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)

PROPOSITION DE L'ACTIVITÉ DE SYNTHÈSE

PRÉSENTÉE

AU SOUS-COMITÉ D'ADMISSION ET D'ÉVALUATION

MAÎTRISE EN INFORMATIQUE DE GESTION

PAR

VASILE STROIAN

JUIN 1998

TABLE DES MATIÈRES

LISTE DES FIGURES ET DES TABLEAUX

Figure Page

Tableau Page

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 existants on remarque l'inaccessibilité à leur base de données ce qui a une influence directe sur la fiabilité de l'information. Le processus d'estimation doit fournir aux managers plus qu'un chiffre, il doit fournir l'information nécessaire pour prendre les meilleures décisions et ils se posent toujours la question de la crédibilité de l'information Selon Louis en estimation il y a toujours une question obsédante pour les gestionnaires « how accurate were the estimates ? » (Louis et al.,1991).

Ce document présente la proposition de l'activité de synthèse dont l'objectif est de développer 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 deux grandes sections. La première expose la problématique à partir d'une revue de la littérature en précisant aussi les objectifs et les limites de notre recherche. La deuxième présente le cadre conceptuel que nous avons adopté et les aspects méthodologiques de la démarche.

La recherche a débuté en été 1997 et sera terminée au mois de novembre 1998.

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

1. LA 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). L'estimation peut être divisée en deux grande 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 déterminée on détermine 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 connaissons pas clairement le processus de développement de logiciel. Dans le processus de l'ingénierie plus traditionnel on peut prédire tous 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 estimer le progrès du 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 commun : 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 système 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 52.7% ont un dépassement de budget ou sont en retard ou ont livré moins de fonctionnalité que prévu initialement. Selon l'étude les dépenses dans le développement de logiciel ont été de plus de 250 milliards $ pour un nombre approximatif 175,000 projets. Seulement 16.2% finissent leur projet à temps et à l'intérieur de budget alors que pour les grandes compagnies il y a seulement 9% donc moins de 1 projet sur 10!. 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 $. Presque la moitié de managers, 48%, pensent que la situation était plus difficile qu'en 1990.

Plusieurs 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; Standish International Group, 1998). 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).

Il y a une littérature abondante sur le « syndrome de 90% » (Baber, 1982; DeMarco, 1982, Synot, 1981) et Baber a donné l'explication suivante :

« [...] estimates of the fraction of work completed (increase) as originally planned until a level of about 80-90% is reached. The programmer's individual estimates then increase only very slowly until the task is actually completed ».

Plusieurs entreprises s'inspirent d'une méthode basée sur l'engagement, qui consiste à demander aux développeurs de s'engager sur une échéance plutôt que d'en faire une estimation rigoureuse (McConnell, 1995). Ils sous-estiment les échéanciers de 20 à 30% (van Genunchten, 1991). L'engagement personnel des développeurs affecte leur estimation d'optimisme et de subjectivité et tend à produire des erreurs d'estimation importantes.

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 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.

Une estimation trop faible peut amener la direction à investir considérablement dans un projet informatique lorsqu'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).

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 doit être basée 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 détient 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.

Hotle mentionne qu'en 1996 la plupart des organisations débutent leurs initiatives en estimation par l'achat d'un progiciel d'estimation. Le marché des outils d'estimations est restreint (Hotle, 1996). Le chiffre total d'affaires ne représente que 5-10 millions $ et presque la moitié est représentée par CHECKPOINT. Parmi les problèmes majeurs des outils on remarque un manque de cohésion, la complexité de l'outil et l'inaccessibilité à leur base de données (Hotle, 1996).

Les organisations qui utilisent les données offertes du domaine commercial rencontrent plusieurs problèmes. Ces types de données représentent une collection d'un large éventail d'industries et sans procédures indépendantes de vérification du processus de collecte de données. Plusieurs firmes affirment avoir de milliers de projets dans leurs banques mais refusent systématiquement à donner des informations précises (Garmus et al., 1996) sur les attributs de leurs données et leurs procédures de validation.

1.3 Les progiciels d'estimations

Les progiciel d'estimation permettent d'estimer l'effort à partir de paramètres comme la taille des différents types de projets, la taille et la composition de l'équipe de développement, la durée de projets 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 a été SPQR/20 (www.spr.com, 1998) en 1985. Un décennie plus tard la grande majorité des outils d'estimation commerciales sur le marché ont ce paramètre existant. 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 de projet : 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 de 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, avec plus de 5400 projets (www.gartnergroup.com, 1998).

1.4 L'approche boîte noire (processus opaque)

Un approche « boîte noire » (voir Figure 1.1) inclut comme intrant des mesures de la ressources, le processus et le produit et comme extrant on retrouve les résultats de l'estimation.

La base de données historique à l'appui de l'estimation n'est pas accessible ce qui réduit automatiquement la crédibilité des résultats et du processus d'estimation. Les équations et les détails des paramètres obtenus ne sont pas connus.

L'appui doit permettre le développement d'étapes d'estimation en fonction de nos besoins particulières. Le résultat ne doit pas être basée seulement sur un chiffre parce que les managers doivent prendre des décisions importantes en se basant sur une information crédible. La crédibilité de chaque composant dans cet approche représente un vrai problème.

Comment un chef de projet pourrait avoir un jugement raisonnable en se basant sur un chiffre seulement ? Comment peut-il convaincre ses managers en se basant sur l'information sur laquelle lui-même a de la difficulté à croire ? Comment peut-on établir le niveau de confiance dans un processus d'estimation alors qu'on ne peut pas l'établir pour ses composants ?

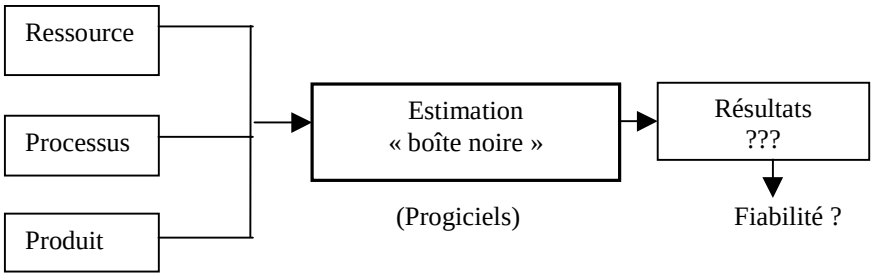

Figure 1.1 Une approche d'estimation « boîte noire »

1.5 Question de la recherche

La question que nous nous posons est la suivante : comment construire un environnement automatisé pour un processus transparent d'estimation fondé sur la base de données de l'International Software Benchmarking Standards Group (ISBSG) ?

Cette question préoccupe plusieurs personnes dont les activités sont liées à l'estimation. Dans le cadre du Laboratoire de recherche en génie logiciels de l'Université du Québec à Montréal plusieurs travaux ont été réalisés dans l'analyse des points de fonction, les méthodes de validation des mesures et les programmes de mesure. Dans le cadre des recherches de ce laboratoire, des analyses ont aussi été faites pour insister sur l'importance d'un processus transparent d'estimation. Un framework pour un processus transparente d'estimation a été aussi mis au point (Abran et al, 1996). Notre recherche constitue le prolongement de ces travaux et pourrait donner lieu à plusieurs recherches dans le domaine de l'estimation du logiciel.

1.6 Objectif de la recherche

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 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 informatique et par les gestionnaires de programme de mesure. Il représenterait un contribution intéressante à ISBSG tant en donnant la valeur ajoutée à leur bas de données.

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

1.8 Les limites de projet

L'environnement est limité à l'utilisation d'une seule base de données, et à l'intérieur de cette base de données l'étude portera sur seulement 23 variables ce qui représente une variable sur trois du total de variables recueillies dans ISBSG. Fait à noter que les 23 variables sont publiques alors que les autres à cause de différentes raisons ne sont pas accessible et leur utilisation n'est pas possible.

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

2. MÉTHODES UTILISÉES POUR RÉALISER NOS OBJECTIFS

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 à le faire.

2.1.1 Processus transparent d'estimation

Dans le processus d'estimation (voir Figure 2.1.) il y a trois grandes composantes (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 qui prend en considération comme intrant la mesure de la ressource, du processus et du produit et comme sortie on retrouve les résultats de la simulation.

La simulation permet le développement du modèle en fonction de nos besoins particulières. L'information ne serait pas basée seulement sur un chiffre parce que l'accès aux données est possible ce qui donne la crédibilité aux résultats obtenus.

Dans une approche transparente le modèle doit être documenté et crédible. Le rapport de confiance doit comprendre un degré de fiabilité pour chaque modèle utilisé.

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 en produisant les résultats finaux de l'estimation.

La crédibilité du processus d'estimation est basée sur la crédibilité de chaque sous-processus d'estimation qui sont à leur tour basé 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 au plus crédible que le moins crédible des composants.

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 importantes. 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'inconnues, d'incertitudes ni de risques et l'information concernant le projet est disponible.

A 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 en littérature comme modèle d'estimation, une expression assez ambiguë, ou modèle de productivité (Abran et al., 1993).

Un modèle de productivité sera considéré empiriquement comme « 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 ce domaine du génie logiciel (Conte, 1986; Verner, 1992). L'approche utilisée pour la construction du modèle COCOMO (Boehm, 1981) peut être considérée comme classique pour la construction de modèles de productivité en génie logiciel. Les équations ont été construites (Conte, 1986) à partir d'une base de

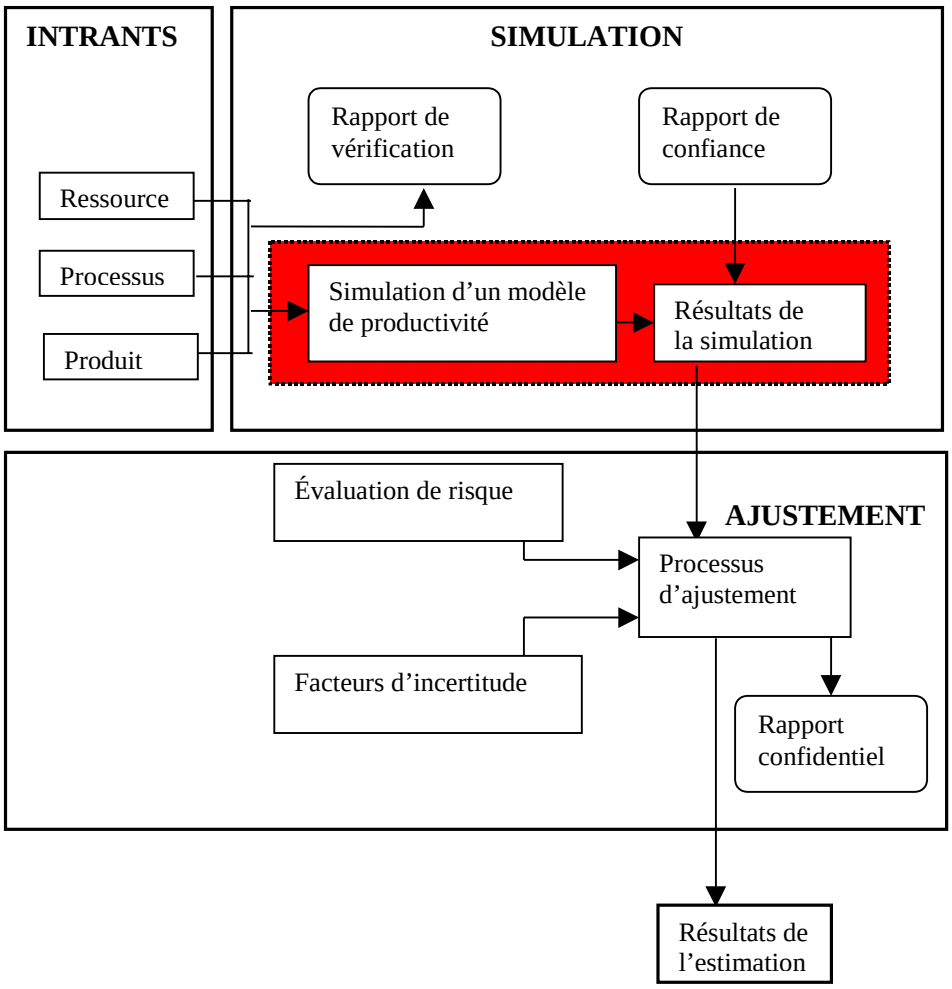

Figure 2.1 Les composants d'un processus transparent d'estimation

données de 63 projets. COCOMO contient 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).

Kemerer a utilisé les données historiques de 15 projets et a comparé les résultats de 4 modèles d'estimation : COCOMO, SLIM, Points de Fonctions et Estimacs, il conclue que malgré ses faiblesses le modèle avec le points de fonction est le « meilleur ».

2.1.3 Mesurer la taille d'un logiciel

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 a 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. A mesure que les langages et techniques de programmation se perfectionnaient et se diversifiaient, l'utilisation du nombre des lignes de code comme principal mesure de la taille d'un logiciel devenait de plus en plus limitatif.

C'est en voulant éviter les limitations et les paradoxes du calcul des lignes de code et s'appuyant indirectement sur l'idée de la mesure de l'effort intellectuel que Allan Albrecht a développé la technique de points de fonction. Cette technique est utilisée pour mesurer la taille et la complexité 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 mesure la taille de chacune de ses caractéristiques et on obtient un nombre que l'on modifie en considérant la complexité globale du projet, 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). Son approche a aussi une vision originale soit celle de mesurer la taille des projets en fonction des prestation livrées à l'usager et non de l'effort technique requis par l'informatique.

2.1.4 Les systèmes d'aide a 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 (Genest, 1996) en trois types: 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 imite 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 des relations entre ces événements. La simulation est un processus de résolution pas à pas ou 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 neitherr well nor consistently ; a useful model produces 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).

2.1.5 Une base de données transparente

En 1990 le groupe australien " Australian Metrics Association " (ASMA) debutait la mise en place d'un groupe de benchmarking avec 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 crée. Aujourd'hui les organisations affiliées sont : ASMA (Australian Software Metrics Association), CIM (Centre d'Intérêt sur les Métriques- Canadian Metrics Associations), SMANZ (Software Measurement Association of New Zealand), UFPUG (United Kindom Function Point Users Group), GUFPI (Italie), NEFPUG (Hollande) et DASMA (Deuschsprachige Andwendrgruppe fur Software Metrik und Aufwandschatzung) et IFPUG (International Function Point Users Group).

La mission du 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 sont utilisés dans une proportion de presque 90% des projets de la base de données.

Dans la base (International Software Benchmarking Group, 1997) on retrouve des informations portant sur l'effort en ressources humaines consacrées au projet, la taille du projet, la qualité du produit livré et l'identification de l'entreprise, en 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 actifs sont les Australiens avec un projet sur deux de la base de données.

La majorité des données du référentiel, soit 83% proviennent d'environnement de développement en 3GL et 4GL. Il y a sept langages associés au 4 GL et dans les premières deux positions on retrouve le langage Natural et SQL et pour la 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, et les services des affaires, banques et manufactures.

Le logiciel gratuit Venturi, est utilisé pour la saisie le données. L'évaluation des données soumis par l'entreprise est faite par deux personnes qui sont membres de ISBSG et l'entreprise reçoit un feedback après 3-4 semaines. La valeur de cette base de données provient du fait quelle est la première grande banque de données disponible pour les chercheurs, et des practiciens sur de projets informatiques au monde.

Plusieurs articles ont été publiés mais l'article de Oligny a retenu notre attention. Dans cet article une approche d'estimation « boîte blanche » a été utilisée (Oligny et al 1997). Étant donné que les attributs de la base de données sont disponibles, des tests statistiques détaillés ont été faits et plusieurs conclusions ont été tirées. Selon les auteurs plusieurs modèles ont été proposés par la littérature jusqu'à date mais COCOMO a retenu le plus d'attention et il est considéré comme un modèle de référence. Avec le critère d'évaluation quantitative, on pourrait observer que le savoir-faire de COCOMO basé sur des données de jusqu'à 20 ans sont très proches de résultats obtenus par un modèle empirique développé à partir de la base de données publique ISBSG qui contient de données sur de projets informatiques développés après 1990.

Un exemple d'un processus d'estimation en utilisant la base de données ISBSG et présenté dans la Figure 2.2. Par rapport au processus d'estimation en utilisant la base de données ISBSG (voir Figure 2.2), l'environnement automatisé ne tiendra pas compte de risque d'affaire ni des incertitudes donc tout le processus de d'ajustement ne sera pas 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. Cependant le résultat de la simulation va contribuer à la fiabilité du rapport de confiance et de vérification.

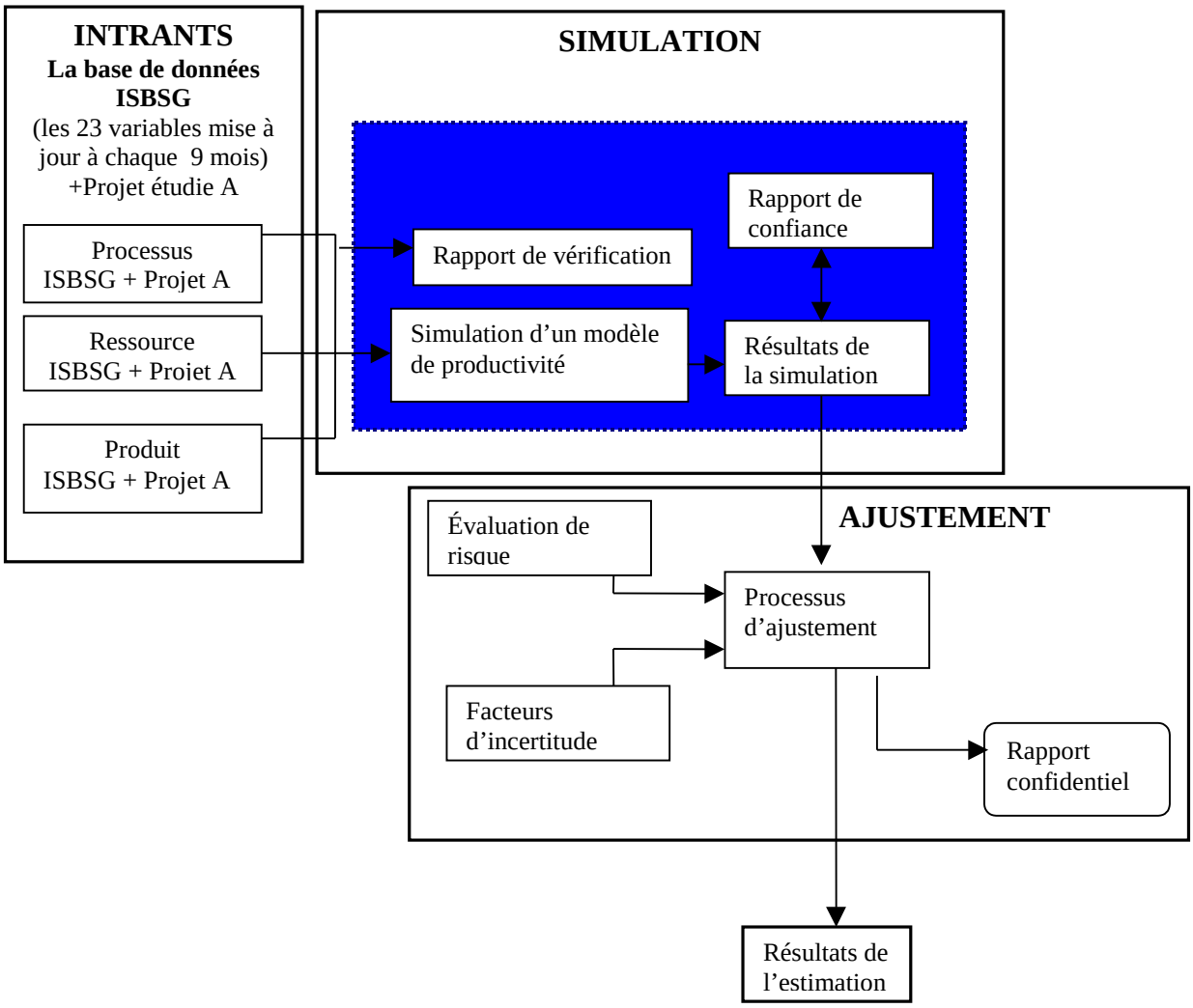

Figure 2.2 Les composants d'un processus d'estimation en utilisant la base de données ISBSG et le Projet A

2.1.6 Techniques statistiques

L'analyse de régression est une technique statique pour l'évaluation analogique de différentes attributs d'un projet. La plupart des modèles d'estimation proposés par la littérature sont basés sur des régressions techniques (Matson et al, 1994).

Le principale avantage de la méthode est de faire référence à un projet déjà réalisé ce qui donne un meilleur fondement dans la réalité. Myers décrit clairement cette approche (Myers, 1989). On pourrait aussi établir et valider statistiquement, par analyse de régression, une relation causale entre le coût d'un projet déjà réalisé et différentes attributs d'un projets comme le langage, la taille, le type de langage, la méthode de développement etc.

Côté, Bourque, Oligny et Rivard (Cote al., 1988) ont identifié et analysé plus de 20 modèles d'estimation. Desharnais a utilisé les analyses statistiques pour mettre au point de modèles de productivité. Les résultats ont été améliorés très nettement en séparant les projets en groupe selon l'environnement informatique dans lequel ils avaient été réalisés (Desharnais, 1988).

2.2 Le projet dans le programme de recherche du laboratoire

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 l'expérimentation et instrumentation».

2.3 Différentes étapes du projet

Cette section décrit les principales étapes qui permettront de mener à bon terme ce projet de recherche. Le programme de maîtrise en informatique de gestion prévoit le dépôt formel d'une proposition de

recherche. L'acceptation de la proposition de recherche constitue le point de départ des étapes qui mèneront au dépôt du rapport final. Le cadre de Basili modifié pour nos recherche est présenté en Appendice A.

2.3.1 Étude de la littérature

Dans notre revue de littérature, nous avons pour objectif de consulter certains livres et des articles sur l'estimation de logiciel et les analyses statistiques. L'étude de la littérature va nous permettre de trouver les variables les plus importantes qui influencent l'estimation et aussi de trouver les meilleure pratique sur la construction de modèles. Il va nous donner une meilleure compréhension de notre sujet et nous permettre de présenter des arguments pour notre environnement automatisé. L'étude est prévue pour toutes les étapes du projet.

2.3.2 Choix d'un langage de programmation

Cette étape va nous permette de mieux comprendre nos besoins et va nous familiariser davantage avec la base de données ISBSG. Cette étape nous permet aussi de faire le choix sur un langage de programmation . Le choix a été Visual Basic Application avec Microsoft Excel qui correspond à nos besoins du point de vue statistique, de l'intégration avec les autres environnements et les ressources nécessaires pour l'avenir.

2.3.3 Identification et réalisation de menus et leur composantes pour l'environnement

La réalisation de l'environnement automatise est prévue pour cet été. Il sera composé de trois menus, dont deux « Évolution » (voir Appendice C), et « Database_ISBSG » (voir Appendice D) seront finalisés en juin 1998 et le troisième, « Simulation », sera finalisé en août 1998.

Dans le menu « Evolution » il y a la possibilité de développer des analyses descriptives pour presque chaque attribut existant dans la base de données ou pour diverses combinaisons des attributs.

Dans le menu « Database_ISBSG » il y a la possibilité de faire des recherches dans la base de données en utilisant plusieurs conditions pour toutes les combinaisons des attributs existants.

Le menu de « Simulation » permettra de développer des modèles d'une façon dynamique et faire diverses analyses statistiques. Dans ce menu il y aura la simulation de modèle de productivité (voir Figure 2.1.). Les graphiques seront développés pour chaque scénario en utilisant plusieurs conditions pour toutes les combinaisons des attributs existants. À l'aide de l'analyse de régression, on peut estimer la valeur des paramètres importants existants dans la base de données. Une fois terminée l'analyse de régression on obtient l'équation de régression et on peut facilement imaginer différents scénarios. Les équations et les détails des paramètres obtenus seront connus. Une fois obtenues un modèle justifiable théoriquement et vérifié statistiquement on peut prévoir le coût de réalisation d'un nouveau projet de même nature qui a des caractéristiques différentes de taille et complexité. Le facteur de corrélation multiple, qui mesure la proportion de variation présente dans les observations et qui est expliquée par l'équation de régression obtenue, sera affiche sur l'écran. Un tableau avec les principaux indicateurs statistiques (minimum, maximum, moyenne, médian, quartile, déviation standard etc.) sera développé pour chaque scénario.

2.3.4 L'élaboration du rapport final

Le rapport final de l'activité de synthèse reprendra la problématique et les fondements conceptuels présentés lors de la proposition de recherche et présentera un sommaire des résultats atteints.

2.4 Faisabilité du projet

2.4.1 Qualification requise

La réalisation de l'activité de synthèse proposée exige à la fois des connaissances en informatique, en gestion et en statistique. Je considère posséder la formation académique et l'expérience nécessaire à la réalisation de cette activité de synthèse. J'ai plus de quatre ans d'expérience en programmation et des études universitaires en génie logiciel.

2.4.2 Ressource nécessaires

La réalisation de cette activité de synthèse requiert principalement certaines ressources matérielles : ordinateur, logiciels etc. Sur ce plan je dispos de tout le matériel nécessaire à la réalisation de nos activités.

2.4.3 Encadrement académique

Cet encadrement académique se traduira en pratique par des rencontres hebdomadaires, avec le directeur ou le codirecteur de l'activité de synthèse d'une heure, au cours desquelles il y aura remise de biens livrables. Chaque bien livrable sera étudié et des commentaires appropriés seront fournis.

Ayant défini la méthode utilisée, nous nous proposons de préciser, dans la prochaine section les échéanciers nécessaires pour réaliser notre recherche.

3. ÉCHÉANCIER

La présente recherche a débuté en été 1997 et se terminera au mois de novembre 1998. Dans ce qui suit, (voir Tableau 2.2), on présente les différentes étapes avec leur échéance respective.

Tableau 2.2

Échéancier pour l'activité de synthèse de Vasile Stroian

ID	Nom de la tâche	Date Début (mois.année)	Date Fin (mois.année)	Durée (heures)
1	Revue de la littérature	07.1997	11.1998	292
2	Développer un prototype	10.1997	06.1998	150
3	Réviser le prototype	06.1998	07.1998	20
4	Réalisation de l'environnement automatisé	06.1997	09.1998	200
5	Rédaction du rapport final	10.1998	11.1998	20
6	Préparation de l'exposé	11.1998	11.1998	10
7	Dépôt du rapport	11.1998	11.1998	8

APPENDICE A

CADRE DE BASILI MODIFIÉ

Tableau 2.1

Cadre de Basili pour l'activité de synthèse de Vasile Stroian

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.	Chef de projets informatique Gestionnaire de programme de mesure Chercheurs en génie logiciel ISBSG et LRGL

Planification		
Etapes 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

APPENDICE B

UN ECHANTILLON DE LA BASE DE DONNEES ISBSG

Project ID	Country	Region	Year Impl	DBMS	Lang Type	Programming Language	Meth	CASE	C/SRV
29015	AST	EUROPE	94	Y	3GL	Unknown	Inhs	Uppr	Y
29017	AST	EUROPE	94	Y	3GL	PL/I	Inhs	Uppr	Unknown
30001	AUS	ASIA	90	Y	4GL	NATURAL	Unknown	Unknown	Unknown
30002	AUS	ASIA	89	Unknown	Unknown	Unknown	Unknown	Unknown	Unknown
30003	AUS	ASIA	91	Y	3GL	COBOL	Unknown	Unknown	Unknown
30004	AUS	ASIA	91	Y	3GL	C	Unknown	Unknown	Unknown
30005	AUS	ASIA	91	Y	3GL	COBOL	Unknown	Unknown	Unknown
30006	AUS	ASIA	91	Y	ApG	Unknown	Unknown	Yes	Unknown
30007	AUS	ASIA	90	Y	ApG	TELON	Unknown	Yes	Unknown
30008	AUS	ASIA	89	Y	4GL	Unknown	Unknown	Unknown	Unknown
30009	AUS	ASIA	91	Y	4GL	Unknown	Unknown	No	Unknown
30010	AUS	ASIA	91	Y	4GL	Unknown	Unknown	No	Unknown
30011	AUS	ASIA	91	Y	4GL	Unknown	Unknown	Unknown	Unknown
30012	AUS	ASIA	92	Y	ApG	TELON	Unknown	Yes	Unknown
30013	AUS	ASIA	91	Y	3GL	COBOL	Unknown	No	Unknown
30014	AUS	ASIA	91	Y	3GL	COBOL	Unknown	No	Unknown
30015	AUS	ASIA	92	Y	3GL	COBOL	Unknown	No	Unknown
30016	AUS	ASIA	92	Y	3GL	C	Unknown	No	Unknown
30017	AUS	ASIA	91	Y	3GL	COBOL	Unknown	No	Unknown

UN ECHANTILLON DE LA BASE DE DONNEES ISBSG (suite)

Time (Months)	Team Size	VAF	FPA Method	Flag	Rate (Hrs/FP)	Effort (Hours)	Size (FP)	Deve-lopment Type	Application Type	Deve-lopment Platform	R
15	11	1,00	IFP	Unknown	8,24	10222	1241	ND/S1	OIS	MR	
20	Unknown	1,00	IFP	Unknown	14,23	5864	412	ND	OIS	MF	
14	4	0,90	IFP	Unknown	5,81	3574	615	ND	Unknown	MF	
12	4	1,04	IFP	Unknown	3,46	3504	1014	ND	Unknown	MF	
Unknown	Unknown	1,00	IFP	Unknown	6,65	545	82	ND/SS	Unknown	MF	
14	8	1,13	IFP	Unknown	18,52	9296	502	ND	Unknown	PC	
13	7	1,16	IFP	Unknown	19,12	16179	846	ND	Unknown	MF	
15	12	1,16	IFP	Unknown	5,79	26408	4562	ND	Unknown	MR	
8	Unknown	0,81	IFP	Unknown	8,40	2597	309	ND	Unknown	MF	
38	32	0,89	IFP	2	54,46	66600	1223	ND	Unknown	MF	
6	Unknown	0,99	IFP	Unknown	1,73	1586	917	ND	Unknown	PC	
5	2	0,86	IFP	Unknown	1,93	528	273	ND	Unknown	PC	
3	2	0,98	IFP	Unknown	1,28	281	220	ND	Unknown	PC	
10	11	0,97	IFP	Unknown	12,15	2660	219	EN	Unknown	MF	
Unknown	Unknown	1,00	IFP	Unknown	8,88	5134	578	ND	Unknown	MF	
Unknown	Unknown	1,00	IFP	Unknown	4,87	419	86	ND	Unknown	MF	
Unknown	Unknown	0,88	IFP	Unknown	1,60	273	171	ND	Unknown	MF	
Unknown	Unknown	1,05	IFP	Unknown	6,85	1035	151	ND/SS	Unknown	PC	
Unknown	Unknown	0,87	IFP	Unknown	4,10	513	125	ND/S1	Unknown	MF	

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, in Metrics in Software Evolution ». vol. *GMD-Bericht* NR. 254, M. Müllerburgh and A. Abran, Eds. Sankt Augustin: R. Oldenbourg Verlag, 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. Desharmais 1996. « A strategy for a credible & audible estimation process ». Note de cours INF-7760, p. 14.

Abran A. et J.M. Desharmais 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., Mai 1994. « Function Points: A Study of their Measurement Processes and Scale Transformations ». *Journal of Systems and Software*, 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 Nov. p. 639-648.

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

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., 1995 « Cost Models for Future Software Life Cycle Processes : Cocomo 2.0 ».

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 ». *J. 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 of Syst. And 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.* Youirdon 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 3/94

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

Fishwick, P. A. and Zeigler B. P., 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 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.

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.

Hoaglin et al. 1991. *Fundaments of Exploratory Analysisi of Variance.* Willey.

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.

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

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

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-1993

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.

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, 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 ». p. 10.

Oligny, S, P. Bourque et A. Abran, 1997. « An empirical assessement of project duration in software engineering ». ESCOM'97.

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. 1991. « Estimeetings : developpement estimates and a front-end process for a large project ». *IEEE* Vol 17, No 8, August, 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. , 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.gartnergroup.com, 1998

www.spr.com, 1998.

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