), de l'analyse de la colonne contribution, il est à noter que (Figure 9.5) :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP) présente la plus grande contribution, soit 63% à la variabilité du degré de satisfaction des utilisateurs;
- le facteur relatif au type de langage de programmation (T-Lang) vient en deuxième position de contribution avec 31%;
- le facteur relatif à la taille du projet (Taille) contribue seulement de 6 % : faible contribution.

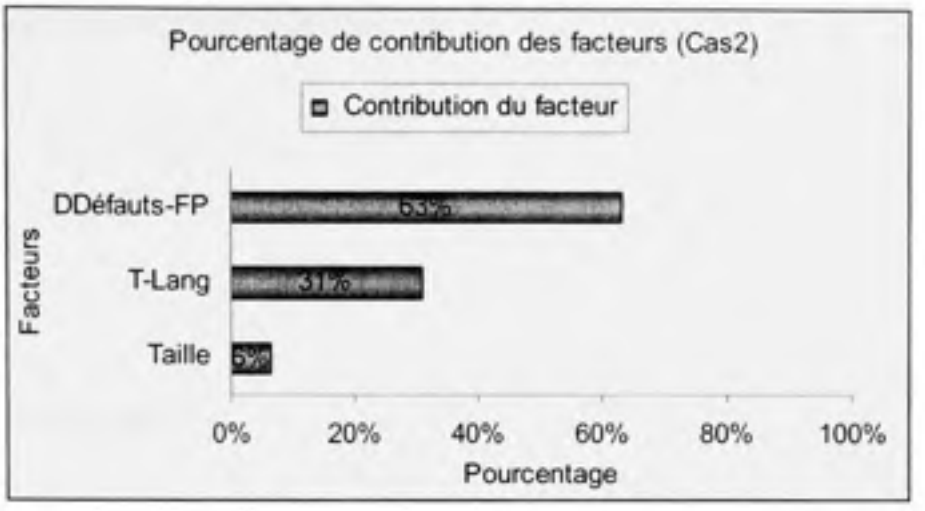

Figure 9.5 ***Pourcentage de contribution des facteurs (Cas 2).***

En outre, en analysant la colonne relative au ratio de variance F, il est à constater que :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP), avec le ratio F = 10, a un effet dominant;
- le facteur relatif au type de langage de programmation (T-Lang), avec F = 5, possède aussi un effet dominant mais inférieur à celui du facteur (DDéfauts-FP).

Le facteur relatif à la taille du projet (Taille) est utilisé (pooling) pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 9.15), les facteurs retenus comme facteurs les plus influents sont (DDéfauts-FP) et (T-Lang). Les niveaux optimums de ces deux facteurs sont respectivement le niveau 1 et le niveau 1. Ainsi, l'équation de prédiction à la condition optimale de ce plan est calculée seulement en fonction de ces deux facteurs. Le facteur (Taille) a été utilisé pour estimer la variance de l'erreur.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{DDéfauts\ -\ FP_1} - \overline{T}\right) + \left(\overline{T - Lang_1} - \overline{T}\right) \tag{9.6}$$

$$\hat{\eta} = 28.41 \text{ avec } \overline{T} = 25.55 \tag{9.7}$$

6) Test de confirmation

Le test de confirmation consiste à effectuer un ensemble d'expériences selon la condition optimale déterminée (dans l'étape 3 de la section 9.7.4.2) pour ce plan et calculer le ratio signal-bruit du test de confirmation. Pour ce cas de plan, vu l'insuffisance de projets disponibles dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale (un seul projet), le test de confirmation ne peut être effectué.

9.7.5 Synthèse

Dans ce plan, de l'analyse du ratio signal-bruit, il en résulte que la densité du nombre de défauts collectés après la fin du projet (lors du premier mois d'opération du logiciel) comme facteur de qualité externe du produit logiciel, est le plus influent des trois facteurs suivi du type de langage de programmation, puis de la taille du projet sur la caractéristique de qualité choisie. Cette caractéristique de qualité correspond au degré de satisfaction des utilisateurs, considéré comme attribut de qualité en utilisation du produit logiciel.

Le degré de satisfaction des utilisateurs dans ce plan prend en considération tous les items du sondage d'ISBSG, à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;

- la qualité de la documentation;
- la facilité d'utilisation;
- la formation fournie;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

Ainsi, l'hypothèse est confirmée : la qualité externe du produit logiciel (mesurée dans ce plan par la variable « Densité du nombre de défauts détectés lors du premier mois d'opération du logiciel après la fin du projet ») influence la qualité en utilisation du produit logiciel.

9.8 Plan d'analyse – Cas 3

Ce plan d'analyse permet d'étudier l'influence de la densité du nombre de défauts détectés après la fin du projet (durant le premier mois d'opération du logiciel), du type de développement et du type de langage de programmation sur le degré de satisfaction des utilisateurs.

Pour ce cas de plan, le degré de satisfaction des utilisateurs est mesuré en fonction des cinq items du sondage d'ISBSG (voir Tableau 9.1), à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

9.8.1 Échantillon de projets d'ISBSG du plan

L'échantillon de projets d'ISBSG de ce plan comprend 23 projets (obtenu par raffinement de « BD-ISBSG-AB-Février-2006» de la section 7.3). Cet échantillon ne contient que les projets ayant tous des informations disponibles (pas de champs vides ou contenant des expressions comme « Some », « Not applicable », etc.) pour tous les facteurs de ce plan et pour sa caractéristique de qualité :

- la densité du nombre de défauts détectés après la fin du projet (durant le premier mois d'opération du logiciel);
- le type de développement;
- le type de langage de programmation;
- le degré de satisfaction des utilisateurs.

9.8.2 Liste des facteurs et leurs niveaux

La liste des facteurs étudiés dans ce cas de plan ainsi que leurs niveaux correspondants sont présentés dans le Tableau 9.16.

Tableau 9.16

Liste des facteurs et leurs niveaux (Cas 3)

Facteurs		Niveaux	Description
Densité du nombre de défauts détectés durant le premier mois d'opération du logiciel - fin du projet (DDéfauts-FP)	A	1	DDéfauts-FP $\leq$ 0.014
		2	DDéfauts-FP > 0.014
Type de développement du logiciel (T-Dev)	B	1	T-Dev = R/A (Redéveloppement /Amélioration)
		2	T-Dev = N (Nouveau développement)
Type de langage de programmation (T-Lang)	C	1	T-Lang = 4GL
		2	T-Lang = 3GL

Chaque facteur possède deux niveaux choisis de la manière suivante :

- pour le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP), les niveaux 1 et 2 représentent respectivement les projets ayant la valeur du facteur inférieure et supérieure à la médiane du facteur dans l'échantillon de projets d'ISBSG de ce plan;
- pour les facteurs relatifs au type développement (T-Dev) et au type de langage de programmation (T-Lang), les niveaux 1 et 2 représentent les catégories de ces deux facteurs.

9.8.3 Choix de la table orthogonale du plan

Ce cas de plan comprend trois facteurs sans aucune interaction et chaque facteur possède deux niveaux. Ainsi, le degré de liberté du plan étudié est égal à trois degrés de liberté (section 7.4.1) et la table orthogonale à choisir est donc L4. Le Tableau 9.17 présente la matrice d'expériences de ce plan.

Tableau 9.17

Matrice d'expériences (Cas 3)

Liste des Essais	**Liste des Facteurs du Plan**		
	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	**Type de développement (T-Dev)**	**Type de langage (T-Lang)**
1	≤ 0.014	R/A	4GL
2	≤ 0.014	N	3GL
3	> 0.014	R/A	3GL
4	> 0.014	N	4GL

9.8.4 Réalisation

9.8.4.1 Exécution

Le Tableau 9.18 montre le détail des projets utilisés dans ce plan d'analyse, lesquels projets au nombre de huit répondent aux conditions des essais du Tableau 9.17. Les autres projets de l'échantillon de projets d'ISBSG de ce plan ne sont pas sélectionnés à cause des contraintes précitées (voir section 9.6.4.1).

Tableau 9.18

Caractéristiques des projets d'ISBSG (Cas 3)

ID projet	Taille du projet	Type de développement	Nombre de défauts - phase fin du projet	Densité du nombre de défauts - phase fin du projet	Type de langage	Degré de satisfaction des utilisateurs
10557	62	Amélioration	0	0.000	4GL	15
31360	244	Amélioration	0	0.000	4GL	11
19561	83	Nouveau développement	0	0.000	3GL	14
23009	182	Nouveau développement	0	0.000	3GL	15
25620	1127	Re-développement	24	0.021	3GL	12
27560	199	Re-développement	5	0.025	3GL	12
23142	44	Nouveau développement	1	0.023	4GL	7
28020	186	Nouveau développement	5	0.027	4GL	13

À chaque essai de la matrice d'expériences de ce plan ne correspond que deux projets d'ISBSG. Ainsi, le même scénario (voir section 9.6.4.1) que les plans précédents est à suivre dans l'affectation des projets du Tableau 9.18 aux essais du Tableau 9.17. L'affectation des projets aux essais est présentée dans le Tableau 9.19.

Tableau 9.19

Affectation des projets aux essais (Cas 3)

Liste des Essais	Liste des Facteurs du Plan			Degré satisfaction des utilisateurs	
	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	Type de développement (T-Dev)	Type de langage (T-Lang)	Y1	Y2
1	≤ 0.014	R/A	4GL	15	11
2	≤ 0.014	N	3GL	14	15
3	> 0.014	R/A	3GL	12	12
4	> 0.014	N	4GL	7	13

9.8.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Le calcul du ratio signal-bruit (en utilisant la formule (7.5)) pour chaque essai du Tableau 9.19 est présenté dans le Tableau 9.20.

Tableau 9.20

Calcul des ratios S/B (Cas 3)

Liste des Essais	Liste des Facteurs du Plan			Degré de satisfaction des utilisateurs		Ratio signal-bruit
	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	Type de développement (T-Dev)	Type de langage (T-Lang)	Y1	Y2	S/B
1	≤ 0.014	R/A	4GL	15	11	21.97
2	≤ 0.014	N	3GL	14	15	23.21
3	> 0.014	R/A	3GL	12	12	21.58
4	> 0.014	N	4GL	7	13	18.81

2) Calcul des effets des facteurs

Le calcul des effets des facteurs de ce plan est présenté dans le Tableau 9.21.

Tableau 9.21

Calcul des effets des facteurs (Cas 3)

	Densité du nombre de défauts détectés - phase fin du projet (DDéfauts-FP)	**Type de développement (T-Dev)**	**Type de langage (T-Lang)**
Moyenne des ratios S/B au niveau 1	22.59	21.78	20.39
Moyenne des ratios S/B au niveau 2	20.20	21.01	22.40
Effet du facteur (Delta)	2.40	0.77	2.01

La Figure 9.6 représente les deux niveaux de chaque facteur du plan d'analyse et leurs valeurs correspondantes de la moyenne des ratios S/B.

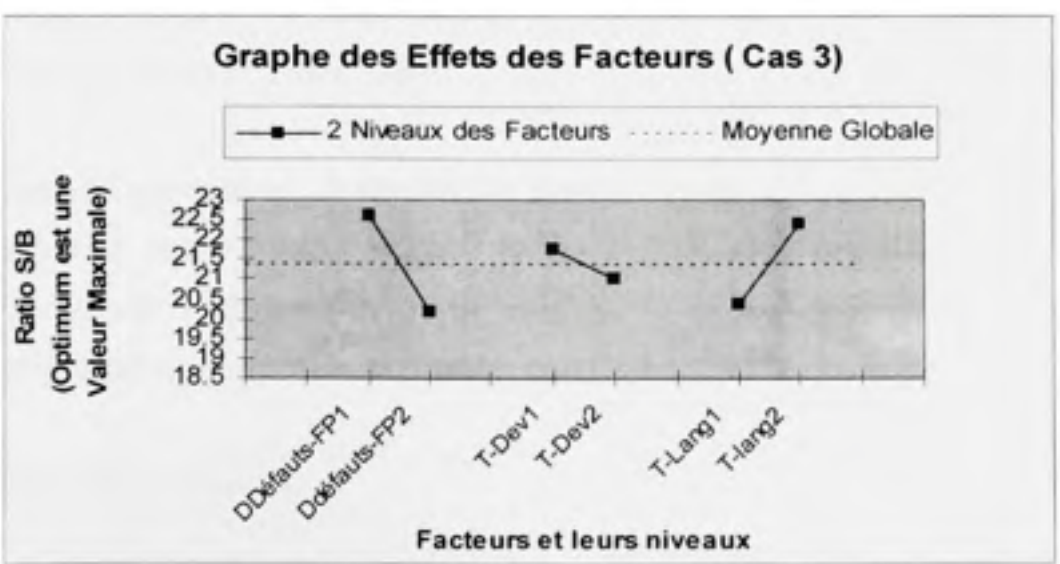

Figure 9.6 ***Graphe des effets des facteurs (Cas 3).***

La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 9.20 : (21.39).

Du Tableau 9.21 et de la Figure 9.6, il est à noter que les facteurs de ce cas de plan possèdent des effets différents sur le degré de satisfaction des utilisateurs :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP) est le plus influent avec un effet de 2.40;
- le facteur relatif au type de langage de programmation (T-Lang) présente une influence moins importante avec un effet de 2.01;
- le facteur relatif au type de développement (T-Dev) possède une faible influence par rapport aux autres facteurs avec un effet de 0.77.

3) Détermination de la condition optimale

Du graphe des effets des facteurs (Figure 9.6) et du Tableau 9.21, les niveaux optimums pour les facteurs de ce plan sont les suivants :

- DDéfauts-FP au niveau 1 : DDéfauts-FP $\leq$ 0.014;
- T-Lang au niveau 2 : T-Lang = 3GL;
- T-Dev au niveau 1 : T-Dev = R/A.

La condition optimale est : la densité du nombre de défauts collectés après la fin du projet (lors du premier mois d'opération du logiciel) est inferieure ou égale à 0.014, le type de langage de programmation est de troisième génération (3GL) et le type de développement est un redéveloppement ou une amélioration.

4) Analyse de la variance

L'analyse de la variance pour ce cas de plan est présentée dans le Tableau 9.22.

Tableau 9.22

Analyse de la variance (Cas 3)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DDéfauts-FP	1	5.74	5.74	55	10
T-Dev	1	0.59	0.59	6	
T-Lang	1	4.04	4.04	39	7
Erreur	0	0			
Total	3	10.37		100	
Erreur estimée	1	0.59	0.59		

De l'analyse de la variance, colonne contribution, il est à observer que (voir Figure 9.7) :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP) présente la plus grande contribution, soit 55% à la variabilité du degré de satisfaction des utilisateurs;
- le facteur relatif au type de langage de programmation (T-Lang) contribue avec 39%, laquelle contribution est aussi importante;
- le facteur relatif au type de développement (T-Dev) ne contribue que de 6 % : faible contribution par rapport aux autres facteurs.

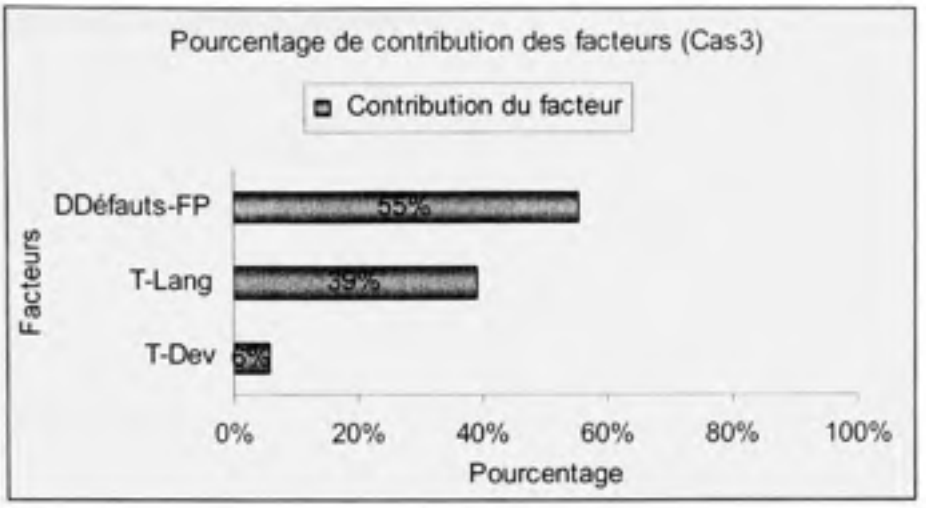

Figure 9.7 ***Pourcentage de contribution des facteurs (Cas 3).***

En outre, de l'analyse de la colonne du ratio de variance F, il est à noter que :

- le facteur relatif à la densité du nombre de défauts détectés après la fin du projet (DDéfauts-FP) a un effet dominant : le ratio F = 10;
- le facteur relatif au type de langage de programmation (T-Lang) a aussi un effet dominant, F = 7, mais moins important que le facteur (DDéfauts-FP).

Le facteur relatif au type de développement (T-Dev) est utilisé (pooling) pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 9.22), les facteurs (DDéfauts-FP) et (T-Lang) sont retenus comme facteurs les plus influents. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 2. Ainsi, l'équation de prédiction à la condition optimale est calculée seulement en fonction de ces facteurs. Le facteur (T-Dev) a été utilisé pour estimer la variance de l'erreur.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{DDéfauts - FP_1} - \overline{T}\right) + \left(\overline{T - Lang_2} - \overline{T}\right) \tag{9.8}$$

$$\hat{\eta} = 23.60 \text{ avec } \overline{T} = 21.39 \tag{9.9}$$

6) Test de confirmation

Cinq projets sont disponibles avec la condition optimale (déterminée dans l'étape 3 de la section 9.8.4.2) dans l'échantillon de projets d'ISBSG de ce plan, regroupés dans le Tableau 9.23.

Tableau 9.23

Projets d'ISBSG - Test de confirmation (Cas 3)

ID projet	Taille du projet	Nombre de défauts détectés - phase fin de projet	Densité du nombre de défauts - phase fin du projet	Type de développement	Type de langage	Degré de satisfaction des utilisateurs
10323	647	6	0.009	Amélioration	3GL	15
28553	788	11	0.014	Re-développement	3GL	12
12922	71	0	0.000	Amélioration	3GL	15
23826	118	0	0.000	Amélioration	3GL	15
31237	146	1	0.007	Amélioration	3GL	18

Le ratio signal-bruit pour le test de confirmation calculé avec la formule (7.5) est : S/B = (23.30).

Par ailleurs, le nombre d'essais de la matrice d'expériences est quatre, le nombre de fois que le niveau 1 du facteur (DDéfauts-FP) est répété dans la matrice d'expériences est deux, le nombre de fois que le niveau 2 du facteur (T-Lang) est répété dans la matrice

d'expériences est deux, le nombre des expériences de vérification est cinq et la variance de l'erreur est égale à 0.59. Ainsi, en utilisant les formules (7.11) et (7.13) :

- l'intervalle de confiance à 95 % de l'erreur de prédiction est : (±1.50);
- l'intervalle de confiance du résultat du ratio S/B est : [22.10, 25.09].

Le ratio S/B du test de confirmation existe dans l'intervalle de confiance du résultat du ratio S/B de ce plan. Ainsi, il est important de prendre en considération les paramètres optimums de ce cas de plan lors de développement du produit logiciel.

9.8.5 Synthèse

Dans ce cas de plan, de l'analyse du ratio signal-bruit, il en résulte que la densité du nombre de défauts collectés lors de la phase de fin du projet (durant le premier mois d'opération du logiciel), comme facteur de qualité externe du produit logiciel, est le plus influent des trois facteurs suivi du type de langage de programmation, puis du type de développement sur la caractéristique de qualité choisie. Cette caractéristique de qualité correspond au degré de satisfaction des utilisateurs, considéré comme attribut de qualité en utilisation du produit logiciel.

Le degré de satisfaction des utilisateurs dans ce plan prend en considération un sous-ensemble d'items du sondage d'ISBSG, à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités;
- la rapidité dans la définition de la solution;
- la rapidité dans la fourniture de la solution.

Ainsi, l'hypothèse est confirmée : la qualité externe du produit logiciel (mesurée dans ce plan par la variable « Densité du nombre de défauts détectés lors du premier mois

d'opération du logiciel après la fin du projet ») influence la qualité en utilisation du produit logiciel.

9.9 Remarques

Comme nous avons indiqué dans la section 8.9 du chapitre 8, la condition optimale déterminée à partir des plans d'analyses est une étape importante de tout processus de production dans le secteur industriel. Toutefois, en génie logiciel, cette condition optimale peut être aussi applicable, même partiellement dans la conception de produits logiciels en prenant en considération les niveaux optimums pour certains facteurs, comme le cas des attributs de qualité étudiés dans ce chapitre. Pour les autres facteurs (types de technologie, types de langage, etc.), le choix d'implémentation des niveaux optimums dépendra des caractéristiques de chaque projet, des contraintes de gestion, etc. Par exemple, en ce qui concerne le choix du type de langage de programmation, Jones (1996, p. 286) souligne que :

> « With more than 500 languages available, most of enterprises find it difficult to select a single language or a set of languages that is optimized for its needs [...] the most appropriate languages can range across all generations. It is not accurate to prescribe fourth generation languages exclusively, since they are not appropriate for many program and system types. »

9.10 Conclusion

Dans ce chapitre, nous avons traité le deuxième volet de la phase 3 de la méthodologie de recherche établie dans le chapitre 5; il s'agit de vérifier l'hypothèse du lien entre la qualité externe et la qualité en utilisation du produit logiciel de la Figure 2.1 de la norme ISO 9126-1. Pour ce faire, nous avons présenté trois plans d'analyses empiriques conçus avec la méthode Taguchi tout en exploitant les données disponibles dans l'extrait de données d'ISBSG de février 2006.

Puisque l'objectif des plans conçus dans ce chapitre est de maximiser la caractéristique de qualité : le degré de satisfaction des utilisateurs, l'analyse des résultats de ces plans a été effectuée avec la méthode d'analyse du ratio signal-bruit de Taguchi dans le cas où *l'optimum est une valeur maximale*.

Le résultat du premier plan montre que le facteur relatif à la densité du nombre de défauts collectés lors de la phase d'implémentation du logiciel est le plus influent des facteurs étudiés dans ce plan sur le degré de satisfaction des utilisateurs. Les résultats du deuxième plan et du troisième plan montrent que le facteur relatif à la densité du nombre de défauts collectés lors de la fin du projet (durant le premier mois d'utilisation du logiciel) est aussi le plus influent sur le degré de satisfaction des utilisateurs. En outre, il en résulte dans ce chapitre que le niveau le plus préférable est d'avoir une faible densité du nombre de défauts détectés lors de la phase d'implémentation/fin du projet afin de maximiser le degré de satisfaction des utilisateurs.

Ainsi, l'hypothèse principale établie dans ce chapitre est confirmée puisque :

- la densité du nombre de défauts détectés de la phase implémentation (fin du projet) représente une mesure de qualité externe du produit logiciel, laquelle mesure permet d'évaluer la sous-caractéristique de qualité « Maturité » de la caractéristique de qualité « Fiabilité » du modèle de qualité externe d'ISO 9126-1;
- le degré de satisfaction des utilisateurs représente une mesure de qualité en utilisation du produit logiciel, laquelle mesure permet d'évaluer la caractéristique de qualité « Satisfaction » du modèle de qualité en utilisation d'ISO 9126-1.

CHAPITRE 10

VÉRIFICATION DU LIEN ENTRE LA QUALITÉ INTERNE ET LA QUALITÉ EN UTILISATION

10.1 Introduction

Ce chapitre traite le troisième volet de la phase 3 de la méthodologie de recherche (chapitre 5); il s'agit de l'objectif 3 de cette recherche, soit le lien entre la qualité interne et la qualité en utilisation du produit logiciel.

Dans un premier temps, nous présentons l'objectif de ce chapitre, l'hypothèse principale à vérifier et la caractéristique de qualité étudiée. Par la suite, nous présentons la liste des facteurs utilisés dans les plans d'analyses conçus avec Taguchi.

L'analyse des résultats de ces plans d'analyses est effectuée en utilisant la méthode d'analyse du ratio signal-bruit (S/B) de Taguchi dans le cas où la caractéristique de qualité correspond à *l'optimum est une valeur maximale*.

10.2 Objectif

L'objectif dans ce chapitre consiste à explorer la relation entre la qualité interne et la qualité en utilisation du produit logiciel. L'intérêt est de déterminer les facteurs de qualité interne du produit logiciel pouvant avoir une influence sur sa qualité en utilisation. Pour ce faire, nous procédons à la conception de plans d'analyses empiriques dont les données exploitées sont disponibles dans l'extrait de données d'ISBSG de février 2006.

Dans la norme ISO 9126-1 (2001a), la différence entre la qualité interne du produit logiciel de sa qualité en utilisation vient du fait que la première (i.e. la qualité interne)

représente l'ensemble des caractéristiques du produit logiciel du point de vue interne et porte sur ses propriétés statiques. La deuxième, qualité en utilisation, représente le point de vue de l'utilisateur final du produit logiciel, laquelle qualité porte sur le résultat de l'utilisation du logiciel et non pas sur ses propres propriétés.

10.3 Hypothèse

L'hypothèse principale à vérifier dans ce chapitre, soit l'objectif 3 de cette recherche, est énoncée comme suit : la qualité interne influence la qualité en utilisation du produit logiciel.

Comme nous avons déjà souligné dans les chapitres 8 et 9, il n'est pas possible d'explorer toutes les relations qui peuvent exister entre les différentes caractéristiques (sous-caractéristiques) du modèle de qualité interne (Figure 2.2) et celles du modèle de qualité en utilisation (Figure 2.3). Toutefois, afin de vérifier la pertinence ou non de cette hypothèse, il est possible dans cette recherche d'explorer les différents facteurs de qualité interne disponibles dans l'extrait de données d'ISBSG de février 2006 pour, par la suite, étudier leurs influences sur la caractéristique de qualité en utilisation considérée. Cette hypothèse sera vérifiée à travers les sous-hypothèses établies et vérifiées dans les sections 10.6 et 10.7 de ce chapitre.

10.4 Caractéristique de qualité

La satisfaction de l'utilisateur final du produit logiciel est devenue l'un des facteurs importants dans la réussite du projet. Non seulement l'intérêt est de développer un logiciel opérationnel, mais un logiciel qui satisfait en plus les exigences de l'utilisateur, voire même les attentes en matière de qualité. Selon Kan (2003, p. 5), « to increase overall customer satisfaction as well as satisfaction with various quality attributes, the quality attributes must be taken into account in the planning and design of the software ». Ainsi, pour satisfaire l'utilisateur final du produit logiciel une fois que le

logiciel est complété et livré, il faut tout d'abord prendre en considération les attributs de qualité à satisfaire lors des premières phases de son cycle de vie, c'est-à-dire les attributs de qualité interne.

Par ailleurs, la qualité interne et la qualité en utilisation peuvent êtres considérées respectivement dans le sens étroit « small q » (q et Q pour désigner qualité) et dans le sens large « big Q » de la définition de qualité proposée par (Kan, 2003). En effet, la qualité du produit logiciel obtenue en mesurant ses propriétés intrinsèques (statiques et dynamiques) constitue le sens étroit de la qualité, alors que la qualité en utilisation représente le sens large de la qualité puisque « quality in use is the combined effect of internal and external quality for the user » (ISO, 2001a, p. 15).

La caractéristique de sortie considérée est le degré de satisfaction des utilisateurs. Le but est de maximiser la caractéristique de qualité, c'est-à-dire maximiser le degré de satisfaction des utilisateurs (voir section 9.4 du chapitre 9 pour la détermination du degré de satisfaction des utilisateurs).

10.5 Liste des facteurs

10.5.1 Diagramme de causes-effets

La Figure 10.1 regroupe un sous-ensemble de facteurs à étudier leurs influences sur la caractéristique de qualité choisie : le degré de satisfaction des utilisateurs. La description de ces facteurs est présentée dans la section suivante.

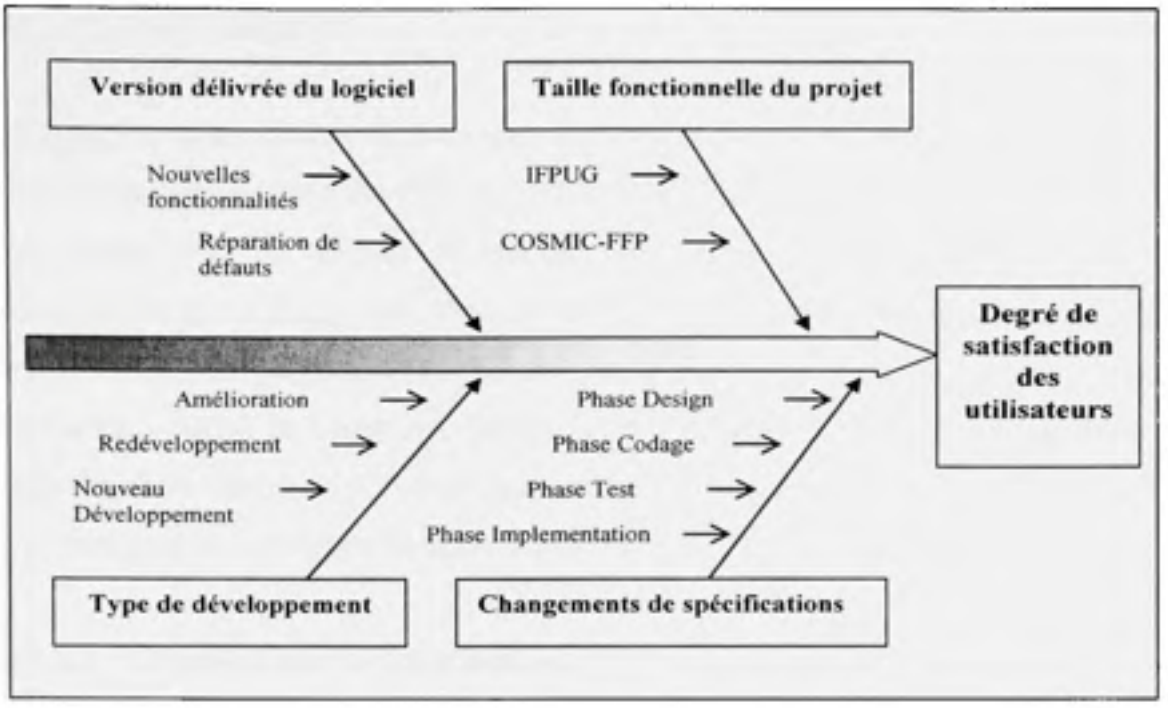

Figure 10.1 ***Diagramme de causes-effets.***

Le choix des facteurs représentés dans le diagramme de causes-effets de ce chapitre est aussi effectué en fonction de l'analyse des réponses du questionnaire sur la qualité du produit logiciel (voir Annexe VI) et dans la limite des données disponibles dans l'extrait d'ISBSG dans sa version de février 2006.

10.5.2 Description des facteurs

10.5.2.1 Type de développement

Trois types de développement sont disponibles dans l'extrait de données d'ISBSG, préalablement décrit dans la section 8.5.2.3, à savoir : nouveau développement, redéveloppement et amélioration.

10.5.2.2 Taille du projet

Les types de taille utilisés dans ce chapitre sont les types de méthodes de mesure de taille fonctionnelle COSMIC-FFP et IFPUG-UFP. L'échantillon de données d'ISBSG avec chaque type de méthode de mesure prise individuellement ne permet pas de concevoir des plans d'analyses. Nous avons donc opté pour l'utilisation des deux types de mesure de taille fonctionnelle ensemble en s'appuyant sur le travail de recherche de (Desharnais, Abran et Cuadrado, 2006) sur la convertibilité entre ces deux types de méthodes (voir section 8.5.2.1) et ainsi, disposer d'un échantillon de données assez important pour la conception de plans d'analyses avec Taguchi.

10.5.2.3 Changements de spécifications

DeMarco (1986, p. 294) souligne que : « the more successful you are in freezing the specification, the more likely your project is to deliver a system that does not fill the true needs of your user ». Ainsi, les changements de spécifications du logiciel peuvent avoir un effet sur la satisfaction des besoins des utilisateurs.

Le nombre de changements de spécifications est collecté tout au long du cycle de développement du logiciel (CDL) d'ISBSG, à savoir : les phases de design, de codage, de test et d'implémentation. Le nombre de changements de spécifications par CDL est calculé en fonction de ces différentes phases, essentiellement en fonction des différents champs de données relatifs au nombre de changements de spécifications disponibles dans l'extrait de données d'ISBSG de février 2006 pour ces différentes phases. Vu le problème de manque de données que présentent ces champs de données, le calcul de la valeur de ce facteur, pour chaque projet d'ISBSG, est effectué de la manière suivante :

- si tous les champs de données relatifs au nombre de changements de spécifications de ces phases sont vides, le projet est éliminé de l'échantillon de projets d'ISBSG du plan étudié;

- si au moins un de ces champs de données contient une donnée (i.e. non vide) et les autres ne contiennent pas de données (i.e. vide), alors ces derniers champs vides sont considérés comme contenant la valeur zéro et le projet est inclus dans l'échantillon de projets d'ISBSG du plan étudié.

Pour une comparaison appropriée des projets de différentes tailles, nous avons normalisé le nombre de changements de spécifications par CDL par la taille du projet (section 10.5.2.2) afin d'obtenir la densité; il s'agit de la densité du nombre de changements de spécifications par CDL.

10.5.2.4 Nombre de versions délivrées du logiciel

Cette information concerne les différents types de versions (release) du logiciel qui ont été délivrées au client ou à l'utilisateur final durant le projet. Deux types se présentent dans l'extrait de données d'ISBSG, à savoir :

- nombre de versions fournissant principalement de nouvelles fonctionnalités;
- nombre de versions fournissant principalement la réparation de défauts.

Là aussi à cause du problème de manque de données, ce facteur représente le nombre de versions tous types confondus pour chaque projet d'ISBSG. Ce facteur est calculé en faisant la somme des deux champs de données précédents. Lors du calcul de ce facteur pour chaque projet d'ISBSG le même principe que celui du facteur précédent (voir section 10.5.2.3) est à suivre, c'est-à-dire il faut que l'un des deux champs de données relatifs au nombre de versions soit non vide afin d'inclure le projet dans l'échantillon de projets d'ISBSG du plan étudié.

10.6 Plan d'analyse – Cas 1

Ce cas de plan permet d'analyser l'influence de la densité du nombre de changements de spécifications par CDL, du type de développement et du nombre de versions du logiciel sur la caractéristique de sortie de ce plan. Cette caractéristique de sortie représente le degré de satisfaction des utilisateurs par rapport à deux items parmi ceux du sondage d'ISBSG (voir Tableau 9.1), à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires.

10.6.1 Échantillon de projets d'ISBSG du plan

Dans ce cas de plan, les projets à sélectionner doivent tous disposer d'informations disponibles (pas de champs vides ou contenant des expressions comme « Not applicable », « Some », etc.) non seulement pour les facteurs de ce plan, mais aussi pour sa caractéristique de sortie, à savoir :

- la densité du nombre de changements de spécifications par CDL;
- le type de développement;
- le nombre de versions du logiciel;
- le degré de satisfaction des utilisateurs.

Avec ces conditions, le nouvel échantillon de projets d'ISBSG (obtenu à partir de l'échantillon « BD-ISBSG-AB-Février-2006» de la section 7.3) pour ce plan ne contient que 13 projets.

10.6.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce plan d'analyse et leurs niveaux correspondants sont présentés dans le Tableau 10.1.

Tableau 10.1

Liste des facteurs et leurs niveaux (Cas 1)

Facteurs		Niveaux	Description
Densité du nombre de changements de spécifications par CDL (DCS-CDL)	A	1	DCS-CDL ≤ 0.023
		2	DCS-CDL > 0.023
Type de développement du logiciel (T-Dev)	B	1	T-Dev = R/A (Redéveloppement/ Amélioration)
		2	T-Dev = N (Nouveau développement)
Nombre de versions tous types confondus (Version)	C	1	Version ≤ 2
		2	Version > 2

Deux niveaux sont choisis pour chaque facteur du plan dont la détermination des niveaux est documentée comme suit :

- pour le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL), la valeur 0.023 représente la médiane de ce facteur dans l'échantillon de projets d'ISBSG de ce plan. Il en est de même pour le facteur nombre de versions (Version) pour la valeur 2. Ainsi, les projets ayant la valeur du facteur inférieure à la médiane représentent le niveau 1 et ceux avec la valeur du facteur supérieure à la médiane représentent le niveau 2;
- pour le facteur relatif au type de développement (T-Dev), les niveaux (1 et 2) représentent les types offerts par ISBSG.

10.6.3 Choix de la table orthogonale du plan

Ce cas de plan comprend trois facteurs avec deux niveaux chacun et sans aucune interaction. Ainsi, le degré de liberté du plan étudié est égal à trois degrés de liberté (voir section 7.4.1). La table orthogonale de Taguchi à choisir est donc la table L4 (quatre essais à traiter). Le Tableau 10.2 montre la matrice d'expériences de ce plan.

Tableau 10.2

Matrice d'expériences (Cas 1)

Liste des Essais	Liste des Facteurs du Plan		
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	**Type de développement (T-Dev)**	**Nombre de versions (Version)**
1	≤ 0.023	R/A	≤ 2
2	≤ 0.023	N	> 2
3	> 0.023	R/A	> 2
4	> 0.023	N	≤ 2

10.6.4 Réalisation

10.6.4.1 Exécution

Le Tableau 10.3 montre le détail des huit projets choisis (des 13 projets) de l'échantillon de projets d'ISBSG de ce plan. Les autres projets ne sont pas utilisés, comme déjà mentionné dans les chapitres 8 et 9, soit par limite de nombre de répétitions ou par non-satisfaction aux conditions des essais de la matrice d'expériences de ce plan.

Le résultat de l'affectation des projets du Tableau 10.3 aux essais de la matrice d'expériences (Tableau 10.2) est présenté dans le Tableau 10.4. En effet, le nombre de répétitions dans ce cas de plan est égal à deux. Ainsi, pour chaque essai du Tableau 10.2 correspondent deux projets du Tableau 10.3. Par exemple, les deux premiers projets (12922 et 31237) correspondent à l'essai numéro 1. Les projets qui suivent (10178 et 19730) correspondent à l'essai numéro 2. Le même scénario est à suivre pour l'affectation des autres projets aux essais 3 et 4.

Tableau 10.3

Caractéristiques des projets d'ISBSG (Cas 1)

ID projet	Taille du projet	Nombre de changements de spécifications - CDL	Densité du nombre changements de spécifications – CDL	Type de développement	Nombre de versions	Degré de satisfaction des utilisateurs
12922	71	0	0.000	Amélioration	1	6
31237	146	3	0.021	Amélioration	2	7
10178	826	1	0.001	Nouveau développement	5	8
19730	795	3	0.004	Nouveau développement	3	6
27560	199	7	0.035	Redéveloppement	13	5
28553	788	39	0.049	Redéveloppement	25	5
23918	34	6	0.176	Nouveau développement	2	6
28020	186	5	0.027	Nouveau développement	1	6

Tableau 10.4

Affectation des projets aux essais (Cas 1)

Liste des Essais	Liste des Facteurs du Plan			Degré satisfaction des utilisateurs	
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	Type de développement (T-Dev)	Nombre de versions (Version)	Y1	Y2
1	≤ 0.023	R/A	≤ 2	6	7
2	≤ 0.023	N	> 2	8	6
3	> 0.023	R/A	> 2	5	5
4	> 0.023	N	≤ 2	6	6

10.6.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Rappelons que l'objectif dans ce cas de plan est de maximiser le degré de satisfaction des utilisateurs. Ainsi, le calcul du ratio signal-bruit (S/B) dans le cas où *l'optimum est une valeur maximale* est effectué selon la formule (7.5). Le résultat du calcul des ratios signal-bruit des essais (lignes) du Tableau 10.4 est présenté dans le Tableau 10.5.

Tableau 10.5

Calcul des ratios S/B (Cas 1)

Liste des Essais	**Liste des Facteurs du Plan**			**Degré de satisfaction des utilisateurs**		**Ratio signal-bruit**
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	Type de développement (T-Dev)	Nombre de versions (Version)	Y1	Y2	S/B
1	≤ 0.023	R/A	≤ 2	6	7	16.18
2	≤ 0.023	N	> 2	8	6	16.64
3	> 0.023	R/A	> 2	5	5	13.98
4	> 0.023	N	≤ 2	6	6	15.56

2) Calcul des effets des facteurs

Dans ce cas de plan, les trois facteurs ont tous deux niveaux chacun. Alors, l'effet d'un facteur est obtenu en calculant la différence entre la moyenne des ratios S/B au niveau 1 du facteur et la moyenne des ratios S/B au niveau 2 du facteur (voir section 7.4.2.2).

Le Tableau 10.6 présente les résultats du calcul des moyennes des ratios S/B pour les deux niveaux des trois facteurs de ce plan et du calcul des effets de ces facteurs.

Tableau 10.6

Calcul des effets des facteurs (Cas 1)

	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	Type de développement (T-Dev)	Nombre de versions (Version)
Moyenne des ratios S/B au niveau 1	16.41	15.08	15.87
Moyenne des ratios S/B au niveau 2	14.77	16.10	15.31
Effet du facteur (Delta)	1.64	1.02	0.56

La Figure 10.2 représente les deux niveaux de chaque facteur du plan d'analyse et leurs valeurs correspondantes de la moyenne des ratios S/B. La moyenne globale représente la moyenne des ratios S/B de tous les essais du Tableau 10.5 : (15.59).

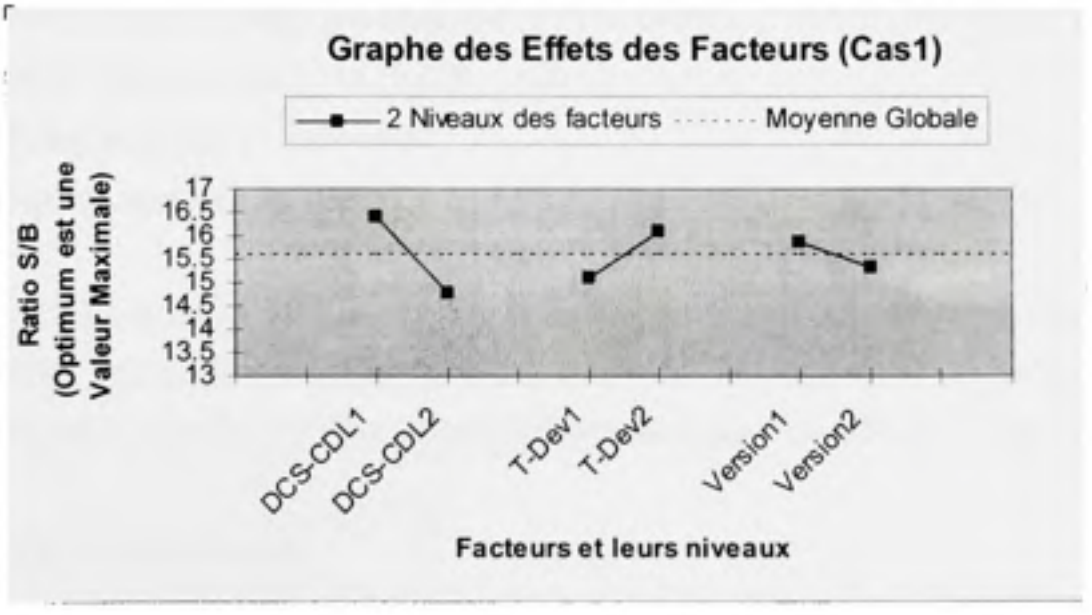

Figure 10.2 ***Graphe des effets des facteurs (Cas 1).***

Du Tableau 10.6 et de la Figure 10.2, il est à remarquer que les facteurs de ce plan présentent des effets différents sur le degré de satisfaction des utilisateurs, à savoir :

- le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) est le plus influent avec un effet de 1.64;
- le facteur relatif au type de développement (T-Dev) vient en deuxième position d'influence avec un effet de 1.02;
- le facteur relatif au nombre de versions (Version) est le moins influent avec un effet de 0.56.

3) Détermination de la condition optimale

La condition optimale représente les niveaux optimums des facteurs, c'est-à-dire les niveaux des facteurs ayant la plus grande moyenne des ratios S/B.

Du calcul des effets des facteurs (Tableau 10.6) et du graphe des effets des facteurs (Figure 10.2), les niveaux optimums pour les trois facteurs de ce plan sont comme suit :

- DCS-CDL au niveau 1 : DCS-CDL $\leq$ 0.023;
- T-Dev au niveau 2 : T-Dev = N;
- Version au niveau 1 : Version $\leq$ 2.

La condition optimale est donc : la densité du nombre de changements de spécifications par CDL est inférieure ou égale à 0.023, le type de développement est un nouveau développement et le nombre de versions délivrées du logiciel est inférieur ou égal à 2.

4) Analyse de la variance

Le Tableau 10.7 présente le calcul de l'analyse de la variance de ce cas de plan d'analyse.

Tableau 10.7

Analyse de la variance (Cas 1)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DCS-CDL	1	2.68	2.68	66	8
T-Dev	1	1.04	1.04	26	3
Version	1	0.32	0.32	8	
Erreur	0	0			
Total	3	4.04		100	
Erreur estimée	1	0.32	0.32		

Du Tableau 10.7, colonne relative à la contribution, il est à noter que (Figure 10.3) :

- le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) contribue pour 66% à la variabilité du degré de satisfaction des utilisateurs;
- le facteur relatif au type de développement (T-Dev) contribue pour 26%;
- le facteur relatif au nombre de versions (Version) ne contribue que de 8% : faible contribution.

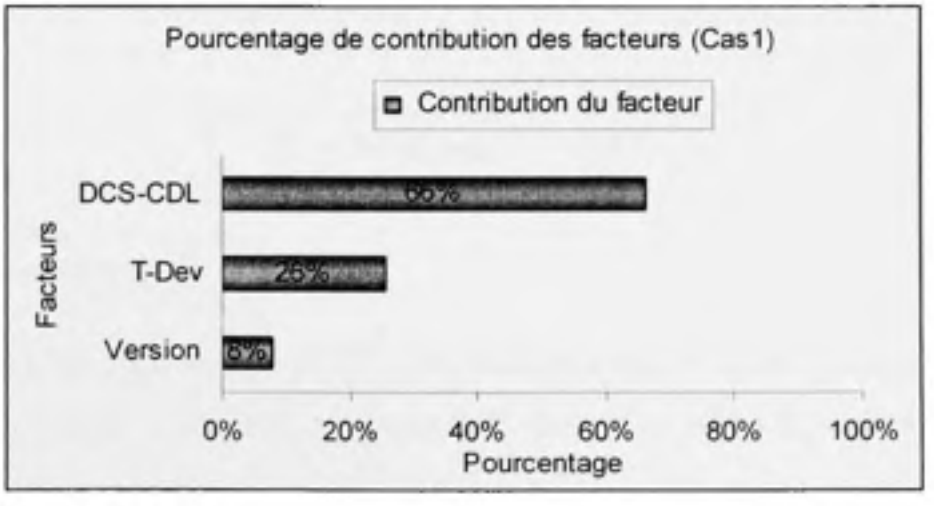

Figure 10.3 ***Pourcentage de contribution des facteurs (Cas 1).***

En outre, de l'analyse de la colonne du ratio de variance F, il est à remarquer que :

- le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) a un effet dominant avec la valeur de F = 8;
- le facteur relatif au type de développement (T-Dev) possède un effet significatif avec F = 3.

Le facteur relatif au nombre de versions (Version) est utilisé (pooling) pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

Du calcul de la variance (Tableau 10.7), seuls les facteurs (DCS-CDL) et (T-Dev) sont retenus comme facteurs les plus influents. Les niveaux optimums pour ces deux facteurs sont respectivement le niveau 1 et le niveau 2. Ainsi, l'équation de prédiction à la condition optimale est calculée seulement en fonction de ces facteurs. Le facteur (Version) a été utilisé pour estimer la variance de l'erreur.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{\text{DCS - CDL}}_{1} - \overline{T}\right) + \left(\overline{T - Dev}_{2} - \overline{T}\right) \tag{10.1}$$

$$\hat{\eta} = 16.92 \text{ avec } \overline{T} = 15.59 \tag{10.2}$$

6) Test de confirmation

Deux projets sont disponibles dans l'échantillon de projets d'ISBSG de ce plan, avec la condition optimale (déterminée à l'étape 3 de la section 10.6.4.2), regroupés dans le Tableau 10.8.

Tableau 10.8

Projets d'ISBSG - Test de confirmation (Cas 1)

ID projet	Taille du projet	Nombre de changements de spécifications -CDL	Densité du nombre changements de spécifications - CDL	Type de développement	Nombre de versions	Degré de satisfaction des utilisateurs
23009	182	4	0.022	Nouveau développement	1	6
32461	11	0	0.000	Nouveau développement	2	4

Le ratio signal-bruit pour le test de confirmation calculé avec la formule (7.5) est : S/B = (13.45).

Par ailleurs, le nombre d'essais de la matrice d'expériences est quatre, le nombre de fois que le niveau 1 du facteur (DCS-CDL) est répété dans la matrice d'expériences est deux, le nombre de fois que le niveau 2 du facteur (T-Dev) est répété dans la matrice d'expériences est deux, le nombre des expériences de vérification est deux et la variance de l'erreur est égale à 0.32. Ainsi, en utilisant les formules (7.11) et (7.13) :

- l'intervalle de confiance à 95 % de l'erreur de prédiction est : (±1.26);
- l'intervalle de confiance du résultat du ratio S/B est : [15.65, 18.18].

Le ratio S/B du test de confirmation n'existe pas dans l'intervalle de confiance du résultat du ratio S/B de ce plan. Toutefois, il est près de cet intervalle de confiance. Ce résultat peut être expliqué par deux points. Le premier point réside dans le manque de données qui a peut-être influencé d'une part le choix des facteurs et de la caractéristique de qualité étudiés dans ce plan et, d'autre part, le calcul des deux champs (DCS-CDL) et (Version) et le choix de leurs niveaux (1 et 2). Le deuxième point réside dans la possibilité de l'existence d'autres facteurs ou interactions qui influencent la caractéristique de sortie de ce plan et qui ne sont pas pris en considération dans ce plan.

10.6.5 Synthèse

Du résultat de ce plan, il en résulte que la densité du nombre de changements de spécifications par cycle de développement du logiciel, comme facteur de qualité interne du produit logiciel, est le plus influent des trois facteurs suivi du type de développement, puis du nombre de versions du logiciel sur la caractéristique de qualité choisie. Cette caractéristique de qualité correspond au degré de satisfaction des utilisateurs, considéré comme attribut de qualité en utilisation du produit logiciel.

Le degré de satisfaction des utilisateurs dans ce plan prend en considération un sous-ensemble d'items du sondage d'ISBSG, à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires.

Ainsi, l'hypothèse est confirmée : la qualité interne du produit logiciel (mesurée dans ce plan par la variable « Densité du nombre de changements de spécifications par cycle de développement du logiciel ») influence la qualité en utilisation du produit logiciel.

10.7 Plan d'analyse – Cas 2

Ce cas de plan analyse l'influence de la densité du nombre de changements de spécifications par CDL, du type de développement et de la taille du projet sur le degré de satisfaction des utilisateurs par rapport à trois items du sondage d'ISBSG (voir Tableau 9.1), à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités.

10.7.1 Échantillon de projets d'ISBSG du plan

Dans ce cas de plan, 13 projets existent dans l'échantillon « BD-ISBSG-AB-Février-2006 » de la section 7.3 avec de l'information disponible (pas de champs vides ou contenant des expressions comme « Not applicable », etc.) pour les facteurs et la caractéristique de sortie de ce plan, à savoir :

- la densité du nombre de changements de spécifications par CDL;
- le type de développement;
- la taille du projet;
- le degré de satisfaction des utilisateurs.

10.7.2 Liste des facteurs et leurs niveaux

Les facteurs étudiés dans ce cas de plan et leurs niveaux correspondants sont présentés dans le Tableau 10.9.

Tableau 10.9

Liste des facteurs et leurs niveaux (Cas 2)

Facteurs		Niveaux	Description
Densité du nombre de changements de spécifications par CDL (DCS-CDL)	A	1	DCS-CDL $\leq$ 0.023
		2	DCS-CDL > 0.023
Type de développement du logiciel (T-Dev)	B	1	T-Dev = R/A (Redéveloppement/ Amélioration)
		2	T-Dev = N (Nouveau développement)
Taille du projet (Taille)	C	1	Taille $\leq$ 190
		2	Taille > 190

Chaque facteur de ce plan possède deux niveaux dont le choix est fait de la manière suivante :

- pour le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL), la valeur 0.023 représente la médiane de ce facteur dans l'échantillon de projets d'ISBSG de ce plan. Ainsi, le niveau 1 et le niveau 2 sont constitués respectivement par les projets ayant la valeur de ce facteur inferieure et supérieure à la médiane;
- pour le facteur relatif à la taille du projet (Taille), la valeur 190 représente une valeur voisinage de la médiane de ce facteur. Ainsi, les projets ayant la valeur de ce facteur inférieure à la valeur voisinage de la médiane correspondent au niveau 1 et les autres projets correspondent au niveau 2;
- pour le facteur relatif au type de développement (T-Dev), les niveaux (1 et 2) représentent les types offerts par ISBSG.

10.7.3 Choix de la table orthogonale du plan

Dans ce cas de plan, le degré de liberté du plan étudié (voir section 7.4.1) est égal à trois degrés de liberté (trois facteurs avec deux niveaux chacun). Alors, la table orthogonale L4 est choisie pour constituer la matrice d'expériences de ce plan (Tableau 10.10).

Tableau 10.10

Matrice d'expériences (Cas 2)

Liste des Essais	**Liste des Facteurs du Plan**		
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	**Type de développement (T-Dev)**	**Taille du projet (Taille)**
1	≤ 0.023	R/A	≤ 190
2	≤ 0.023	N	>190
3	> 0.023	R/A	>190
4	> 0.023	N	≤ 190

10.7.4 Réalisation

10.7.4.1 Exécution

Le Tableau 10.11 montre le détail des huit projets choisis (des 13 projets de l'échantillon de projets d'ISBSG de ce plan) et qui correspondent aux conditions des essais de la matrice d'expériences (Tableau 10.10). Les autres projets ne sont pas utilisés pour les mêmes contraintes présentées dans la section 10.6.4.1.

Tableau 10.11

Caractéristiques des projets d'ISBSG (Cas 2)

ID projet	Taille du projet	Nombre de changements de spécifications - CDL	Densité du nombre changements de spécifications - CDL	Type de développement	Degré de satisfaction des utilisateurs
23826	118	0	0.000	Amélioration	9
31237	146	3	0.021	Amélioration	11
10178	826	1	0.001	Nouveau développement	11
19730	795	3	0.004	Nouveau développement	9
25620	1127	64	0.057	Redéveloppement	8
27560	199	7	0.035	Redéveloppement	8
23918	34	6	0.176	Nouveau développement	9
28020	186	5	0.027	Nouveau développement	9

Le nombre de répétitions pour ce cas de plan est égal à deux. Ainsi, le même scénario que la section 10.6.4.1 est à suivre dans l'affectation des huit projets du Tableau 10.11 aux quatre essais du Tableau 10.10. Le résultat de cette affectation des projets aux essais est présenté dans le Tableau 10.12.

Tableau 10.12

Affectation des projets aux essais (Cas 2)

Liste des Essais	Liste des Facteurs du Plan			Degré satisfaction des utilisateurs	
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	Type de développement (T-Dev)	Taille du projet (Taille)	Y1	Y2
1	≤ 0.023	R/A	≤190	9	11
2	≤ 0.023	N	>190	11	9
3	> 0.023	R/A	>190	8	8
4	> 0.023	N	≤190	9	9

10.7.4.2 Analyse et Interprétation

1) Calcul du ratio signal-bruit (S/B)

Le calcul du ratio signal-bruit (effectué en utilisant la formule (7.5)) pour chaque essai du Tableau 10.12 est présenté dans le Tableau 10.13.

Tableau 10.13

Calcul des ratios S/B (Cas 2)

Liste des Essais	Liste des Facteurs du Plan			Degré de satisfaction des utilisateurs		Ratio signal-bruit
	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	Type de développement (T-Dev)	Taille du projet (Taille)	Y1	Y2	S/B
1	≤ 0.023	R/A	≤190	9	11	19.87
2	≤ 0.023	N	>190	11	9	19.87
3	> 0.023	R/A	>190	8	8	18.06
4	> 0.023	N	≤190	9	9	19.08

2) Calcul des effets des facteurs

Le calcul des effets des trois facteurs de ce plan est présenté dans le Tableau 10.14.

Tableau 10.14

Calcul des effets des facteurs (Cas 2)

	Densité du nombre de changements de spécifications - CDL (DCS-CDL)	**Type de développement (T-Dev)**	**Taille du projet (Taille)**
Moyenne des ratios S/B au niveau 1	19.87	18.97	19.48
Moyenne des ratios S/B au niveau 2	18.57	19.48	18.97
Effet du facteur (Delta)	1.30	0.51	0.51

La Figure 10.4 présente les moyennes des ratios S/B des deux niveaux des facteurs étudiés dans ce cas de plan.

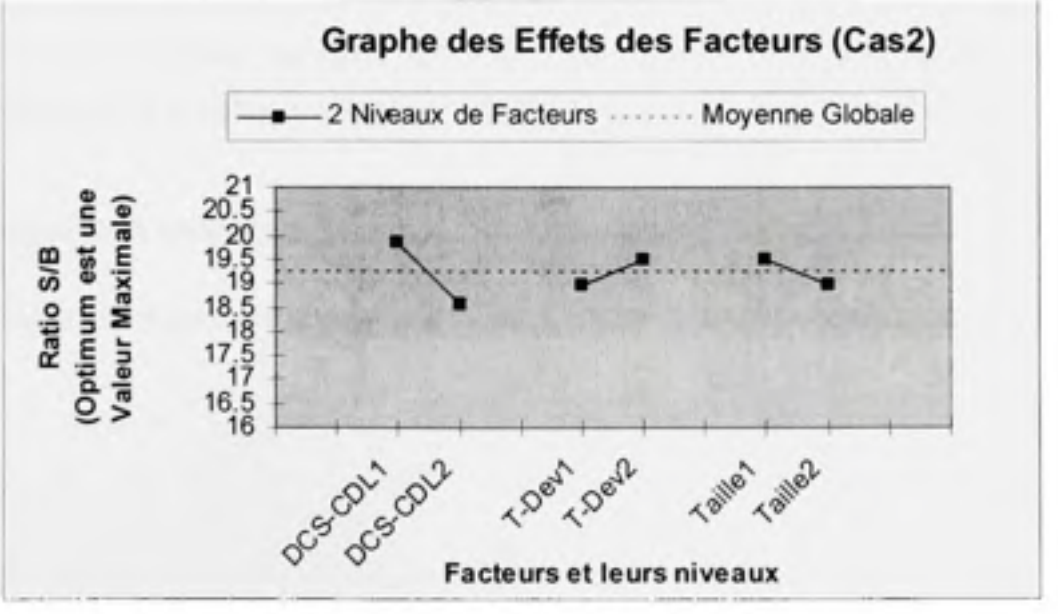

Figure 10.4 ***Graphe des effets des facteurs (Cas 2).***

La moyenne globale représente la moyenne de tous les essais du Tableau 10.13 : (19.22).

Du Tableau 10.14 et de la Figure 10.4, il apparait que les facteurs étudiés dans ce cas de plan possèdent des effets différents sur le degré de satisfaction des utilisateurs, à savoir :

- le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) est le plus influent avec un effet de 1.30;
- les facteurs relatifs au type de développement (T-Dev) et à la taille du projet (Taille) présentent une faible influence avec un effet de 0.51 chacun.

3) Détermination de la condition optimale

À partir du Tableau 10.14 et de la Figure 10.4, les niveaux optimums des facteurs de la condition optimale sont les suivants :

- DCS-CDL au niveau 1 : DCS-CDL ≤ 0.023;
- T-Dev au niveau 2 : T-Dev = N;
- Taille au niveau 1 : Taille ≤ 190.

La condition optimale est donc : la densité du nombre de changements de spécifications par CDL est inferieure ou égale 0.023, le type de développement est un nouveau développement et la taille du projet est inférieure ou égale à 190 points de fonction.

4) Analyse de la variance

Le Tableau 10.15 présente le calcul de l'analyse de la variance pour ce cas de plan.

Tableau 10.15

Analyse de la variance (Cas 2)

Facteurs/ Source de variation	Degré de liberté	Somme des carrés	Variance	Contribution en (%)	Ratio de variance F
DCS-CDL	1	1.68	1.68	76	6
T-Dev	1	0.26	0.26	12	
Taille	1	0.26	0.26	12	
Erreur	0	0			
Total	3	2.20		100	
Erreur estimée	2	0.52	0.26		

Du Tableau 10.15, colonne relative à la contribution, il est à constater que (Figure 10.5) :

- le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) possède la plus grande contribution, soit 76 % à la variabilité du degré de satisfaction des utilisateurs;
- les facteurs relatifs au type de développement (T-Dev) et à la taille du projet (Taille) contribuent tous les deux de 24%, soit une contribution de 12% chacun.

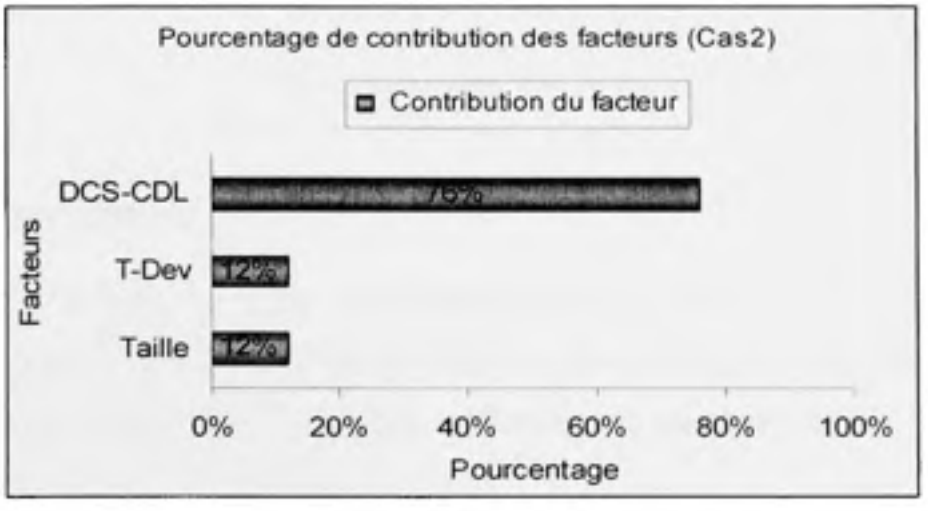

Figure 10.5 ***Pourcentage de contribution des facteurs (Cas 2).***

En outre, de l'analyse de la colonne relative au ratio de variance F, il est à noter que le facteur relatif à la densité du nombre de changements de spécifications par CDL (DCS-CDL) possède un effet dominant avec la valeur de F = 6.

Les deux autres facteurs relatifs au type de développement (T-Dev) et à la taille du projet (Taille) sont utilisés pour constituer la variance de l'erreur estimée.

5) Calcul de l'équation de prédiction

De l'analyse de la variance, le facteur le plus influent du plan d'analyse étudié est le facteur (DCS-CDL) et son niveau optimum est le niveau 1. Ainsi, l'équation de prédiction à la condition optimale pour ce cas de plan est calculée seulement en fonction de ce facteur. Les facteurs (T-Dev) et (Taille) sont utilisés pour estimer la variance de l'erreur.

Le résultat prévu à la condition optimale est :

$$\hat{\eta} = \overline{T} + \left(\overline{\text{DCS - CDL}_1} - \overline{T}\right) \tag{10.3}$$

$$\hat{\eta} = 19.87 \text{ avec } \overline{T} = 19.22 \tag{10.4}$$

6) Test de confirmation

Dans ce cas de plan, en raison de l'insuffisance de projets (un seul projet) dans l'échantillon de projets d'ISBSG de ce plan avec la condition optimale (déterminée à l'étape 3 de la section 10.7.4.2), le test de confirmation ne peut être effectué.

10.7.5 Synthèse

Dans ce cas de plan, de l'analyse du ratio signal-bruit, il en résulte que la densité du nombre de changements de spécifications par cycle de développement du logiciel, comme facteur de qualité interne du produit logiciel, est le plus influent des trois facteurs. Les facteurs relatifs au type de développement et à la taille du projet viennent en deuxième position avec des influences égales sur le degré de satisfaction des utilisateurs, considéré comme attribut de qualité en utilisation du produit logiciel.

Le degré de satisfaction des utilisateurs dans ce plan prend en considération un sous-ensemble d'items du sondage d'ISBSG, à savoir :

- la satisfaction des objectifs établis;
- la satisfaction des exigences d'affaires;
- la qualité des fonctionnalités.

Ainsi, l'hypothèse est confirmée : la qualité interne du produit logiciel (mesurée dans ce plan par la variable « Densité du nombre de changements de spécifications par cycle de développement du logiciel ») influence la qualité en utilisation du produit logiciel.

10.8 Remarques

Comme nous avons mentionné dans les sections 8.9 et 9.9 des chapitres 8 et 9, la condition optimale déterminée à partir des plans d'analyses est importante puisqu'elle permet de déterminer les niveaux optimums des facteurs étudiés, lesquels niveaux sont à prendre en considération lors de nouveaux projets logiciels.

Cependant, pour le premier cas de plan d'analyse de ce chapitre, le test de confirmation n'a pas permis d'appuyer la solution recommandée pour les raisons précitées (à l'étape 6 de la section 10.6.4.2). L'une des raisons consiste en l'existence d'autres facteurs qui

peuvent influencer la sortie étudiée. Ainsi, le fait que les experts d'ISO 9126 considèrent que la qualité en utilisation soit l'effet combiné de la qualité interne et de la qualité externe pour l'utilisateur (ISO, 2001c) et non seulement de la qualité interne laisse supposer que ces facteurs peuvent être des facteurs de qualité externe. En effet, à cause de l'insuffisance de données disponibles dans l'extrait d'ISBSG, aucun cas de plan d'analyse n'a pu être conçu permettant d'étudier l'influence des facteurs de ces deux types de qualité (interne et externe) ensemble sur la qualité en utilisation. Le but est de vérifier cette supposition et ainsi confirmer l'absence de lien direct entre la qualité interne et la qualité en utilisation dans les modèles de la Figure 2.2 et de la Figure 2.4 d'ISO 9126.

Par ailleurs, l'intérêt est de savoir si les facteurs de qualité interne influencent la qualité en utilisation. En effet, le calcul des effets des facteurs étudiés et le calcul de l'ANOVA donnent de façon générale les facteurs influents de ceux non influents. Ces informations relatives au degré d'influence de chaque facteur sont utiles pour le génie logiciel en tant que discipline, qui vise à atteindre un certain niveau de maturité, de diverses manières. À titre d'exemple, établir les bonnes pratiques en matière de qualité du produit logiciel, lesquelles pratiques une fois appliquées permettront aux concepteurs et aux développeurs d'assurer la production de logiciels de qualité.

10.9 Conclusion

Dans ce chapitre nous avons traité le volet 3 relatif à l'hypothèse du lien entre la qualité interne et la qualité en utilisation du produit logiciel de la phase 3 de la méthodologie de recherche (chapitre 5). Il est important de noter que les experts d'ISO 9126 n'ont pas formulé l'existence de liens directs entre ces deux types de qualité.

Pour vérifier l'hypothèse de l'influence de la qualité interne sur la qualité en utilisation du produit logiciel, nous avons présenté deux plans d'analyses conçus avec la méthode

Taguchi tout en exploitant l'extrait de données d'ISBSG de février 2006, mis à notre disposition pour des fins de recherche.

L'objectif des plans d'analyses de ce chapitre est de maximiser le degré de satisfaction des utilisateurs. Ainsi, l'analyse des résultats de ces plans a été effectuée avec la méthode d'analyse du ratio signal-bruit de Taguchi dans le cas où la caractéristique de qualité correspond à *l'optimum est une valeur maximale*.

Les résultats des deux plans montrent que le facteur relatif à la densité du nombre de changements de spécifications par cycle de développement du logiciel est le plus influent des facteurs étudiés dans ce chapitre sur le degré de satisfaction des utilisateurs.

Ainsi, l'hypothèse principale établie dans ce chapitre est confirmée vu que :

- la densité du nombre de changements de spécifications par cycle de développement du logiciel représente une mesure de qualité interne du produit logiciel permettant d'évaluer la sous-caractéristique « Aptitude » de la caractéristique « Capacité fonctionnelle » du modèle de qualité interne d'ISO 9126-1;
- le degré de satisfaction des utilisateurs représente une mesure de qualité en utilisation du produit logiciel permettant d'évaluer la caractéristique de qualité « Satisfaction » du modèle de qualité en utilisation d'ISO 9126-1.

CONCLUSION

En génie logiciel, l'importance de la qualité est reconnue et le problème de produire des logiciels de qualité est étudié par beaucoup de chercheurs individuels. Il existe également quelques rares normes en qualité du logiciel, telle la norme ISO 9126 sur la qualité du produit logiciel.

Cette norme internationale ISO 9126-1 propose trois modèles de qualité pour les trois types de qualité à satisfaire afin d'assurer la production de logiciels de qualité; il s'agit de la qualité interne, de la qualité externe et de la qualité en utilisation. Les experts d'ISO prennent pour acquis l'existence de liens entre ces trois types de qualité (Figure 2.1), lesquels liens demeurent néanmoins des propositions théoriques dont les fondements n'ont pas été démontrés en utilisant des approches scientifiques reconnues incluant des expérimentations ou des études empiriques.

Le but principal de cette thèse est d'explorer la pertinence des relations entre les modèles de qualité d'ISO 9126 dans la production de logiciels de qualité. Afin de démontrer la pertinence ou non des relations entre ces modèles, les trois objectifs spécifiques de cette recherche ont été de démontrer, par des études empiriques, si les relations prises pour acquises par ISO 9126 sont supportées par des données empiriques. Il s'agit des relations entre :

1. La qualité interne et la qualité externe.
2. La qualité externe et la qualité en utilisation.
3. La qualité interne et la qualité en utilisation.

Pour remplir ces objectifs nous avons procédé dans cette thèse par trois grandes phases : exploration, préparation et plan d'analyse empirique. Ces phases, résumées au fil de cette section sont importantes l'une par rapport à l'autre.

Bilan du Travail

La première phase de cette thèse, **la phase d'exploration**, a été partagée en deux étapes : exploration globale et exploration détaillée.

La première étape, étape d'exploration globale (chapitres 1, 2, 3 et 4), a porté sur les différents axes principaux de ce projet de recherche : la qualité du produit logiciel, la série ISO 9126, les référentiels de données en génie logiciel et la méthode Taguchi de conception de plans d'expériences. De cette exploration et de la revue de la littérature, il en ressort en général que la définition de la qualité dans la norme ISO 9126-1 représente la seule définition qui englobe tous les aspects de la qualité. Le modèle de qualité d'ISO 9126-1 est le modèle le plus développé parmi les modèles de la littérature du logiciel et tient compte de la qualité durant tout le cycle de vie du logiciel. De la revue de la littérature sur la série ISO 9126, il en résulte que les mesures proposées dans les rapports techniques ISO TR 9126-2 à 4 nécessitent des améliorations et des vérifications afin qu'elles reflètent adéquatement les caractéristiques mesurées.

Les études empiriques et l'expérimentation représentent de bons moyens pour améliorer les pratiques du génie logiciel et, par la suite, contribuer à la maturité d'une telle discipline. Toutefois, l'expérimentation repose sur la conception de plans d'expériences. Dans la littérature, plusieurs méthodes de conception de plans d'expériences sont disponibles, incluant la méthode Taguchi nommé ainsi du nom de son concepteur le Dr. Taguchi. Cette méthode Taguchi est basée sur des pratiques industrielles et statistiques dont les apports pour le génie logiciel résident dans la réduction du coût, l'amélioration de la qualité dès la conception et l'amélioration des connaissances. De la revue de la littérature sur la méthode Taguchi, il en ressort que c'est une méthode qui n'est pas réservée qu'au domaine du génie industriel et que son utilisation dans le domaine du logiciel en est à ses débuts.

D'autre part, tout plan d'expériences nécessite une réalisation afin de remplir ses objectifs, laquelle réalisation ne peut s'effectuer sans une collecte de données. En effet, la réalisation des expériences dans un milieu de calibre industriel est préférable. Cependant, collecter les données dans un milieu industriel est confronté à un ensemble d'obstacles : le temps, le coût et la disponibilité du partenaire industriel. Ainsi, compte tenu de ces obstacles et de l'absence d'opportunités d'expérimentations réelles, nous avons exploré un ensemble de référentiels de données disponibles dans la littérature en génie logiciel.

Parmi ces référentiels, nous avons choisi d'utiliser dans cette recherche le référentiel de données industrielles de l'International Software Benchmarking Standards Group - ISBSG - puisque ce référentiel offre différentes catégories d'informations relatives au projet logiciel, et ce pour les différentes phases de son cycle de vie. L'extrait de ce référentiel de données d'ISBSG-Release 9 de 2005 de plus de 3000 projets était disponible à l'École de technologie supérieure pour des fins de recherche.

En résumé, cette étape d'exploration globale nous a permis d'approfondir notre problématique de recherche et ses principaux axes. De notre revue de la littérature effectuée dans cette étape, il en résulte qu'aucun travail de recherche n'avait abordé cette problématique avec une démarche expérimentale et statistique.

La deuxième étape, étape d'exploration détaillée (chapitre 6), a porté sur un ensemble d'analyses de la série des documents d'ISO 9126 (modèles de qualité et mesures) par rapport au questionnaire ISBSG de collecte de données et l'extrait de données du référentiel d'ISBSG-Release 9 de 2005.

De la mise en correspondance (de haut niveau et détaillée) des modèles de qualité d'ISO 9126 par rapport au questionnaire de collecte de données d'ISBSG, nous avons constaté

que le questionnaire d'ISBSG collecte des informations pour les trois types de qualité de la norme ISO 9126 de la qualité du produit logiciel. Il en ressort aussi que :

- la section « Processus du projet » peut fournir de l'information pour deux des trois types de qualité : interne et externe;
- la section « Fin du projet » peut fournir de l'information pour deux des trois types de qualité : externe et en utilisation;
- les autres sections, « Technologie », « Personne et Effort du travail » et « Produit » ne peuvent fournir de l'information pour aucun des trois modèles de qualité d'ISO 9126 et les informations de la section « Taille fonctionnelle du projet » sont utiles pour des objectifs de normalisation lors du calcul des ratios en relation avec la qualité.

De l'analyse de l'extrait de données d'ISBSG-Release 9 de 2005, nous avons relevé qu'un nombre minime de données relatives à la qualité du logiciel est rendu disponible automatiquement par ISBSG dans cet extrait. Cependant, ces données ne concernent qu'une partie des types de qualité d'ISO 9126. En fait, dans la pratique, ISBSG ne fournit pas tout le contenu de son référentiel, mais rend disponible, à un coût supplémentaire raisonnable, un échantillon (i.e., extrait) de son référentiel à l'industrie et aux chercheurs. Ces derniers peuvent accéder à des données spécifiques de recherche à condition qu'ils documentent et soumettent une demande d'accès à ces données additionnelles. Pour ce projet de recherche, il a fallu que nous soumettions une demande d'accès aux données spécifiques à notre recherche auprès de l'organisation ISBSG, laquelle demande a permis d'obtenir un nouvel extrait de données d'ISBSG de février 2006.

De la mise en correspondance (de haut niveau et détaillée) des rapports techniques ISO TR 9126-2 à 4 et du questionnaire d'ISBSG, il a été constaté que dans ISBSG il y a des données de qualité disponibles que pour quelques caractéristiques et sous-caractéristiques des modèles de qualité d'ISO 9126 :

- la capacité fonctionnelle, la fiabilité et la maintenabilité à travers les sous-caractéristiques suivantes : l'aptitude, la maturité et la facilité de modification pour la qualité interne et la qualité externe du produit logiciel;
- la satisfaction pour la qualité en utilisation du produit logiciel.

Les autres caractéristiques et sous-caractéristiques des modèles de qualité d'ISO 9126 (Figures 2.2 et 2.3) ne sont mentionnées ni explicitement ni implicitement dans le questionnaire d'ISBSG.

En outre, en dépit des différences que présentent ISO 9126 et ISBSG, nous avons relevé certaines similarités, lesquelles correspondent aux mesures de base du logiciel. Il est important de noter que ces mesures de base similaires ne représentent qu'un petit ensemble (soit 6) de l'ensemble des mesures de base des rapports techniques ISO TR 9126-2 à 4. Nous avons mené ensuite une analyse de ces mesures de base similaires d'ISO 9126 et son correspondant d'ISBSG à travers des exemples de mesures proposées dans les rapports techniques d'ISO 9126. Cette analyse a révélé un manque de clarté dans la définition de l'attribut mesuré et que rares sont les attributs mesurés ayant un standard de défini aussi bien dans ISO 9126 que dans ISBSG.

Pour remédier à cette insuffisance et assurer la répétabilité et la reproductibilité du résultat de la mesure, nous suggérons aussi bien pour ISBSG que pour ISO 9126 de :

- identifier clairement les mesures de base puisqu'elles représentent la base de toute mesure dérivée (i.e., composition de plusieurs mesures de base);
- fournir une définition claire de l'attribut mesuré et dans le cas échéant décrire de façon consistante sa méthode de mesure;
- utiliser des standards du génie logiciel et d'autres domaines de la science qui offrent des définitions standards à propos des termes et des concepts relatifs au domaine de la mesure.

En résumé, les travaux entrepris dans cette deuxième étape de la phase d'exploration, étape d'exploration détaillée, se sont avérés importants et nécessaires pour notre recherche. Il en résulte que nous disposons d'une base de données (extrait de données d'ISBSG de février 2006) qui couvre les trois types de qualité d'ISO 9126 et que nous avons pu identifier un ensemble de mesures propres à ISBSG pour évaluer la qualité du produit logiciel durant tout le cycle de vie du logiciel. Ces mesures sont identifiées à partir des mesures de base similaires entre ISO 9126 et ISBSG, et dont la structure de documentation est partiellement arrimée à celle des rapports techniques d'ISO TR 9126-2 à 4. Ces mesures ont par la suite été utiles dans la conception des plans d'analyses empiriques avec la méthode Taguchi afin de vérifier les liens entre les trois types de qualité de la norme ISO 9126-1.

Les résultats de cette étape sont également utiles de différentes manières pour les publics suivants :

- pour l'industrie du logiciel en général afin que les organisations puissent comparer leurs données de qualité de logiciel par rapport au référentiel international d'ISBSG;
- pour les chercheurs dans la préparation de leurs plans de recherche détaillés afin de demander à ISBSG les d'informations concernant la qualité du logiciel;
- pour l'industrie du logiciel afin d'analyser et mettre en application des modèles d'ISO 9126 de la qualité du logiciel;
- pour l'organisation ISBSG elle-même pour améliorer l'alignement de leurs standards de collecte de données par rapport aux modèles ISO 9126 de la qualité du logiciel.

Dans la deuxième phase de cette recherche, **phase de préparation**, nous avons procédé à la préparation de l'environnement des plans d'analyses empiriques (chapitre 7). Cette préparation a porté sur trois points.

Le premier point correspond à la sélection des données à exploiter à partir du nouvel extrait de données d'ISBSG de février 2006. De cette préparation, il en ressort qu'en

plus de la catégorie des données de qualité du logiciel, les différentes autres catégories d'informations en relation avec le projet logiciel sont aussi à prendre en considération lors de la conception des plans d'analyses avec Taguchi.

Le deuxième point correspond à la préparation des données elles-mêmes de cet extrait de données d'ISBSG : il s'agit de la vérification de la qualité et de la complétude de ces données. Seuls les projets qualifiés par les gestionnaires d'ISBSG ayant un bon degré d'intégrité de données collectées (projets avec code A et B) sont retenus pour constituer l'échantillon de projets d'ISBSG avec lequel nous pouvons mener les expérimentations. Il en résulte aussi de cette préparation que le problème de données manquantes parmi les différents champs de données de cet échantillon de projets (A et B) exige une autre étape de raffinement, et ce, pour chaque plan d'analyse conçu.

Le troisième point est une description détaillée des étapes à suivre lors de l'analyse des résultats des plans d'expériences avec l'analyse du ratio signal-bruit de Taguchi ainsi que les calculs nécessaires.

La dernière phase de cette recherche, **phase de plan d'analyse empirique**, représente la phase de conception des plans d'analyses avec la méthode Taguchi tout en exploitant l'extrait de données d'ISBSG de février 2006 afin de démontrer la pertinence ou non des relations prises pour acquises par la norme ISO 9126 entre ses trois types de qualité.

Pour vérifier la première hypothèse du lien entre la qualité interne et la qualité externe de la norme ISO 9126-1, nous avons conçu trois plans d'analyses (chapitre 8). L'ensemble des facteurs étudiés dans ces plans sont : la densité du nombre de changements de spécifications (phase design/codage), la taille fonctionnelle du projet, la plateforme de développement, le type de développement du logiciel et la stabilité de l'équipe de développement. Chaque plan d'analyse permet d'étudier l'influence d'un sous-ensemble de ces facteurs (trois facteurs sans aucune interaction et chaque facteur

possède deux niveaux) sur la caractéristique de qualité du plan. Cette caractéristique de qualité représente la densité du nombre de défauts collectés lors de la phase de test, et ce, pour les trois plans. L'objectif de chaque plan d'analyse est de minimiser la caractéristique de qualité, c'est-à-dire minimiser la densité du nombre de défauts collectés lors de la phase de test.

De l'analyse des résultats de ces trois plans en utilisant l'analyse du ratio signal-bruit de Taguchi dans le cas où *l'optimum est une valeur minimale*, il en ressort que le facteur relatif à la densité du nombre de changements de spécifications de la phase de codage/ design possède la plus grande influence, parmi les facteurs étudiés, sur la densité du nombre de défauts détectés lors de la phase de test. Il en résulte aussi que le niveau préférable est d'avoir une faible densité du nombre de changements de spécifications lors de la phase de codage/design afin de minimiser la densité du nombre de défauts détectés lors de la phase de test du produit logiciel. Ainsi, l'hypothèse du lien entre la qualité interne et la qualité externe est confirmée puisque :

- la densité du nombre de changements de spécifications lors de la phase de codage (design) représente une mesure de qualité interne du produit logiciel, laquelle mesure permet d'évaluer la sous-caractéristique de qualité « Aptitude » de la caractéristique de qualité « Capacité fonctionnelle » du modèle de qualité interne d'ISO 9126-1;
- la densité du nombre de défauts détectés lors de la phase de test représente une mesure de qualité externe du produit logiciel, laquelle mesure permet d'évaluer la sous-caractéristique de qualité « Maturité » de la caractéristique de qualité « Fiabilité » du modèle de qualité externe d'ISO 9126-1.

Pour vérifier la deuxième hypothèse du lien entre la qualité externe et la qualité en utilisation de la norme ISO 9126-1, nous avons conçu trois plans d'analyses (chapitre 9). L'ensemble des facteurs étudiés dans ces plans sont : la densité du nombre de défauts collectés (phase implémentation/fin du projet), la taille fonctionnelle du projet, le type de langage de programmation, le type de développement du logiciel et l'expérience du

chef du projet. Chaque plan d'analyse permet d'étudier l'influence d'un sous-ensemble de ces facteurs (trois facteurs sans aucune interaction et chaque facteur possède deux niveaux) sur la caractéristique de qualité du plan. Cette caractéristique de qualité représente le degré de satisfaction des utilisateurs, et ce pour les trois plans. L'objectif de chaque plan d'analyse est de maximiser la caractéristique de qualité, c'est-à-dire maximiser le degré de satisfaction des utilisateurs.

De l'analyse des résultats de ces trois plans en utilisant l'analyse du ratio signal-bruit de Taguchi dans le cas où *l'optimum est une valeur maximale*, nous avons remarqué que le facteur relatif à la densité du nombre de défauts collectés de la phase implémentation/ fin du projet possède le plus grand impact, parmi les facteurs étudiés, sur le degré de satisfaction des utilisateurs. En outre, nous avons constaté que le niveau préférable est d'avoir une faible densité du nombre de défauts collectés lors de la phase implémentation/fin du projet afin de maximiser le degré de satisfaction des utilisateurs. Ainsi, l'hypothèse du lien entre la qualité externe et la qualité en utilisation est confirmée étant donné que :

- la densité du nombre de défauts collectés de la phase implémentation (fin du projet) représente une mesure de qualité externe du produit logiciel, laquelle mesure permet d'évaluer la sous-caractéristique de qualité « Maturité » de la caractéristique de qualité « Fiabilité » du modèle de qualité externe d'ISO 9126-1;
- le degré de satisfaction des utilisateurs représente une mesure de qualité en utilisation du produit logiciel, laquelle mesure permet d'évaluer la caractéristique de qualité « Satisfaction » du modèle de qualité en utilisation d'ISO 9126-1.

Pour vérifier la troisième hypothèse du lien entre la qualité interne et la qualité en utilisation, nous avons conçu deux plans d'analyses (chapitre 10). La caractéristique de qualité de ces plans représente le degré de satisfaction des utilisateurs. L'ensemble des facteurs étudiés dans ces plans sont : la densité du nombre de changements de spécifications par cycle de développement du logiciel (durant les phases de design,

codage, test et d'implémentation), la taille fonctionnelle du projet, le nombre de versions délivrées du logiciel et le type de développement du logiciel. Chaque plan d'analyse permet d'étudier l'influence d'un sous-ensemble de ces facteurs (trois facteurs sans aucune interaction et chaque facteur possède deux niveaux) sur la caractéristique de qualité du plan. L'objectif de chaque plan d'analyse est de maximiser la caractéristique de qualité, c'est-à-dire maximiser le degré de satisfaction des utilisateurs.

De l'analyse des résultats de ces deux plans en utilisant l'analyse du ratio signal-bruit de Taguchi dans le cas où *l'optimum est une valeur maximale*, il en résulte que le facteur relatif à la densité du nombre de changements de spécifications par cycle de développement du logiciel est le plus influent des facteurs étudiés sur le degré de satisfaction des utilisateurs. Ainsi, l'hypothèse du lien entre la qualité interne et la qualité en utilisation est confirmée vu que :

- la densité du nombre de changements de spécifications par cycle de développement du logiciel représente une mesure de qualité interne du produit logiciel permettant d'évaluer la sous-caractéristique « Aptitude » de la caractéristique « Capacité fonctionnelle » du modèle de qualité interne d'ISO 9126-1;
- le degré de satisfaction des utilisateurs représente une mesure de qualité en utilisation du produit logiciel permettant d'évaluer la caractéristique de qualité « Satisfaction » du modèle de qualité en utilisation d'ISO 9126-1.

Il est à signaler que le choix des facteurs (et leurs niveaux) et de la caractéristique de qualité étudiés dans chaque cas de plan (des chapitres 8, 9 et 10) a été effectué en fonction de la disponibilité des données (non vide) pour ces facteurs et cette caractéristique de qualité dans l'extrait d'ISBSG. De plus, les résultats de ces cas de plan sont obtenus en étudiant quelques variables, en utilisant un petit échantillon de projets et parfois sans test de confirmation. Ceci vient du fait que les données manquantes sont présentes avec des degrés différents dans l'extrait d'ISBSG.

Il est à signaler aussi que la méthode Taguchi est utilisable pour étudier aussi bien un petit qu'un grand ensemble de facteurs dans des expérimentations aussi bien avec un petit qu'avec un grand échantillon de données.

Par ailleurs, il est important de noter que la condition optimale déterminée à partir des plans d'analyses conçus dans cette dernière phase de recherche (dans les chapitres 8, 9 et 10) est importante. En fait, la condition optimale permet de déterminer les niveaux optimums des facteurs étudiés, lesquels niveaux sont à prendre en considération lors de nouveaux projets logiciels. Bien que cette condition optimale soit la base de tout processus de production dans le secteur industriel, son application en génie logiciel n'est pas toujours facile vu la variabilité des contraintes du projet logiciel. Néanmoins, cette condition optimale donne une valeur informative aux concepteurs et aux développeurs du logiciel sur les pratiques à prendre en considération et celles à éviter. Lesquelles pratiques sont utiles pour la discipline du génie logiciel afin de contribuer à sa maturité.

En résumé, les travaux entrepris dans cette dernière phase, phase de plan d'analyse empirique, ont permis de vérifier les hypothèses des liens entre les trois types de qualité de la norme ISO 9126-1.

En conclusion, les trois objectifs de cette recherche sont accomplis. Néanmoins, un ensemble de limites se sont présentées.

Limites de la recherche

De façon générale, les limites de cette recherche sont présentées comme suit :

- la disponibilité d'un partenaire industriel avec lequel on peut appliquer tout le processus d'expérimentation qui débute par la conception du plan d'expériences, passe par la réalisation de ces expériences pour enfin aboutir aux résultats;
- le faible nombre de données relatives à la qualité du produit logiciel collectées par ISBSG à travers ses questionnaires de collectes de données. Cette limitation a

empêché d'effectuer d'autres travaux permettant d'étudier d'autres relations entre d'autres caractéristiques de qualité des modèles de qualité d'ISO 9126-1;

- le problème de données manquantes dans les champs de données de la base de données d'ISBSG qui a donné lieu à d'autres limites. En effet, ce problème de données manquantes a limité le choix des facteurs étudiés, des niveaux de ces facteurs et de la caractéristique de qualité à prendre en considération dans chaque plan d'analyse : ceci a limité l'utilisation des tables orthogonales de Taguchi permettant d'étudier plusieurs facteurs et plusieurs niveaux. En outre, chaque essai de la table orthogonale (matrice d'expériences) est une combinaison particulière des niveaux des facteurs du plan étudié : le manque de données a limité la taille de l'échantillon de projets d'ISBSG du plan étudié et il a réduit le nombre de répétitions pour chaque essai de la matrice d'expériences. Enfin, il résulte aussi de ce problème de données manquantes le manque de test de confirmation.

En dépit des limites rencontrées dans cette recherche, les apports de cette recherche sont importants et un ensemble de contributions ont été réalisées.

Contributions de la recherche

L'intérêt de cette recherche réside dans le développement de fondements expérimentaux bien documentés pour valider (ou infirmer) les relations entre les modèles de qualité proposés dans ISO 9126.

L'originalité de cette thèse porte sur la vérification de la pertinence ou non des relations entre les trois types de qualité d'ISO 9126, en adaptant la méthode de conception de plans d'expériences de Taguchi au contexte des études empiriques et en exploitant le référentiel de données industrielles de l'organisation ISBSG.

Les résultats de cette recherche seront utiles aux organismes de normalisation en général soucieux d'améliorer les assises rationnelles et scientifiques de leurs publications. Ces

résultats devraient également être utiles à l'industrie pour leur permettre de faire des choix parmi les modèles de qualité et les centaines de mesures proposées dans la série des documents ISO 9126.

En outre, les résultats de cette thèse pourraient contribuer à préparer des suggestions d'améliorations aux modèles et documents d'ISO 9126 de façon à progressivement mieux satisfaire les besoins de l'industrie en matière de la qualité du logiciel et contribuer progressivement à la maturité de la discipline de l'ingénierie du logiciel. Ces résultats pourraient contribuer également à préparer des suggestions d'améliorations à l'organisation ISBSG de telle sorte que son questionnaire collecte plus de données de qualité du logiciel permettant d'évaluer les différentes caractéristiques (sous-caractéristiques) des modèles de qualité d'ISO 9126-1.

Ce projet de recherche a permis, à moyen terme, de démontrer l'utilité et le bénéfice de la combinaison des deux disciplines du génie : industriel et logiciel dans l'évaluation de la qualité du logiciel en se basant sur ISO 9126 et la méthode Taguchi.

La principale contribution de cette recherche est la vérification des hypothèses des liens de la norme ISO 9126-1 entre les trois types de qualité : la qualité interne affecte la qualité externe qui affecte, à son tour, la qualité en utilisation du produit logiciel est effectivement justifiée dans ce travail de recherche.

Les autres contributions de cette recherche se résument comme suit :

- vérification jusqu'à quel point le questionnaire d'ISBSG tient compte des trois types de qualité définis dans la norme ISO 9126-1. Pour cela, nous avons aligné les différentes parties du questionnaire d'ISBSG par rapport à la qualité interne, la qualité externe et la qualité en utilisation de la norme ISO 9126-1;
- identification des caractéristiques de qualité interne, externe et en utilisation couvertes par le questionnaire d'ISBSG à travers l'identification des données de

qualité du questionnaire d'ISBSG et leurs mesures correspondantes dans les rapports techniques ISO TR 9126-2 à 4;

- analyse du référentiel de données d'ISBSG afin de déterminer les données de qualité disponibles pour évaluer les trois types de qualité du produit logiciel d'ISO 9126. Nous avons proposé des mesures propres à ISBSG à base de ces données. Ces mesures seront utiles aussi bien pour les chercheurs que pour les gens de l'industrie du logiciel;
- adaptation de la méthode Taguchi de conception de plans d'expériences, d'ordre industriel, au contexte d'analyse empirique en génie logiciel, laquelle adaptation a porté sur l'étape de paramètres de design de la stratégie hors production (off-line) de contrôle de la qualité de Taguchi.

Perspectives d'avenir

Des travaux entrepris dans cette thèse peuvent s'ensuivre plusieurs projets de recherche sur les différents axes de cette thèse : la méthode Taguchi, les mesures et les modèles de qualité d'ISO 9126 et le référentiel d'ISBSG non seulement dans le domaine de la qualité du logiciel, mais aussi dans les différents domaines de connaissances de la discipline du génie logiciel.

En particulier, notre approche d'adaptation de la méthode Taguchi au contexte des études empiriques pourrait servir de base pour d'autres travaux de recherche aussi bien empiriques qu'expérimentaux étant donné que la seule différence réside dans la collecte de données. En fait, cette approche fournit toutes les étapes nécessaires permettant de bien mener une étude empirique ou expérimentale en génie logiciel.

Les nouvelles voies de recherche que nous suggérons sont :

- en présence de bases de données plus complètes disposant d'un grand nombre de mesures de qualité du produit logiciel : mener d'autres études empiriques afin d'approfondir les résultats obtenus des plans d'analyses de cette recherche;
- en collaboration avec l'industrie : élaborer des études expérimentales permettant d'étudier d'autres caractéristiques (sous-caractéristiques) des modèles de qualité d'ISO 9126 dans la vérification des liens entre les trois types de qualité : interne, externe et en utilisation;
- en collaboration avec les experts de la série ISO 9126 dans sa nouvelle version : développer pour les modèles de qualité de la norme ISO 9126-1 des modèles de relations montrant les relations entre les différentes caractéristiques (sous-caractéristiques) des modèles de qualité d'ISO 9126;
- en collaboration avec l'organisation ISBSG : améliorer le questionnaire de collecte de données d'ISBSG afin qu'il prend en considération les différentes caractéristiques (sous-caractéristiques) des modèles de qualité d'ISO 9126;
- en utilisant la méthode Taguchi : utiliser d'une part toutes les étapes de la stratégie de contrôle de qualité hors production (off-line) de Taguchi et, d'autre part, la stratégie de contrôle de qualité en production (on-line) de Taguchi dans le processus de développement de logiciel de qualité.

Nous espérons que cette recherche et ces nouvelles voies de recherche une fois effectuées permettront de contribuer à améliorer la maturité du génie logiciel en général et ses standards en particulier, car l'expérimentation permet de valider le contenu des standards et d'appuyer leur crédibilité; cela encouragera les entreprises à les utiliser et à appliquer les standards du logiciel comme c'est le cas de beaucoup de standards du domaine de l'industrie, tels que : ISO 9000.

ANNEXE I

LISTE DES PUBLICATIONS

1. Cheikhi Laila, Alain Abran et Lopez Miguel. 2005. « Analysis of the Designs of Coupling Measures », 15th International Workshop on Software Measurement - IWSM'2005, p. 435-454. Aachen, Germany: Shaker Verlag.
2. Cheikhi, Laila, Alain Abran et Luigi Buglione. 2006. « ISBSG Software Project Repository & ISO 9126: An Opportunity for Quality Benchmarking ». In the European Journal for the Informatics Professional, vol. 7, n° 1, p. 46-52.
3. Cheikhi, Laila, Alain Abran et Luigi Buglione, 2006. « El repositorio de proyectos software ISBSG & ISO 9126: una oportunidad para medir la calidad ». In Novatica Journal , 2006 , pp. 41-47.
4. Cheikhi, Laila, Alain Abran et Witold Suryn. 2006. Harmonization of Usability Measurement in ISO Software Engineering Standards. In the IEEE International Symposium on Industrial Electronics (ISIE'06).
5. Cheikhi, Laila, Alain Abran et Luigi Buglione. 2007. « The ISBSG Software Project Repository - An Analysis from the ISO 9126 Quality Perspective ». ASQ Software Quality Professional Journal, vol. 9, n° 2, p. 4-24.
6. Abran Alain, J. Garbajosa et Laila Cheikhi. 2007. « Estimating the Test Volume and Effort for Testing and V&V ». International Workshop in Software Measurement - International Conference on Software Process and Product Measurement (IWSM-Mensura).

ANNEXE II

MODÈLES DE QUALITÉ DU PRODUIT LOGICIEL

Tableau II.1

Modèle de qualité de (McCall, 1977)

Use	Quality Factors	Quality Criteria
Product revision	Maintainability	Simplicity, Conciseness, Self-descriptiveness, Modularity, Consistency
	Flexibility	Self-descriptiveness, Expandability, Generality, Modularity
	Testability	Simplicity, Instrumentation, Self-descriptiveness, Modularity
Product operations	Correctness	Traceability, Completeness, Consistency
	Efficiency	Execution efficiency, Storage efficiency
	Reliability	Consistency, Accuracy, Error tolerance, Simplicity
	Integrity	Access control, Access audit
	Usability	Operability, Training, Communicativeness
Product transition	Portability	Self-descriptiveness, Software system independence, Machine independence, Modularity
	Reusability	Self-descriptiveness, Generality, Modularity, Software system independence, Machine independence
	Interoperability	Modularity, Communication commonality, Data commonality

Tableau II.2

Modèle de qualité de (Boehm, 1976)

Primary uses	**Intermediate constructs**	**Primitive constructs**
As-is Utility	Reliability	Self-Containedness, Accuracy, Completeness, Robustness/integrity, Consistency
	Efficiency	Accountability, Device Efficiency, Accessibility
	Human Engineering	Robustness/integrity,Accessibility, Communicativeness
Portability		Device-Independence, Self-Containedness
Maintainability	Testability	Accountability, Accessibility , Structuredness, Communicativeness, Self-Descriptiveness,
	Understandability	Consistency, Self-Descriptiveness, Structuredness, Conciseness, Legibility
	Modifiability	Structuredness, Augmentability

Tableau II.3

Modèle de qualité de (Dromey, 1996)

Software product	**Product properties**	**Quality attributes**
Implementation	Correctness	Functionality, Reliability
	Internal	Efficiency, Maintainability, Reliability
	Contextual	Reusability, Portability, Maintainability, Reliability
	Descriptive	Usability, Maintainability, Portability, Reusability

ANNEXE III

STRUCTURE DÉTAILLÉE DE L'EXTRAIT DE DONNÉES D'ISBSG-RELEASE 9 DE 2005

Categories	Fields
Project ID	- Project ID
Rating (2)	- Data Quality Rating - Unadjusted Function Point Rating
Sizing (4)	- Count approach - Functional Size - Adjusted function points - Value Adjustment factor
Effort (2)	- Summary work effort - Normalized work effort
Productivity (4)	- Reported delivery rate (afp) - Project delivery rate (ufp) - Normalized productivity delivery rate (afp) - Normalized productivity delivery rate (ufp)
Schedule (11)	- Project elapsed time - Project inactive time - Implementation date - Project activity scope - Effort plan - Effort specify - Effort design - Effort build - Effort test - Effort implement - Effort unphased
Quality (4)	- Minor defects: Defects reported in the first month of use of the software: The number of Minor defects delivered. - Major defects: Defects reported in the first month of use of the software: The number of Major defects delivered. - Extreme defects: Defects reported in the first month of use of the software: The number of Extreme defects delivered. - Total defects delivered: Total defects reported

Categories	Fields
	in the first month of use of the software: The Total number of defects (minor, major and extreme), or where no breakdown is available, the single value is shown here.
Grouping Attributes (6)	- Development type - Organization type - Business Area type - Application type - Package customization - Degree of customization
Architecture (7)	- Architecture - Client server - Client roles - Server roles - Type of server - Client/server description - Web development
Documents &Techniques (16)	- Planning documents - Specification documents - Specification techniques - Design documents - Design techniques - Build products - Build activity - Test documents - Test activity - Implementation documents - Implementation activity - Development techniques - Functional sizing technique - FP standard - FP standards all - Reference table approach
Project Attributes (23)	- Development platform - Language type - Primary programming language - 1st Hardware - 2nd Hardware - 1st Operating system - 2nd Operating system - 1st Language - 2nd Language

Categories	Fields
	- 1st Data base system - 2nd Data base system - 1st Component server - 2nd Component server - 1st Web server - 2nd Web server - 1st Message server - 2nd Message server - 1st Debugging tool - 1st Other platform - 2nd Other platform - CASE tool used - Used methodology - How methodology acquired
Product Attributes (4)	- User base: business units - User base: locations - User base: concurrent users - Intended Market
Effort Attributes (6)	- Recording method - Resource level - Maximum team size - Average team size - Ratio of project work effort: non-project Effort - Percentage of uncollected work effort
Size Attributes (8)	- Input count - Output count - Enquiry count - File count - Interface count - Added count - Changed count - Deleted count
Size other than FSM (2)	- Lines of code - LOC not Statements

ANNEXE IV

ANALYSE DE L'EXTRAIT DE DONNÉES D'ISBSG-RELEASE 9 DE 2005

This annexe presents an analysis of the quality-related data fields in the ISBSG MS-Excel data extract (Release 9 of 2005) available to practitioners and researchers.

1. First level of data preparation

As recommended in (Déry et Abran, 2005), two verification steps must be carried out prior to using data for data analysis: data quality verification and data completeness verification. The first step is carried out by the ISBSG repository manager, who analyzes the data collected from the questionnaires and then rates the project data collected. This rating information is recorded in a data field: the **Data Quality Rating (DQR)**. The admissible values for this data field (ISBSG, 2005) are:

- "**A** = The data submitted was assessed as being sound with nothing being identified that might affect its integrity.
- **B** = The submission appears fundamentally sound but there are some factors which could affect the integrity of the submitted data.
- **C** = Due to significant data not being provided, it was not possible to assess the integrity of the submitted data.
- **D** = Due to one factor or a combination of factors, little credibility should be given to the submitted data."

It is thus advisable for analysis purposes to consider only those projects having a DQR equal to A or B (e.g. the data collected have a high degree of integrity). The number of projects, with their corresponding data quality rating, is presented in Table IV.1. The 232 projects with a C or D quality rating will be dropped from further analysis, leaving a sample of **2792** projects with an A or B data quality rating.

Table IV.1

Project Classification by DQR (Source: ISBSG R9)

Data Quality Rating	No. of projects	Percentage (%)
A	734	24.27
B	2058	68.06
C	128	4.23
D	104	3.44
Total	**3024**	**100**

2. Second level of data preparation

Another step is required in data preparation: in the ISBSG repository, the quality-related data fields are not mandatory, which means, for instance, that a number of projects might not have any data about defects. Table IV.2 presents the number of projects with, or without, information about defects for a period of one month after the date on which the software began operation, grouped by:

- Minor Defects → Software usable.
- Major Defects → Part of the software unusable.
- Extreme Defects → All the software unusable.

The columns in Table IV.2 correspond to:

- (1) the number of projects without any information (blank fields);
- (2) the number of projects with information (non blank values or with the word '*some*' instead of a numerical value);
- (3) the number of projects with zero defects – as a subgroup of (2);
- (4) the maximum number of defects registered in the MS-Excel data extract for a defect severity type.

Table IV.2

Number of Projects (A/B) by Defect Severity type (ISBSG R9)

Quality	Blanks (1)	NonBlanks (2)	0 defects (3)	Max (4)
Minor defects	2485	307	116	682
Major defects	2403	389	177	725
Extreme defects	2556	236	184	1272
Total defects	2270	522	158	2554

In particular, 2270 projects had blanks in all the defect severity-type fields; these projects are also dropped from further analysis. This leaves only **522** projects with some quality-related information available for further analysis.

Furthermore, it was observed that, of these 522 projects, 103 had a zero value in all three types of defect fields (e.g. minor = 0, major = 0 and extreme = 0), that is, about 20%. This might be real information, but the zero value might also be caused by poor data entry, and some organizations might have entered a zero value instead of leaving the field blank for a missing value. To be on the safe side for this analysis, we elected to drop these 103 projects from further analysis. Moreover, 55 additional projects were dropped for the same reason; they had only zero and/or a blank in the three types of defect fields (for example, minor = blank, major = 0 and extreme = blank). Finally, three projects did not have numerical values in the defect fields; the word 'some' or a '?' was inserted for the number of defects; these three projects were also dropped. This left **361** projects available for quality-related analysis.

3. Analysis of the distribution of defects by projects

For these 361 projects, with quality-related information about defects during the first month of the software's operation, a detailed distribution by defect type is graphically shown in Figures IV.1 to IV.4:

- Minor defects: 200 projects with a minimum of 0 defects and a maximum of 682 (and 161 with blanks) – Figure IV.1;
- Major defects: 252 projects with a minimum of 0 defects and a maximum of 725 (and 109 with blanks) – Figure IV.2;
- Extreme defects: 130 projects with a minimum of 0 defects and a maximum of 1272 (and 231 with blanks) – Figure IV.3;
- Total defects: all 361 projects with a minimum of 1 defect and a maximum of 2554 – Figure IV.4.

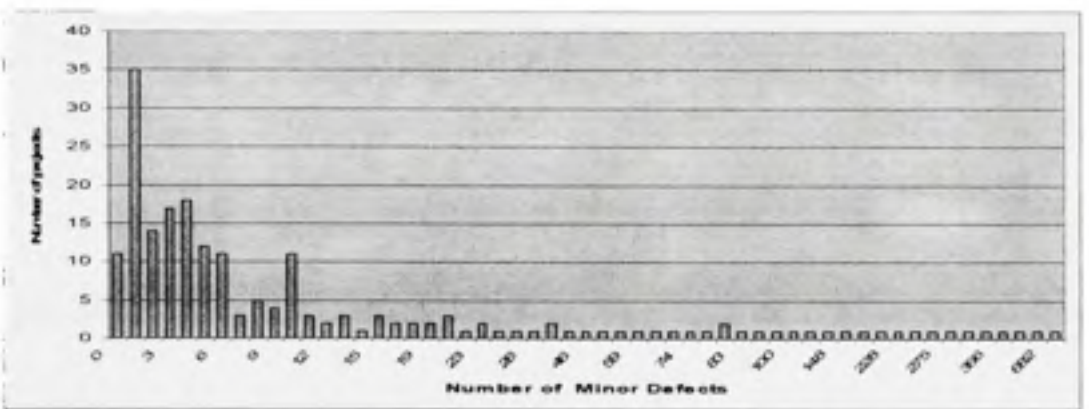

Figure IV.1 ***Project classification by Number of Minor Defects (200 projects).***

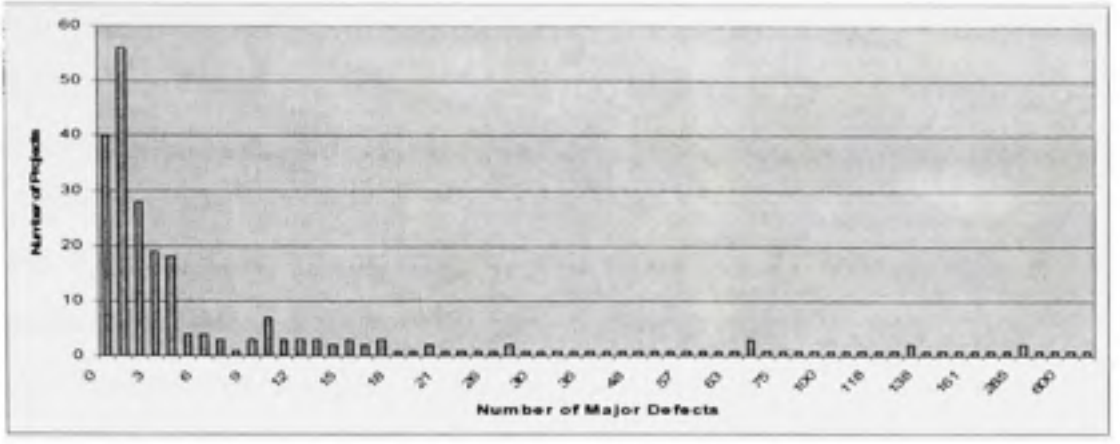

Figure IV.2 ***Project classification by Number of Major Defects (252 projects).***

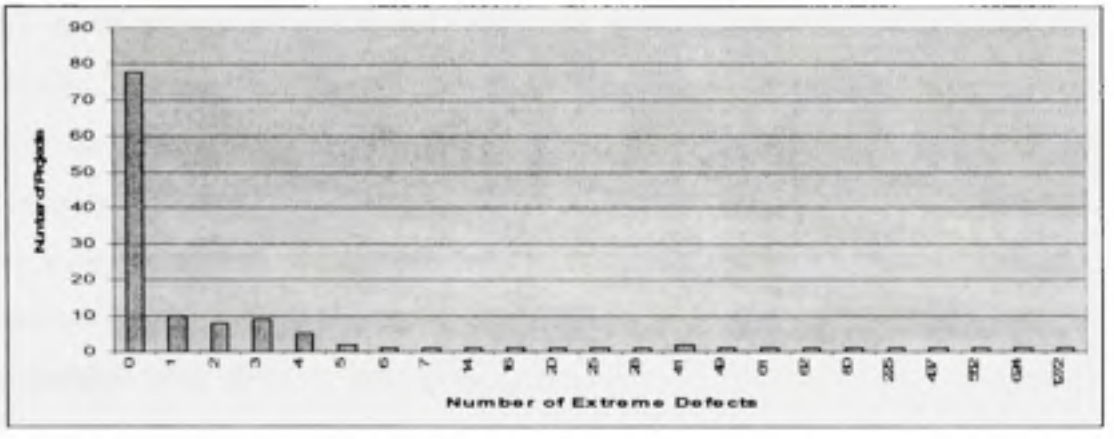

Figure IV.3 ***Project classification by Number of Extreme Defects (130 projects).***

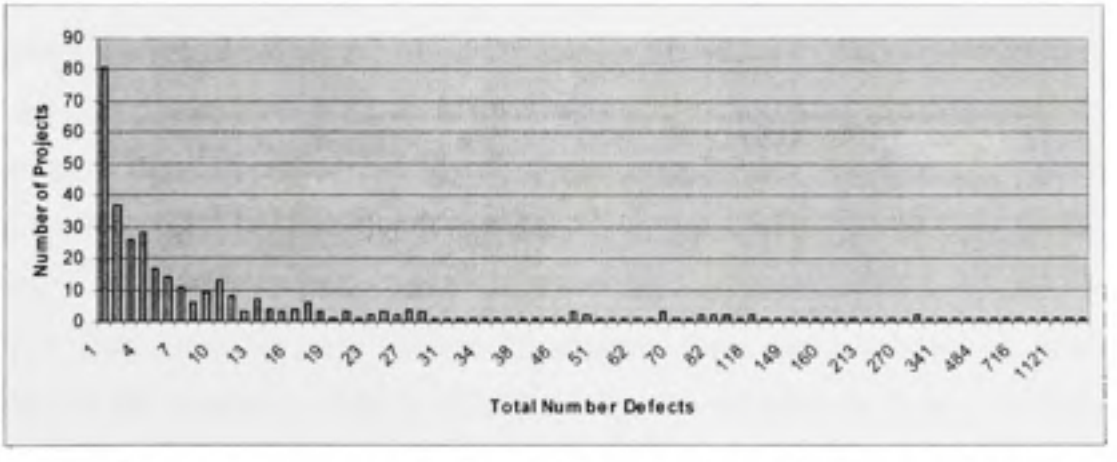

Figure IV.4 ***Project classification by Total Number of Defects (361 projects).***

From Figures IV.1 to IV.4, it appears clearly that there is a wide variation in defects (minor, major, extreme and total) per project during the software's first month of use.

- For Minor defects, 71% of projects have fewer than 11 defects, while 29% of projects have between 11 and 682 defects (Figure IV.1);
- For Major defects, 73% of projects have fewer than 11 defects, whereas 27% of projects have between 11 and 725 defects (Figure IV.2);
- For Extreme defects, 86% of projects have fewer than 6 defects, while 14% of projects have between 6 and 1272 defects (Figure IV.3);
- For Total defects, 67% of projects have fewer than 11 defects, whereas 33% of projects have between 11 and 2554 defects (Figure IV.4).

What can be noted here is that the percentage of projects having fewer than 11 defects (all types) is higher than that for the others. This can be an indicator of good software quality once the software has been delivered to its users. However, it does not provide information about whether or not the development process did indeed produce *in process* quality directly, or whether there was significant rework required to correct the in-process quality issues. For those interested in in-process quality, they could repeat this analysis with defects data from specification to implementation or installation phases.

4. Analysis of defect density

In the previous section, the analysis of the number of defects for the projects did not take into account project size. With the ISBSG MS-Excel data extract, it is possible to have information about the functional size of the projects. In fact, the data size is retrieved from the field Adjusted Function Points (AFP), which is based on different approaches to sizing the ISBSG projects; those taken for this analysis are NESMA, IFPUG and Mark-II. Here again, for some of these 361 projects, there was no information as to data size and/or the number of defects (blanks), so they are excluded from the distribution of defect density; that is, the number of projects is different:

- Minor defects: 190 projects with a minimum of 0 defects and a maximum of 682;
- Major defects: 237 projects with a minimum of 0 defects and a maximum of 725;
- Extreme defects: 124 projects with a minimum of 0 defects and a maximum of 1272;
- Total defects: 339 projects with a minimum of 1 defect and a maximum of 2554.

This left 339 projects (reduced from 361) available for analyzing defect density.

4.1. Total Defect density – Sample of 339 projects

One of the ways to judge the quality of a project is to measure its defect density – the ratio of the number of defects to the size of the project - expressed in Number of Defects per AFP. Therefore, to obtain a general view of the quality of the ISBSG projects, Figure IV.5 presents the defect density of the Total Number of Defects (minor, major, extreme or total) for the 339 projects.

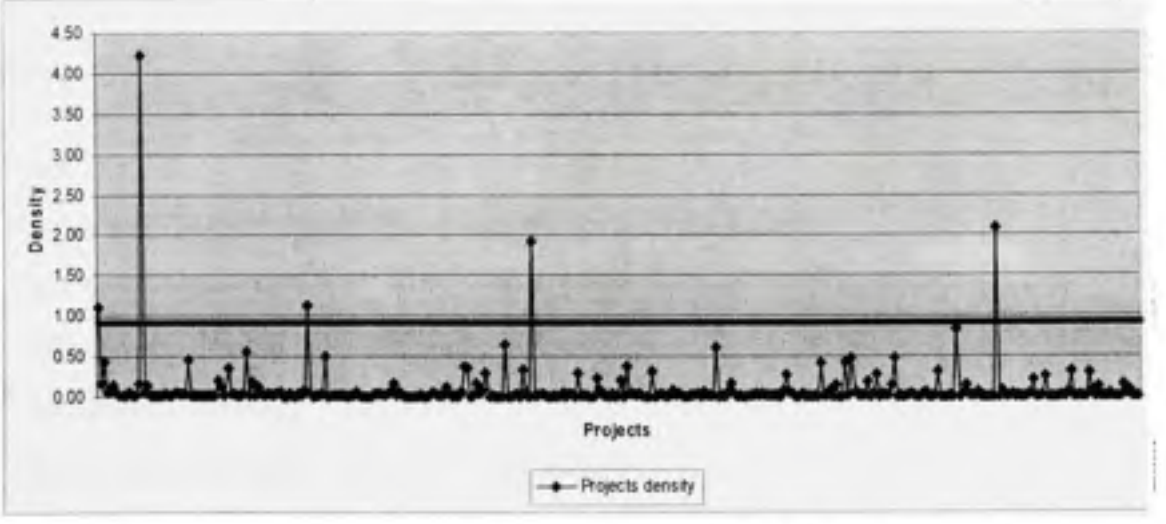

Figure IV.5 ***Defect density for Total Number of Defects (339 projects).***

An assumption has been made here, which is that projects with a defect density close to 0 are of good quality, those between 0.01<=defect density <= 0.09 are of medium quality and those > = 0.1 are of poor quality, when using this assessment scale[10]. The data about defect density reveals that 15% of the projects are of good quality, 70% are of medium quality and 15% are of poor quality (Figure IV.6).

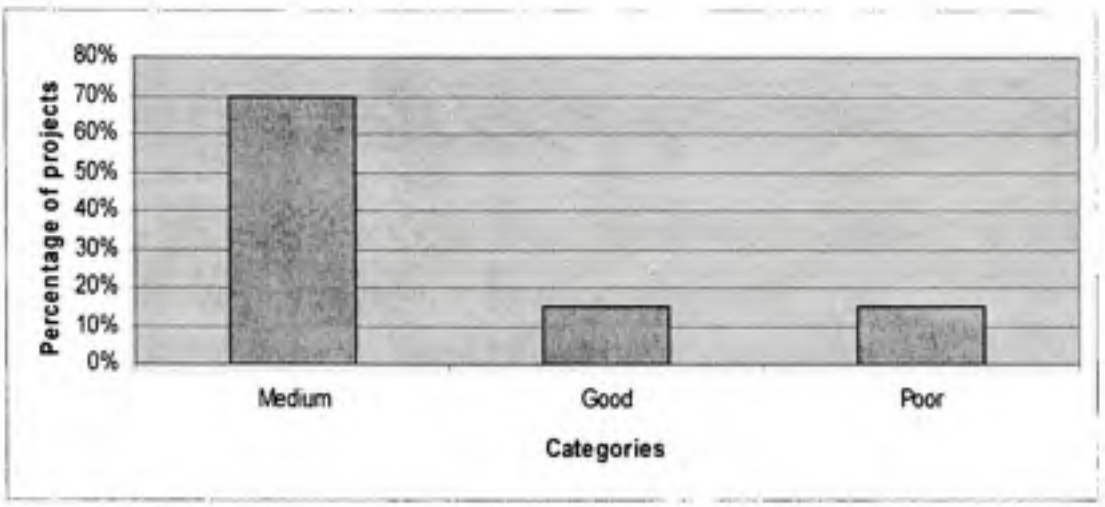

Figure IV.6 ***Distribution of poor, medium and good quality projects (339 projects).***

Moreover, a set of 5 projects has been identified graphically from Figure IV.5 with a defect density higher than 1 (above the red line), and they are:

- the project with size = 38 AFP, total defects = 161 and defect density = 4.24;
- the project with size = 1225 AFP, total defects = 2554 and defect density = 2.08;
- the project with size = 250 AFP, total defects = 484 and defect density = 1.94;
- the project with size = 116 AFP, total defects = 130 and defect density = 1.12;
- the project with size = 9 AFP, total defects = 10 and defect density = 1.11.

[10] Of course, organizations must define assessment scales that meet their own quality requirements.

Another assumption has been made, again about the 5 projects with a defect density higher than 1: they could be considered to be of the worst quality, unless the defects are minor, that is, they do not make the software unusable in any way. Since, for all five projects, the defects are not all minor, the quality of these projects is therefore considered to be the worst of this data set.

4.2. Defect density – projects of size over 1000 AFP

In this sample of 339 projects, there are 278 with fewer than 1000 AFP and 61 with 1000 or more AFP. The defect density of these 61 larger projects is presented next, in Figures IV.7 to IV.10, to show the distribution of projects using size data and number of defects respectively:

- Minor defects (Figure IV.7): 43 projects with a minimum of 10 defects and a maximum of 682;
- Major defects (Figure IV.8): 52 projects with a minimum of 0 defects and a maximum of 725;
- Extreme defects (Figure IV.9): 31 projects with a minimum of 0 defects and a maximum of 1272;
- Total defects (Figure IV.10): 61 projects with a minimum of 1 defect and a maximum of 2554.

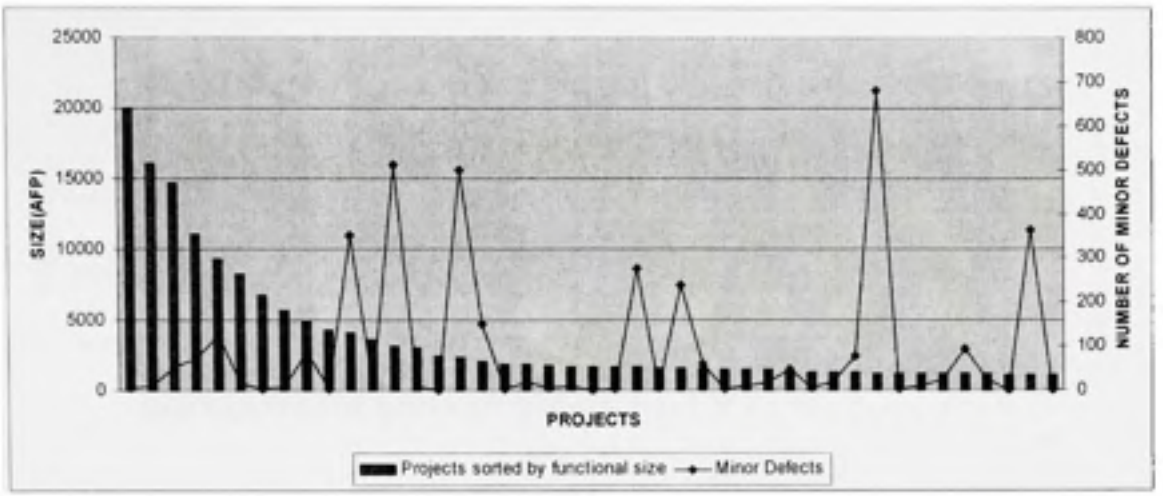

Figure IV.7 ***Number of Minor Defects and Size per Project (43 projects).***

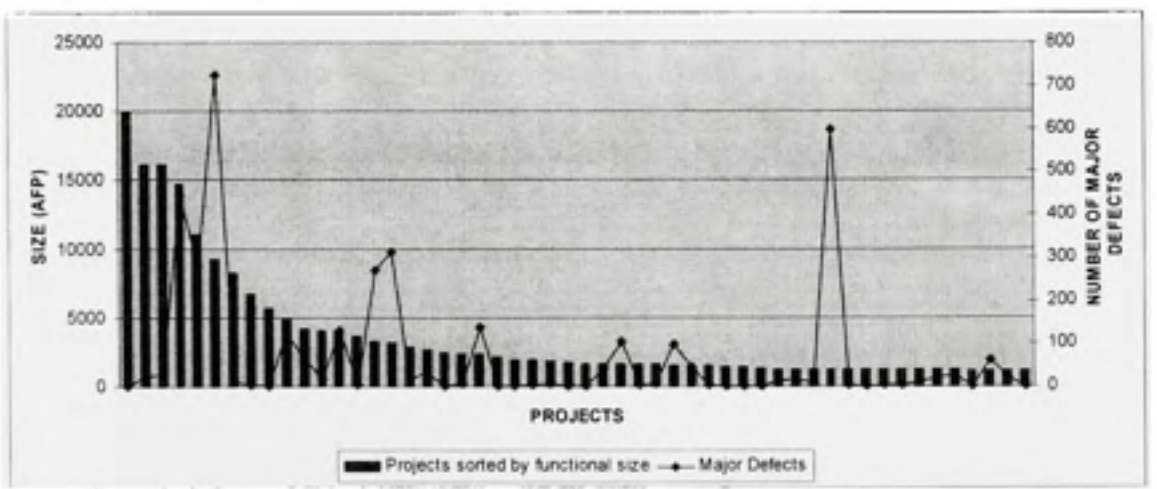

Figure IV.8 ***Number of Major Defects and Size per Project (52 projects).***

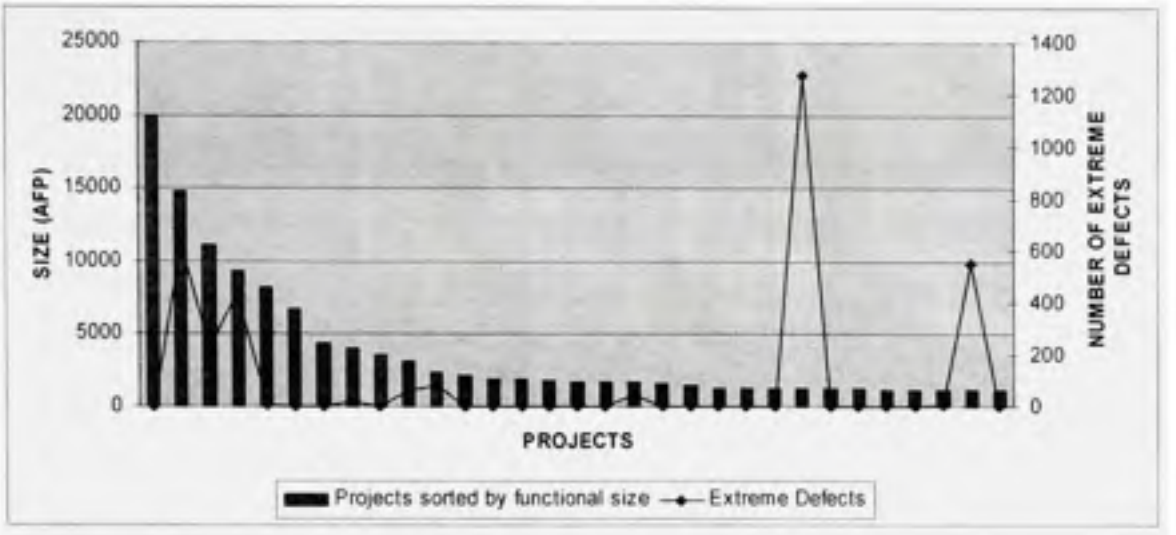

Figure IV.9 ***Number of Extreme Defects and Size per Project (31 projects).***

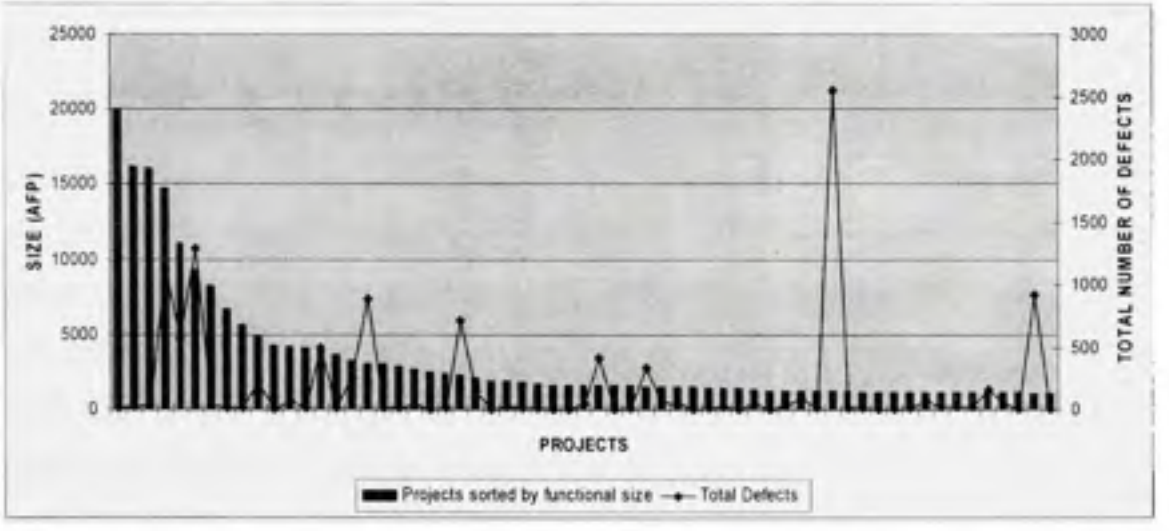

Figure IV.10 ***Total Number of Defects and Size per Project (61 projects).***

By taking into account project size, it can be observed from Figures IV.7 to IV.10 that: 10 projects out of 43 have a higher number of minor defects relative to their size (Figure IV.7), 8 projects out of 52 have a higher number of major defects relative to their size (Figure IV.8), 2 projects out of 31 have a higher number of extreme defects relative to their size (Figure IV.9) and 7 projects out of 61 have a higher total number of defects and are small (Figure IV.10). See Practitioners Guidance box.

GUIDANCE FOR PRACTITONERS
Such information can be used to develop a Quality Improvement Initiative as follows: Let's assume these numbers represent actual projects within a single organization, such as "your own" development organization. From Figures 13 to 16, a software development organization would then immediately ask the question: What do I need to do to improve my development process so that I can reduce or eliminate projects exhibiting a higher number of defects, detected during the first month of the software's operation, relative to their size? The suggested steps can be: - Select these 27 projects cases identified in Figures 13 to 16 with high defects ratios relative to their size. - Root Cause Analysis to identify why these cases occurred. - Analysis of defects by each phase of the software life cycle.

It can also be observed that the largest project (20000 AFP) has only a few defects of all types (minor=6, major=1, extreme=3, total=10), while the project with the highest number of defects of all types (minor=682, major=600, extreme=1272, total=2554) is much smaller (1225 AFP). In such cases, one can postulate that the project with the higher size is well controlled (or that the defects are not adequately collected and therefore the number of defects detected is small[11]) while the project with the smaller size and the higher number of defects is not well controlled during the whole software life cycle.

[11] Such very large projects submitted to ISBSG with a small number of defects should be cross-checked by the ISBSG Repository Manager Report.

ANNEXE V

MESURES DE QUALITÉ DE PRODUIT LOGICIEL PROPRES À ISBSG

Part V.I: Internal quality measurements

Table V.I.1

Characteristic: Functionality – Subcharacteristic: Suitability

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Specification Changes Density (SCD-CDL)	How stable is the specification during the development life cycle?	SS= A/B With A= Ad+Ab+At+Ai	Specification changes/Cfsu		The smaller the better and the software is more stable
		A=Number of specification changes raised during CVL.	Specification changes		
		Ad=Number of specification changes raised during Design	Specification changes	Count the number of specification changes raised during -Design (Qn:30)	
		Ab=Number of specification changes raised during Build	Specification changes	Count the number of specification changes raised during Build (Qn:36)	
		At=Number of specification changes raised during Test	Specificationc hanges	Count the number of specification changes raised during Test (Qn:41)	
		Ai=Number of specification changes raised during Implementation	Specificationc hanges	Count the number of specification changes raised during Implementation (Qn: 47)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Specification Changes Density (SCD)	How many specification changes raised during a defined period activity?	SCD= A/B	Specification changes/Cfsu		-The higher is the better in the first software development phases. -When comparing projects, the smaller is the better
		A=Number of specification changes raised during Design/Build/Test/Implementation	Specification changes	Count the number of specification changes raised during: Design (Qn:30)/Build (Qn:36)/Test (Qn:41)/Implementation (Qn: 47)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	
Note: Specification Changes Density measure can be applied to each phase taken alone (design, build, test, and implementation) or combined.					

Table V.I.2

Characteristic: Reliability - Subcharacteristic: Maturity

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Defect Density (DD)	How many defects were detected in reviewed product?	DD= A/B	Defect/Cfsu		-The higher is the better in the first software development phases. -When comparing two or more projects, the smaller is the better
		A= Number of defect detected during Specification/ Design/ Build	Defect	Count the number of defects detected during Specification (Qn:24)/ Design (Qn:29)/Build (Qn:35)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	
Note: Defect density measure can be applied to each phase taken alone (specification, design, build (code review/inspection)) or combined					

Table V.I.3

Characteristic: Maintainability – Subcharacteristic: Changeability

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Resolution/ Rework Effort (RE)	Can the maintainer easily change the software to resolve defects detected during defined period activity?	RE= A/B	Hour/defect		The smaller the better
		A=Number of hours taken to correct detected defect during Specification/ Design/Build	Hour	Count number of hours taken to correct detected defect during Specification(24) /Design(Qn:29)/ Build (Qn:35)	
		B= Number of detected defects during Specification/ Design/Build	Defect	Count number of detected defect during Specification(24) /Design(Qn:29)/ Build (Qn:35)	
Note: Resolution/Rework Effort measure can be applied to each phase taken alone (specification, design, build (code review/inspection)) or combined.					

Part V.II: External quality measurements

Table V.II.1

Characteristic: Functionality – Subcharacteristic: Suitability

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Changed Functionality Density (CFD-Global)	How stable is the functionality after entering operation ?	SF= A/B A=Aa+Ac+Ad	Cfsu/Cfsu		The smaller the better and the software is more stable
		A= Software enhancement functional size	Cfsu	A= Functional size of enhancement (added, changed, deleted functionality) (Qn: 104)	
		Aa= Functional size of added functionality	Cfsu	Functional size of added functionality during enhancement (Qn:101)	
		Ac= Functional size of changed functionality	Cfsu	Functional size of changed functionality during enhancement (Qn:102)	
		Ad= Functional size of deleted functionality	Cfsu	Functional size of deleted functionality during enhancement (Qn:103)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	
Name of the measure	**Purpose of the measure**	**Numerical assignments**	**Unit of measurement**	**Measurement method**	**Interpretation of measured value**
Changed Functionality Density (CFD)	How many functionality were changed (added, changed, deleted) after the software operation ?	CFD= A/B	Cfsu/Cfsu		-The smaller the better. -It depends on whether it's added, changed or deleted functionality.
		A= Functional size of changed functionality after operation during enhancement	Cfsu	Functional size of changed functionality during enhancement: added functionality (Qn:101)/ changed functionality (Qn:102)/ deleted functionality (Qn:103)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	
Note: Changed functionality density measure can be applied to each type (added, changed, deleted) alone.					

Table V.II.2

Characteristic: Reliability - Subcharacteristic: Maturity

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Defect Density (DD)	How many defects were detected durin g defined trial period?	DD= A/B	Defect/Cfsu		-The smaller the better at the end stage of software development phases. -When comparing two or more projects, the smaller is the better
		A= Number of defect detected during Build/Test/ Implementatio n /Operation	Defect	Count the number of defects detected during Build (Qn:35)/ Test (Qn:40)/ Implementation (Qn:46)/ Operation (Qn: 126)	
		B= Functional size of the software product	Cfsu	Functional size of the software product using COSMIC-FFP (Qn:95)	
Note: Defect density measure can be applied to each phase taken alone (build (unit testing), test, implementation, operation) or combined.					

Table V.II.3

Characteristic: Maintainability - Subcharacteristic: Changeability

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
Resolution/ Rework Effort (RE)	Can the maintainer easily change the software to resolve defects detected during defined period activity?	RE= A/B	Hour/defect		The smaller the better
		A=Number of hours taken to correct detected defect during Build/Test/ Implementation	Hour	Count number of hours taken to correct detected defect during Build (Qn:35)/ Test (Qn:40)/ Implementation (Qn:46)	
		B= Number of detected defects during Build/Test/ Implementation	Defect	Count number of detected defect during Build (Qn:35)/ Test (Qn:40)/ Implementation (Qn:46)	
Note: Resolution/Rework Effort measure can be applied to each phase taken alone (build (unit testing), test, implementation) or combined.					

Part V.III: Quality in use measurements

Table V.III.1

Characteristic: Satisfaction

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
User Satisfaction Degree (USD-Global)	How satisfied is the user?	TUSQ = MSO+ MBR+ QF+ QD+ EU+ TG+SDS+ SPS			The higher the better
		MSO= Meet Stated Objectives	Likert Scale	Response to the question: Did the project meet stated objectives? (Qn:128)	
		MBR=Meet Business Requirements	Likert Scale	Response to the question : Did the project meet business requirements (Qn:128)	
		OF= Quality of Functionality	Likert Scale	Response to the question: Quality expectations for the software? (Qn:128)	
		OD= Quality of Documentation	Likert Scale	Response to the question: Quality expectations for user documentation (Qn:128)	
		EU= Ease-of-Use	Likert Scale	Response to the question: Ease of use requirements for the software? (Qn:128)	
		TG =Training Given	Likert Scale	Response to the question: Was sufficient training or explanation given? (Qn:128)	
		SDS= Speed of Defining Solution	Likert Scale	Response to the question: Schedule for planning & specification? (Qn:128)	
		SPS= Schedule of Providing Solution	Likert Scale	Response to the question: Schedule for design, build, test & implement? (Qn:128)	
Note: The satisfaction of the user about two or n-features can be done by summing the results of the two or n-features					

Name of the measure	Purpose of the measure	Numerical assignments	Unit of measurement	Measurement method	Interpretation of measured value
User Satisfaction Degree (USD)	How satisfied is the user with specific software features?	USQ= A A represent one of the 8 questions of the ISBSG user satisfaction survey	Likert Scale	Response to question related to feature (Qn:128) : -Meet stated objectives / -Meet business requirements/ -Quality expectations for the software / -Quality expectations for user documentation / -Ease-of- use requirements for the software -Training or explanation given / -Schedule for planning & specification / -Schedule for design, build, test & implement	The higher rating scale the better

Notes:

- The number (x) of the question (Qn) in the ISBSG questionnaire is represented by (Qn: x).
- The COSMIC-FFP is taken as example of functional size measurement. The others IFPUG, Mark-II and NESMA are also applicable.

ANNEXE VI

QUESTIONNAIRE SUR LA QUALITÉ DU PRODUIT LOGICIEL

Ce document se veut une présentation d'un ensemble de questions relatives à la qualité du produit logiciel, en particulier les différents facteurs influençant la qualité du produit logiciel durant son cycle de vie. Ces questions sont groupées par des catégories clés, à savoir la technologie, le type de développement, le nombre de défauts, le nombre de changements.

Veuillez répondre à ces questions en se basant sur vos connaissances pratiques et vos expériences dans le domaine du logiciel.

Dans un premier lieu, nous avons dressé un ensemble de questions relatives à votre profil actuel au sein de votre organisation et vos expériences précédentes dans le domaine du génie logiciel. Ces informations restent confidentielles et ne seront utilisées que pour des fins d'analyses, ou si nous avons besoin de plus d'éclaircissements sur les réponses fournies.

Vu l'importance de ces informations pour notre recherche, veuillez vous assurer d'avoir bien compris la question et par la suite y répondre clairement. Si vous avez des précisions ou des commentaires sur des questions les inscrire aussi.

Nous vous remercions d'avance de votre collaboration.

Laila Cheikhi

Projet de recherche sous la supervision du Dr. Alain Abran
Directeur du laboratoire de recherche en génie logiciel – GÉLOG
École de Technologie Supérieure
Laboratoire du Recherche en Génie Logiciel

Questions relatives au Profil Personnel

1. Quel poste occupez-vous actuellement au sein de votre organisation ?

   ☐ Chef du projet ☐ IT opérations

   ☐ Membre d'une équipe de développement ☐ Gestionnaire

   Autres :.....

2. En quoi consiste votre tâche de travail durant le cycle de développement du logiciel ?

   ☐ Planification du logiciel ☐ Test

   ☐ Spécification du logiciel ☐ Implémentation ou Installation

   ☐ Conception du logiciel ☐ Gestion du projet

   ☐ Programmation et Test unitaire ☐ Maintenance du logiciel

   Autres :.....

3. Quelle est votre expérience dans le domaine du logiciel ?

   Dans votre établissement de travail actuel (en année).....: ☐

   En général (en année).....: ☐

   Si vous êtes un chef de projet, combien de projets vous avez géré ?..: ☐

4. Utilisez-vous des processus ou standards de qualité de logiciel au sien de votre établissement de travail ?

   ☐ ISO 9126

   ☐ CMM

   ☐ Séries ISO 9000

   Autres :.....

Instructions sur les modalités de réponses

1. Chaque ligne correspond à un facteur ou paramètre relatif aux différentes informations sur le projet logiciel.
   - Facteurs contrôlés : facteurs ayant un grand impact sur la qualité du produit logiciel. Cet impact se manifeste positivement ou négativement selon leurs présences ou absences.
   - Facteurs bruits (facteurs non contrôlés) : facteurs influençant la qualité du produit logiciel et ils sont difficilement contrôlables ou incontrôlables.

2. Trois types de qualité du produit logiciel sont identifiés durant son cycle de vie :
   - Qualité interne (QI) : correspond à la qualité du produit logiciel durant les phases de planification, de spécification, de design et de codage du produit logiciel, c'est-à-dire la qualité du produit logiciel à base de ses propriétés intrinsèques et en dehors de toute exécution.
   - Qualité externe (QE) : correspond à la qualité du produit logiciel durant les phases de test et d'opération du produit logiciel, c'est-à-dire la qualité du produit logiciel à base de son comportement lors de son exécution.
   - Qualité à l'utilisation (QIU) : correspond à la qualité du produit logiciel de point de vue utilisateur final, c'est-à-dire le degré de satisfaction des besoins de l'utilisateur.

3. L'objectif est de savoir si ces facteurs ont un impact sur l'une ou les deux types de qualité suivantes :
   - la qualité externe du produit logiciel, c'est-à-dire lors de son exécution. En particulier, l'impact des ces facteurs sur le nombre de défauts durant l'exécution du produit logiciel.
   - la qualité de l'utilisation, c'est-à-dire le degré de satisfaction de l'utilisateur du produit logiciel livré.

4. Pour chaque facteur, deux zones sont fournies, une pour la qualité externe et l'autre pour la qualité à l'utilisation et à l'intérieur de chaque zone trois réponses sont fournies : Oui, Non et Ne sait pas. Une seule réponse doit être choisie. Par exemple :
   - Inscrivez « Oui » lorsque vous êtes certain que le facteur a effectivement un impact sur le nombre de défauts lors de l'exécution du logiciel. Si c'est le cas indiquez le degré d'impact : Grand ou Faible.
   - Inscrivez « Non » lorsque vous êtes certain que le facteur n'a d'aucune manière un impact sur le nombre de défauts lors de l'exécution du logiciel.
   - Inscrivez « Ne sait pas » lorsque vous ne savez pas quoi répondre.

5. Pour toute remarque ou information supplémentaire, veuillez l'écrire dans la section commentaire relative au facteur concerné.
6. Il est important que vos réponses soient à base de vos expériences et vos pratiques dans le domaine du génie logiciel.

Questionnaire sur la qualité du produit logiciel

Objectif : Indiquez si ces facteurs ont-ils un impact sur :

- le nombre de défauts lors de l'exécution du logiciel
  et/ou
- le degré de satisfaction des utilisateurs

1) Taille du projet (Exprimée en LOC ou Points de fonctions – IFPUG ou COSMIC)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

2) Effort dépensé pour le développement du projet

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

3) Type de développement (Nouveau développement, Redéveloppement, Amélioration)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

4) Personnalisation (de certain package lors du développement)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

5) Architecture du projet (Stand alone, Client service, Multi Tier,..)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6) Technologie utilisée

6.1) Plateforme de développement (PC, Main Frame, Multi Platform, etc.)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.2) Type de langage de programmation (3GL, 4GL, etc.)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.3) Langage de programmation (Java, SQL, C++, etc.)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.4) Système de Base de données

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.5) Utilisation des Outils de débogage

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.6) Utilisation des Outils CASE

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.7) Utilisation d'une méthodologie développement

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.8) Catégorie d'acquisition de la méthodologie (Purchased, in-house, combined)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

6.9) Utilisation d'un processus standard (CMM, CMMI, ISO, etc.)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

7) Changements de spécifications

7.1) Nombre de changements de spécifications de la phase : Design

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

7.2) Nombre de changements de spécifications de la phase : Codage

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

7.3) Nombre de changements de spécifications de la phase : Test

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

7.4) Nombre de changements de spécifications de la phase : Implémentation (ou installation)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

8) Défauts détectés

8.1) Nombre de défauts détectés durant la phase : Spécification

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

8.2) Nombre de défauts détectés durant la phase : Design

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

8.3) Nombre de défauts détectés durant la phase : Codage

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

8.4) Nombre de défauts détectés durant la phase : Test

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

8.5) Nombre de défauts détectés durant la phase : Implémentation

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

9) Effort de résolution ou de la refaite des défauts (Rework effort) (en heure)

9.1) Effort de résolution ou de la refaite des défauts détectés lors de la phase de : Spécification

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

9.2) Effort de résolution ou de la refaite des défauts détectés lors de la phase de : Design

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

9.3) Effort de résolution ou de la refaite des défauts détectés lors de la phase de : Codage

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

9.4) Effort de résolution ou de la refaite des défauts détectés lors de la phase de : Test

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

9.5) Effort de résolution ou de la refaite des défauts détectés lors de la phase de : Implémentation

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

10) Expériences

10.1) Expérience des membres de l'équipe dans le « end users'area of business »

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

10.2) Expérience d'IT opérations

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

10.3) Expérience des membres de l'équipe dans le développement du logiciel

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

10.4) Expérience du chef de projet

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

10.5) Changement de membres de l'équipe

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

11) Versions du projet

11.1) Nombre de nouvelle version (nouvelles fonctionnalités)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

11.2) Nombre de version de réparation de défauts

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

12) Taille fonctionnelle des améliorations

12.1) Nombre de nouvelles fonctions ou de fonctions ajoutées (exprimé en Unadjusted Function points ou en Cosmic Fnction Points)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

12.2) Nombre de fonctions modifiées (exprimé en Unadjusted Function points ou en Cosmic Fnction Points)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

12.3) Nombre de fonctions supprimées (exprimé en Unadjusted Function points ou en Cosmic Fnction Points)

	Impact sur Nombre défauts lors de l'exécution du logiciel				Impact sur Degré de satisfaction des utilisateurs			
	Oui		Non	Ne sait pas	Oui		Non	Ne sait pas
Degré impact	Grand	Faible			Grand	Faible		
Commentaires								

Merci de votre collaboration

Analyse des résultats du questionnaire

Tableau VI.1

Facteurs et leurs impacts sur le nombre de défauts lors de l'exécution du logiciel

Catégories d'Impact	Facteurs
Impact : 4G	-Type de développement -Langage de programmation -Nombre de défauts détectés de la phase : Codage -Nombre de défauts détectés de la phase : Test -Nombre de défauts détectés de la phase : Implémentation -Effort de résolution/refaite pour les défauts de la phase : Codage -Effort de résolution/refaite pour les défauts de la phase : Test -Effort de résolution/refaite pour les défauts de la phase : Implémentation
Impact : 3G, 1F	-Taille du projet -Effort -Type de langage de programmation -Utilisation d'une méthodologie développement -Utilisation d'un processus standard -Nombre de changements de spécifications de la phase : Codage -Nombre de défauts détectés de la phase : Spécification -Nombre de défauts détectés de la phase : Design -Effort de résolution/refaite pour les défauts de la phase : Spécification -Effort de résolution/refaite pour les défauts de la phase : Design -Expérience des membres de l'équipe dans développement -Changement de membres de l'équipe -Nombre de nouvelles fonctions ou ajoutées -Nombre de version de réparation de défauts
Impact : 3G	-Architecture du projet -Utilisation des outils de débogage -Nombre de changements de spécifications de la phase : Design -Nombre de changements de spécifications de la phase : Test -Nombre de changements de spécifications de la phase : Implémentation -Expérience d'IT opérations -Nombre de fonctions modifiées
Impact : 2G, 2F	-Expérience des membres de l'équipe dans le « end users'area of business » -Nombre de nouvelles versions

Catégories d'Impact	Facteurs
Impact : 2 G, 1 F	-Plateforme de développement -Expérience du chef de projet
Impact : 1G, 2F	-Utilisation des outils CASE -Nombre de fonctions supprimées
Impact : 1G, 1F	-Personnalisation -Catégorie d'acquisition de la méthodologie
Impact : 1F	-Système de Base de données

Tableau VI.2

Facteurs et leurs impacts sur le degré de satisfaction des utilisateurs

Catégories d'Impact	Facteurs
Impact : 4G	-Nombre de défauts détectés durant la phase : Implémentation -Effort de résolution/refaite pour les défauts de la phase : Implémentation
Impact : 3G, 1F	-Type de développement -Effort de résolution/refaite pour les défauts de la phase : Spécification -Nombre de version de réparation de défauts
Impact : 3G	-Effort -Architecture du projet -Type de langage de programmation -Langage de programmation -Nombre de changements de spécifications de la phase : Implémentation -Nombre de défauts détectés de la phase : Codage -Nombre de défauts détectés de la phase : Test -Effort de résolution/refaite pour les défauts de la phase : Codage -Effort de résolution/refaite pour les défauts de la phase : Test -Nombre de nouvelles fonctions ou ajoutées -Nombre de fonctions modifiées
Impact : 2 G, 1 F	-Taille du projet -Plateforme de développement -Nombre de changements de spécifications de la phase : Codage -Nombre de défauts détectés de la phase : Spécification -Nombre de défauts détectés de la phase : Design -Effort de résolution/refaite pour les défauts de la phase : Design -Expérience des membres de l'équipe dans le « end users'area

Catégories d'Impact	Facteurs
	of business » -Expérience d'IT opérations -Expérience des membres de l'équipe dans développement -Expérience du chef de projet
Impact : 2G	-Utilisation des outils de débogage -Utilisation d'une méthodologie développement -Utilisation d'un processus standard -Nombre de changements de spécifications de la phase : Design -Nombre de changements de spécifications de la phase : Test -Changement de membres de l'équipe
Impact : 1G, 3F	-Nombre de nouvelle version
Impact : 1G, 2F	-Nombre de fonctions supprimées
Impact : 1G, 1F	-Utilisation des outils CASE - Catégorie d'acquisition de la méthodologie
Impact : 1F	-Personnalisation
Impact : 0	-Système de Base de données

BIBLIOGRAPHIE

Abran, Alain, Rafa E. Al-Qutaish et Juan J. Cuadrado-Gallego. 2006. « Investigation of the Metrology Concepts within ISO 9126 on Software Product Quality Evaluation ». In *Proceedings of the 10th WSEAS International Conference on Computers - ICComp'2006*. p. 864-872. Athens, Greece: WSEAS Press.

Abran, Alain, Rafa E. Al-Qutaish, Jean-Marc Desharnais et Naji Habra. 2005. « An Information Model for Software Quality Measurement with ISO Standards ». In *Proceedings of the International Conference on Software Development - SWDC-REK*. p. 104-116. Reykjavik, Iceland: University of Iceland Press.

Abran, Alain, J. Garbajosa et Laila Cheikhi. 2007. « Estimating the Testing Volume and Effort for Testing and V & V ». In *IWSM-Mensura 2007*. p. 216-234. Baleares, Spain: Universitat de les Illes Baleares.

Abran, Alain, Adel Khelifi, Witold Suryn et Ahmed Seffah. 2003a. « Usability Meanings and Interpretations in ISO Standards ». Software Quality Journal, . vol. 11, n° 4, p. 325-338.

Abran, Alain, Martin Kunz, Reiner R. Dumke et Luigi Buglione. 2003b. « A Prototype Web-based Implementation of the QEST Model ». In *Proceedings of the 13th International Workshop on Software Measurement - IWSM'2003*. p. 82-92. Aachen, Germany: Shaker Verlag.

Abran, Alain, Miguel Lopez et Naji Habra. 2004. « An Analysis of the McCabe Cyclomatic Complexity Number ». In *Proceedings of the 14th International Workshop on Software Measurement - IWSM'2004*. p. 391-405. Aachen, Germany: Shaker Verlag.

Abran, Alain, et Asma Sellami. 2004. « Analysis of Software Measures Using Metrology Concepts - ISO 19761 Case Study ». In *the 6th International Conference on Enterprise Information Systems - ICEIS'2004: the International Workshop on Software Audits and Metrics - SAM'2004*. <http://www.gelog.etsmtl.ca/publications/pdf/801.pdf>. Consulté le 13 Juillet 2007.

Abran, Alain, Asma Sellami et Witold Suryn. 2003. « Metrology, Measurement and Metrics in Software Engineering ». In *Proceedings of the 9th International Software Metrics Symposium - METRICS'2003*. p. 2-11. Sydney, Australia, 22-23 Sept. 2003. Los Alamitos (CA), USA: IEEE Computer Society Press.

Abreu, Fernando Brito e, et Rogério Carapuça. 1994. « Candidate Metrics for Object-Oriented Software within a Taxonomy Framework ». Journal of Systems and Software, North-Holland, Elsevier Science, vol. 26, n° 1, p. 87-96.

Akingbehin, Kiumi. 2005a. « A quantitative supplement to the definition of software quality ». In *Proceedings of the Third Annual ACIS International Conference on Software Engineering Research, Management and Applications*. p. 348-352. Los Alamitos (CA), USA: IEEE Computer Society.

Akingbehin, Kiumi. 2005b. « Taguchi-based metrics for software quality ». In *Proceedings of the Fourth Annual ACIS International Conference on Computer and Information Science*. p. 713-716. Los Alamitos (CA), USA: IEEE Computer Society.

Al-Kilidar, Hiyam, Karl Cox et Barbara Kitchenham. 2005. « The Use and Usefulness of the ISO/IEC 9126 Quality Standard ». In *the International Symposium on Empirical Software Engineering*. p. 126-132. The Institute of Electrical and Electronics Engineers.

Al-Qutaish, Rafa E., et Alain Abran. 2005. « An Analysis of the Design and Definitions of Halstead's Metrics ». In *Proceedings of the 15th International Workshop on Software Measurement - IWSM'2005*. p. 337-352. Aachen, Germany: Shaker Verlag.

Basili, Victor R. 1996 «The Role of Experimentation in Software Engineering: Past, Current, and Future ». In *Proceedings of the 18th International Conference on Software Engineering*. p. 442-449.

Basili, Victor R., Richard W. Selby et David H. Hutchens. 1986. « Experimentation in Software Engineering ». IEEE Transactions on Software Engineering, vol. 12, n° 7, p. 733-743.

Beauchamp, Yves. 2005. *Maîtriser Ses Procédés: Les méthodes Taguchi et traditionnelles*. Montréal, Canada: École de Technologie Supérieure.

Bhatti, Shahid Nazir. 2005. « Why Quality? ISO 9126 Software Quality Metrics (Functionality) Support by UML suite ». ACM SIGSOFT Software Engineering Notes, vol. 30, n° 2, p. 1-5.

Boehm, B. W., J. R. Brown, H. Kaspar, M. Lipow, G. McLeod et M. Merritt. 1978. *Characteristics of Software Quality*. Amsterdam, The Netherlands: North Holland Publishing, 169 p.

Boehm, Barry W. 1981. *Software Engineering Economics*. Upper Saddle River, NJ, USA: Prentice-Hall, 767 p.

Boehm, Barry W., J. R. Brown et M. Lipow. 1976. « Quantitative evaluation of software quality ». In *Proceedings of the 2nd international conference on Software engineering*. p. 592-605. Los Alamitos (CA), USA: IEEE Computer Society.

Bourque, Pierre, Marcela Maya et Alain Abran. 1996. « A Sizing Measure for Adaptive Maintenance Work Products ». En ligne. <http://www.gelog.etsmtl.ca/publications/pdf/84.pdf>. Consulté le 30 Juillet 2007.

Bourque, Pierre, S. Oligny, Alian Abran et B. Fournier. 2007. « Developing Project Duration Models in Software Engineering ». Journal of Computer Science and Technology, vol. 22, n° 3, p. 348-357.

Bourque, Pierre, Sibylle Wolff, Robert Dupuis, Amsa Sellami et Alain Abran. 2004. « Lack of Consensus on Measurement in Software Engineering: Investigation of Related Issues ». In *14th International Workshop on Software Measurement IWSM-Metrikon 2004* p. 321-333. Konigs Wusterhausen, Magdeburg, Germany: Springer-Verlag. <http://www.gelog.etsmtl.ca/publications/pdf/836.pdf>. Consulté le 13 Juillet 2007.

Briand, Lionel, John Daly et Jurgen Wuest. 1997. « A unified framework for cohesion measurement in object-oriented systems ». In *Proceedings of 4th International Software Metrics Symposium (ISMS)*. p. 43-53.

Briand, Lionel, John Daly et Jurgen Wuest. 1999. « A Unified Framework for Coupling Measurement in Object-Oriented Systems ». IEEE Transactions on Software Engineering, vol. 25, n° 1, p. 91-121.

Briand, Lionel, Prem Devanbu et Walcelio Melo. 1997. « An Investigation into Coupling Measures for C++ ». In *Proceedings of 19th International Conference on Software Engineering (ICSE)*. p. 412-421. Boston, USA.

Cesarone, John. 2001. « The Power of Taguchi : You've heard of design of experiments and taguchi methods.; Now understand when it's appropriate to use each method ». Institute of Industrial Engineers (IIE) Solutions, vol. 33, n° 11.

Cheikhi, Laila, Alain Abran et Lopez Miguel. 2005. « Analysis of the Designs of Coupling Measures: A Case Study ». In *15th International Workshop on Software Measurement - IWSM'2005*. p. 435-454. Aachen, Germany: Shaker-Verlag.

Cheikhi, Laila, Alain Abran et Witold Suryn. 2006. « Harmonization of Usability Measurement in ISO Software Engineering Standards ». In *IEEE International Symposium on Industrial Electronics (ISIE'06)*. p. 3246-3251. IEEE Industrial Electronics Society (ICE).

Chidamber, Shyam, et Chris Kemerer. 1991. « Towards a Metrics Suite for Object-Oriented Design ». In *Conference Proceedings on Object-Oriented Programming, Systems, Languages and Applications*. p. 197- 211. ACM Press New York, NY, USA.

Chidamber, Shyam, et Chris Kemerer. 1994. « A metrics suite for object oriented design ». IEEE Transactions on Software Engineering, vol. 20, n° 6, p. 476-493.

Côté, Marc-Alexis. 2005. « An Analysis of Quality Models as a Foundation for Software Quality Engineering ». M.Eng. Thesis, Montreal, Canada, Dept. of Software Engineering and IT. École de technologie supérieure.

Côté, Vianney, Ghislaine Métivier, Pierre Bourque et Jean-Philippe Jacquet. 1996. « Caractérisation des logiciels industriels de gestion ». In *Génie Logiciel, Ates des Neuvième journées internationales, Le génie logiciel et ses applications (GL96)*. p. 92-102. Paris. <http://www.gelog.etsmtl.ca/publications/pdf/80.pdf>. Consulté le 30 Août 2007.

Cukic, Bojan. 2005. « Guest Editor's Introduction: The Promise of Public Software Engineering Data Repositories ». IEEE Software, vol. 22, n° 6, p. 20-22.

Curtis, Bill. 1980. « Measurement and Experimentation in Software Engineering ». Proceedings of the IEEE, vol. 68, n° 9, p. 1144-1157.

DeMarco, T. 1986. *Controlling Software Projects: Management, measures and estimation*. New York, NY, USA: Prentice Hall PTR, 296 p.

Déry, David, et Alain Abran. 2005. « Investigation of the Effort Data Consistency in the ISBSG Repository ». In *15th International Workshop on Software Measurement – IWSM 2005*. p. 123-136. Aachen, Germany: Shaker Verlag.

Desharnais, Jean-Marc, Alain Abran et Juan Cuadrado. 2006. « Convertibility of Function Points to COSMIC-FFP: Identification and Analysis of Functional Outliers ». En Ligne. <www.gelog.etsmtl.ca/publications/pdf/1035.pdf>. Consulté le 19 Novembre 2007.

Desharnais, Jean-Marc, France Paré, Marcela Maya et Denis St-Pierre. 1997. « Implementing a Measurement Program in Software Maintenance - An Experience Report Based on Basili's Approach ». En Ligne.

<http://www.gelog.etsmtl.ca/publications/pdf/221.pdf>. Consulté le 13 Juillet 2007

Dolado, J.J., D. Rodriguez, J. Riquelme, F. Ferrer-Troyano et J.J. Cuadrado. 2007. « A Two-Stage Zone Regression Method for Global Characterization of a Project Database». En ligne. <http://www.isbsg.org/isbsg.nsf/weben/Recent%20Research%20Papers>. Consulté le 12 Septembre 2007.

Dromey, R. G. 1995. « A model for software product quality ». IEEE Transactions on Software Engineering, vol. 21, n° 2, p. 146-162.

Dromey, R. G. 1996. « Concerning the Chimera [software quality] ». IEEE Software, vol. 13, n° 1, p. 33-43.

Fenton, Norman E., et Shari Lawrence Pfleeger. 1997. *Software Metrics: A Rigorous and Practical Approach*, 2nd ed. Boston, MA, USA: PWS Publishing Company, 656 p.

Galin, Daniel. 2004. *Software Quality Assurance : From theory to implementaion*. Harlow, England ; New York Pearson/Addison-Wesley, 590 p.

Gil, Blanca, et Witold Suryn. 2005. « ISO/IEC9126-3 Internal Quality Measures: Are They Still Useful? ». In *11th International Conference on Human-Computer Interaction (HCI)*. Las Vegas, Nevada, USA: Lawrence Erlbaum Associates, Inc.

Ian, H.Witten, et Eibe Frank. 2005. *Data Mining: Practical machine learning tools and techniques*. Californie, États-Unis: 2nd Edition, Morgan Kaufmann, San Francisco, 371 p. <http://www.cs.waikato.ac.nz/ml/weka/>. Consulté le 15 Octobre 2007.

IEEE. 1998. *Standard for a software quality metrics methodology*. IEEE Std. 1061-1998 New York, NY, USA: the Institute of Electrical and Electronics Engineers, 26 p.

IEEE. 2004. *Guide to the Software Engineering Body Of knowledge- SWEBOK*. Los Alamitos, California: IEEE Computer Society, 204 p. <http://www.swebok.org/pdfformat.html>. Consulté le 9 Juillet 2007.

ISBSG. 2005. « International Software Benchmarking Standards Group - Data Extract Release 9 ». In. CD-ROM obtained on request from ISBSG.

ISBSG. 2006a. « International Software Benchmarking Standards Group- Data Extract of February 2006 ». In. CD-ROM obtained on request from ISBSG.

ISBSG. 2006b. « International Software Benchmarking Standards Group- Glossary of Termes Version 5.9 ». En Ligne. <http://www.isbsg.org/Isbsg.Nsf/weben/Development%20&%20Enhancement>. Consulté le 26 Juillet 2007.

ISBSG. 2006c. « International Software Benchmarking Standards Group-Data Collection Questionnaire New development, Redevelopment or Enhancement Sized Using COSMIC-FFP Function Points Version 5.9 ». En Ligne. <http://www.isbsg.org/Isbsg.Nsf/weben/Development%20&%20Enhancement>. Consulté le 26 Juillet 2007.

ISO. 1991. *Software Product Evaluation - Quality Characteristics and Guidelines for their Use*. ISO/IEC IS 9126, Geneva, Switzerland: International Organization for Standardization.

ISO. 1993. *International Vocabulary of Basic and General Terms in Metrology (VIM)*. Geneva, Switzerland: International Organization for Standardization.

ISO. 2001a. *Information technology - Product Quality - Part 1: Quality Model*. ISO/IEC 9126-1, Geneva, Switzerland: International Organization for Standardization, 25 p.

ISO. 2001b. *Software Engineering - Software Measurement Process*. ISO/IEC 15939, Geneva, Switzerland: International Organization for Standardization, 39 p.

ISO. 2001c. *Technologies de l'information- Qualités de produits logiciels- Partie 1: Modèle de qualité*. ISO/IEC 9126-1, Geneva, Switzerland: International Organization for Standardization.

ISO. 2003a. *Information Technology - Product Quality - Part 2: External Metrics*. ISO/IEC TR 9126-2, Geneva, Switzerland: International Organization for Standardization, 86 p.

ISO. 2003b. *Information Technology - Product Quality - Part 3: Internal Metrics*. ISO/IEC TR 9126-3, Geneva, Switzerland: International Organization for Standardization, 62 p.

ISO. 2003c. *ISO/IEC 19761: Software Engineering - COSMIC-FFP - A Functional Size Measurement Method*. Geneva, Switzerland: International Organization for Standardization, 17 p.

ISO. 2003d. *ISO/IEC 20926: Software Engineering - IFPUG Unadjusted Functional Size Measurement Method*. Geneva, Switzerland: International Organization for Standardization, 17 p.

ISO. 2004. *Information Technology - Product Quality - Part 4: Quality in Use Metrics*. ISO/IEC TR 9126-4, Geneva, Switzerland: International Organization for Standardization, 59 p.

Jabir, et J. W. Moore. 1998. « A search for fundamental principles of software engineering ». Computer Standards & Interfaces, vol. 19, n° 2, p. 155-160.

Jacquet, Jean-Philippe, Alain Abran et Robert Dupuis. 1997. « Une analyse structurée des méthodes de validation de métriques ». In *Génie legiciel, Dixièmes journées internationales «Le génie logiciel et ses applications» (GL97)* Paris. <http://www.gelog.etsmtl.ca/publications/pdf/283.pdf>. Consulté le 13 Juillet 2007.

Jacquet, Jean-Philppe, et Alain Abran. 1997. « From Software Metrics to Software Measurement Methods: A Process Model ». In *Proceedings of the 3rd International Symposium and Forum on Software Engineering Standards - ISESS'1997*. p. 128-135. Los Alamitos (CA), USA: IEEE Computer Society Press.

Jones, Capers. 1996. *Applied Software Measurement- Assuring Productivity and Quality*, 2nd ed. États-Unis; New York, N.Y. : McGraw-Hill, 618 p.

Jung, Ho-Won, Seung-Gweon Kim et Chang-Shin Chung. 2004. « Measuring Software Product Quality: A Survey of ISO/IEC 9126 ». IEEE Software, vol. 21, n° 5, p. 88-92.

Kan, Stephen H. 2003. *Metrics and models in software quality engineering* 2nd ed. Massachusetts: Boston : Addison-Wesley, 528 p.

Kanchana, B., et V. V. S. Sarma. 1999. « Software quality enhancement through software process optimization using Taguchi methods ». In *IEEE Conference and Workshop on Engineering of Computer-Based Systems (ECBS '99)*. p. 188-193.

Kitchenham, Barbara, et Shari Lawrence Pfleeger. 1996. « Software Quality : The Elusive Target ». IEEE Software, vol. 13, n° 1, p. 12-21.

Koscianski, André, et João Candido Bracarense Costa. 1999. « Combining Analytical Hierarchical Analysis with ISO/IEC 9126 for a Complete Quality Evaluation Framework ». In *Proceedings of the 4th IEEE International Symposium and*

Forum on Software Engineering Standards - ISESS'1999. p. 218-226. Los Alamitos (CA), USA: IEEE Computer Society Press.

Li, Wei, et Sally Henry. 1993. « Maintenance metrics for the object oriented paradigm ». In *Proceedings of the First International Software Metrics Symposium*. p. 52-60. Baltimore, Maryland.

Liu, Zhiyong. 2004. « Taguchi approach for Performance Evaluation of Service-Oriented Software Systems ». Master Thesis, Ontario- Canada, University of Windsor.

Lorenz, Mark, et Jeff Kidd. 1994. *Object-oriented software metrics: a practical guide*. États-Unis: Englewood Cliffs, N.J.: PTR Prentice Hall, 146 p.

Losavio, F., L. Chirinos, A. Matteo, N. Lévy et A. Ramdane-Cherif. 2004. « ISO quality standards for measuring architectures ». Journal of Systems and Software, vol. 72, n° 2, p. 209-223.

Maxwell, Katrina D. 2002. *Applied Statistics for Software Managers*. Coll. « Software Quality Institue ». Upper Saddle River, N.J.: Prentice Hall PTR, 333 p.

McCall, J. A., P. K. Richards et G. F. Walters. 1977. *Factors in Software Quality, Volumes I, II, and III*. USA: US Rome Air Development Center Reports, US Department of Commerce.

McGarry, John, David Card, Cheryl Jones, Beth Layman, Elizabeth Clark, Josef Dean et Fred Hall. 2002. *Practical Software Measurement: Objective Information for Decision Makers*. États-Unis: Boston : Addison-Wesley , c2002, 277 p.

Morisio, Maurizio, Michel Ezran et Colin Tully. 2002. « Success and failure factors in software reuse ». IEEE Transactions on Software Engineering, vol. 28, n° 4, p. 340-357.

Oman, Paul, et Shari Lawrence Pfleeger. 1997. *Applying software metrics*. États-Unis: Los Alamitos, Calif.: IEEE Computer Society Press, 321 p.

Paulk, Mark Christopher. 2005. « An Empirical Study of Process Discipline and Software Quality ». University of Pittsburgh, 442 p.

Peace, Glen Stuart. 1993. *Taguchi methods : a hands-on approach*. Reading, Mass. : Addison-Wesley, 522 p.

Pfleeger, Shari Lawrence, et Joannw M. Altee. 2005. *Software engineering : theory and practice.* , 3rd ed. New Jersey: Upper Saddle River, N.J. Pearson Prentice-Hall, 716 p.

Pfleeger, Shari Lawrence, Norman Fenton et Stella Page. 1994. « Evaluating software engineering standards ». IEEE Computer, vol. 27, n° 9, p. 71-79.

Phadke, Amit Ashok. 2004. « Predicting Open-Source Software Quality Using Statistical and Machine Learning Techniques ». MSc thesis, Mississippi, MS, USA, Department of Computer Science and Engineering, Mississippi State University.

Phadke, Madhav Shridhar. 1989. *Quality Engineering Using Robust Design*. Englewood Cliffs, New Jersey Prentice Hall, 334 p.

Pillet, Maurice. 1997. *Les plans d'expériences par la méthode Taguchi*. Paris, France: Éditions d'Organisation, 330 p.

Pressman, Roger S. 2004. *Software Engineering: A Practitioner's Approach*, 6th ed. New York, NY, USA: McGraw-Hill, 880 p.

PROMISE. 2005. « The PROMISE Repository of Software Engineering Databases ». En Ligne. <ftp://ftp.ics.uci.edu/pub/machine-learning-databases/DOC-REQUIREMENTS>. Consulté le 26 Juillet 2007.

Rao, Baru S., et N. L. Sarda. 2003. « Effort drivers in maintenance outsourcing- An experiment using Taguchi's methodology ». In *Proceedings of the Seventh European Conference on Software Maintenance and Reengineering (CSMR'03)*. p. 271-280. IEEE Computer Society.

Ross, Pillip J. 1996. *Taguchi techniques for quality engineering : loss function, orthogonal experiments, parameter and tolerance design* 2nd ed. New York, N.Y. : McGraw-Hill, 329 p.

Roy, Ranjit K. 1990. *A primer on the Taguchi method* New York, N.Y. : Van Nostrand Reinhold, , 247 p.

Roy, Ranjit K. 2001. *Design of experiments using the Taguchi approach : 16 steps to product and process improvement*. New York, N.Y. : J. Wiley and Sons, 538 p.

Schneidewind, Norman F. 2002. « Body of Knowledge for Software Quality Measurement ». IEEE Computer, vol. 35, n° 2, p. 77-83.

Schulmeyer, G. Gordon, et James I. McManus. 1999. *Handbook of Software Quality Assurance*. Upper Saddle River, N.J. : Prentice Hall PTR, 712 p.

Sellami, Asma. 2005. « Processus de Vérification des Mesures de Logiciels selon la Perspective de Métrologie ». Ph.D Thesis, Montréal-Canada, Université du Québec.

Seshadri, Rajani, K. R. Kannan et V. Sathish. 2004. « Use of Taguchi DOE in Software process improvement ». In *International Workshop on Software Systems (IWSS 2004)*. Istanbul,Turkey

Shirabad, Jelber Sayyad, et Tim Menzies. 2005. « The PROMISE Repository of Software Engineering Databases ». En Ligne. <http://promise.site.uottawa.ca/SERepository>. Consulté le 26 Juillet 2007.

Souvay, Pierre. 2002. *Plans d'expériences : Méthode Taguchi*. Paris, France: Association française de normalisation (AFNOR), 53 p.

Strike, Kevin, Khaled El Emam et Nazim Madhavji. 2001. « Software Cost Estimation With Incomplete data ». IEEE Transactions on Software Engineering, vol. 27, n° 10, p. 890-908.

Summers, Donna C.S. 2006. *Quality*, 4th ed. Upper Saddle River, New Jersey: Pearson Prentice Hall, 819 p.

Suryn, Witold, Alain Abran et Alain April. 2003. « ISO/IEC SQuaRE: The Second Generation of Standards for Software Product Quality ». In *Proceedings of the 7th IASTED International Conference on Software Engineering and Applications*. Calgary (AB), Canada: ACTA Press.

Suryn, Witold, Alain Abran, Pierre Bourque et Claude Laporte. 2002. « Software Product Quality Practices - Quality Measurement and Evaluation using TL9000 and ISO/IEC 9126 ». In *Proceeding of 10th International Workshop on Software Technology and Engineering Practice - STEP'2002* ( ). p. 156-162. Los Alamitos (CA), USA: IEEE Computer Society Press.

Suryn, Witold, et Claude Y Laporte. 2005. « IQUAL- Ingénierie de la Qualité du Produit Logiciel ». En Ligne. <http://www.gelog.etsmtl.ca/publications/pdf-presentations/973.pdf>. Consulté le 9 Juillet 2007.

Taguchi, Genichi, Subir Chowdhury et Yuin Wu. 2004. *Taguchi's quality engineering handbook*. New Jersey: Hoboken, N.J.: Wiley, 1662 p.

Tsai, Hsien-Tang, Herbert Moskowitz et Lai-Hsi Lee. 2003. « Human resource selection for software development projects using Taguchi's parameter design ». European Journal of Operational Research, vol. 151, n° 1, p. 167-180.

Wolff, Sibylle. 1999. « La place de la mesure au sein des principes fondamentaux du génie logiciel ». M.Eng. Thesis Montréal- Canada., Université du Québec à Montréal. <http://www.gelog.etsmtl.ca/publications/pdf/438.pdf>. Consulté le 10 Novembre 2007.

Yang, Kai, et Basem El-Haik. 2003. *Design for Six Sigma: a roadmap for product development*. New York McGraw-Hill, 624 p.

Zhou, Jia, Kendra Cooper et I-Ling Yen. 2006. « QoS Data Collection: An Approach to Assist Predictable QoS Behavior Modeling in Component Based Development ». In *PROMISE: Predictive Models in Software Engineering Workshop, ICSE 2006, September 24, 2006.* <http://unbox.org/promise/2006/papers/ZhouCooperYen.pdf>. Consulté le 10 Novembre 2007.