

Distr.  
GÉNÉRALE

CES/AC.71/2003/3  
10 février 2003

ANGLAIS et FRANÇAIS seulement

**COMMISSION DE STATISTIQUE et  
COMMISSION ÉCONOMIQUE POUR  
L'EUROPE (ONU)  
CONFÉRENCE DES STATISTICIENS  
EUROPÉENS**

**COMMISSION EUROPÉENNE**

**OFFICE STATISTIQUE DES  
COMMUNAUTÉS EUROPÉENNES  
(EUROSTAT)**

**ORGANISATION DE COOPÉRATION ET DE DÉVELOPPEMENT  
ÉCONOMIQUES (OCDE) DIRECTION DES STATISTIQUES**

**Réunion mixte CEE/Eurostat/OCDE sur la gestion des systèmes d'information statistique  
(Genève, 17-19 février 2003)**

Point I: Mesures à prendre pour améliorer la qualité au niveau de la gestion de la TI

**CADRE DE LA QUALITÉ DU LOGICIEL**

**Communication sollicitée**

Émanant de Statistique Canada<sup>1</sup>

***Résumé:** Statistique Canada, bureau national de la statistique du Canada, est tenu de fournir à ses clients des statistiques de la plus haute qualité. L'organisme s'appuie sur de nombreuses politiques et lignes directrices pour aider ses employés à produire des données fiables. L'utilisation de solutions de grande qualité fondées sur les technologies de l'information (TI) est une composante importante de la réalisation de cet objectif. Bien que beaucoup d'efforts aient été consacrés à la description des caractéristiques de la qualité des données, peu de travaux ont été effectués pour décrire la qualité du logiciel. Ce document donne un aperçu de la qualité du logiciel à Statistique Canada en proposant une définition générale de ce qu'est un logiciel de qualité. Il illustre l'importance des caractéristiques de la qualité du logiciel et de leurs interrelations, et décrit les différentes perspectives des intervenants concernés. Le document indique également les répercussions qu'ont les coûts, le calendrier et les caractéristiques des produits sur les compromis auxquels il faut en arriver pour assurer la qualité du logiciel, en plus de décrire d'autres éléments qui influent sur cette qualité (personnes, processus, produit et technologie). Il aborde également les différentes techniques qui permettent d'améliorer la qualité du logiciel réalisé et utilisé à Statistique Canada.*

**MOTS CLÉS :** Logiciel, amélioration de la qualité.

---

<sup>1</sup> Établi par Matt Edgar, Mel J. Turner (Mel.Turner@statcan.ca).

## I. INTRODUCTION

1. Statistique Canada, qui œuvre dans le secteur de l'information, ne fabrique aucun bien (automobiles ou ameublement) mais produit de l'information statistique. Pour s'assurer de l'efficacité de la collecte, de l'analyse, du traitement et de la diffusion de cette information, Statistique Canada utilise une variété de systèmes d'information. Un système d'information inclut les éléments suivants :

- données (valeurs descriptives, spatiales, numériques ou logiques utilisées dans le système);
- métadonnées (données qui décrivent les autres données utilisées dans le système);
- documentation (plans, procédures, manuels qui facilitent l'utilisation de cette information);
- personnes (chargées des tâches de système);
- matériel (technologie utilisée pour le traitement et le stockage de l'information);
- logiciel (information qui commande le matériel pour effectuer des tâches particulières).

2. Il est crucial pour notre survie d'assurer la qualité de tous les aspects du système d'information :

*La confiance dans la qualité de l'information produite est une question de survie pour un organisme de statistique. Si l'information produite est douteuse, la crédibilité de l'organisme est remise en question et sa réputation à titre de source indépendante et objective d'information fiable est compromise. Le soin apporté à la qualité de l'information est donc une préoccupation centrale pour la gestion d'un bureau national de la statistique. (Source : Brackstone 1999)*

3. Statistique Canada, bureau national de la statistique du Canada, est tenu de fournir à ses clients des statistiques de la plus haute qualité. L'organisme s'appuie sur de nombreuses politiques et lignes directrices pour aider ses employés à produire des données fiables.

4. Beaucoup d'efforts ont été consacrés à la description des caractéristiques de la qualité des données. Il existe des lignes directrices concernant la qualité des données recueillies et l'information traitée à Statistique Canada (Statistique Canada, 1997a). *Le Cadre d'assurance de la qualité permet de documenter les processus actuels utilisés pour gérer la qualité au sein de la structure organisationnelle tout en tenant compte des risques et des possibilités identifiés par l'organisme* (Statistique Canada, 1998). Malheureusement, peu de travaux ont été effectués pour décrire la qualité des autres aspects des systèmes d'information. Ce document met l'accent sur les logiciels et sur l'importance de leur qualité à Statistique Canada.

5. Pour être en mesure d'offrir de l'information de haute qualité, Statistique Canada investit des sommes substantielles en technologie de l'information (matériel et logiciel). Chaque année, l'organisme consacre des parties importantes de son budget à la réalisation et à la maintenance de logiciels. Il est donc essentiel de bien comprendre les caractéristiques de la qualité du logiciel pour être en mesure de produire effectivement des solutions logicielles de qualité. D'autre part, la qualité n'est pas une valeur absolue et elle exprime des réalités diverses selon les différents contextes. Quoi qu'il en soit, on doit faire le maximum pour produire du logiciel de grande qualité.

6. La section II du document définit la qualité du logiciel en termes généraux pour établir une base de discussions plus détaillées. Le résumé des caractéristiques de la qualité du logiciel et de leurs interrelations, à la section III, permet d'étayer cette définition. Il y a plusieurs façons d'envisager la qualité du logiciel. La section IV décrit différentes perspectives possibles et illustre leurs interrelations.

7. Une variété de contraintes influent sur la qualité du logiciel :

- Le coût est important pour l'efficacité et la rentabilité d'un organisme tel que Statistique Canada.
- Par leur nature, les enquêtes doivent respecter des calendriers et des délais stricts essentiels à la réussite de leur exécution.

- La complexité ou le degré de difficulté de la réalisation du logiciel sont intimement liés à la complexité de nos domaines d'enquête et aux demandes sans cesse croissantes pour des produits de grande qualité qui s'appuient sur des technologies de cueillette, de transformation et de diffusion d'information qui évoluent rapidement.

8. La description des contraintes, à la section V, inclut une discussion sur la nécessité de faire des compromis et d'établir des priorités au moment de décider des objectifs de la qualité.

9. Une fois qu'un projet logiciel est lancé, plusieurs éléments peuvent influencer sur la qualité du résultat. La section VI décrit en détail l'influence de ces éléments (personnes, processus, produit et technologie) sur la qualité du logiciel livré aux clients. La section VII inclut une brève liste de certaines techniques disponibles pour améliorer le logiciel réalisé et utilisé à Statistique Canada.

## II. DÉFINITION DE LA QUALITÉ DU LOGICIEL

10. D'aucuns prétendent que « *la qualité est une question de perception* » (Davis 1995). L'expérience montre que chacun perçoit la qualité d'une manière qui lui est propre. Ce concept est donc difficile à définir et à mesurer. Lorsqu'on demande à des personnes de définir la qualité d'un logiciel, elles décrivent habituellement un ensemble de critères d'évaluation (p. ex., exactitude, souplesse, robustesse et fiabilité). Ces critères, toutefois, ne désignent que des *caractéristiques* d'un logiciel de grande qualité. Ces caractéristiques, également désignées sous le nom d'attributs ou de dimensions, ne permettent pas de saisir l'essence même de ce qui définit la qualité du logiciel.

11. L'IEEE<sup>2</sup> propose la définition suivante de la qualité : « mesure dans laquelle un système, une composante ou un processus répondent aux besoins ou aux attentes d'un client ou d'un utilisateur » (norme 610.12-1990). Dans le cas d'un logiciel, cette définition suppose que la qualité se mesure en fonction de la manière dont celui-ci répond adéquatement aux besoins des clients. Toutefois, il n'est pas facile de déterminer et de mesurer ces besoins. Pour un logiciel donné, Wiegers (1999) indique que, de manière générale, les utilisateurs ont certaines attentes implicites non formulées dans l'énoncé des exigences. Donc, répondre uniquement aux exigences ne permet pas nécessairement de satisfaire aux attentes des utilisateurs sur le plan de la qualité. De plus, qu'en est-il des réalisateurs? Il va de soi qu'ils ont également leur mot à dire sur ce que constitue un logiciel de qualité. Après tout, il leur revient de créer la meilleure solution possible aux problèmes des utilisateurs.

12. Compte tenu de ce qui précède, le présent document propose la définition suivante en matière de qualité de logiciel :

La **qualité d'un logiciel** désigne la mesure dans laquelle celui-ci répond aux exigences énoncées et implicites de l'ensemble des intervenants.

13. Cette définition contient l'essentiel de la définition que propose l'IEEE, et est équivalente à la définition de la qualité des données adoptée par Statistique Canada. Elle reconnaît qu'il faut à la fois répondre aux exigences énoncées et implicites pour satisfaire aux besoins des clients. Les utilisateurs, les réalisateurs et tous les autres intervenants ont leur propre perception de la qualité du logiciel et ont donc leur mot à dire concernant la description des caractéristiques.

14. Il peut être relativement facile de caractériser les attentes (énoncées et implicites) en matière de qualité du logiciel, mais la liste des caractéristiques semble sans fin. Le défi consiste à choisir parmi les caractéristiques celles qui sont les plus importantes pour les intervenants.

## III. CARACTÉRISTIQUES DE LA QUALITÉ DU LOGICIEL

15. Le mandat de Statistique Canada est de « recueillir, compiler, analyser, dépouiller et publier des renseignements sur la vie économique et sociale et sur les conditions générales du pays et de ses

---

<sup>2</sup> IEEE - Institute of Electrical and Electronics Engineers

citoyens » (Statistique Canada 2000). Pour permettre à l'organisme de réussir son mandat, les méthodologistes et économistes sont chargés de faire en sorte que l'information produite soit de la plus grande qualité. Le Cadre d'assurance de la qualité (Statistique Canada 2002) établit très clairement les grandes lignes de ces objectifs. Le cadre propose également le concept de définition de la qualité des données en définissant six dimensions de la qualité des données. Le tableau 3-1 donne un résumé de ces six dimensions.

**Tableau 3-1 : Six dimensions de la qualité des données**

Pertinence	Par pertinence des données statistiques, on entend la mesure dans laquelle les besoins réels des clients sont satisfaits.
Exactitude	Par exactitude des données statistiques, on entend la mesure dans laquelle l'information décrit bien le phénomène qu'elle doit mesurer.
Actualité	L'actualité des données statistiques correspond au délai entre le point de référence (ou la fin de la période de référence) auquel se rapportent l'information et la date à laquelle les données sont disponibles.
Accessibilité	Par accessibilité des données statistiques, on entend la facilité avec laquelle on peut se les procurer du bureau national de la statistique.
Intelligibilité	Par intelligibilité des données statistiques, on entend la disponibilité de renseignements supplémentaires et de métadonnées nécessaires à l'interprétation et à l'utilisation appropriées de ces données.
Cohérence	Par cohérence des données statistiques, on entend la mesure dans laquelle celles-ci peuvent être jumelées à d'autres renseignements statistiques dans un vaste cadre analytique au fil du temps.

(Source : Brackstone 1999)

16. Ces dimensions constituent un excellent point de départ pour la liste des caractéristiques servant à décrire la qualité du logiciel. Elles tendent à mettre l'accent sur les caractéristiques externes plutôt qu'internes. Les caractéristiques externes concernent la façon dont les utilisateurs du logiciel en perçoivent la qualité. Les caractéristiques internes représentent les attributs de la qualité du logiciel qui reflètent la mesure dans laquelle un produit logiciel est bien construit. Dans *Peopleware* (DeMarco 1999), l'auteur précise que les normes de qualité internes sont très élevées lorsqu'elles sont définies par les réalisateurs. Cela sous-entend que leur perception (interne) de la qualité est différente de celle (externe) des autres intervenants.

17. Le *Cadre des technologies de l'information* (CTI) (Statistique Canada 1996) est l'un des plus récents documents à aborder les questions de logiciel au sein de notre organisation. Il donne un aperçu de la liste des objectifs stratégiques que la Direction de l'informatique juge importants pour assurer une saine infrastructure informatique à Statistique Canada. Le CTI donne un aperçu des objectifs sur le plan de la disponibilité, de l'interfonctionnement, de l'utilisation, de la fiabilité, de la sécurité et de la souplesse des solutions logicielles. Ces objectifs reflètent, à un haut niveau, le besoin pour un bureau national de la statistique d'obtenir un effet de levier efficace de ses investissements informatiques. Le cadre propose une bonne combinaison de caractéristiques à la fois du point de vue de l'utilisateur et de celui du réalisateur, tout en mettant l'accent sur le niveau externe.

18. L'absence d'autres documents d'orientation à Statistique Canada pose un défi pour la détermination des caractéristiques que nos réalisateurs jugent importantes. Nous devons donc nous tourner vers l'industrie du logiciel. James McCall, alors qu'il était en poste à la General Electric Company, a rédigé un des premiers documents de discussion de l'industrie du logiciel sur les facteurs de qualité du logiciel (McCall 1977, Walters & McCall 1979). Souvent appelé le « modèle GE » ou le « modèle McCall », ce modèle vise d'abord les réalisateurs et est utilisé durant le processus de réalisation. Le document *Characteristics of Software Quality* (Boehm 1978) est une autre tentative de l'industrie de définir la qualité du logiciel. Il a fréquemment été utilisé comme document de référence et décrit plusieurs attributs de la qualité du logiciel.

19. La norme ISO 9126 (ISO/IEC 9126:1991) décrit dans une liste les six dimensions de la qualité du logiciel. Cette norme d'évaluation de produit vise à éliminer toute confusion entre les utilisateurs et les réalisateurs. Plus récemment, le *Guide to the Software Engineering Body of Knowledge-SWEBOK* (IEEE 2000) a adopté la norme ISO 9126 pour la description du domaine des connaissances en matière de qualité du logiciel. D'autres sources (McConnell 1993, Wiegers 1999) proposent leur propre ensemble de caractéristiques selon la perspective du réalisateur.

20. Le tableau 3-2 tente de résumer ces attributs dont certains sont liés à l'exécution (point de vue de l'utilisateur) et d'autres, à la réalisation (point de vue du réalisateur). Selon le cas, certains des attributs auront une plus grande importance pour certains intervenants. La section IV explore les différents points de vue que les personnes adoptent lorsqu'elles décident de l'importance des attributs. La section V décrit l'incidence des contraintes sur les compromis et les priorités associés à ces caractéristiques de qualité.

**Tableau 3-2 : Sommaire des caractéristiques de la qualité du logiciel**

<b>Attribut</b>	<b>Réalisation / Définition exécution</b>	
Adaptabilité	Réalisation	Facilité avec laquelle le logiciel peut être modifié pour tenir compte des changements de l'environnement d'exploitation.
Applicabilité	Réalisation	Mesure dans laquelle le logiciel peut être utilisé, sans modification, dans des applications et des environnements autres que ceux pour lesquels il a été spécifiquement conçu.
Disponibilité	Les deux	Facilité avec laquelle le logiciel peut être diffusé, obtenu ou installé.
Clarté	Réalisation	Facilité avec laquelle un réalisateur ou un responsable de la maintenance est en mesure de comprendre la conception et la construction d'un système. La clarté concerne davantage la compréhension de bas niveau que l'intelligibilité, qui est de niveau supérieur. Elle inclut la présence de renseignements supplémentaires et des métadonnées nécessaires à l'interprétation et à l'utilisation du logiciel.
Conformité	Réalisation	Mesure dans laquelle le logiciel est conforme aux interfaces standard (de droit ou de facto) et peut interfonctionner avec d'autres applications logicielles. Également désignée sous le nom d'interfonctionnement.
Exactitude	Réalisation	Mesure dans laquelle les spécifications, la conception et la mise en œuvre du logiciel sont exemptes de tout problème.
Maintenabilité	Réalisation	Effort nécessaire à l'exécution de modifications, y compris les corrections, les améliorations ou l'adaptation du logiciel aux changements (exigences et spécifications fonctionnelles).
Performance	Exécution	Niveau de rendement (durée d'exécution) du logiciel et quantité de ressources utilisées (mémoire, stockage) dans des conditions spécifiques. Également désignée sous le nom d'efficience.
Portabilité	Réalisation	Capacité de transférer le logiciel d'une organisation, d'un matériel ou d'un environnement logiciel à un autre.
Précision	Exécution	Mesure dans laquelle le logiciel fonctionne exactement comme il a été conçu. La précision est différente de l'exactitude; elle détermine la mesure dans laquelle le logiciel effectue le travail pour lequel il a été créé plutôt que la mesure dans laquelle il a été correctement créé.
Ponctualité	Réalisation	Le logiciel a-t-il été acquis ou mis en œuvre au moment opportun? Bien que non nécessairement un attribut interne du logiciel, peut être critique pour mesurer un service de réalisation.
Fiabilité	Exécution	Mesure dans laquelle on peut s'attendre que le logiciel remplit la fonction pour laquelle il a été conçu, avec la précision requise.

Attribut	Réalisation / exécution	Définition
Réutilisabilité	Exécution	Facilité avec laquelle on peut utiliser des parties d'un système dans d'autres systèmes ou dans un autre contexte.
Robustesse	Exécution	Capacité du logiciel à maintenir son niveau de rendement en présence de données erronées ou d'un environnement de système pénible.
Sécurité	Exécution	Mesure dans laquelle on peut contrôler l'accès au logiciel ou aux données par des personnes non autorisées. Également désignée sous le nom d'intégrité.
Pertinence	Exécution	Mesure dans laquelle le logiciel répond aux besoins des clients; indique également s'il est adapté à l'utilisation.
Testabilité	Réalisation	Facilité avec laquelle le logiciel peut être testé pour y déceler des défauts et s'assurer qu'il répond aux spécifications.
Intelligibilité	Exécution	Facilité avec laquelle on peut comprendre un système logiciel.
Facilité d'utilisation	Exécution	Facilité d'utilisation du logiciel.

(Sources : Boehm 1978, ISO 1991, McConnell 1993, STC 1996, Walters & McCall 1979, Wiegers 1999)

## A. Cadre de la qualité

21. Le cadre ci-dessous concerne les attributs des dimensions de la qualité relevés dans les ouvrages sur le logiciel et utilisés pour décrire nos sorties statistiques tout en maintenant la distinction entre les points de vue des réalisateurs (attributs de réalisation) et des utilisateurs (attributs d'exécution). Les six dimensions de la qualité sont redéfinies ci-dessous en fonction de la qualité du logiciel :

- *Pertinence* : Puisqu'un logiciel est un processus d'automatisation d'une théorie, d'un algorithme ou d'un processus quelconque, la pertinence permet d'établir son applicabilité ou la mesure dans laquelle sa fonction est adaptée à un but particulier. On présume que le niveau de pertinence d'une application personnalisée est élevé alors que celui d'un système généralisé, pour une situation donnée, peut être inférieur.
- *Exactitude* : On peut évaluer la mise en œuvre d'un logiciel en fonction de son *exactitude* ou de la mesure dans laquelle il respecte fidèlement la théorie ou la spécification. Ce qui importe, du point de vue de l'utilisateur, est la *précision* ou l'exactitude du résultat (qui peut être liée en partie à la qualité des données d'entrée et non uniquement au caractère approprié du logiciel).
- *Actualité* : On dit qu'un logiciel est actuel s'il peut être livré au moment opportun et qu'il est en mesure de s'exécuter à un rythme approprié à l'application pour laquelle il a été créé. Ces deux aspects permettent de mesurer la *disponibilité*, quoique dans des perspectives différentes.
- *Accessibilité* : L'accessibilité concerne la mesure dans laquelle le public cible peut facilement repérer et utiliser la technologie logicielle. Cette dimension peut inclure des attributs tels que la disponibilité dans le commerce, l'existence de pratiques commerciales (p. ex., l'émission de licence d'entreprise) ou l'accessibilité générale à la formation, des outils et des pratiques liés à la technologie. Du point de vue de l'utilisateur, l'accent est mis davantage sur la sécurité et la fiabilité de l'accès plutôt que sur la *facilité d'utilisation*, abordée en détail ci-dessous.

- *Intelligibilité* Le public cible est-il en mesure de bien comprendre le logiciel et de l'appliquer? Pour cette dimension, les points de vue du réalisateur et de l'utilisateur sont très différents. Pour le réalisateur, la clarté de la documentation et du code du logiciel est essentielle au soutien de l'activité de maintenance et d'amélioration. Pour l'utilisateur, la facilité d'apprentissage et d'utilisation sont des attributs clés.
- *Cohérence :* Cette dimension concerne le niveau de normalisation adopté par le logiciel et son application. Si le logiciel est conforme aux normes de l'industrie, il pourra facilement être utilisé dans des environnements techniques différents, interfonctionner avec d'autres composantes logicielles et être plus facilement réutilisable dans différents contextes opérationnels. La capacité de réutilisation a une incidence positive sur la qualité du logiciel, qui devient de plus en plus robuste et exempt d'erreur au fur et à mesure que son utilisation se propage. Du point de vue de l'utilisateur, un logiciel très répandu offre également deux avantages : transférabilité des compétences d'utilisateur et réduction des besoins de formation spécialisée.

22. Le tableau ci-dessous donne un sommaire de la correspondance des descripteurs. Cette correspondance n'est pas exacte et le tableau contient également des indicateurs positifs et négatifs d'interférence, ou interdépendance. On aborde dans la prochaine section l'incidence de ces indicateurs sur les compromis concernant la qualité.

**Tableau 3-3 : Correspondance des caractéristiques de la qualité du logiciel**

Attributs de la qualité du logiciel		Caractéristique du cadre						Commentaires
		Pertinence	Exactitude	Actualité	Accessibilité	Intelligibilité	Cohérence	
<b>Attributs de réalisation</b>								<b>Point de vue du réalisateur</b>
	Applicabilité	l			s	s	t	Solution d'un problème particulier.
	Exactitude		l				s	Conformité aux spécifications.
	Ponctualité, disponibilité			l	s			Achèvement dans le respect des délais et du budget.
	Adaptabilité, testabilité			t	l		s	Conception technique professionnelle.
	Maintenabilité, clarté	s			s	l	s	Valeur durable.
	Portabilité, conformité	t		t			l	Conformité aux normes de l'industrie.
<b>Attributs d'exécution</b>								<b>Point de vue de l'utilisateur</b>
	Pertinence	l			s	s	t	Adaptation au besoin.
	Précision		l				s	Fiabilité des données.
	Rendement, disponibilité			l	s			Aide à l'exécution du travail.
	Fiabilité, sécurité			t	l		s	Logiciel fiable.
	Intelligibilité, facilité d'utilisation	s			s	l	s	Facilité d'apprentissage et d'utilisation.
	Réutilisabilité, robustesse	t		t			l	Compétences utilisables dans d'autres contextes.
l	<i>Attribut et (ou) caractéristique primaire.</i>							
s	<i>L'attribut soutient ou améliore la caractéristique du cadre.</i>							
t	<i>L'attribut peut entrer en conflit avec la caractéristique du cadre ou lui nuire.</i>							

23. Bien que l'on puisse définir de nombreux attributs de la qualité du logiciel, ils ne seront pas toujours complémentaires. McConnell (1993) a signalé à juste titre qu'il y aura des conflits entre les tentatives de maximisation de certaines caractéristiques. Par exemple, de manière générale, un logiciel très performant atteint ce niveau de qualité en utilisant des techniques qui, d'autre part, en réduisent la portabilité. Certains outils génériques permettent d'améliorer la cohérence du logiciel mais peuvent ne pas convenir à une application particulière. Ces types de conflits sont fréquents dans la plupart des projets logiciels.

24. Par contre, d'autres attributs se complètent très bien. Une solution qui répond aux besoins de l'utilisateur est en général très accessible. Pour l'utilisateur, l'intelligibilité d'une composante logicielle aura une incidence positive sur sa cohérence. Du point de vue du réalisateur, la testabilité améliore la cohérence alors que le temps nécessaire pour la réaliser peut avoir une incidence négative sur l'actualité du logiciel.

#### IV. PERSPECTIVES

25. On peut décrire la qualité du logiciel en utilisant une variété de caractéristiques, comme il a été mentionné à la section précédente. On peut donc conclure qu'il est possible que la qualité puisse «*être classée selon un certain nombre de perspectives ou de points de vue*» (Gillies 1992, p. 9). Souvent, chaque intervenant développe sa propre perspective par rapport à un produit logiciel. Les intervenants peuvent être soit des «*utilisateurs*», soit des «*réalisateurs*». Toutefois, cette approche pourrait être trop restrictive pour les besoins de Statistique Canada. Dans les faits, notre environnement de travail est plus diversifié et exige une meilleure méthode pour le décrire.

26. Les classes ci-dessous désignent les intervenants types d'un projet logiciel à Statistique Canada. La liste a été adaptée à partir de listes similaires tirées du livre *Software Quality: Theory and Management* (Gillies 1992, Section 1.3) et du document *CTI : Gestion des projets* (Statistique Canada 1993, section 3).

***Parrain de projet.*** Le parrain de projet (ou gestionnaire de programme) désigne la personne, le comité ou la division qui finance le projet logiciel. Le parrain est habituellement le directeur de la division qui utilisera la solution. Cet intervenant est souvent éloigné des problèmes quotidiens de mise en œuvre et désire la *réussite* du projet afin de justifier les dépenses encourues. Il évalue la réussite du projet tant en fonction de la livraison du produit dans le respect des délais et du budget que de critères de qualité généraux.

***Gestionnaire de projet.*** Le gestionnaire de projet est responsable des objectifs et (ou) des produits livrables du projet. Son objectif est de produire un logiciel fiable et facile à maintenir et qui répond aux besoins du client. Toutefois, il doit composer avec des contraintes de budget et d'échéances qui exigeront des compromis.

***Chef de projet de TI*** Un chef de projet de TI est affecté à la plupart des projets logiciels. Il est responsable de surveiller les travaux quotidiens des réalisateurs. Il s'assure que ces derniers connaissent les exigences fonctionnelles et techniques du projet. Il aide également le gestionnaire de projet en assurant la liaison avec les clients afin de bien comprendre leurs besoins et leurs priorités.

***Réalisateur.*** Le réalisateur est la personne qui crée le logiciel.

***Utilisateur final.*** Les utilisateurs finals sont habituellement ceux qui interviennent le moins souvent au cours du processus de réalisation. Toutefois, ce sont eux qui doivent utiliser le produit final. En bout de ligne, leur taux de satisfaction déterminera la mesure dans laquelle l'investissement dans le projet était justifié.

27. Chacun de ces intervenants possède son propre point de vue sur la qualité du logiciel et ces différents points de vue peuvent être conflictuels.

*J'ai déjà assisté à une réunion post-mortem sur un projet de traitement d'enquête récemment terminé. La première question posée fut la suivante : « Qui parmi vous estime que ce projet est une réussite? » Un des cadres hiérarchiques a répondu spontanément : « Oui, ce fut une réussite! Nous avons livré le logiciel à temps. » De ce point de vue, il s'agissait de la plus haute priorité. Toutefois, une des réalisatrices invitées à titre d'auditeur a indiqué qu'elle n'était pas d'accord : « Comment pouvez-vous dire ça? Le logiciel est un fouillis total et il n'y a aucune documentation. La conception n'est pas adaptable aux modifications prévues l'an prochain. Il y a plus de 30 000 lignes de code, dont la majorité est excessivement difficile à comprendre. » Son point de vue était complètement différent. Elle se souciait peu que le logiciel ait été livré à temps. Sa priorité consistait à s'assurer que le système était facile à maintenir et réutilisable pour les enquêtes de l'année suivante. Tous les participants à la réunion ont eu leur mot à dire et la discussion s'est orientée sur d'autres aspects du projet. Ces aspects étaient perçus comme des réussites par certains et comme des échecs par d'autres. Il a été possible d'en venir à un consensus, mais il a fallu débattre longtemps.*

28. Ce scénario illustre exactement la façon dont chaque intervenant perçoit différemment la qualité du logiciel. Il est important de réaliser que chaque point de vue est valable et représente simplement une perspective différente. Pour aider à comprendre dans quelle mesure chaque caractéristique est importante pour chaque classe, le tableau 5-1 indique un profil des qualités importantes pour chaque intervenant. Il ne s'agit pas du seul profil possible mais plutôt d'un profil type. Il vaut la peine de bien comprendre ces attentes dès le début d'un projet, au moment où il est possible de les gérer.

**Tableau 5-1 : Priorités types de qualité du logiciel**

	Parrain de projet	Gestionnaire de projet	Chef de projet de TI	Réalisateur	Utilisateur final
<b>Pertinence</b>	Pertinence		Applicabilité		Pertinence
<b>Exactitude</b>		Précision	Exactitude	Exactitude	Précision
<b>Actualité</b>	Ponctualité	Ponctualité	Disponibilité	Rendement	Disponibilité
<b>Accessibilité</b>		Adaptabilité	Testabilité	Sécurité	Fiabilité
<b>Intelligibilité</b>	Clarté		Maintenabilité		Facilité d'utilisation
<b>Cohérence</b>	Robustesse	Réutilisabilité	Portabilité	Conformité	Réutilisabilité

## V. CONTRAINTES

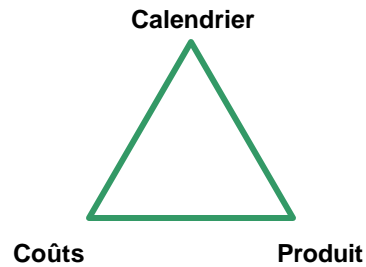
29. Comme il a été mentionné précédemment, les intervenants du projet doivent décider des aspects pour lesquels on doit faire des compromis et des priorités à attribuer aux caractéristiques, en ordre d'importance.

30. Dans le cas des projets de TI à Statistique Canada, il y a toujours des contraintes évidentes liées au calendrier et aux coûts. McConnell (1996) avance un argument convaincant lorsqu'il mentionne que le triangle des contraintes coûts-calendrier-*qualité* n'est pas aussi pertinent que le triangle coûts-calendrier-*produit*. Dans ce dernier cas, le produit inclut la qualité et tous les autres attributs du logiciel, y compris les fonctions, la complexité, le taux de défektivité, etc.

31. Pour maintenir le triangle, il faut assurer un équilibre entre calendrier, coûts et produit. De nombreux projets logiciels appuient un programme statistique particulier (c.-à-d. une enquête quelconque). L'essence même des enquêtes suppose qu'elles doivent être effectuées à des moments très précis. Cette règle est valable dans tous les cas, qu'il s'agisse d'une enquête mensuelle, trimestrielle ou annuelle, ou d'une partie du Recensement. Faire des compromis dans le cas du calendrier de réalisation d'un tel logiciel est virtuellement impossible. Par contre, d'autres projets s'accommodent de calendriers plus souples et peuvent survivre à des modifications importantes, sans incidence négative sur la réussite du projet.

32. Pour équilibrer le triangle des contraintes, il est particulièrement important de pouvoir *réutiliser* des composantes du logiciel. La réutilisation d'une composante existante permet de stabiliser les coûts, de respecter des délais stricts et d'assurer une fonction essentielle. Même lorsqu'il faut modifier les exigences pour qu'elles correspondent à la composante réutilisable, ce type de compromis se révèle souvent intéressant.

**Figure 5-1. Triangle des contraintes du logiciel**



(Source : McConnell 1996)

33. À Statistique Canada, la source de financement des projets logiciels constitue parfois un facteur de détermination des compromis possibles sur le plan des coûts. Un budget illimité permettrait d'obtenir un logiciel dont la qualité serait de beaucoup supérieure. Toutefois, à titre d'organisme responsable, nous avons l'obligation, par respect pour les contribuables, d'utiliser efficacement les fonds disponibles.

34. Les budgets de certains projets logiciels sont fixes et couvrent une partie ou l'ensemble d'un exercice (avril à mars), alors que ceux d'autres projets peuvent s'étendre sur plusieurs années. Certains projets reçoivent leurs budgets de sources particulières et doivent utiliser ce financement pour atteindre des objectifs particuliers. D'autres obtiennent leurs budgets de sources de financement plus larges et ont beaucoup plus de contrôle sur la quantité et l'étalement des ressources qui leur sont consacrées.

35. Dans la plupart des cas, malgré un financement et des délais fixes, il demeure possible de produire un logiciel de grande qualité. Pour obtenir un logiciel de qualité, il faut planifier le processus de réalisation tout comme il faut planifier la conception de la méthodologie pour obtenir des données de qualité. Les mêmes contraintes de coûts et de calendrier existent dans les deux cas. On peut atteindre un niveau de qualité élevé si l'on consent à recourir à de nouvelles pratiques fondées sur d'excellents principes éprouvés. Toutefois, il faut un engagement de tous les intervenants et non seulement de ceux qui favorisent le recours à ces pratiques.

#### **A. Compromis et priorités**

36. Une fois connues les contraintes associées à la qualité du logiciel, il faut déterminer les caractéristiques que les intervenants du projet jugent les plus importantes. Compte tenu de la diversité (et, parfois, des conflits) des points de vue, ce processus permet de définir un ensemble de caractéristiques qui conviennent le mieux aux besoins globaux du projet de TI. Ainsi, on s'assure de tenir compte de la perspective de chacun des intervenants et que tous aient à cœur de voir le projet atteindre ses objectifs de qualité.

37. En attribuant des priorités à certaines caractéristiques, les intervenants auront une meilleure idée de leurs attentes à l'égard du produit fini. La connaissance de la date d'échéance éventuelle d'une enquête (p. ex., la date de mise à la poste) permet de déterminer à quel niveau du produit logiciel des compromis sont possibles. Par exemple, on peut réduire la complexité de certaines fonctions et la priorité accordée à l'efficacité et à la robustesse de la solution. Si une solution de TI doit être très facile à maintenir (p. ex., pour permettre de futures améliorations prévues), on doit alors accorder une priorité plus élevée à la lisibilité et la testabilité. Dans le cas d'un logiciel que les utilisateurs finals ne connaissent pas bien (p. ex., un nouveau système de comptabilisation des heures), la facilité d'utilisation et la sécurité deviennent importantes. Les réalisateurs du logiciel quittent le projet? Alors l'interprétabilité (c.-à-d., notes, spécifications de conception, manuels d'utilisateur et structures de fichier) du logiciel devient importante puisqu'il faut assurer la continuité du projet.

38. Les intervenants doivent tenir compte des préoccupations sur le plan des contraintes, des compromis nécessaires et des priorités. Ils seront alors mieux en mesure de connaître les caractéristiques de qualité offertes par les produits logiciels.

## **VI. INFLUENCES SUR LA QUALITÉ DU LOGICIEL**

39. Une fois qu'un projet a été lancé, plusieurs facteurs influent sur la qualité du produit fini. Quel que soit le type de projet, il subit l'influence des quatre facteurs importants suivants : les personnes, le processus, le produit et la technologie.

*Les personnes deviennent efficaces rapidement ou lentement. Un processus soit permet de maximiser le temps des personnes, soit crée des obstacles les uns après les autres. Un produit est soit défini d'une manière telle qu'il se crée presque lui-même, soit défini d'une façon qui nuit aux meilleurs efforts de ses créateurs. Une technologie soit contribue aux travaux de réalisation, soit nuit aux meilleures tentatives du réalisateur. (Source : McConnell 1996)*

40. Il est possible, quoique non désirable, de s'attarder davantage à un facteur au détriment des autres. Il est toutefois préférable de les envisager tous comme des dimensions différentes et de les gérer efficacement si l'on veut assurer la qualité du logiciel.

### **A. Personnes**

41. Les personnes assignées à un projet de réalisation de logiciel ont une grande influence sur sa réussite. Ce sujet est très complexe et nous ne pouvons pas l'aborder adéquatement dans ce document (voir la section Lectures complémentaires). En résumé, on peut dire que les activités de sélection du personnel et d'organisation et de motivation de l'équipe peuvent avoir une incidence majeure sur le résultat d'un projet logiciel. De plus, le personnel doit être sensibilisé aux aspects, pratiques et méthodes d'assurance de la qualité du logiciel et bien les maîtriser.

### **B. Processus**

42. Les processus de réalisation et de maintenance des produits logiciels ont été étudiés en profondeur. Des organismes tels que l'Institut de génie logiciel (SEI), le Project Management Institute (PMI) et l'Institute of Electrical and Electronics Engineers (IEEE), ainsi que d'autres organisations, les ont examinés et ont documenté les preuves qui justifient leur utilisation. Les secteurs clés (KPA) du modèle de stabilisation des capacités (Capability Maturity Model<sup>SM</sup> - CMM) (SEI 1993) expliquent en détail un ensemble de processus qui aident à améliorer la qualité des produits logiciels. Certains des processus les plus efficaces du modèle de stabilisation sont les suivants :

- Assurance de la qualité du logiciel
- Gestion des exigences
- Examens par les pairs
- Planification de projet logiciel
- Suivi et surveillance de projet
- Gestion de la configuration du logiciel

43. Dans certaines situations, Statistique Canada utilise efficacement plusieurs de ces processus. Toutefois, il faudrait davantage de normalisation, de formation et de partage d'expériences. Le Groupe de travail sur les pratiques exemplaires logicielles de SC a pour objectif de s'intéresser à ces préoccupations.

### **C. Produit**

44. La taille du produit logiciel est l'aspect qui influe le plus sur sa qualité. Il est plus difficile d'assurer la qualité d'un projet logiciel de grande envergure. Pour les plus petits projets, cet objectif est plus facile à atteindre. Pour assurer la réussite des projets à coûts et calendrier fixes, on doit

habituellement ajuster la taille et la portée du logiciel. Bien estimer la taille du produit permet de déterminer un ensemble de fonctions réalistes pour le logiciel.

#### **D. Technologie**

45. Il faut toujours choisir le bon outil pour un projet de TI. Cette responsabilité relève normalement des réalisateurs et des chefs de projet de TI puisqu'ils possèdent le plus d'expérience dans le domaine des technologies de l'information. Le choix d'un outil ou d'une technique plus efficace peut grandement accroître la productivité du réalisateur ainsi que la qualité du logiciel. Il vaut la peine d'envisager l'utilisation de technologies telles que Visual Basic, Java et les méthodologies orientées objet, ainsi que la réutilisation du logiciel. Le recours à des approches standard à l'échelle du Bureau (c.-à-d., utilisation accrue de SAS et moins grande diversité de logiciels) permet aux réalisateurs d'utiliser leurs compétences pour de nombreux types différents de projets logiciels. Plus les technologies de réalisation sont productives, plus les responsables disposent de temps pour réaliser les objectifs de qualité du logiciel.

### **VII. TECHNIQUES D'AMÉLIORATION DE LA QUALITÉ DU LOGICIEL**

*L'assurance de la qualité du logiciel est un programme d'activités planifié et systématique conçu pour faire en sorte qu'un système donné possède toutes les caractéristiques désirées. Bien qu'il semble que la meilleure façon de réaliser un produit de haute qualité soit de mettre l'accent sur le produit lui-même, dans le secteur de l'assurance de la qualité du logiciel, la meilleure approche consiste à mettre l'accent sur le processus. (McConnell 1993)*

46. Il est important de savoir où concentrer ses efforts lorsque l'on tente de réaliser un logiciel de grande qualité. Comme le mentionne McConnell, il faut favoriser le processus et non nécessairement le produit. Cette section décrit certaines techniques qui permettent d'améliorer la qualité du logiciel. Les descriptions donnent uniquement un aperçu des techniques. Pour plus de détails, consultez les documents indiqués en référence dans la section Lectures complémentaires.

47. **Objectifs de la qualité du logiciel.** L'établissement d'objectifs explicites de qualité sous-entend que chaque intervenant dans le projet logiciel connaît les objectifs visés. Si tous les intervenants connaissent ces objectifs (et que ceux-ci sont raisonnables), il est fort probable qu'ils arriveront à les réaliser. L'identification explicite de tous les intervenants et une bonne connaissance de leurs priorités sur le plan de la qualité constituent une étape importante de tout projet logiciel. Il s'agit d'une technique puissante, que l'on peut facilement utiliser au début d'un projet et qui doit être révisée au besoin.

*Au cours d'une réunion préliminaire de l'un de mes récents projets, le gestionnaire de projet a informé les intervenants qu'il fallait réutiliser le logiciel existant dans la mesure du possible. Le parrain ne désirait pas « réinventer la roue » et disait vouloir s'en tenir au modèle de données conceptuelles actuel lorsque cela est possible. La priorité élevée accordée à la réutilisabilité a permis à tous les participants au processus de conception d'envisager toutes les options en fonction de cette priorité. Plusieurs conceptions ont été envisagées; toutefois, celle retenue pour le projet était celle qui favorisait la réutilisation d'une grande partie des composantes et des concepts existants. Les intervenants connaissaient les objectifs de qualité et ont été en mesure de s'y conformer et de les intégrer à leur travail.*

48. **Activité explicite d'assurance de la qualité** Assurer la qualité du logiciel doit être prioritaire. Lorsque la direction investit dans un processus explicite d'assurance de la qualité, elle indique clairement à tous les intervenants que la qualité est une priorité. La base de connaissances en matière d'assurance de la qualité est considérable. De nombreuses techniques assurent la qualité au cours du processus de réalisation. Il suffit simplement de les apprendre, de les mettre en pratique et de les adapter.

49. **Stratégie de test.** Trop fréquemment, la responsabilité des tests est laissée à la discrétion des réalisateurs et des utilisateurs. Les réalisateurs effectuent rarement des essais efficaces de leur propre travail puisqu'ils manquent simplement d'objectivité pour repérer et signaler les erreurs. Les utilisateurs remplissent ce rôle efficacement mais, malheureusement, ils sont également susceptibles de ne pas relever certaines déficiences. Ils mettent souvent l'accent sur des scénarios qu'eux seuls jugent

importants et qui ne sont pas exhaustifs. Plusieurs erreurs non détectées se glissent donc dans le logiciel. Dans les pires scénarios, les réalisateurs effectuent peu de tests en mentionnant que cette responsabilité relève de l'utilisateur. D'autre part, les responsables de certains projets adoptent une stratégie de tests qu'ils respectent pendant toute la durée du projet. Ils peuvent ainsi identifier en détail tous les scénarios qui doivent faire l'objet de tests, déterminer qui en est responsable puis documenter les résultats. Habituellement, de tels projets logiciels atteignent une plus grande qualité globale.

50. **Lignes directrices en matière de génie logiciel.** La base de connaissances dans le domaine du génie logiciel est tout simplement famineuse. Néanmoins, il est essentiel de posséder des connaissances sur certaines pratiques et certains concepts fondamentaux (sans égard au projet ou à la technologie). Ces lignes directrices s'appliquent à l'ensemble du processus de réalisation et incluent les conventions d'attribution de noms, l'analyse des besoins, la conception de système, la création du logiciel et les essais de système. En communiquant ces lignes directrices à votre équipe de réalisation, vous améliorerez leur productivité ainsi que la qualité résultante du logiciel.

51. **Examens techniques.** Les erreurs repérées tôt durant le processus sont faciles et moins coûteuses à corriger. Les examens techniques formels et informels sont deux méthodes parmi les plus efficaces d'amélioration de la qualité du logiciel. Les examens informels effectués par les réalisateurs permettent d'identifier des erreurs et des omissions évidentes avant l'examen formel. Les examens formels remplissent également ce rôle en plus de permettre l'identification d'erreurs critiques bien avant qu'elles ne puissent causer de problème aux autres intervenants. En plus de réduire les erreurs, les examens techniques permettent d'améliorer le code source. Ils peuvent améliorer la qualité de certaines caractéristiques du code source telles que la modularité, la cohésion et le couplage, le style, la lisibilité, la nature générale des interfaces, etc.

52. **Gestion du risque.** La gestion des risques qui peuvent se manifester au cours d'un projet a un effet positif sur la qualité de celui-ci. L'identification et la suppression fiables et prévisibles des risques permettent d'obtenir un logiciel également fiable et prévisible.

53. **Procédures de contrôle des modifications.** Des modifications non contrôlées peuvent très facilement entraîner une dégradation de la qualité du logiciel. Plusieurs procédures différentes utilisées par les réalisateurs et les chefs de projet de TI de Statistique Canada permettent de réduire les répercussions des modifications sur le logiciel. La taille de la majorité des projets entrepris à SC est suffisamment petite pour permettre une gestion informelle des modifications qui donne des résultats satisfaisants. Par contre, l'absence de toute procédure formelle de contrôle des modifications pourrait facilement entraîner l'effondrement de certains projets de plus grande envergure. Selon la taille et la complexité de votre projet, une procédure connue et documentée de contrôle des modifications vous permettra d'éviter les embûches.

54. **Mesure des résultats.** La qualité du logiciel est liée aux estimations utilisées pour en déterminer la taille et établir l'ampleur des ressources nécessaires à sa réalisation. Ces estimations sont davantage précises si elles s'appuient sur une expérience antérieure. À Statistique Canada, plusieurs employés ont participé à un projet durant un certain nombre de cycles d'enquête. Ils sont en mesure d'offrir des estimations relativement précises. Toutefois, nous devons composer avec un environnement où on encourage les professionnels de participer à une programme d'affectation de développement en rotation, donc, nous avons besoins des meilleurs méthodes d'enregistrer et communiquer l'expérience. En mesurant et enregistrant les résultats des projets de réalisation de logiciel, nous pouvons éviter les problèmes associés ou roulement élevé du personnel.

55. **Prototypage.** Utilisé fréquemment à Statistique Canada, le prototypage d'un produit logiciel très tôt au début de sa réalisation donne aux intervenants un premier aperçu de la solution proposée. Il est très utile pour recueillir des commentaires des utilisateurs et mieux cibler les exigences du logiciel. Son seul inconvénient est de laisser croire trop facilement que le prototype est le produit final. La création de prototypes ne devrait pas exiger trop de ressources et ceux-ci devraient être détruits dès qu'ils ont permis

d'atteindre l'objectif visé. La connaissance acquise au cours du processus de prototypage peut servir ensuite à créer le produit logiciel réel.

56. En examinant cette liste de techniques, les observations ci-dessous s'imposent :
- il existe de nombreuses techniques que l'on peut utiliser pour améliorer la qualité du logiciel;
  - ces techniques offrent toutes des approches raisonnables de réalisation et de maintenance du logiciel;
  - certaines d'entre elles sont utilisables immédiatement, sans l'aide d'outils complexes (ou coûteux); et
  - l'amélioration de la qualité du logiciel relève davantage des personnes et du processus que du produit et de la technologie.

## VIII. CONCLUSION

57. Ce document vise à donner un aperçu de la qualité du logiciel à Statistique Canada. Un certain nombre de caractéristiques aident à définir ce qu'est la qualité du logiciel et à mieux quantifier les objectifs de qualité des projets de réalisation et de maintenance de logiciel du Bureau.

58. La qualité du logiciel peut être envisagée selon de nombreux points de vue différents. Du parrain de projet aux utilisateurs finals, tous possèdent une perspective qui leur est propre. Il est essentiel d'en tenir compte lorsqu'on décide des caractéristiques jugées importantes pour la qualité globale du logiciel.

59. Plusieurs conflits peuvent surgir lorsque l'on tente de réaliser et de maintenir un logiciel de grande qualité. Les coûts et calendriers fixes exigent des intervenants qu'ils attribuent des priorités à certaines caractéristiques et acceptent les compromis qui en découlent.

60. Les influences qu'exercent les personnes, le processus, le produit et la technologie peuvent être utilisées à notre avantage et permettre de produire un produit logiciel de grande qualité. Apprendre à tirer avantage de tous les aspects de ces influences constitue un atout majeur pour la réalisation des objectifs de qualité des projets de TI.

61. L'amélioration de la qualité du logiciel requiert l'utilisation de techniques saines et pratiques. Nous disposons de plusieurs de ces techniques qui peuvent influencer immédiatement sur le produit et qui exigent un minimum d'effort.

## IX. LECTURES COMPLÉMENTAIRES

### A. Références:

**Boehm**, B. [et al]. 1978. *Characteristics of Software Quality*. New York: North Holland. Chapter 3.

**Brackstone**, G. 1999. "Managing Data Quality in a Statistical Agency." *Survey Methodology* (Statistics Canada Catalogue no. 12-001-XIB). Ottawa: Minister responsible for Statistics Canada, (December 1999): pg. 139-149.

**DeMarco**, T. and Lister, T. 1999. *Peopleware: Productive Projects and Teams, 2nd ed.* New York: Dorset House. pg. 19-22.

**Gillies**, A. 1992. *Software Quality: Theory and Management*. London: Chapman. pg. 4-29.

**Institute of Electrical and Electronics Engineers** (IEEE). 2000. *Guide to the Software Engineering Body of Knowledge (Stone Man Version 0.7)*. <http://www.swebok.org/>: Released April 2000. (Accessed October 2, 2000).

**McCall, J.** [et al] 1977. *Factors in Software Quality*. NTIS AD-A049-014, Rome Air Development Center.

**McConnell, S.** 1993. *Code Complete*. Redmond, WA: Microsoft Press. pg. 557-571.

**McConnell, S.** 1996. *Rapid Development*. Redmond, WA: Microsoft Press. pg. 11-17, 126-127.

**Software Engineering Institute (SEI).** 1993. *Key Practices of the Capability Maturity Model, Version 1.1* (CMU/SEI-93-TR-25). Carnegie Mellon University. Pittsburgh, PA.

**Statistics Canada.** 1993. *Information Technology Framework (ITF): Project Management*. Informatics Branch. Released October 1993.

**Statistics Canada.** 1996. *Information Technology Framework (ITF)*. Informatics Branch. Released March 1996.

**Statistics Canada.** 2001. *Information Technology Framework (ITF)*. Informatics Branch. Released June 2001.

**Statistics Canada.** 2002. *Quality Assurance Framework. Methods and Standards Committee*, Statistics Canada.

**Statistics Canada.** 1998. *Statistics Canada Quality Guidelines: Third Edition - October 1998*, (Statistics Canada Catalogue no. 12-539-XIE). Ottawa: Minister responsible for Statistics Canada: pg. 74-80.

**Statistics Canada.** 2000. *The Statistics Act, section 3a*. Internal Communications Network (ICN). [http://www.statcan.ca/10/10\\_006\\_e.htm](http://www.statcan.ca/10/10_006_e.htm): Last Modified February 10, 2000, STC Intranet. (Read October 13, 2000).

**Walters, G. and McCall, J.** 1979. "Software Quality Metrics for Life Cycle Cost-Reduction." *IEEE Transactions on Reliability*. Vol. R-28, No. 3 (August 1979). pg. 212-219.

**Wieggers, K.** 1999. *Software Requirements* Redmond, WA: Microsoft Press. pg. 195-206.

## **B. Aspects liés à Peopleware**

**Constantine, L.** 1995. *Constantine on Peopleware*. Englewood Cliffs, NJ: Yourdon Press

**DeMarco, T and Lister, T.** 1999. *Peopleware: Productive Projects and Teams, 2nd ed.* New York: Dorset House.

## **C. Stratégies de tests**

**Myers, G.** 1979. *The Art Of Software Testing*. New York, Toronto. Wiley.

## **D. Examens de logiciel**

**Yourdon, E.** 1985. *Structured Walkthroughs*. New York, NY: Yourdon Press

## **E. Techniques de réalisation de logiciel**

**McConnell, S.** 1993. *Code Complete*. Redmond, WA: Microsoft Press.

**McConnell, S.** 1996. *Rapid Development*. Redmond, WA: Microsoft Press.

Tous les ouvrages mentionnés ci-dessus sont disponibles à la bibliothèque de Statistique Canada (<http://lib2.statcan.ca/>). La Bibliothèque possède un fond de documents sur la technologie de l'information très riche en ouvrages de toutes sortes (dont certains sont disponibles également en français) sur la réalisation de logiciel de qualité. La consultation de ce fond a été précieuse pour la recherche nécessaire à la préparation de ce document.

- - - - -