UNIVERSITÉ DU QUÉBEC À MONTRÉAL

UN ENVIRONNEMENT AUTOMATISÉ POUR UN PROCESSUS TRANSPARENT D'ESTIMATION FONDÉ SUR LA BASE DE DONNÉES DU *INTERNATIONAL SOFTWARE BENCHMARKING STANDARDS GROUP* (ISBSG)

RAPPORT D'ACTIVITÉ DE SYNTHÈSE

PRÉSENTÉ

AU SOUS-COMITÉ D'ADMISSION ET D'ÉVALUATION

MAÎTRISE EN INFORMATIQUE DE GESTION

PAR

VASILE STROIAN

AVRIL 1999

Rapport final de l’activité de synthèse approuvée par :

ALAIN ABRAN Directeur de recherche	**PIERRE BOURQUE** Codirecteur de recherche

REMERCIEMENTS

Je remercie tout particulièrement mes directeurs de recherche M. Alain Abran et M. Pierre Bourque pour leur support constant, leurs compétences et leur critiques toujours constructives qui m'ont permis de donner le meilleur de moi-même.

Je dois aussi remercier à M. Robert Dupuis, directeur de la Maîtrise en Informatique de gestion qui m'a appuyé tout au long de mes études de maîtrise.

Un remerciement à toute l'équipe du Laboratoire de recherche en gestion des logiciels de l'UQAM, tout particulièrement à Michèle Hébert pour leur disponibilité tout au long de ce projet.

Finalement je remercie ma fille, Marie-Louise et ma femme, Adriana que j'ai privé de mon attention durant ces innombrables heures consacrer à cette activité.

SOMMAIRE

La présente activité de synthèse s'inscrit dans le cadre du programme de maîtrise en informatique de gestion offert par l'Université du Québec à Montréal.

L'objectif de cette activité de synthèse est de développer un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données historiques de l'International Software Benchmarking Standards Group (ISBSG). Ce développement s'inscrit dans le cadre du programme de recherche du Laboratoire de recherche en gestion des logiciels de l'UQAM.

La problématique de recherche que nous avons proposée de résoudre est la suivante : comment transformer un processus « boîte noire » d'estimation en un processus « boîte transparente » fondé sur la base de données de l'International Software Benchmarking Standards Group (ISBSG) ?

Plusieurs études montrent que plus de deux tiers des projets de développement informatique dépassent très largement leurs délais ce qui engendrent des pertes considérables pour les entreprises. La décision de commencer ou de continuer un projet est souvent influencée par les estimations d'effort de développement. Malgré l'importance de bien estimer les gestionnaires ne possèdent pas d'outils capables de leur fournir l'information crédible au cours du processus d'estimation. Les bases de données historiques à l'appui des progiciels d'estimation ne sont pas accessibles, ce qui ne permet pas de déterminer la crédibilité des résultats et du processus d'estimation. Les progiciels actuels sont de type « boîte noire » ce qui limite l'analyse des résultats et ne permet pas d'en déterminer la fiabilité.

Avec les opportunités offertes par l'existence de la base de données ISBSG, qui est la première grande base de données de projets informatiques disponible pour les chercheurs et les praticiens nous avons mis au point un environnement automatisé pour un processus transparent d'estimation. Cet environnement améliore le processus d'estimation des projets informatiques et il représente une contribution intéressante à ISBSG en donnant une valeur ajoutée à leur base de données.

TABLE DES MATIÈRES

SOMMAIRE iii

INTRODUCTION 1

1. PROBLÉMATIQUE 2

1.1 Les impacts des mauvaises estimations 2
1.2 Les problèmes de disponibilité de base de données historiques 4
1.3 Les progiciels d'estimations 5
1.4 L'approche boîte noire (processus opaque) 6
1.5 Problématique de la recherche 8
1.6 Objectif de l'activité de synthèse 8
1.7 Les utilisateurs de l'environnement 9

2. MÉTHODOLOGIE 10

2.1 Cadre conceptuel 10
2.1.1 Processus transparent d'estimation ..... .10
2.1.2 Modèles de productivité 12
2.1.3 Les systèmes d'aide à la décision et la simulation 14
2.1.4 Une base de données transparentes 15
2.1.5 Techniques statistiques 18
2.2 Le langage de programmation utilisé pour la réalisation de l'outil 21
2.3 Le projet dans le programme de recherche du laboratoire 22

3. PRÉSENTATION ET ANALYSE DES RESULTATS 23

3.1 Les étapes de l'estimation avec l'environnement automatisé 23
3.2 Présentation de résultats 25
3.3 Cadre de Basili modifié 26

4. CONCLUSION 29

RÉFÉRENCES 36

APPENDICE A
LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU DATBASE_ISBSG 32

APPENDICE B
LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU "EVOLUTION" 33

APPENDICE C
LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU "SIMULATION"..... 34

APPENDICE D
LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC L'OUTIL..... 35

LISTE DES FIGURES ET DES TABLEAUX

Figure Page

1.1 Une approche d'estimation «boîte noire» 7
2.1 Les composants d'un processus transparent d'estimation 11
2.2 Les composants d'un processus d'estimation en utilisant la base de données ISBSG et le Projet A. 17
2.3 Les étapes d'une analyse statistique 20
3.1 Les étapes de l'estimation par rapport au processus transparent d'estimation 24
3.2 Présentation de résultats sur le site WEB 27

Tableau Page

1.1 L'information nécessaire pour produire l'estimation 4
1.2 Les suggestions pour améliorer le processus d'estimation 5
3.1 Cadre de Basili pour l'activité de synthèse 28
4.1 Échéancier de l'activité de synthèse 31

INTRODUCTION

L'estimation de projet est un élément déterminant du processus décisionnel des investissements (Abran et Robillard, 1993). Les résultats des modèles d'estimation sont utilisés pour la planification des ressources et de l'échéancier de projet. Parmi les problèmes majeurs des outils logiciels pour aider au processus d'estimation on remarque l'inaccessibilité à leur base de données ce qui limite l'analyse de la fiabilité des résultats produits en extrant de ces logiciels d'estimations. Les gestionnaires se posent toujours la question de la crédibilité de l'information et le processus d'estimation doit fournir aux managers plus qu'un nombre, il doit fournir l'information nécessaire pour prendre les meilleures décisions. Selon Louis (Louis et al.,1991) en estimation il y a toujours une question importante pour les gestionnaires « how accurate were the estimates ? ».

Ce document présente le rapport de l'activité de synthèse dont l'objectif est de développer un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données de l'International Software Benchmarking Standards Group (ISBSG).

Ce document se divise en trois grandes sections. La première section expose la problématique à partir d'une revue de la littérature en précisant aussi les objectifs et les limites de notre recherche. La deuxième section présente le cadre conceptuel que nous avons adopté et les aspects méthodologiques de la démarche. La troisième section présente l'environnement automatisé qui a été développé pour rencontrer les objectifs fixés.

La recherche a débuté à l'été 1997 et a été terminée au mois de mars 1999.

Le travail a été réalisé sous la direction de Monsieur Alain Abran et Monsieur Pierre Bourque.

1. PROBLÉMATIQUE

1.1 Les impacts des mauvaises estimations

Au début des années 1990 Robert Glass remarquait : « [...] if there is one management danger zone to mark above all others, it is software estimation » (Glass, 1991).

Le processus d'estimation doit fournir aux développeurs une planification raisonnable de ce qu'ils devraient faire (DeMarco, 1982) et une information crédible et vérifiable aux gestionnaires pour qu'ils puissent prendre les meilleures décisions (Abran et Desharnais, 1996).

Le processus d'estimation peut être divisé en deux grandes étapes : la première est de déterminer la taille et la complexité du logiciel et la deuxième est de déterminer l'effort requis pour développer un tel logiciel. Une fois la taille et la complexité déterminées on estime l'effort en fonction de la productivité de chaque entreprise. L'estimé de l'effort est ensuite utilisé pour la planification des ressources et du calendrier de projet.

La gestion de projets logiciels diffèrent des autres types de gestion de projet sur au moins quatre points (Brooks, 1987) : la complexité, l'invisibilité, la conformité et les changements. Le développement de logiciel est plus complexe parce que nous nous ne comprenons pas clairement le processus de développement de logiciel. Dans le processus de l'ingénierie plus traditionnelle, on peut mieux prédire les stades du développement alors qu'en génie logiciel, le modèle du cycle de développement comme celui de la cascade ne sont que des représentations simplifiées. Le logiciel selon Brooks « is invisible and unvisualisable » (Brooks, 1987). Le logiciel est intangible, on ne peut pas le voir ni le toucher et le chef de projet doit s'en remettre à la documentation pour évaluer le progrès du projet de développement. Le chef d'un projet de construction de bâtiment peut voir le produit se développer et si la planification ne suit pas le cheminent initialement prévu les effets sur le produit sont immédiatement visibles. Il est difficile de mesurer un produit intellectuel alors que par analogie il est facile de mesurer un produit physique: la température, le poids etc. Les grands projets

logiciels sont souvent très différents de ce qui déjà a été fait et alors les expériences antérieures ne sont pas d'un grand secours pour prédire le coût du projet. Chaque grand développement de logiciel est souvent un projet techniquement innovateur.

Dans une étude effectuée par Standish Group en 1995 (The Standish Group, 1998), aux États-Unis plus de 31 % des projets ont été arrêtés et 53% ont un dépassement de budget ou sont en retard ou ont livré moins de fonctionnalités que prévu initialement. Selon cette étude les dépenses en développement de logiciel ont été de plus de 250 milliards $ pour un nombre approximatif 175,000 projets. Seulement 16% finissent leur projet à temps et à l'intérieur du budget ce qui représente moins de un projet sur 6 ! Les 80,000 projets arrêtés ont coûté au gouvernement et aux entreprises privées 81 milliards $ en 1995 et les opportunités d'affaires perdues qui ne sont pas mesurables sont estimées en trillions $.

D'autres études montrent que près de deux tiers des projets dépassent très largement leurs délais (Garmus et Herron, 1996, Ingram,1994 ; Jones, 1997 ; Lederer et Prasad,1993). Les projets sont généralement en retard sur la date de livraison de 25 à 50%, et l'importance de ce retard est proportionnelle à la taille du projet (Jones, 1994). Une étude effectuée en 1991 sur plus de trois cents projets a révélé que les retards étaient rarement comblés (van Genunchten, 1991).

L'évaluation « à priori » d'un projet de développement des logiciels est une préoccupation majeure (Londeix, 1987 ; Fenton, 1994). L'estimation de l'effort de développement est une des activités importantes effectuées par un chargé de projet. Les estimations influencent considérablement le déroulement du projet et le logiciel qui en découle. La décision de commencer ou de continuer un projet est souvent influencée par les estimations d'effort de développement.

Une estimation trop faible peut amener la direction à investir considérablement dans un projet informatique alors qu'elle aurait pu obtenir un meilleur rendement ailleurs. Selon Boehm :

« The software undersizing problem is our most critical road block to accurate software cost estimation [...] there are no magic formulas that we can use to overcome the software undersizing problem. In absence of any such formula, it is important to understand the major sources of the software undersizing problem » (Boehm, 1981).

D’autre part, une estimation trop élevée peut amener une entreprise à abandonner un projet qui aurait pu réduire ses dépenses, augmenter ses revenus, améliorer son image auprès de la clientèle, etc. L'estimation de l’effort nécessaire pour réaliser un projet est importante aussi quand on doit soumissionner pour obtenir un contrat de développement : on n'obtiendra pas le contrat si le montant de la soumission est plus élevé que celui des autres mais on risque d’avoir un déficit à combler si on sous-estime l’effort requis pour compléter le contrat.

1.2 Les problèmes de disponibilité de base de données historiques

Selon Kitchenham un processus d’estimation devrait être basé sur une base de données historique, sur l’utilisation de méthodes de développement adéquates et sur une définition claire des besoins (Kitchenham, 1996).

La plupart des organisations ne possède pas leur propre base de données sur les projets (Hotle, 1996) parce qu’ils ne réalisent que quelque projets par année et il faut accumuler des données sur plusieurs années. De plus, moins de 15% de toutes les organisations ont implanté un programme de mesure.

Dans un article publié en 1993 par Lederer et Prasad dans le "Journal of Information Technology", les estimateurs ont été demandés quelle est l’information nécessaire pour faire compléter l’estimation mais qui manque habituellement (Lederer et Prasad, 1993). Le **Tableau 1.1** présente les résultats. Le **Tableau 1.2**. présente leur recommandation pour améliorer le processus.

Tableau 1.1 L'information nécessaire pour produire l’estimation

Information désirée	Score
«Rules of thumb» (productivité, effort)	21
Les ressources disponible et leur compétences	18
Base de données sur les projets passés	18
Feedback sur les projets estimés	17

Tableau 1.2 Les suggestions pour améliorer le processus d'estimation

Mesures	Nombre de mentions
Cadre qui inclut les résultats métriques basés sur les anciens projets	9
Feedback sur les estimations passées	6
«Wider range of work »	2
Feedback sur les projets estimés	2

1.3 Les progiciels d'estimation

Les progiciels d'estimation permettent d'estimer l'effort à partir de paramètres comme la taille estimée des différents types de projets, la composition de l'équipe de développement, l'échéance et d'autres variables.

Dans le travail réalisé par Mendes *et al.* on trouve une liste de progiciels d'estimation (Mendes et al., 1997) ; il est à remarquer que le premier outil ayant pris en considération les points de fonctions comme mesure de la taille fonctionnelle des logiciels a été SPQR/20 (www.spr.com, 1998) en 1985. Un décennie plus tard la grande majorité des progiciels d'estimation sur le marché utilisent ce paramètre. Plusieurs progiciels ont aussi la possibilité d'avoir une estimation détaillée et compatible avec Microsoft Project, Project Manager etc.

SPR (Software Productivity Research) a deux progiciels d'estimation : SPR KnowledgePLAN 2.0 et CHECKPOINT. Ils affirment que leur base de données contient de l'information sur 6700 projets (Jones, 1997) et de multiples variables sur le processus de développement, la technologie, le personnel et tous les environnements majeurs de développement.

SLIM (Software Lifecycle Management) est un progiciel du Putnam. En septembre 1996, ils affirmaient que la base de données contenait 3885 projets (Putnam et al., 1997).

Gartner Group affirment que leur base de données Real Decision contient des données qui proviennent de plus de 500 organisations, accumulées sur plus de 20 ans, et contient des informations sur plus de 5400 projets (www.gartnergroup.com, 1998).

Hotle mentionne qu'en 1996 la plupart des organisations débutent l'implantation d'un processus d'estimation par l'achat d'un progiciel d'estimation. La taille du marché des progiciels d'estimation est petite (Hotle, 1996) : le chiffre total d'affaires ne représente que 5-10 millions $ et presque la moitié est représentée par CHECKPOINT mis en marché par SPR (Hotle, 1996). Parmi les problèmes majeurs de ces progiciels on remarque un manque de cohésion, la complexité de l'outil et l'inaccessibilité à leur base de données (Hotle, 1996). Les quelque firmes de progiciels d'estimation affirment avoir des milliers de projets dans leurs base de données mais refusent systématiquement de donner des informations précises (Garmus et al., 1996) sur les attributs de leurs données et leurs procédures de validation.

1.4 L'approche boîte noire (processus opaque)

Une approche « boîte noire » (voir **Figure 1.1**) reçoit en intrant l'estimé de la taille et de la complexité des logiciels à produire selon un certain processus et par certaines ressources produit en extrant les résultats de l'estimation.

La base de données historique à l'appui du progiciel d'estimation n'est pas accessible, ce qui ne permet pas de déterminer la crédibilité des résultats et du processus d'estimation. Les équations et les détails des paramètres obtenus ne sont pas documentés pour les utilisateurs de ces progiciels.

Les modèles d'estimation sont préprogrammés dans les progiciels, et ne sont pas transparents pour les utilisateurs. Les utilisateurs n'ont donc pas d'information sur la structure des modèles d'estimation utilisés et ils ne peuvent pas connaître le type de modèle (avec paramètres déterminés heuristiquement

ou par méthodes statistiques) ou la qualité statistique des résultats. De plus les analyses graphiques sont « préfabriqués » et il n'y a pas la possibilité d'avoir de scénario personnalisé conforme à nos besoins. La recherche scientifique en utilisant les progiciels commerciaux devient presque impossible.

Comment un chef de projet peut-il poser un jugement raisonnable en se basant sur un nombre seulement ? Comment peut-il convaincre ses patrons à partir d'information non vérifiable par lui ? Comment peut-on établir le niveau de confiance d'un processus d'estimation alors qu'on ne peut pas l'établir pour ses composants internes?

Le résultat d'un processus d'estimation ne doit pas être basé seulement sur un nombre parce que les gestionnaires doivent prendre des décisions importantes en se basant sur une information crédible. La crédibilité de chaque composant dans cette approche représente un vrai défi.

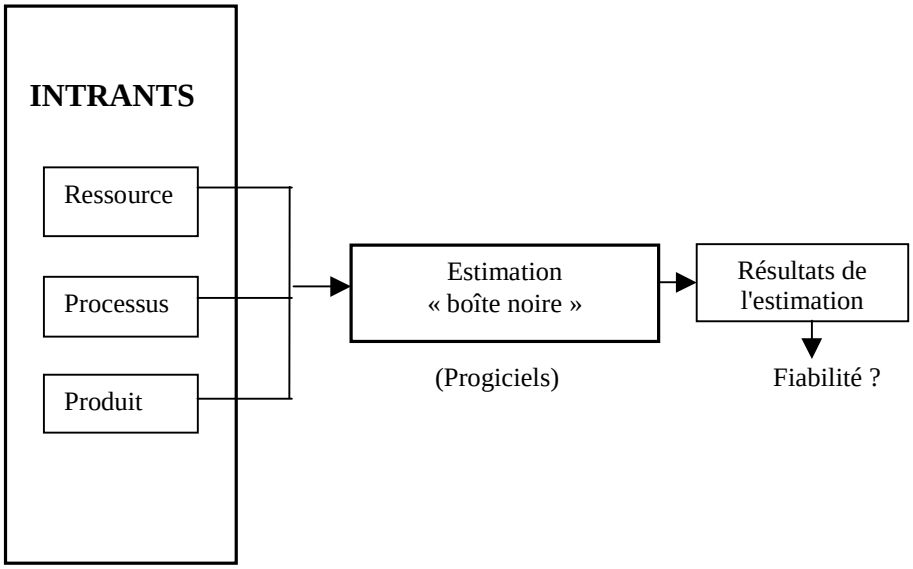


Figure 1.1 Une approche d'estimation « boîte noire » (Abran et Desharnais, 1996)

1.5 Problématique de la recherche

La problématique de recherche que nous avons proposée de résoudre est la suivante : comment transformer un processus « boîte noire » d'estimation en un processus « boîte transparente » fondé sur la base de données de l'International Software Benchmarking Standards Group (ISBSG) ?

Cette problématique préoccupe plusieurs personnes dont les activités sont liées à l'estimation. Dans le cadre du Laboratoire de recherche en gestion des logiciels de l'Université du Québec à Montréal, plusieurs travaux ont été réalisés sur l'analyse des points de fonction, les méthodes de validation des mesures du logiciel et les programmes de mesure.

Dans le cadre des recherches de ce laboratoire, des analyses ont aussi été faites pour décrire un processus transparent d'estimation. Un cadre pour un processus transparent d'estimation a été ainsi mis au point (Abran et Desharnais, 1996). Notre recherche constitue le prolongement de ces travaux et pourrait donner lieu à plusieurs recherches subséquentes dans le domaine de l'estimation du logiciel.

1.6 Objectif de l'activité de synthèse

L'objectif de la présente activité de synthèse est de développer un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données historiques de l'International Software Benchmarking Standards Group (ISBSG).

Mon objectif personnel est de mieux comprendre le processus de l'estimation, d'améliorer mes connaissances au niveau de l'analyse statistique, de réaliser de la programmation dans un langage qui est de plus en plus demandé sur le marché du travail et de contribuer à l'amélioration du processus d'estimation.

1.7 Les utilisateurs de l'environnement

Cet environnement pourrait être utilisé par les chefs de projets informatiques et par les gestionnaires de programme de mesure. Il représente une contribution intéressante à ISBSG en donnant une valeur ajoutée à leur base de données.

Cet environnement pourrait aussi être utilisé comme environnement de recherche et d'enseignement par les membres du Laboratoire de recherche en gestion des logiciels de l'Université du Québec à Montréal. D'autres chercheurs en génie logiciel pourraient aussi être intéressés.

Ayant défini les problèmes et les objectifs qui nous préoccupent ici, nous nous proposons de préciser, dans la prochaine section les méthodes qui ont été utilisées pour réaliser nos objectifs.

2. MÉTHODOLOGIE

2.1 Cadre conceptuel

Un projet est un ensemble relativement complexe d'activités et de tâches (Genest et Nguyen, 1995), toujours orientées vers un objectif précis et connu au départ; cet objectif correspond à la réalisation d'un extrant concret, un produit nouveau; la livraison de ce produit concrétise l'atteinte de l'objectif du projet.

Le Petit Robert définit un modèle comme « une représentation simplifiée d'un processus d'un système ». Un modèle nous permet de simuler les caractéristiques essentielles d'un projet sans être obligé de le réaliser.

2.1.1 Processus transparent d'estimation

Dans le processus d'estimation (voir **Figure 2.1.**) il y a trois grandes parties (Abran et al., 1996) : les intrants mesurables, le sous-processus de simulation et le sous-processus d'ajustement.

Le sous-processus de simulation inclut un modèle de simulation de la productivité ayant comme intrant des mesures des ressources, du processus et du produit et comme extrant les résultats de la simulation ainsi que de l'information sur la fiabilité de ses résultats de la simulation (le rapport de « confiance » dans la **Figure 2.1**).

Dans une approche transparente d'estimation, le modèle de simulation doit être documenté et crédible. Le rapport de confiance doit comprendre un degré de fiabilité pour chaque résultat de la simulation.

Le sous-processus d'ajustement prend en considération les résultats de la simulation comme intrant d'une part et les facteurs de risque et d'incertitude d'autre part pour ensuite produire les résultats finaux de l'estimation.

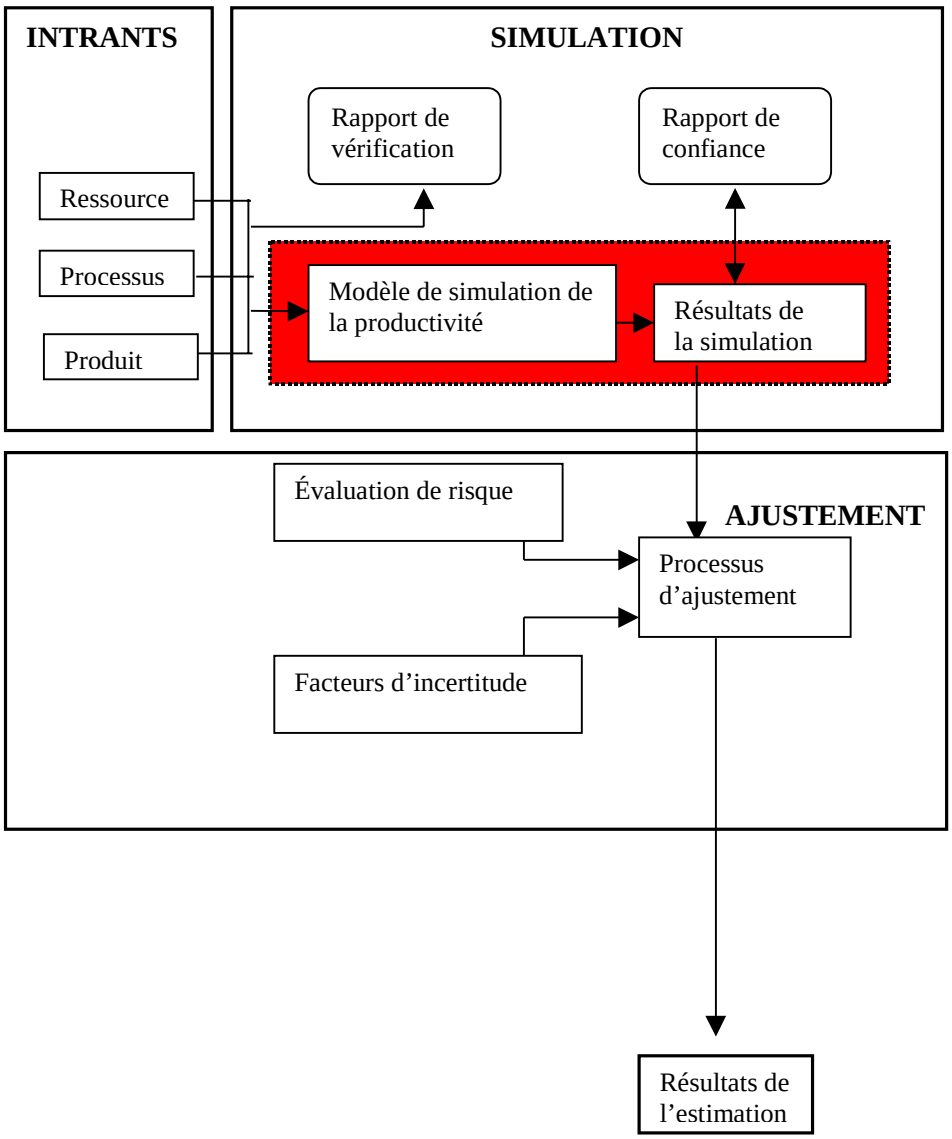


Figure 2.1 Les composants d'un processus transparent d'estimation (Abran et Desharnais, 1996)

La crédibilité du processus d'estimation est basée sur la crédibilité de chaque sous-processus d'estimation qui à leur tour sont basées sur la crédibilité de chaque composant. Le résultat final ne peut pas être plus crédible que la crédibilité de chaque composant et il est tout aussi crédible que le moins crédible des composants.

Park décrit clairement le flux d'information et les activités qui sont associées au processus d'estimation. Il insiste beaucoup sur l'acquisition de connaissance et sur l'importance de l'expérience en ce qui concerne la prise de décision dans un processus d'estimation. Les décisions ne doivent pas être basées seulement sur les résultats du modèle de simulation, elles doivent aussi tenir compte des principes de base énoncés en 1975 et qui restent toujours valables (Park,1989) :

« 1. Estimates are made by people not by models. They required reasoned judgements and commitments to organisational goals that cannot be delegated to any automated process.

2. All estimates are based on comparaisons. When people estimate, they evaluate how something is like, and how it is unlike, things that they or others have seen before.

3. Before people can estimate, they must require knowledge. They must collect and quantify information from other projets, so that they can place their comparative evaluations, on demonstrably sound footings. »

2.1.2 Modèles de productivité

Au début d'un projet il y a un nombre considérable d'inconnus et de facteurs d'incertitudes et de risques importants. Au fur et à mesure que le projet avance les difficultés spécifiques sont progressivement éliminées et à la fin du projet il n'existe plus d'inconnus, d'incertitudes ni de risques et toute l'information concernant le projet est disponible.

À la fin d'un projet il est possible de construire des modèles pour analyser la relation entre le produit développé et les intrants au processus de développement, c'est-à-dire le ratio de productivité. Il s'agit d'un modèle « à posteriori », nommé le plus souvent dans la littérature modèle d'estimation, une expression assez ambiguë, ou modèle de productivité (Abran et al., 1993).

Un modèle de productivité sera considéré empiriquement « acceptable » s'il est capable de rencontrer le critère d'erreur relative moyenne de +/- 25% pour 75% des observations (Abran et al., 1993). Ce critère d'évaluation est recommandé par plusieurs auteurs spécialisés dans le domaine du génie logiciel (Conte, 1986; Verner, 1992).

L'approche utilisée pour la construction du modèle COCOMO (Boehm, 1981) peut être considérée comme classique en ce qui concerne la construction de modèles de productivité en génie logiciel. Les équations ont été construites à partir d'une base de données de 63 projets (Conte, 1986). COCOMO utilisent plusieurs concepts des modèles de productivité qui l'ont précédé dans les années 1970 et plusieurs des modèles dans les années 1980-90 s'en sont fortement inspirés (Abran et al., 1993).

Au cours des années 70 et au début des années 80, c'est le nombre de lignes de code qui constituait la mesure la plus fréquemment utilisée comme intrant à l'estimation de projet. Cette mesure était satisfaisante à cette époque parce que les principaux langages informatiques utilisés présentaient de nombreuses similarités du point de vue de la productivité. Au fur et au mesure que les langages et les techniques de programmation se perfectionnaient et se diversifiaient, l'utilisation du nombre de lignes de code comme principale mesure de la taille d'un logiciel devenait de plus en plus limitative.

C'est en voulant éviter les limitations et les paradoxes du calcul du nombre de code qu'Allan Albrecht a développé la méthode des points de fonction. Cette méthode de mesure est utilisée pour mesurer la taille fonctionnelle des projets (Albrecht, 1983). Les points de fonction sont calculés en comptant les caractéristiques suivantes du logiciel : entrées et sorties externes, interactions avec l'utilisateur, interfaces externes et fichiers utilisés par le système. On identifie et compte chacune de ces caractéristiques et on obtient une taille non ajustée, taille qui est ensuite pondérée, en se basant sur une gamme de facteurs comme le taux de réutilisation, la performance etc.

Plusieurs organisations se servent de cette mesure pour construire des modèles de productivité (Abran et *al*., 1993 ; ISBSG, 1996).

2.1.3 Les systèmes d'aide à la décision et la simulation

Les systèmes d'aide à la décision ont pour fonction essentielle d'effectuer des analyses et des évaluations visant à aider le gestionnaire ou l'analyste à évaluer les effets prévisibles d'une décision que l'entreprise envisage de prendre. Les systèmes d'aide à la décision peuvent être groupés en trois types (Genest, 1996): les systèmes interactifs d'aide à la décision; les systèmes d'optimisation et de simulation et les systèmes expert. Ils sont particulièrement bien adaptés pour répondre aux besoins des gestionnaires pendant les phases d'évaluation du processus de décision (Waterson, 1994).

Dans un système d'optimisation, la réalité à étudier est modélisée par des équations mathématiques qui décrivent les caractéristiques et le fonctionnement de l'opération étudiée; un modèle conçu pour permettre l'optimisation d'une opération complexe peut comporter des centaines ou des milliers de variables et d'équations.

Les systèmes de simulation imitent le système réel en reproduisant pas à pas son comportement. Dans un système de simulation, la réalité à étudier est modélisée par l'identification de tous les événements qui se produisent au cours de l'opération étudiée ainsi que les relations entre ces événements. La simulation est un processus de résolution pas à pas où les équations disent comment calculer le pas temporel suivant mais non comment aller directement vers n'importe quel instant futur.

La simulation d'un modèle d'un système réel ne donne pas une solution générale mais plutôt l'évolution séquentielle des variables de l'état du système en fonction des coefficients qu'on lui a assigné. La simulation donne une autre solution numérique si le modèle est expérimenté avec des conditions différentes. L'expérience de simulation porte sur un modèle du système réel plutôt que sur le système réel lui-même comme dans le cas d'une véritable expérience.

La capacité de manipuler la complexité permet aux modèles de simulation d'aborder les problèmes d'un point de vue plus holistique que les méthodes qui ne peuvent pas traiter aisément plusieurs variables. Les avantages sont bien mis en évidence par Hormer :

« The most important advantage of a simulation model is its ability to « play out» the dynamic consequences of a given set of assumptions in a way the human mind can do neither well nor consistently ; a useful model produces scenarios that are both realistic and explainable in the policymaker's own terminology. In addition, a simulation model provides an experimental arena for discovering the source of real-life problems and evaluating alternative policy options in relatively little time and with litle cost » (Hormer, 1983).

2.1.4 Une base de données transparente

En 1990 le groupe australien « Australian Software Metrics Association » (ASMA) débutait la mise en place d'un groupe de benchmarking ayant comme objectif la collecte et l'analyse des données de productivité provenant de projets informatiques.

En 1994, plusieurs associations nationales se sont réunies pour développer des normes de benchmarking dans le domaine de la mesure du logiciels et ISBSG (International Software Benchmarking Standards Group) est créé. Aujourd'hui les organisations affiliées sont : ASMA (Australian Software Metrics Association), CIM (Centre d'Intérêt sur les Métriques- Canadian Interest Group in Software Measurement Association), UKSMA (United Kindom Software Metrics Associations), GUFPI (Italie), NESMA (Hollande) et DASMA (Deuschsprachige Andwendrgruppe fur Software Metrik und Aufwandschatzung) et IFPUG (International Function Point Users Group).

La mission de ISBSG est : « To promote the use of measurement in the development and support of software in order to improve processes, products and services through education, liaison with other professional groups, provision of services, involvement in standard setting and research » (www.asma.org.au/a_fevent.htm, 1998) .

Les entreprises qui contribuent à cette base de données doivent utiliser obligatoirement un programme de mesure pour leur projet. Les points de fonctions avec la méthode IFPUG sont utilisés dans une proportion de presque 90% des projets de la base de données.

Dans cette base de données (International Software Benchmarking Group, 1997), on retrouve des informations portant sur l'effort en jours personne consacrées au projet, la taille du projet, la qualité du produit livré et l'entreprise, au total il y a environ 80 attributs. Dans la version 1997 il y a 396 projets qui proviennent d'Asie, d'Amérique du Nord et d'Europe. Les plus grands contributeurs sont les Australiens avec un projet sur deux de la base de données, suivi du Canada.
La majorité des projets de la base de données, soit 83%, sont de projets réalisés avec des environnements de développement en 3GL et 4GL. Il y a sept langages de programmation associés au 4 GL et dans les deux premières positions on retrouve les langages Natural et SQL. Pour les langages 3GL on retrouve les deux versions de COBOL avec 67% des projets. La plupart de projets proviennent des organisations de types suivants: administration publique, finance, affaire et manufacturier.

Venturi, un logiciel disponible gratuitement sur le Web, est utilisé pour la saisie le données. L'évaluation des données soumises par l'entreprise est faite par deux personnes qui sont membres de ISBSG et l'entreprise reçoit un feedback après 3-4 semaines. L'évaluation est basée sur un niveau de confiance accordé à l'exactitude et la complétude des données. Un échelle d'évaluation à trois niveaux est utilisée. La valeur de cette base de données provient du fait quelle est la première grande base de données de projets informatiques disponible pour les chercheurs et les praticiens.

Quelques articles ont été publiés suite à des analyses statistiques du contenu de cette base de données, l'article de Oligny et *al.* « Developing project duration models in software engineering » a retenu notre attention. Dans cet article une approche d'estimation « boîte transparente » a été utilisée (Oligny et al, 1997). Étant donné que les attributs de la base de données sont disponibles, des statistiques détaillés ont été faits.

Un processus transparent d'estimation en utilisant la base de données ISBSG est présenté dans la **Figure 2.2**. Le logiciel développé dans ce travail ne tient pas compte des risques d'affaire ni des incertitudes; de plus le processus d'ajustement n'est inclus dans l'environnement. Le rapport de vérification et le rapport de confiance qui se retrouvent dans le module de simulation ne feront pas partie de l'environnement que nous allons automatiser, cependant le résultat de la simulation va contribuer à leur préparation.

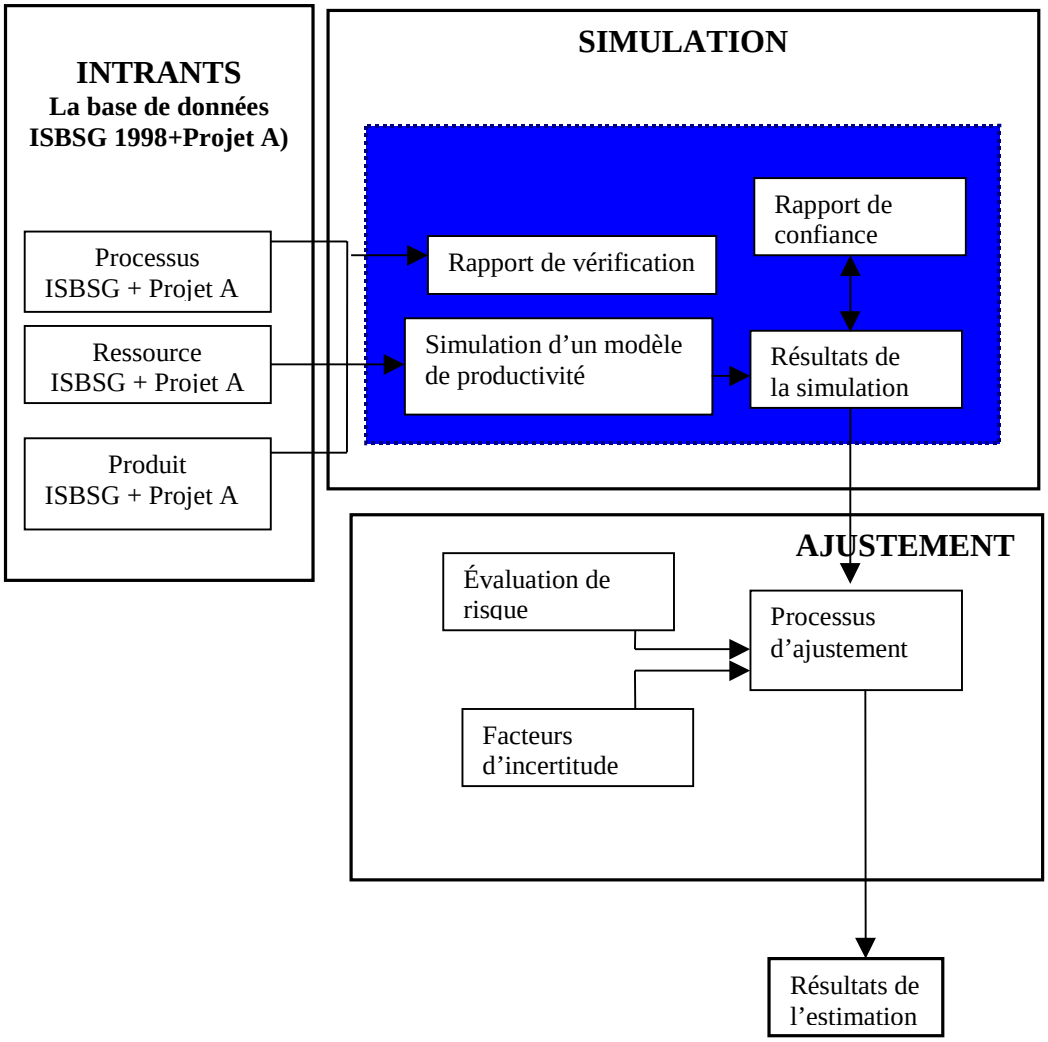


Figure 2.2 Les composants d'un processus d'estimation en utilisant la base de données ISBSG et le projet A

2.1.5 Techniques statistiques et les étapes de l'analyse statistique

La statistique s'intéresse à généraliser à de grands ensembles (populations) les conclusions tirées des résultats obtenus avec des ensembles beaucoup plus restreints (échantillons). Une des parties de la statistique consiste à apprendre comment on prend une bonne décision concernant un grand groupe (population) après avoir étudié les informations recueillies pour un petit groupe de cette population (échantillon). Un échantillon est un groupe relativement petit et choisi scientifiquement de manière à représenter le plus fidèlement possible une population.

La plupart des modèles d'estimation proposés par la littérature sont basés sur les techniques statistiques des régressions (Matson et *al.*; 1994). On peut établir et valider statistiquement, par analyse de régression, une relation causale entre le coût d'un projet déjà réalisé et différentes attributs de ce projet comme le langage de programmation, la taille, le type de langage, la méthode de développement etc. La variable dont on veut expliquer les valeurs inconnues est appelée variable dépendante ou expliquée et les autres variables dont les valeurs sont connues et qui servent à expliquer les valeurs la variable dépendante, sont appelées variables indépendantes ou explicatives.

En général dans une analyse statistique (voir **Figure 2.3**) on commence par l'identification du sujet de l'étude et la population à laquelle cette étude est destinée. Pour le faire on restreint au minimum le nombre de questions à cause de la difficulté de recueillir les données et de les analyser. Il faut choisir les questions significatives avec lesquelles on peut tirer des conclusions valables. Dans un outil d'estimation le sujet est bien identifié mais il reste à faire le choix des variables indépendantes ou explicatives : quels attributs d'un projet informatique sont importants pour l'analyse ?

Ensuite il faut décider si l'expérience s'adresse à toute la population ou à un échantillon. Pour la plupart des expériences, on utilise un échantillon. On généralise pour la population les résultats obtenus pour l'échantillon par le biais de l'inférence statistique. À cette étape l'échantillon dans un outil d'estimation correspond aux choix des entreprises qui contribuent avec leurs projets à la base de données historiques.

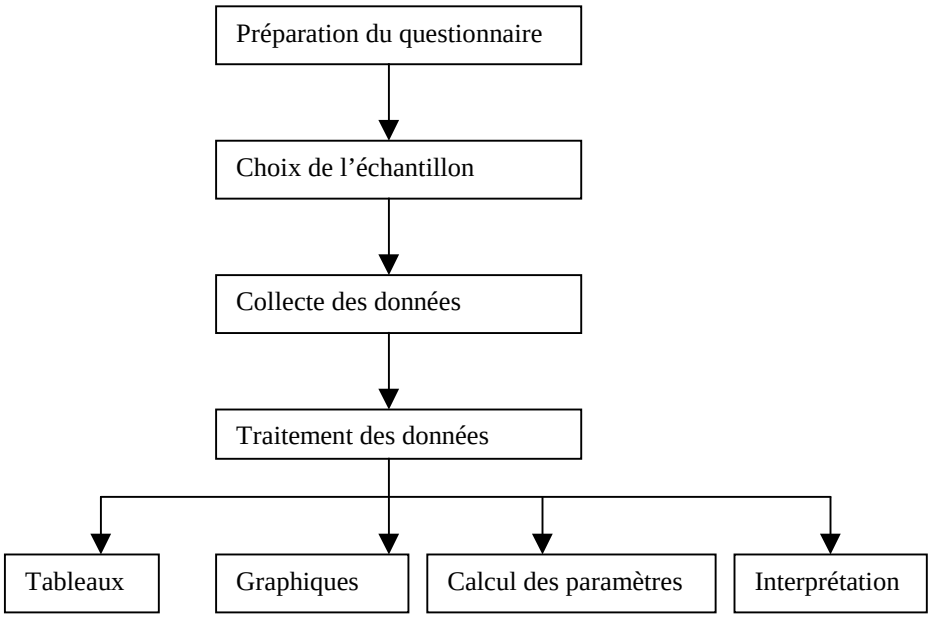


Figure 2.3 Les étapes d'une analyse statistique

Il faut déterminer les résultats à recueillir, comment on va les recueillir et quels critères serviront à l'interprétation. Il est essentiel de déterminer avant l'expérience les critères permettant d'interpréter les résultats. Il est à noter que les critères permettant d'analyser les résultats sont connus avant la cueillette des données.

Recueillir les données représente une étape longue et laborieuse dans la plupart des expériences statistiques. Dans notre cas, la cueillette de données est faite avec le logiciel Venturi depuis plus de 8 ans.

Analyser les résultats, tirer des conclusions et déterminer la fiabilité des résultats représente la dernière étape d'une analyse statistique.

Il est important de déterminer la fiabilité des résultats sinon le rapport statistique est sans signification et n'a aucune valeur scientifique. Les progiciels actuels sont de type « boîte noire » ce qui limite l'analyse des résultats et ne permet pas d'en déterminer la fiabilité.

Les méthodes d'analyse statistique utilisées sont presque toutes des méthodes « univariées » dans le sens qu'elles permettent d'étudier les échantillons en fonction d'une seule variable ou caractéristique. Cependant dans nos analyses on est intéressé à étudier plusieurs variables qui ont plusieurs caractéristiques des unités de cet échantillon. Une telle suite des analyses statistiques univariées ne peut pas permettre d'expliciter les relations existantes entre ces variables : seules les méthodes statistiques dites « multivariées » permettent d'étudier des populations considérées simultanément ou conjointement et de vérifier l'existence des relations pouvant exister entre ces variables.

Par régression on entend un modèle mathématique cherchant à expliquer la variabilité d'un phénomène mesurable par celle d'autres facteurs également mesurables, et choisis pour leur valeur explicative. Le modèle le plus simple et le plus utilisé est le modèle linéaire qui présente un grand intérêt théorique et pratique.

Dans une étude de régression au niveau mathématique il faut trouver la spécification du modèle, faire l'estimation du modèle, le vérifier et ensuite l'appliquer.

Le modèle de régression le plus simple est celui où on utilise une seule variable explicative X pour expliquer les variations de la variable Y et où la relation qu'on essaye de construire entre X et Y prend la forme d'une droite: on parle alors de modèle de régression linéaire simple. La relation fonctionnelle de type statique est définie généralement par une équation de la forme suivante: $Y=ß_0+ ß_1X+E$ où Y est la variable dépendante ou expliquée à caractère aléatoire, X est la variable indépendante ou explicative, $ß_0$ et $ß_1$ sont les coefficients de la régression théorique et E est une variable aléatoire qui prend en compte l'existence d'autres influences que X ou Y.

Pour construire le modèle de régression il faut sélectionner un échantillon parmi la population. On détermine ensuite b_0 et b_1 qui correspondent aux coefficients de régression empiriques servant à estimer les coefficients de régression théorique $ß_0$ et $ß_1$. Le modèle de régression linéaire empirique

prend la forme suivante: $Y= b_0 + b_1X$. On cherche à obtenir la droite de régression empirique qui s'ajuste le mieux à l'ensemble de l'échantillon. La droite déterminée s'appelle la droite des moindres carrés.

On évalue par la suite la validité de la droite de régression empirique obtenue par la méthode des moindres carrés. Il faut que les hypothèses de base soient respectées, est-ce qu'elles sont vraiment ? Une façon de le déterminer est d'étudier les comportements des erreurs empiriques ou résidus.

Une fois qu'on s'est assuré de la validité du modèle de régression linéaire simple on peut utiliser ce modèle. La construction d'un modèle a une grande importance pour la prise de décision. Lorsqu'on se prépare à prendre une décision, on doit fréquemment effectuer quelques prévisions conditionnelles, et dans cette perspective, les modèles de régression peuvent être très utiles. Ils permettent d'expliquer la variabilité d'un phénomène mesurable par celle d'autres facteurs également mesurables choisis pour leur valeur explicative.

2.2 Le langage de programmation utilisé pour la réalisation de l'outil

Le langage de programmation utilisée a été Visual Basic Application (VBA) qui est un produit relativement récent, dont les racines sont ancrées dans la préhistoire de l'informatique. C'est une des multiples évolutions du langage BASIC (Beginner All Purpose Symbolic Instructions Code) qui a vu le jour dans les années 60. Le BASIC a eu beaucoup de succès parce qu'il était simple à comprendre et facile à apprendre. Les premiers PC construits par IBM comportaient une version du langage Basic intégrée en mémoire morte. En 1992 Visual Basic pour Windows apparaît sur le marché avec son environnement de programmation entièrement graphique et qui offrait toutes les fonctions dont on a besoin pour la mise au point des applications Windows.

Les besoins d'automatisation dans les applications Office ont été en forte demande ce qui a poussé Microsoft à concevoir une version spéciale du Visual Basic destinée spécialement pour les applications Office qui a été intégrée pour la première fois dans Excel 5, sous le nom de Visual Basic Application. Dans la version actuelle VBA est devenu un environnement de programmation à part

entière doté des mêmes structures de langage, des mêmes outils d'édition du code et des mêmes possibilités de programmation que Visual Basic 4.0. VBA est fondé sur le principe que les applications manipulent des objets et qu'elles sont elle-mêmes constituées d'éléments auxquels on peut accéder comme des objets. Chaque objet possède un jeu de propriétés et on peut lui appliquer des méthodes. Une plage de cellules devient un objet, et c'est la même chose pour un tableau croisé dynamique ou un graphique. Leur propriétés peuvent être lues ou modifiées.

Parmi les raisons pour lesquelles nous utilisons VBA on peut énumérer : les fonctions automatisée de l'outil qui représente les raisons principales pour l'utilisation du langage VBA, la gestion de la base de données avec l'utilisation des objets serait bien améliorée, la facilité de manipuler les graphiques en tant qu'objet etc.

2.3 Le projet dans le programme de recherche du Laboratoire de recherche en gestion des logiciels

Les quatre axes de recherche du laboratoire sont : l'efficience et l'efficacité des programmes de mesure, les points de fonction (expérimentation et instrumentation), la mesure de la maintenance du logiciel et la réutilisation fonctionnelle. Notre projet de recherche s'insère dans les axes de recherche suivantes : « L'efficience et l'efficacité des programmes de mesure » et dans l'axe « Les points de fonction, (expérimentation et instrumentation) ».

Ayant défini la méthode qui a été utilisée, nous nous proposons de préciser, dans la prochaine section un sommaire des résultats atteints.

3. PRÉSENTATION ET ANALYSE DES RESULTATS

3.1 Les étapes de l'estimation avec l'environnement automatisé

Pour l'analyse d'un projet les étapes sont fonction des besoins et de l'expérience de l'estimateur. En ce qui concerne l'utilisation de l'environnement automatisé développé dans le présent travail nous avons identifié 6 étapes qui sont généralement valables :

1. Identifier le contexte du développement du projet : le langage de programmation, la plate-forme de développement, les méthodes utilisées etc.

2. Fixer soit la taille ou soit la durée du projet comme point de départ. Dans le menu « Simulation » il y a la possibilité de trouver la durée à partir de la taille comme constante ou la taille à partir de la durée comme constante.

3. Commencer l'analyse avec le menu « Database_ISBSG » pour réussir à déterminer des échantillons susceptibles d'être utilisés dans une analyse.

4. Analyser les statistiques descriptive du menu « Evolution » pour mieux connaître les échantillons susceptibles d'être utilisé dans l'analyses et trouver éventuellement d'autres alternatives.

5. Construire des modèles de productivité et d'estimation dans le menu « Simulation ».

6. Faire des analyses de la qualité des modèles et des résultats produits par ces modèles dans le menu « Simulation ».

Ayant identifié les étapes de l'estimations avec l'outil nous faisons maintenant le liaison avec le processus d'estimation qui a été présentée dans le sous-chapitre 2.1.1 et on spécifie pour chaque module les étapes correspondantes.

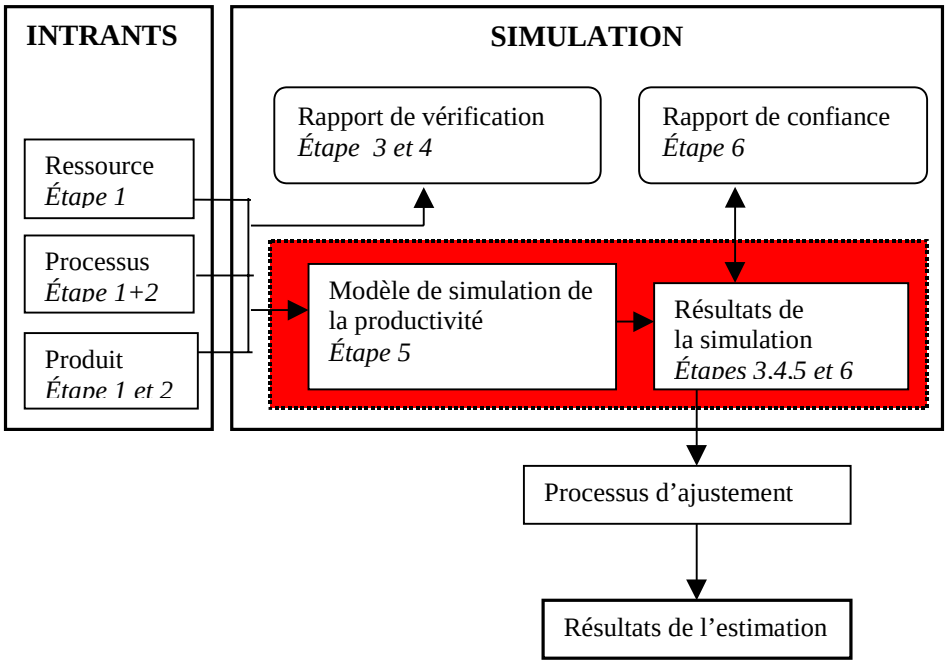


Figure 3.1 Les étapes de l'estimation par rapport au processus transparent d'estimation

3.2 Présentation de résultats

L'environnement automatisé contient trois menus : "Database_ISBSG", "Evolution" et "Simulation". Il n'y a pas de séquence obligatoire de menus dans l'environnement automatisé pour réaliser une estimation. Le chef de projet pourrait commencer son analyse dans n'importe quel menu et utiliser les fonctions du logiciel à sa manière.

Dans le menu « Database_ISBSG » il y a la possibilité de faire des recherches dans la base de données selon plusieurs combinaisons de conditions sur les attributs existants et aussi la possibilité de choisir seulement les attributs qui nous intéressent. Son rôle est de nous permettre de mieux comprendre le contenu de la base de données et de nous guider dans le choix des échantillons qui seront utilisés dans les analyses subséquentes.

Dans le menu « Evolution » on peut effectuer des analyses statistiques descriptives pour presque chaque attribut existant dans la base de données ou pour diverses combinaisons d'attributs et selon les conditions spécifiées par l'utilisateur. Ces analyses statistiques descriptives nous guident encore une fois dans le choix des échantillons pour les analyses subséquentes et nous aident à comprendre les forces et faiblesses des échantillons sélectionnés.

Le menu de « Simulation » permet de développer des modèles de productivité d'une façon dynamique. À l'aide de l'analyse de régression, on peut estimer la valeur des paramètres importants existants dans la base de données et nous pouvons simuler différents scénarios. Les équations et les détails des paramètres sont tous affichés sur les graphiques produits par l'outil. Une fois obtenu un modèle justifiable théoriquement et vérifié statistiquement on peut prévoir le coût de réalisation d'un nouveau projet. Un tableau avec les principaux indicateurs statistiques (le minimum, le maximum, la moyenne, la médiane, les quartiles, l'écart-type, l'asymétrie, la variance, l'intervalle de confiance, kurtosis etc.) est dynamiquement affiché pour chaque scénario, ainsi que le graphique du nuages de points et celui de la distribution des valeurs. L'analyse de « benchmarking » est un bon mécanisme de comparaison par rapport à d'autres projets, ce qui pourrait être un élément motivateur pour une équipe de développement qui se situe parmi les meilleurs 25 % ou un stimulant pour les autres équipes qui

veulent être parmi dans les 25% les plus performants. Avec l'analyse de benchmarking les gestionnaires seront en mesure d'effectuer des choix plus réalistes au niveau des ressources et des performances attendues.

Étant donné qu'il y a un grand nombre des graphiques nécessaires pour la présentation des résultats nous avons considéré qu'il serait plus facile de les présenter en utilisant les avantages offerts par un éditeur HTML : nous avons donc développé un site WEB à cette fin qui se trouve à l'adresse suivante: http:\\www.lrgl.ca\personal/stroian/isbsg.htm

Dans ce site WEB qui fait partie intégrante de ce rapport d'activité de synthèse nous avons inclus un exemple concret (d'un développement de projet) et on explique les fonctions de chaque menu. Finalement nous y avons identifié les composantes du processus d'estimation tel que présentées dans la section 2.1.1. (voir **Appendice A** à **D**).

3.3 CADRE DE BASILI

Le cadre expérimentale de Basili a été utilisé par les auteurs pour présenter un sommaire de la méthodologie de recherche et des résultats qui en découlent. Dans le **Tableau 3.1** nous présentons le cadre de Basili utilisé pour la réalisation de cette activité de synthèse.

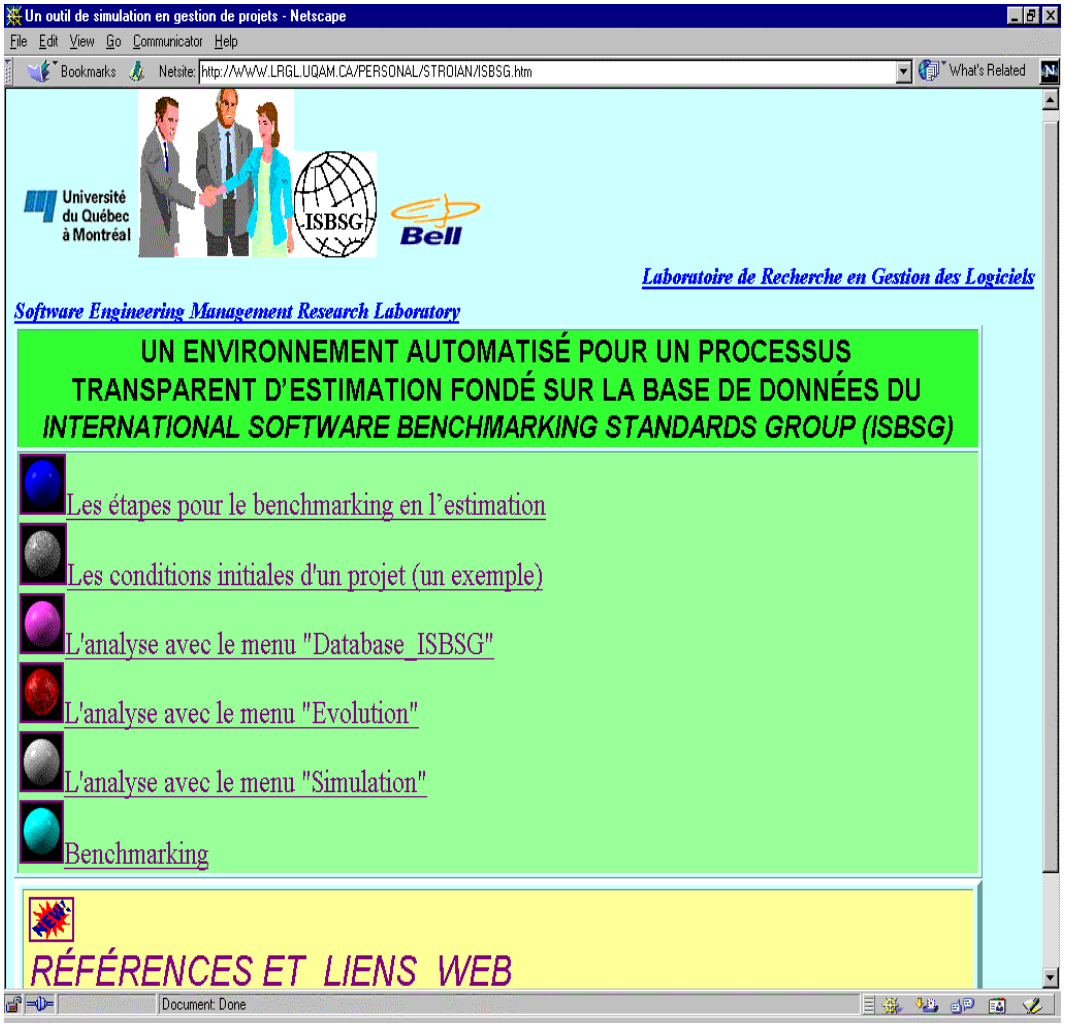


Figure 3.2 Présentation de résultats sur le site WEB à l'adresse suivante:
http://www.lrgl.uqam.ca/personal/stroian/isbsg.htm

Tableau 3.1 Cadre de Basili modifié pour cette activité de synthèse

Définition			
Motivation	Objet	But	Utilisateurs
Aider à améliorer le processus d'estimation des projets informatiques.	Un processus transparent pour l'estimation des projets informatiques.	Développer un environnement automatisé pour un processus transparent d'estimation.	Chefs de projet informatique Gestionnaires de programme de mesure Chercheurs en génie logiciel ISBSG et LRGL

Planification		
Étapes du projet	Intrants au projet	Livrable du projet
Développer un prototype : Database_ISBSG Évolution	Revue de littérature Le langage VB La base de données ISBSG	Prototype 0.1
Réviser le prototype	Prototype 0.1	Prototype 1.0
Réalisation de l'environnement automatisé : Database_ISBSG Évolution Simulation	Prototype 1.0 Le langage VB La base de données ISBSG Revue de littérature	L'environnement automatisé de l'estimation
Rédaction du rapport final Préparation de l'exposé Dépôt du rapport	Revue de littérature	Rapport Site WEB

4. CONCLUSIONS

Ce chapitre conclut la présente activité de synthèse et une réflexion personnelle sur le déroulement de cette étude est ajoutée.

Ce travail vous a présenté le développement d'un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données transparente de l'International Software Benchmarking Standards Group (ISBSG).

Le développement de logiciel est plus complexe parce que nous nous ne comprenons pas clairement le processus de développement. Le logiciel est intangible, on ne peut pas le voir ni le toucher et le chef de projet doit s'en remettre à la documentation pour évaluer le progrès d'un projet de développement.

Plusieurs études ont démontré que la plupart des projets informatiques ont un dépassement de budget ou sont en retard ou ont livré moins de fonctionnalités que prévues initialement (ce qui produit des pertes importantes pour les entreprises).

La décision de commencer ou de continuer un projet est souvent influencée par les estimations d'effort de développement. Malgré l'importance de bien estimer l'effort, les gestionnaires ne possèdent pas d'outils capables de leur fournir l'information crédible pour cette tâche. Il en résulte que les estimations sont loin de la réalité.

Les bases de données historiques à l'appui des progiciels d'estimation ne sont pas accessibles, ce qui ne permet pas de déterminer la crédibilité des résultats et du processus d'estimation. Les modèles d'estimation sont préprogrammés dans les progiciels et les utilisateurs n'ont pas d'information sur la structure des modèles d'estimation utilisés et ils ne peuvent pas connaître le type de modèle ou la qualité statistique des résultats.

Le résultat d'un processus d'estimation ne doit pas être uniquement un nombre parce que les gestionnaires doivent prendre des décisions importantes en se basant sur une information crédible.

L'environnement automatisé développé dans le présent travail permet aux gestionnaires d'utiliser un processus transparent d'estimation, ce qui leur permet de mieux comprendre le contexte des résultats obtenus.

La problématique de recherche que nous avons proposée de résoudre est la suivante : comment transformer un processus « boîte noire » d'estimation en un processus « boîte transparente » fondé sur la base de données de l'International Software Benchmarking Standards Group (ISBSG).

Avec l'environnement automatisé développé, nous avons réussi á transformer un processus d'estimation « boîte noire » en un processus « boîte transparente » ce qui pour les chefs de projet informatique, les gestionnaires de programme de mesure et les chercheurs en génie logiciel représente un outil mieux adapté à leurs besoins.

Il serait intéressant que d'autre chercheurs ou praticiens poursuivent la recherche et le développement en ce qui concerne l'environnement transparent d'estimation pour développer d'autres modèles de productivité et utiliser d'avantage les fonctions statistiques.

Le **Tableau 4.1** montre les principales étapes de l'activité de synthèse. Les informations sur le « réel » sont approximatives mais confirme que l'effort de développement du logiciel a été sous-estimé.

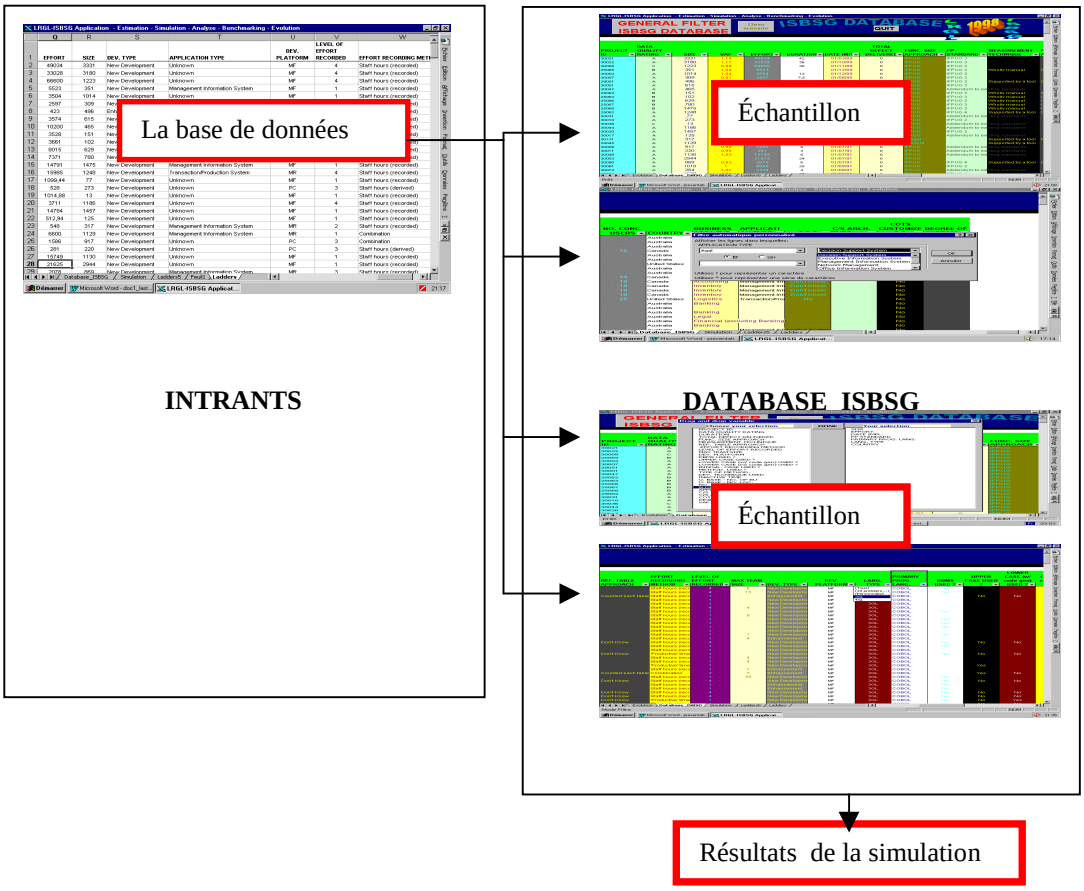
Finalement le travail m'a permis à comprendre le processus d'estimation et de faire une analyse plus en profondeur sur plusieurs aspects statistiques ce qui ma aidé sur le marché de travail. Mes objectifs personnels ont été atteints.

Tableau 4.1 Échéancier de l'activité de synthèse

ID	Nom de la tâche	Durée prévue (heures)	Durée réelle (heures)
1	Revue de la littérature	292	350
2	Développer un prototype	150	250
3	Réviser le prototype	20	20
4	Réalisation de l'environnement automatisé	200	350
5	Rédaction du rapport final	20	60
6	Préparation de l'exposé	10	10
7	Dépôt du rapport	8	8
	TOTAL	700	1048

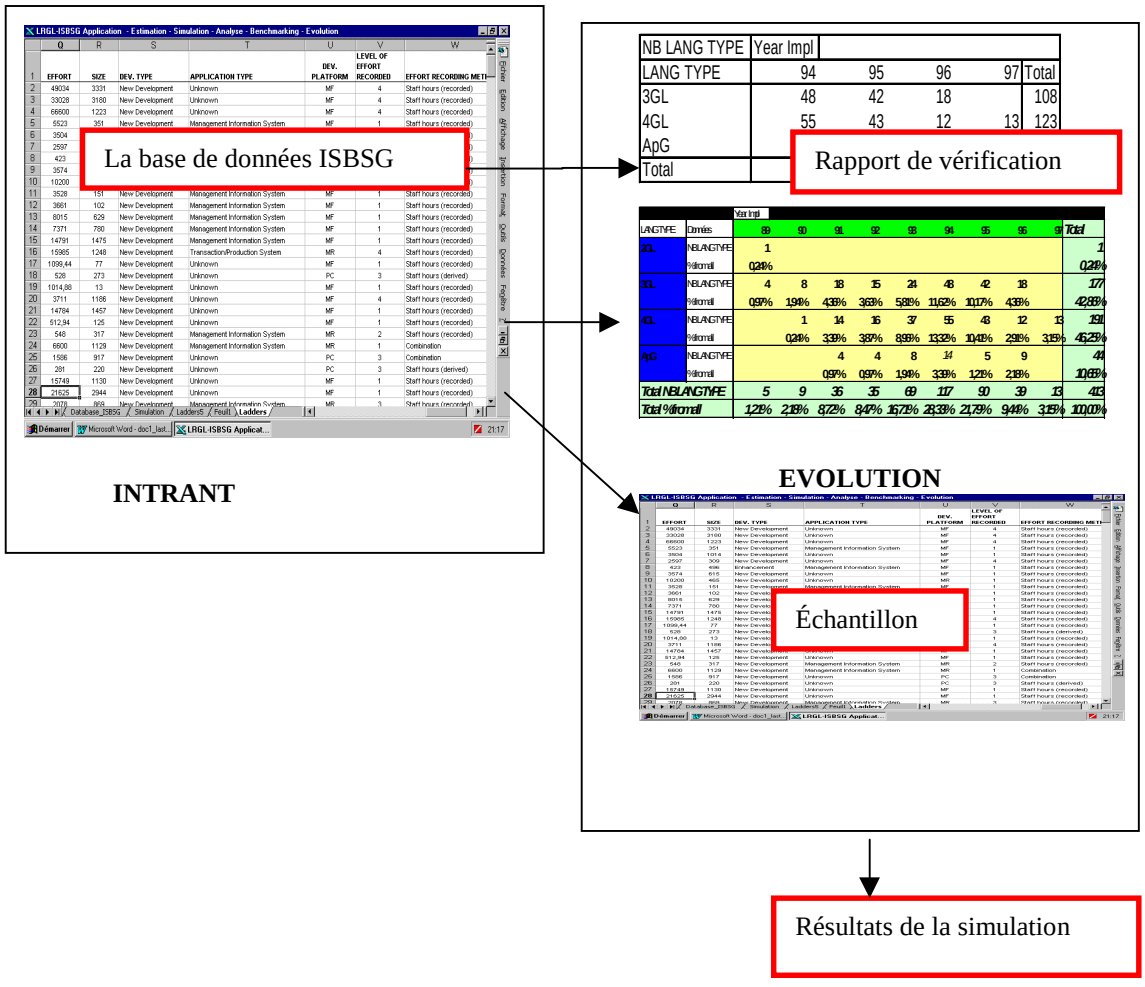
APPENDICE A

LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU DATBASE_ISBSG



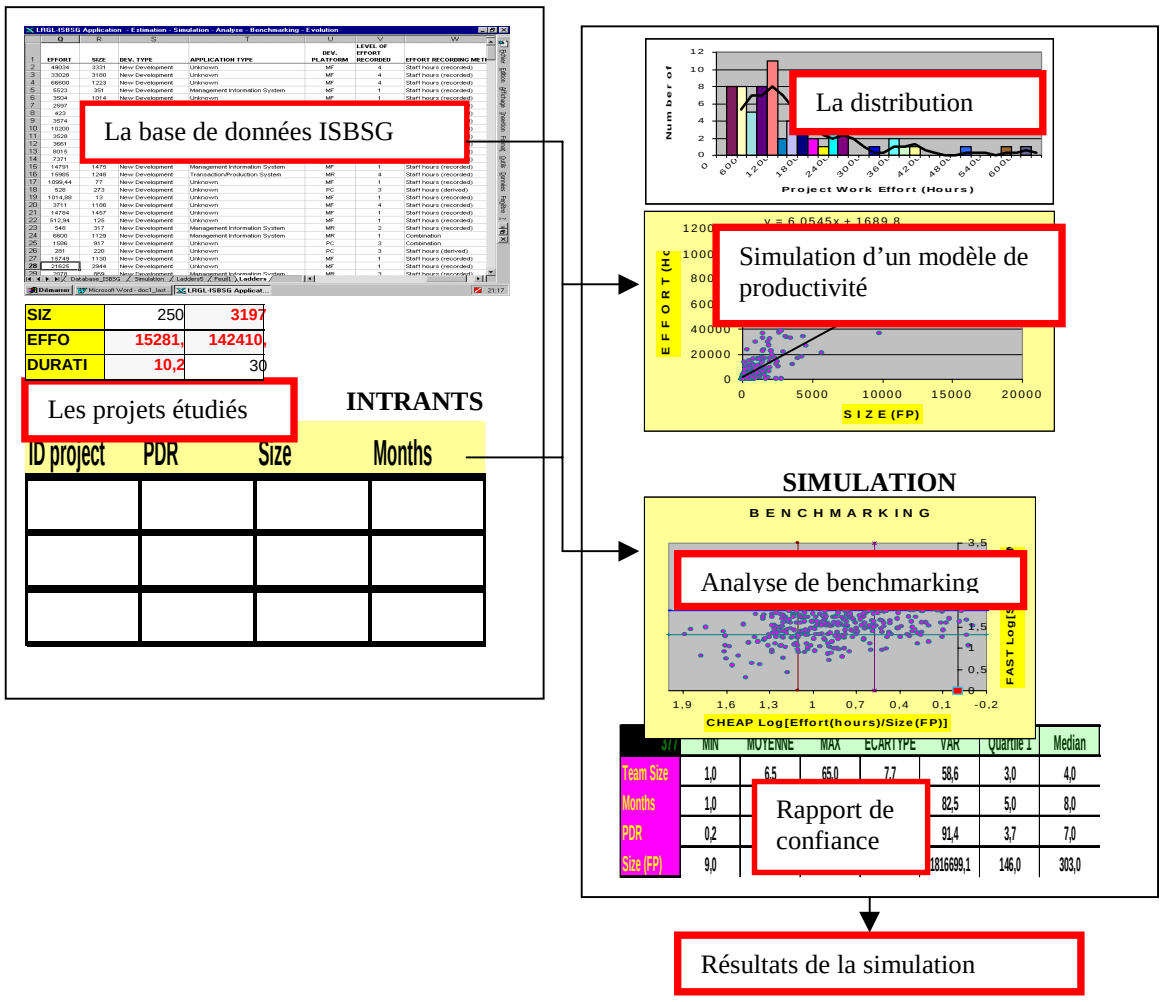
APPENDICE B

LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU EVOLUTION



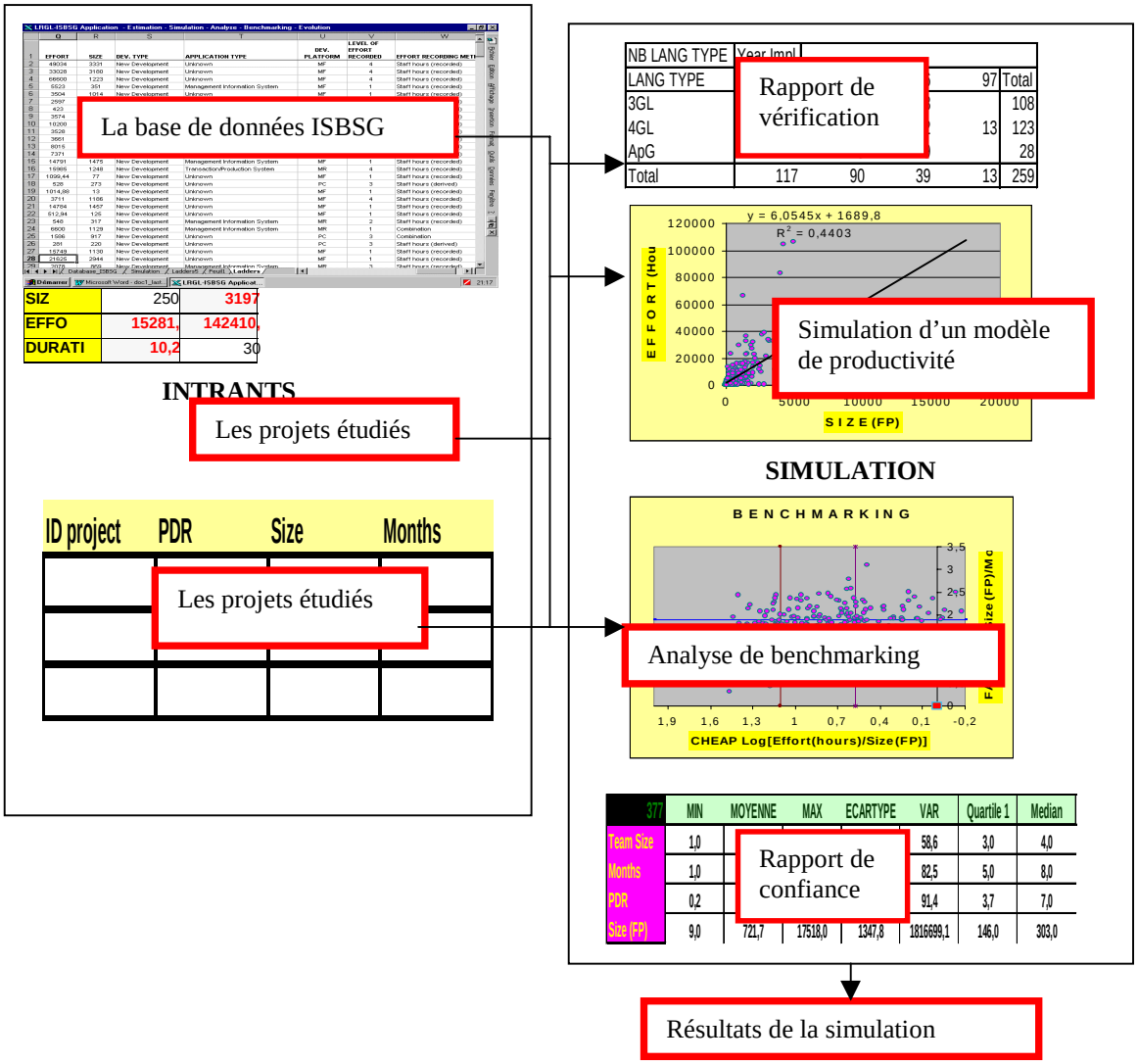
APPENDICE C

LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC LE MENU SIMULATION



APPENDICE D

LES COMPOSANTS D'UN PROCESSUS TRANSPARENT D'ESTIMATION SUITE À UNE ANALYSE AVEC L'OUTIL



RÉFÉRENCES

Abdel-Hamid, T.K. 1991. *Software Projet Dinamics : an integrated approach.* Englewood Cliffs, N.J., Prentice Hall, p. 227.

Abran, A. 1995. « Function Point-Based Production Models ». *Metrics in Software Evolution*. Vol. *GMD-Bericht* NR. 254, p. 223-246.

Abran, A. 1994. « Analyse du processus de mesure des points de fonction ». *Département de génie électrique et de génie informatique*. Montréal : École polytechnique de Montréal, p. 405.

Abran A. et J.M. Desharnais, 1996. « A strategy for a credible & audible estimation process ». Note de cours INF-7760, p. 14.

Abran A. et J.M. Desharnais. 1995. « Implanter un programme de mesures en génie logiciel : des étapes pour éviter les risques d'échec ». *L'expertise informatique*, Vol. no. 3.

Abran, A. et P.N. Robillard, 1994. « Function Points : A Study of their Measurement Processes and Scale Transformations ». *Journal of Systems and Software*, Vol. 25, no 2, p. 171-184.

Abran, A. et P.N. Robillard, 1993. « Reliability of Function Points Productivity Model for Enhancement Projects ». Presented at *Conference on Software Maintenance*, Montreal, Quebec, Canada.

Abran, A. et P.N. Robillard., 1993. « Analyse Comparative de la fiabilité des points de fonction comme modèle de productivité ». *ICO* Vol.4, no 3-4.

Albrecht, A.J. et J.E.Gaffney. 1983. « Software Function, Source Lines of Code, and Developmentffort Prediction : A Software Science Validation ». *IEEE Transaction on Software Engineering*., Vol.SE-9, no.6, p. 639-648.

Baber, R.L., 1982. *Software Reflected.* New York, NY : North Holland Publishing Company.

Basili et al, 1986. « Experimentation in software engineering », *IEEE Transactions on software engineering*, Vol. SE-9, No.7 p.733-743

Benbasat, I. and I. Vessey. 1980. « Programmer and Analyst Time/Cost Estimation. ». *MIS Quaterly*, Vol.4, No.2, p. 31-43.

Boehm, B. W., 1981. *Software Engineering Economics*.. Englewood Cliffs, NJ : Prentice Hall,Inc.

Boehm, B. W. et *al*. 1995 « Cost Models for Future Software Life Cycle Processes : Cocomo 2.0 ». *Annals of Software Engineering Special Volume on Software Process and Product Measurement.*The Netherlands : Science Publishers.

Bourque, P. 1988. «Développement d'un modèle statistique d'estimation du nombre de lignes de code fondé sur des métriques de spécification ». *Master's Thesis*, Université de Sherbrooke, p. 138.

Bourque, P. et V. Côté,. 1991. « An Experiment in Software Sizing with Structured Analysis Metrics ». *Journal Systems Software*, p. 159-172.

Brooks, F.P. 1987. « No silver bullet : essence and accidents of software engineering ». *IEEE Computer*, April, p 10-19.

Carr, M. 1997. «Software effort and schedule estimation : a case study ». WP97/02 *Departement of Information Systems*, City University of Hong Kong, p. 22.

Côte, V. et al., 1988. « Software Metrics : An overview of recent results ». *Journal Systems Software*, No.8, p : 121-131.

DeMarco, T. 1974. *Controlling Software Projects.* New York, NY : McGraw-Hill

DeMarco, T. 1982. *Controlling Software Projects : Management, Measurement & Estimation.* Yourdon Press. Computing Series.

Denis, C. et D. Kinlaw Ed,. 1992. *Continuous Improvement and Measurement for Total Quality*. Pleiffer Company, p. 251.

Desharnais J-M. 1988. « Analyse statistique de la productivité de projets de développement en informatique à partir de la techniques des points de fonctions », *Master's Thesis*, UQAM.

Driscoll, A.J., 1977. « Software visibility and the program manager ». *Defense Systems Managements Review*, p : 12-17.

Dupuis, R, 1994. *Méthodologie de la recherche appliquée MIG 9100.* Notes de cours, Université du Québec à Montréal, Département d'informatique, p. 138.

Fenton, N. 1994. «Software Measurement : A necessary scientific basis ». IEEE Transaction on Software Engineering, Vol.20, no.3, p.1999-206

Fishwick, P. A. et P. A. Luker. 1991. *Simulation Modeling and Analysis*, Springer Verlag.

Fishwick, P. A. and B. P.Zeigler, 1992. « A Multimodel Methodology for Qualitative Model Engineering » *ACM Transactions on Modeling and Computer Simulation*, Vol.2. no.1, p. 52-81.

Finnie G.R. et al. 1997. « A comparaison of software effort estimation techniques : Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models ». *Journal of Systems Software*, No. 39 , p. 281-289.

Garmus, G. et D. Herron. 1996. « Effective Early Estimation ». *Software Development*, July.

Genest, B. A. 1996. *Technologies et systèmes d'information dans l'entreprise.* Les Éditions Sigma Delta, Vol 2.

Genest, B.A. et T.H. Nguyen. 1995. *Principes et techniques de la gestion de projets.* Les Éditions Sigma Delta .

Glass, R. 1991. *Building Quality Software*. Prentice Hall.
Gilbert, N. 1978. *Statistiques*. Les Éditions HRW Ltée, Montréal p.384.

Gulledge T.R. et al. 1997. « Estimation problems in rate-augmented learning curves ». *IEEE transaction on engineering management* . Vol. 44, no. 1, p. 91-98.

Heiat, A et N. Heiat. 1997. « A model for estimating efforts required for developing smale-scale business applications ». *Journal Systems Software,* Vol.39, p. 7-14.

Host, M. et C. Wohlin. 1997. « A subjective effort estimation experiment ». *Information and Software Technology,* Vol.39, p. 755-762.

Hormer, J.B. 1983. *A Dynamic Model for Analyzing the Emergence of New Medical Technologies.* Unpublished Ph.D. disertation MIT Cambridge MA.

Hotle, M. 1996. « Understanding and improving the estimation process Application Developpement & Management Strategies (ADM) ». *Strategic Analysis Report Gartner- Group,* p.31.

Ingram, T., 1994. «Managing Client/Server and Open Systems Projects : A 10 Year Study of 62 Mission-Critical Project ». P*roject Management Journal* . Juin.

International Software Benchmarking Group (ISBSG). 1997. *Workshop documentation Release – 4*. Melbourne, Australie.

Jones, C. 1991. *Applied Software Measurement : Assuring Productivity and Quality.* New York : McGraw-Hill.

Jones, C. 1994. *Assement and Control of Software Risk.* Englewood Cliffs. N.J, Yourdon press.

Jones, C. 1996. « Our worst current development practices ». *IEEE Software,* 13 p. 102-104.

Jones, C. 1997. *Applied Software Measurement : Assuring Productivity and Quality.* New York : McGraw-Hill.

Knepell, L. P. et D.C. Aragno. 1993. « *Simulation Validation a Confidence Assement Methodology* ».

Kleijnen, J.P.C. et Willem van G. 1992. *Simulation : a statistical perspective.* J. Wiley, p. 241.

Kitchenham, B. et K. Kansala. 1993. « Inter-item Correlations among Function Points ». IEEE, No.4.

Kitchenham, B.A. 1996. « Estimation-a personal view ». *Proceedings of the 7th European Software Control and Metrics conference*, Wilmslow, UK, p. 185-189.

Lederer, A. et J. Prasad. 1993. «Systems Development And Cost Estimating Chalenge and Guidelines ». *Information Systems Management,* Fall.

Lederer, A. et J. Prasad, 1993. « Information systems cost estimating ». Journal of Information Technology, Vol. 8 p. 22-33.

Lessard, S. et *al*, 1993. *Statistique concepts et méthodes*. Masson éditeur, p.421.

Londeix, B. 1997. « Three-Point Estimation Techniques ». *SEER Technologies* p. 27

Londeix, B. 1987. *Cost Estimation for Software Development*, Addison-Wesley, Workingam. UK.

Louis, M.T., W. Borchering, et W.R. Hudgings, 1991. « Estimeetings :Development Estimates and a Front –End Process For a Large Project ». IEEE, Vol. 8.

McConnell, S. 1995. *Rapid Development*. Microsoft Press.

MacDonell, S.G. 1994. « Comparative review of fonctional complexity assessement methods for effort estimation ». *Software Engineering Journal*, May.

Martel, J-M, N. Nadeau. 1988. *Statistique en gestion et en économie*. Gaétan Morin Éditeur, Montreal, p. 621.

Matson, J.E. , B.E. Barrett, et J.M. Mellichamp. 1994. « Software Development Cost Estimation Using Function Points. IEEE Transaction on Software Engineering ». Vol 20, no. 4, p. 275-287.

Mertes, K.R. 1996. « Calibration of the checkpoint model to the space and Missile Systems Center (SMC) Software Database ». *Master's Thesis*, Air Force Institute of Technologie, Ohio, p. 119.

Myers, B. 1989. « Allow plenty of time for large-scale software ». *IEEE Software*, Vol. 1. p. 92-98.

Osborne, M. R. et Watts, R. O. 1977. *Simulation and modelling*, Prentice-Hall International p. 208.

Oligny S, P. Bourque. A. Abran. et B. Fournier. 1997. « Developing project duration models in software engineering ». Université du Québec á Montréal. p. 10.

Oligny, S, P. Bourque. et A. Abran. 1997. « An empirical assessement of project duration in software engineering ». *Presented at The Eight European Software Control and Metrics Conference (ESCOM'97)*, Berlin, Germany.

Praehofer, H. 1991. « Systems Theoretic Formalisms for Combined Discrete-Continuous System Simulation ». *International Journal of General Systems*, Vol. 19, no.3, p. 219-240.

Park, R.E. et B. Korda,. Peter1989. « Making Estimates Credible-or, Why Should I believe What You are Telling Me ? ». *Journal of Parametrics*, Vol 9, no. 2.

Park, R.E. et al. 1994. « Software cost and schedule estimating : a process improvement initiative » *Software Process Measurement Project*, Software Engineering Institute, Special Report, p. 95.

Putnam, H.L. et al. 1997. *Industrial Strength Software Effective Management Using Measurement. IEEE Computer Society*, p.309.

Ramsey, C. L. et V.R Basili. 1989. « An evaluation of Expert Systems for Software Engineering Managenment ». *IEEE Transaction on Software Engineering*, vol 15. No.6.

Shoval, P. et *al.* 1997. « A combination of the Mk—II Function Points software estimation method with the Adidda methodology for systems analysis and design ». *Information and Software Technology* 39, p. 855-865.

Synott, W.R. et *al*. 1981. *Information Ressource Management*. New York, NY : John Wiley & Sons.

Symons, C.R. 1988. « Function-point analysis : difficulties and improvements ». *IEEE Trans. Software Enginneering*, Vol. 14, no. 1, p. 2-11.

Steven, V.S. 1996. « Calibration of the Softcost-R Software Cost Model to the space and Missile Systems Center (SMC) Software Database ». *Master's Thesis*, Air Force Institute of Technologie, Ohio, p. 86.

Taff, L.M. et *al.* 1991. « Estimeetings : developpement estimates and a front-end process for a large project ». *IEEE Transaction on Software Engineering (ISO)*, Vol. 17, no 8, p. 839-849.

Thayer, R.H. 1979. « Modelling a Software Engineering Project Management System ». Unpublished Ph.D. dissertation, University of California, Santa Barbara, CA.

The Standish Group. 1995. « CHAOS ». http://www.standishgroup.com/chaos.html.

van Genuchten, M. et H. Koolen. 1991. « On the Use of Software Cost Models ». *Information & Management*, Vol. 21, p. 37-44.

Waterson, K. 1994. « The Changing World of EIS », Byte, Vol. 19, no. 6, p. 183-193.

Watson , I. et F. Marir, , 1994. « Case-based Reasoning : A Review ». *The Knowledge Engineering Review*, Vol. 9. no.4, p. 327-354.

Weiss, D.M. 1979. « Evaluation Software Decveloppement by Error Analysis ». *Journal of Systems and Software*, Vol.1, p. 57-70.

Wheeler, D.J. et D.S Chambers. 1986. « Understanding Statistical process and control ». Knoxville.

www.asma.org.au/a_fevent.htm, 1998.

www.lrgl.uqam.ca/personal/stroian/isbsg.htm, 1999.

www.gartnergroup.com, 1998

www.spr.com, 1998.