UNIVERSITÉ DU QUÉBEC À MONTRÉAL

AUTOMATISATION DU MODÈLE QEST

MÉMOIRE
PRÈSENTÉ
COMME EXIGENCE PARTIELLE
DE LA MAITRISE EN INFORMATIQUE :
GÉNIE LOGICIEL

PAR
MYRIAM BEN EZZEDDINE

AVRIL 2000

REMERCIEMENTS

Je remercie tous ceux qui m'ont aidé de près ou de loin à réaliser ce mémoire et en particulier mon directeur de recherche M. Alain Abran, professeur au Laboratoire de recherche en gestion des logiciels à l'Université du Québec à Montréal, par son encadrement et son apport significatif à la réalisation de ce mémoire.

Tous mes remerciements à M. Luigi Buglione de European Software Institute pour son aide précieuse.

Je voudrais remercier aussi Michèle Hébert, François Cossette et tous les amis du Laboratoire de recherche en gestion des logiciels de l'UQAM qui m'ont aidé à mener à bien ce travail.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ACRONYMES

CV	Valeur de la caractéristique (Charactéristic Value)
E	Domaine économique
MQL	Taux moyen de qualité (Mean Quality Level)
P(E)	Priorité selon le domaine économique
P(S)	Priorité selon le domaine social
P(T)	Priorité selon le domaine technique
PW	Pondération des priorités (Priority Weight)
QF	Facteur de qualité (Quality Factor)
QL	Taux d'objectivité des mesures (Quantity Level)
S	Domaine social
SCV	La moyenne des points de vues des participants pour chaque caractéristiques
SSV	Somme des valeurs des sous-caractéristiques (Sum of Subcharacteristics Value)
T	Domaine technique
TCV	Valeur totale des caractéristiques (Total Characteristics Value)
W	Poids (Weight)

RÉSUMÉ

Luigi Buglione, chercheur à l'ESI (European Software Institute) en Espagne et Alain Abran, professeur et directeur du Laboratoire de recherche en gestion des logiciels à l'UQAM, ont développé le modèle QEST(Quality factor, Economic, Social and Technical dimensions) pour représenter en trois dimensions la performance d'un projet de développement d'un logiciel. Ce modèle intègre, en une seule représentation, les dimensions économique, social et technique pour mesurer et évaluer la qualité d'un logiciel tout en considérant les aspects quantitatif et qualitatif. La base théorique géométrique de ce modèle multidimensionnel est très solide, mais forcément très complexe. De plus, les procédures d'utilisation sont non seulement manuelles, mais également nombreuses et sujettes par conséquent à des erreurs de manipulation. Au niveau pratique, ce modèle est donc difficile à comprendre et présente certains inconvénients.

Ce modèle très complexe ne pouvait donc être utilisé que par quelques rares experts du domaine au niveau international. Une utilisation plus courante en industrie demandait donc que toutes les parties complexes du modèle QEST soient identifiées et même cachées en arrière plan de façon à créer une interface beaucoup plus conviviale pour les utilisateurs de façon à leur permettre de focuser leur attention sur la collecte de données, somme toute relativement simple chacune, mais dont la manipulation subséquente était hautement complexe, et par conséquent très problématique.

Le but de ce mémoire consiste à concrétiser ce modèle en une application sur Internet qui permet de faciliter la compréhension et l'utilisation de QEST et d'automatiser le traitement des fonctions manuelles.

CHAPITRE I

INTRODUCTION

De nos jours, le domaine de l'informatique est en perpétuelle évolution et il y a de plus en plus de systèmes et logiciels sur le marché. Pour évaluer la qualité de ces produits, il est nécessaire d'avoir des fonctions d'analyse qui nous permettent de distinguer les bons systèmes des mauvais selon les besoins des entreprises et des consommateurs. Le meilleur moyen pour définir la qualité et la performance d'un système est d'utiliser un ensemble de fonctions qui permettent d'évaluer quantitativement cette performance, lesquelles fonctions peuvent être regroupées dans un modèle, lequel modèle devrait être ensuite automatisé par un outil logiciel.

Le modèle QEST est un modèle qui nous permet d'avoir une représentation quantitative de la qualité et de la performance. Ce modèle QEST peut également nous informer sur la performance d'un système par rapport à la performance maximale potentielle en considérant les côtés économique, social et technique. Notre travail consiste à faciliter l'utilisation de ce modèle.

1.1. LOGICIEL DE MESURAGE

Les entreprises de développement de logiciels et de systèmes sont confrontées à plusieurs questions telles que :

- Est-ce que les résultats désirés et les objectifs sont atteints?
- Les clients sont-ils satisfaits de nos produits et services?
- Les coûts de production peuvent-ils être réduits?
- Comment perfectionner le produit selon les besoins des clients et les nouvelles technologies?

Pour répondre à ces questions nous avons besoin de certaines informations qui ne peuvent être obtenues que par mesurage c'est-à-dire par compréhension quantitative du logiciel. Pour ceci on utilise, dans le domaine de l'ingénierie, des méthodes qui sont basées sur des modèles et des théories. Le mesurage n'est pas une tâche à prendre à la légère et pouvant être facilement effectuée sans la connaissance au préalable de certains principes pour le choix des mesures et de la façon d'établir un programme de mesure dans une organisation.

Les mesures sont de plus en plus reconnues comme importantes dans la prise de décisions des organisations durant le cycle de vie des systèmes ou logiciels. Elles peuvent être utilisées pour planifier et contrôler les coûts d'un projet, sa qualité, ses ressources, etc. Elles permettent aux gestionnaires de prendre des décisions sur le statut et la performance du projet ainsi que sur les programmes d'amélioration de sa qualité.

D'autre part, les logiciels de mesure sont des outils logiciels capables de donner les informations nécessaires pour pouvoir, par la suite, prendre les décisions adéquates au cours du cycle de vie du système à mesurer. Ils aident à cerner plus facilement les faiblesses du produit afin de cibler les fonctions à améliorer.

Les outils logiciels de mesure doivent satisfaire certaines caractéristiques, par exemple :

- les mesures doivent être robustes et précises;
- la collecte des données pour les mesures doit être facile et simple;
- les valeurs des mesures doivent être estimables pour pouvoir être par la suite comparées avec les mesures actuelles.

Les principes d'un « bon » modèle d'évaluation selon M. Hatfield sont les suivants :

- les mesures doivent correspondre aux buts de l'organisation; requièrent des définitions opérationnelles claires;
- elles doivent tenir compte des points de vues et des besoins des utilisateurs, techniciens et gestionnaires du logiciel;
- les mesures doivent être objectives; doivent s'appliquer durant le cycle de vie du logiciel.

De plus, dans une organisation, il est important qu'une personne ou un groupe de personnes prenne la responsabilité de ce modèle d'évaluation pour identifier les mesures à collecter, contrôler leurs cohérence avec le projet et l'organisation et fournir des données historiques que les gestionnaires peuvent utiliser pour comparer, planifier et contrôler leurs projets.

1.2. BUT DES PROGRAMMES DE MESURE

Un programme de mesure en génie logiciel est un ensemble structuré de mesures liées à des buts et des objectifs quantifiables. La mise en place d'un programme de mesure doit être vue comme un projet, avec des buts précis, un plan pour rencontrer les objectifs et des ressources pour mettre en œuvre le plan. Un programme de mesure fournit aux informaticiens et aux gestionnaires des informations qui les aident à prendre des décisions, comprendre, contrôler et améliorer le système, établir des objectifs quantifiables, identifier l'endroit où les changements sont nécessaires [Desharnais, 1994].

Les organisations ayant un outil de mesure performant bénéficient de ces avantages :

- un aperçu global sur le développement du logiciel;
- capacité de quantifier les décisions;

- meilleurs planification et contrôle des projets;
- meilleure compréhension du processus de développement du logiciel et de l'environnement de développement;
- identification des domaines d'amélioration;
- amélioration de la communication au sein de l'organisation.

1.3. LE MESURAGE

Le mesurage n'est pas très répandu dans le développement des logiciels et systèmes. Il est considéré comme une fonction, parmi beaucoup d'autres, dans le processus de développement du logiciel. Le mesurage d'un système est lui même un système qui n'est pas moins complexe, bien au contraire [Morisio, 1999].

Dans le Capability Maturity Model du Software Engineering Institute, le mesurage est une clé pour comprendre et contrôler les systèmes au moyen de données quantitatives. Le mesurage est une application difficile et demande un effort et du savoir-faire. Une organisation qui veut établir un programme de mesurage se heurte à des problèmes pour sélectionner un nombre raisonnable de mesures, à partir des métriques proposées dans la littérature, et pour trouver les outils nécessaires pour les calculer et les collecter.

Les processus de développement varient d'une organisation à une autre et une organisation peut même utiliser différents modèles de développement. Vu que ces modèles se différencient par leurs complexités, leurs objectifs finaux et leurs contraintes, il n'est pas possible d'utiliser le même ensemble de mesures pour tous les systèmes. Les standards tels que IEEE et ISO reconnaissent qu'il est impossible de définir un ensemble fini de mesures [Morisio, 1999] et ils laissent aux utilisateurs le soin de déterminer eux-mêmes les métriques qui leur semblent les plus appropriées. Basili et Rombach définissent l'approche GQM (Goal Question Metric) pour sélectionner et définir les mesures adéquates. GQM est un moyen de définir les buts à atteindre dans un projet. Une fois ces buts spécifiés, ils sont utilisés pour

générer des questions sur la démarche à suivre pour les atteindre. Ces questions permettent à l'utilisateur d'identifier les mesures dont il a besoin pour y répondre.

La figure1.1, nous présentons une vue globale du processus d'un logiciel de mesure de qualité.

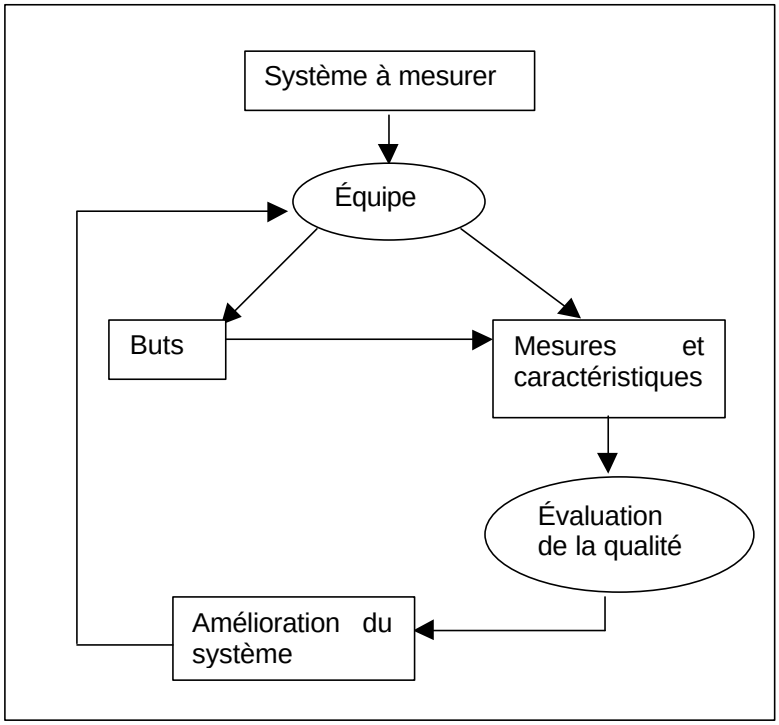

Figure 1.1 Processus général d'un logiciel de mesure de qualité

1.4. LE MODÈLE QEST

Luigi Buglione, chercheur à l'ESI (European Software Institute) en Espagne et Alain Abran, professeur et directeur du Laboratoire de recherche en gestion des logiciels à l'UQAM, ont développé une méthode pour représenter en trois dimensions la performance de logiciels et de systèmes. Cette méthode est appelée QEST (Quality factor, Economic, Social and Technical dimensions) et elle donne le moyen

d'intégrer en une seule représentation les dimensions économique, sociale et technique pour mesurer et évaluer un logiciel en considérant l'aspect quantitatif et qualitatif.

Trois études ont été réalisées sur la mesure multidimensionnelle de la performance des logiciels. La première utilise une approche vectorielle [Gonzales, 1995], la deuxième se base sur une représentation cubique [Hatfield, 1995] et la troisième utilise les graphes de Kiviat [Donaldson et al., 1997].

Le modèle QEST est un modèle ouvert permettant une représentation multidimensionnelle de plusieurs mesures. Il permet d'exprimer la performance d'un système comme une combinaison de mesures spécifiques et non prédéterminées par le modèle même, pour chacune des trois dimensions distinctes mais inter reliées qui sont : économique, sociale et technique. La performance est donnée par une seule valeur qui intègre ces trois dimensions.

Le modèle QEST se base sur la spécification et l'évaluation de la qualité. La spécification de la qualité signifie l'établissement des besoins de la qualité du système. Une organisation utilisant QEST doit définir les ratios à utiliser ainsi que leurs valeurs dans le système et leurs valeurs maximale et minimale pour chacun d'entre eux. De plus, les personnes chargées de donner leurs opinions sur le système à évaluer sont conviées à donner leurs points de vues sur les différentes caractéristiques de qualité du système.

Dans QEST, l'évaluation de la qualité et la performance permet de vérifier si un projet est en bonne position par rapport à la performance maximale et nous permet d'analyser les résultats pour savoir quel domaine (économique, social ou technique) est à reconsidérer pour améliorer la qualité. Pour obtenir la valeur de la performance, une représentation géométrique et numérique en trois dimensions est utilisée.

Pour arriver au résultat de notre mesure, plusieurs traitements importants de calcul doivent être effectués tel que la normalisation des mesures, la détermination des coordonnées et équations des hyperplans, calcul de volume, etc.

Les procédures d'implantation du modèle QEST sont basées sur les étapes du cycle PMAI (Plan, Measure, Assess, Improve) que nous résumons en trois parties dans les sous-sections suivantes.

1.4.1. Collecte des données et leurs valeurs

Les organisations peuvent choisir leur propres composants pour chacune des dimensions économique, sociale et technique, selon leurs besoins et leurs buts.

Il faut aussi établir la liste des mesures, par exemple en utilisant les normes ISO 9126 et les points de fonction (pour mesurer la taille fonctionnelle du logiciel). Pour la dimension sociale, ces mesures seront dégagées à partir d'un questionnaire structuré selon les principes de statistiques et de la méthodologie de recherche sociale, lequel questionnaire aura été rempli précédemment par les utilisateurs du logiciel.

QEST exige de diviser les besoins de qualité d'un système en ratios (à partir des mesures), caractéristiques et sous-caractéristiques de qualité dans le cadre du modèle de qualité proposé par ISO 9126. Une fois que les valeurs des ratios sélectionnées pour chacune des trois dimensions et les points de vues des personnes sur les sous-caractéristiques du système sont ajoutés, QEST, en traitant ces données va exprimer le facteur de qualité et la performance du système à mesurer comme étant la combinaison de celles-ci.

1.4.2. Calcul du QF

La qualité d'un produit est l'ensemble des caractéristiques ou services que possède ce dernier. Le facteur de qualité permet l'évaluation qualitative d'un logiciel en considérant les trois domaines c'est-à-dire les opinions des utilisateurs, des développeurs et des gestionnaires sur la qualité du système. Pour arriver à la valeur du facteur de qualité, nous devons remplir plusieurs tableaux et faire un certain nombre de calcul.

1.4.3. Calcul de la Performance

Connaissant le facteur de qualité QF, il est ensuite possible de calculer la performance à l'aide du modèle QEST en considérant les trois points de vues E, S et T. Le traitement mathématique pour atteindre la valeur de la performance est particulièrement complexe et monotone.

1.5. CONCLUSION

Les différentes étapes du modèle QEST ne sont pas faciles à comprendre ni à pratiquer. La base théorique géométrique de ce modèle multidimensionnel est très solide, mais forcément très complexe. De plus les procédures d'utilisation sont également non seulement manuelles, mais également nombreuses et sujettes par conséquent à des erreurs de manipulation. Nous détaillons ces procédures dans le chapitre 2 en énumérant les inconvénients de celles-ci et en déterminant notre objectif de faciliter l'utilisation du modèle. Nous présentons aussi les méthodes utilisées par QEST dans le chapitre 3. Par la suite, dans le chapitre 4, nous spécifions notre démarche qui nous a permis de simplifier le modèle QEST pour les utilisateurs. Nous consacrons aussi une partie sur les détails de l'implantation sur

technologie web et aux difficultés rencontrées lors de la programmation. Enfin, le dernier chapitre nous donne une idée globale sur tout notre travail et les travaux futurs qui permettront d’améliorer la méthode QEST.

CHAPITRE II

ANALYSE DES PROCÉDURES D'IMPLANTATION DU MODÈLE QEST

La performance est définie comme étant le degré que possède un système ou un composant d'accomplir ses fonctions désignées selon les contraintes données [IEEE, 1990].

Dans le modèle QEST, la mesure de la performance est donnée par l'intégration de la mesure du processus, qui est exprimée par la productivité brute ajustée par un facteur de qualité exprimé par une évaluation subjective de cette dimension de qualité. Ces mesures nécessitent un traitement de données assez important avec l'utilisation de plusieurs tableaux.

Le modèle QEST, tel que défini initialement par Luigi Buglione et Alain Abran, est d'une utilisation manuelle et peu pratique. Il requiert des efforts d'attention et de calcul considérables de la part de l'utilisateur. Dans ce chapitre, nous présentons les étapes du modèle, les inconvénients qui sont présents dans le déroulement de QEST et l'objectif de réaliser un modèle QEST automatisé afin d'en faciliter l'utilisation par un grand nombre de gestionnaires.

2.1. INCONVÉNIENTS DU MODÈLE QEST

Dans ce qui suit, nous décrivons la complexité du modèle QEST : la gestion de plusieurs tableaux, le nombre important d'opérations mathématiques pour le calcul de la valeur de la performance et l'effort nécessaire pour accomplir ces traitements.

2.1.1. Utilisation des tableaux selon les procédures Buglione et al.

Pour calculer la performance d'un système, le modèle QEST propose d'utiliser un ensemble de six tableaux dans lesquels sont inscrits les différents résultats obtenus à chaque étape.

Le premier tableau (Tableau 2.1), consiste à définir les mesures à utiliser. Ces mesures sont collectées à partir de questionnaires et selon les opinions des utilisateurs des trois domaines : économique, social et technique. Pour chaque mesure, nous avons à déterminer son unité de mesure et sa valeur mesurée dans le projet.

Tableau 2.1 Liste des mesures du projet

Numéro	Mesures	Définition	Valeurs
M1			
M2			
. . .	. . .	. . .	. . .
Mn			

Dans le deuxième tableau (Tableau 2.2), il faut construire tous les ratios qui seront utilisés pour un modèle QEST. Pour chaque mesure, il faut préciser la ou les mesures les plus importantes qui vont permettre d'établir ces ratios. Les mesures les plus importantes sont sélectionnées à partir des questionnaires. Plus une mesure est choisie par les utilisateurs, plus elle est importante. Dans la colonne 5, l'utilité de chaque ratio est décrite. Ensuite, il faut préciser dans la colonne 6 la ou les dimensions auxquelles appartiennent ces ratios. Dans les colonnes 7 et 8 il faut fixer les valeurs minimales et maximales que peuvent avoir chaque ratio, celles-ci seront par la suit utilisées pour la normalisation des valeurs du projet qui auront auparavant été inscrites dans la colonne 9. Les résultats de normalisation doivent être mis dans

la colonne 10 du tableau 2. Valeur normalisée = (Valeur projet – Valeur min.) / (Valeur max. – Valeur min.)

La dernière colonne du tableau 2.2 concerne le processus d'amélioration de la qualité impliqué par le ratio, et précisé par le modèle d'évaluation utilisé par une organisation.

Tableau 2.2 Sélection des ratios et des valeurs normalisées

Num [1]	Num [2]	✓ [3]	Nom Ratio [4]	Description [5]	Dimension (E,S,T) [6]	Rmin [7]	Rmax [8]	Valeur projet [9]	Valeur ratio [10 = 9–7 / 8-7]	Processus [11]
M1	M1									
	M2									
	…									
	Mn									
M2	M1									
	M2									
	…									
	mn									
…	…									
Mn	M1									
	M2									
	…									
	Mn									

Une fois ces deux premiers tableaux complétés, il faut passer au troisième (Tableau 1.3). Ce tableau est divisé en trois, en fonction des domaines. Il permet de traiter chacun des domaines individuellement. Tout d'abord, il faut inscrire les ratios de la colonne 4 du tableau 2.2 à la première colonne du tableau 2.3. Puis à la colonne 2, il faut déterminer la participation relative de chaque ratio dans l'évaluation de la performance, celle-ci doivent être représentée par les poids en pourcentage. La somme des colonnes des poids pour chaque domaine ne doit pas dépasser la valeur de QL (Quantity Level). QL est la contribution des poids assignés par rapport aux informations objectives dans le calcul de la qualité du système.

Ensuite, il faut calculer la valeur finale de chaque ratio qui correspond à la multiplication de la colonne 2 par la colonne 3. Une fois ces valeurs déterminées, les sommes de ces dernières pour chaque domaine sont aussi à calculer.

L'étape suivant pour le tableau 2.3 consiste à préciser les seuils d'acceptabilité par l'équipe de mesure et de mettre les résultats dans la colonne 5. Ceci consiste en la détermination des valeurs de référence avec lesquelles les valeurs du projet seront comparées pendant la phase d'évaluation. Enfin, il faut calculer la différence entre les valeurs finales et les valeurs seuils à répartir dans la colonne 6.

Tableau 2.3 Calcul des valeurs selon les domaines

Nom ratio [1]	Poids [2]	Valeur ratio [3]	Valeur finale [4 = 2 * 3]	Seuils d'acceptabilité [5]	Valeurs ˇ [6 = 5 - 4]
économique					
E1					
E2					
E3					
Social					
S1					
S2					
technique					
T1					
T2					
T3					

Le quatrième tableau (Tableau 2.4), permet de regrouper les différentes opinions des personnes interviewées sur les sous-caractéristiques de qualité collectées du système. La méthode d'assignation des points de vues est détaillée dans la section 3.5. Le tableau 2.4 illustre un exemple utilisant deux représentants du domaine économique, deux du domaine social et un représentant du domaine technique.

Pour chaque utilisateur, il faut ajouter une colonne. Dans la colonne qui suit, il faut calculer le nombre de personnes qui ont donné leur avis sur la sous-caractéristique

correspondante. Dans la colonne 9 du tableau 2.4 on inscrit la somme des points de vues, c'est-à-dire la somme des colonnes 3, 4, 5, 6 et 7. L'avant-dernière colonne est consacrée à la moyenne des opinions déterminée en divisant chaque valeur de la colonne 9 par la valeur correspondante de la colonne 8. La dernière colonne du tableau consiste à évaluer le pourcentage d'acceptation des avis pour chaque sous-caractéristique. On obtient cette évaluation par la division des valeurs de la colonne 8 avec le nombre total des personnes interviewées.

Tableau 2.4 Collecte des points de vues pour le calcul de QF

Caractéristique [1]	Sous-caractéristique [2]	E1 [3]	E2 [4]	S1 [5]	S2 [6]	T1 [7]	Nombre personnes [8]	Somme [9=3+4+5+6+7]	Moyenne [10=9/8]	% d'acceptation [11= 8 / personnes interviewées]
C1	Ss 1									
	Ss2									
C2										
…	….									
C6										
	Ss21									

Après avoir terminer de remplir le quatrième tableau, il faut passer au suivant qui est destiné à effectuer le calcul des priorités des caractéristiques. Ce dernier est présenté par deux tableaux : 2.5.a et 2.5.b comme le montre la figure 2.2. Dans ce cinquième tableau, il faut indiquer dans les colonnes 2, 4 et 6 du tableau les priorités des caractéristiques selon les domaines. Ces priorités sont déduites des colonnes des points de vues du quatrième tableau. Les étapes de calcul pour trouver les priorités sont décrites dans le paragraphe 3.5.1.2 du chapitre 3. Par la suite, une pondération (w) est assignée pour chaque priorité. La détermination de cette dernière est définie dans le chapitre 3 au paragraphe 3.5.1.3. Les pondérations calculées sont inscrites dans le tableau 2.5.b de la figure 2.2 et dans les colonnes 3, 5, 7 du tableau 2.5.a selon la priorité correspondante. Pour la colonne 8 du tableau

2.5.a, on calcule la somme des pondérations pour chaque caractéristique et dans les différents domaines; il s'agit d'additionner les valeurs des colonnes 3, 5 et 7. À la dernière colonne du tableau 2.5.a, il faut calculer la moyenne des pondérations pour chaque caractéristique.

Tableau 2.5.a

Caractéristique [1]	p(E) [2]	W [3]	p(S) [4]	W [5]	p(T) [6]	W [7]	Somme [8= 3+5+7]	Moyenne [9= 8/ nombre de domaines]
C1								
C2								
C3								
C4								
C5								
C6								

Tableau 2.5.b

Rang	Poids
p1	
p2	
p3	
p4	
p5	
p6	

Figure 2.2 Cinquième tableau : Détermination des pondérations des priorités

Enfin, le dernier tableau (Tableau 2.6) permet de calculer le facteur de qualité. À la première colonne il faut placer les sous-caractéristiques et à la deuxième, quatrième et sixième il faut cocher cochons les sous-caractéristiques correspondant à chaque domaine qui ont été retenues par les utilisateurs. À la troisième, cinquième et septième colonnes il faut copier respectivement les colonnes 2, 4 et 6 du tableau 2.5.a. Les informations de la colonne 8 sont les mêmes que celles de l'avant dernière colonne du tableau 2.5.a. Il faut calculer calculons dans la colonne 9 la somme des SCV (colonne 8) pour chaque caractéristique. La colonne 10 est

identique à la dernière colonne du tableau 2.5.a. Et enfin, nous terminons cette bataille avec les tableaux avec la colonne 11, dans laquelle il faut multiplier les valeurs de la colonne 9 avec celles de la colonne 10. À la ligne TCV il faut calculer la somme des valeurs de la colonne 9.

À partir de la valeur de TCV et de MQL (MQL = 100 - QL), il faut calculer la valeur du facteur de qualité QF.

$$QF = (TCV – MQL) * 0.01$$

Tableau 2.6 Calcul de QF et de TCV

Sous-caractéristique [1]	E [2]	P(E) [3]	S [4]	P(S) [5]	T [6]	P(T) [7]	SCV [8]	SSV [9]	PW [10]	CV [11= 9*10]
Ss1										
Ss2										
Ss21										
									TCV	
									QF	

Nous remarquons bien la complexité qui réside dans l'utilisation des tableaux. Une seule erreur commise lors de la manipulation d'un tableau, que ce soit une confusion entre les lignes et les colonnes ou une faute de copiage des nombres et notre valeur finale sera faussée. Que dire aussi d'une faute de calcul manuel

2.2.2. Traitement mathématique

En plus des opérations simples d'addition, de multiplication et de normalisation que nous avons présentées pour les tableaux dans le paragraphe précédent, d'autres formes de calculs beaucoup plus complexes sont aussi utilisées.

À partir de la valeur du facteur de qualité, le calcul de la valeur de la performance est entamé. C'est à cette étape que le calcul commence à se compliquer, car des coordonnées, des hauteurs, surfaces et volumes dans un tétraèdre sont à calculer.

Comme il faut considérer trois domaines, un graphique en trois dimensions s'impose et, mieux encore, un tétraèdre dont le sommet représente la performance maximale que peut avoir un logiciel. Les trois dimensions (E,S,T) dans l'espace correspondent à la base du tétraèdre dont on suppose que le vecteur ES est situé sur l'axe des x. Le sommet de la pyramide représente le plus haut niveau de la performance. La figure 2.3 ci-dessous illustre cette représentation pyramidale.

Pour faciliter les calculs, le tétraèdre choisi pour le modèle QEST est un tétraèdre régulier c'est-à-dire que la longueur des côtés est égale à 1. Ainsi, les valeurs des trois dimensions (e,s,t) doivent être représentées par des valeurs normalisées entre 0 et 1 (1 étant la valeur maximale pour chaque dimension).

La valeur de chaque dimension d'un projet spécifique est donnée par la somme des poids de la liste des n, valeurs normalisées des mesures représentatives pour la dimension concernée. En d'autres termes, il s'agit de calculer la somme pour chaque domaine des valeurs de la colonne 4 du tableau 3. Chaque valeur est placée sur l'arête correspondante, pour obtenir l'hyperplan (Q_e, Q_s, Q_t) comme le montre la figure 3. L'hyperplan (Q_e, Q_s, Q_t) représente une mesure de performance tridimensionnelle.

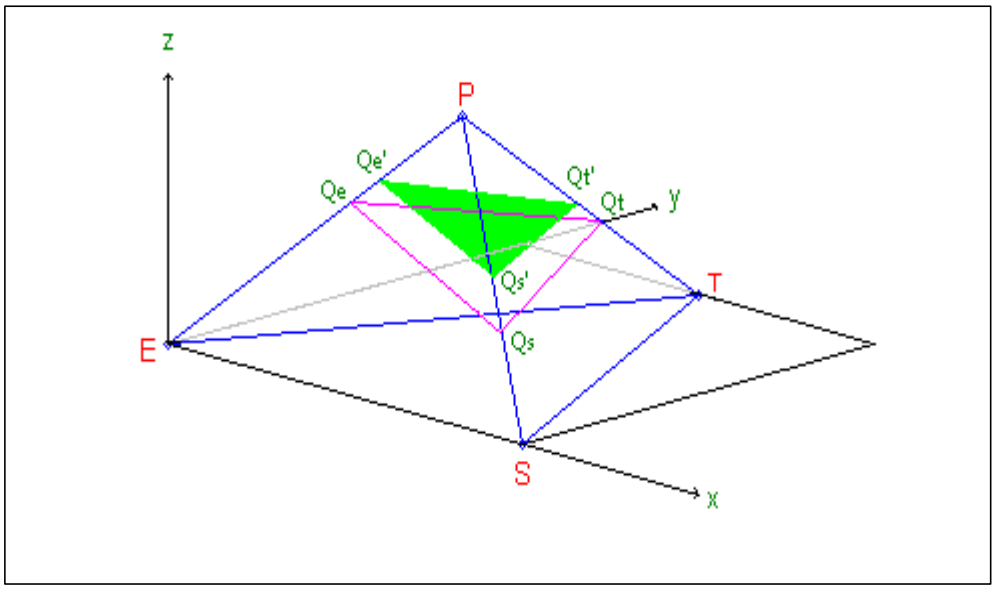

Figure 2.3 Représentation pyramidale

La distance entre E (S et T) et Q_e (Q_s et Q_t) est la mesure de la dimension économique e (sociale s et technique t).

Comme la mesure de qualité tridimensionnelle (QF) est un facteur supplémentaire affectant la représentation de la mesure de la productivité, il est possible d'établir le plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) qui représente la somme des mesures de productivité et de qualité. Ce plan montre si la valeur de la qualité du logiciel est supérieure ou inférieure à la qualité cible prédéterminée pour chaque dimension. Ainsi, la performance est représentée par le volume de la section du tétraèdre au dessous de l'hyperplan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) divisé par le volume de tout le tétraèdre.

p = Volume (E, S, T, $Q_{e'}$, $Q_{s'}$, $Q_{t'}$) / Volume total = 1 – Volume($Q_{e'}$, $Q_{s'}$, $Q_{t'}$, P) / Volume total

e' = e + QF où e est la valeur de productivité économique;
s' = s + QF où s est la valeur de productivité sociale;
t' = t + QF où t est la valeur de productivité technique.

Pour calculer l'équation de l'hyperplan ($Q_{e'}$,$Q_{s'}$,$Q_{t'}$), on a besoin des coordonnées des points E, S, T, P et le point H qui est le centre de la base du tétraèdre (E, S, T).

Comme les valeurs sont normalisées :

E = (0, 0, 0) ; S = (1, 0, 0) ; T = (1/2, $\sqrt{3}$ /2, 0) ; P= (1/2, 1/(2$\sqrt{3}$), $\sqrt{6}$ /3) ;
H = (1/2, 1/(2$\sqrt{3}$), $\sqrt{6}$ /3) ;

Ensuite, il faut calculer les coordonnées des points Q_e, Q_s, Q_t, $Q_{e'}$, $Q_{s'}$, $Q_{t'}$:

Q_e = E + e EP = (1/2 e, 1/(2$\sqrt{3}$) e, $\sqrt{6}$ /3 e)
Q_s = S + s SP = (1 -1/2 s, 1/(2$\sqrt{3}$) s, $\sqrt{6}$ /3 s)
Q_t = T + t TP = (1/2 , $\sqrt{3}$ /2 – t/$\sqrt{3}$, $\sqrt{6}$ /3 t)
$Q_{e'}$ = E +e' EP = (1/2 e', 1/(2$\sqrt{3}$) e', $\sqrt{6}$ /3 e')
$Q_{s'}$ = S + s' SP = (1 -1/2 s', 1/(2$\sqrt{3}$) s', $\sqrt{6}$ /3 s')
$Q_{t'}$ = T + t' TP = (1/2 ,$\sqrt{3}$ /2 – t'/$\sqrt{3}$, $\sqrt{6}$ /3 t')

Selon la règle de Sarrus, l'équation du plan est donnée sous la forme :

$$aX + bY + cZ + d = 0.$$

Les coefficients du plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) sont les suivants :

a = (s't' – s'-e't' + e') / $\sqrt{2}$
b = (s' – 2e's' + e' – 2t' + s't' +e't') / $\sqrt{6}$
c = (3 – 2e' – 2s' – 2t' +e's' + e't'+s't') / 2$\sqrt{3}$
d = e's' + e't' – e' –e's't') / $\sqrt{2}$

Pour calculer p, il faut tout d'abord calculer le volume total et le volume de la section ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$, P) :

Volume total = $\sqrt{2}$ /12
Volume ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$, P) = (A – h) / 3 , où A est l'aire de l'hyperplan($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) et h la hauteur de la pyramide ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$, P).

$h = | aX + bY + cZ + d| / \sqrt{a^2 + b^2 + c^2}$

$A = \sqrt{sp\,(sp - k)\,(sp - l)\,(sp - m)}$, où sp est le semi périmètre du triangle ($Q_{e'}$,$Q_{s'}$,$Q_{t'}$) et k, l, m sont les distances des trois côtés d'un même triangle.

$k = |Q_{e'}Q_{s'}| = \sqrt{1 + e'^2 + s'^2 - e' - s' - e's'}$

$l = |Q_{e'}Q_{t'}| = \sqrt{1 + e'^2 + t'^2 - e' - t' - e't'}$

$m = |Q_{s'}Q_{s'}| = \sqrt{1 + s'^2 + t'^2 - s' - t' - s't'}$

Cette méthode de calcul de performance présente quelques exceptions. Il y a sept cas dans lesquels la valeur de la performance p ne peut être déterminée.

Si e' = 1 ou t' = 1 ou s' = 1, dans notre représentation graphique on obtient comme résultat de performance une portion d'une des surfaces du tétraèdre. Seule l'aire du plan peut être calculée.

Dans le cas où e' = s' = 1 ou e' = t' = 1 ou s' = t' = 1, le résultat graphique est un segment sur l'un des côtés du tétraèdre et on ne peut calculer que la distance.

La dernière exception donne la performance maximum. On l'obtient quand e' = s' = t' = 1, car elle est représentée par le sommet du tétraèdre.

D'autres mesures de performance peuvent être calculées, ces mesures sont basées sur les mesures de distance ou d'aire plutôt que sur le volume. Mais la performance calculée à l'aide du volume reste la plus significative, car le volume donne une idée plus générale que les autres mesures [Buglione et al., 1999].

2.2. PROBLÈMATIQUE : FAIBLESSES

La difficulté du modèle QEST proposé par Buglione et al. réside non seulement dans la quantité mais aussi dans la compréhension des étapes du modèle, la gestion des différents tableaux et les calculs compliqués.

Dans les publications concernent QEST [Buglione et al., 1999], la méthodologie suivie pour expliquer le modèle n'est pas simple. Le lecteur s'y retrouve avec grand difficulté :

- les divers mots techniques employés qui se ressemblent mais qui ne signifient pas la même chose, par exemple SSV, SCV et TCV;
- les différents formulaires et tableaux utilisés auxquels il faut à chaque fois se référer pour se rappeler de quoi il s'agit;
- les six tableaux à remplir qui peuvent avoir plus de dix lignes et dix colonnes chacun, et dont pour avoir les informations à inscrire dans chaque colonne, il faut revoir certaines autres colonnes d'autres tableaux pour y faire un certain nombre de calculs;
- le traitement mathématique à faire qui regroupe de simples équations arithmétiques et d'autres beaucoup plus compliqués telles que des calculs vectoriels et de volume dans un tétraèdre.

De plus, pour bien comprendre et appliquer le modèle, un ensemble de pré-requis est exigé. Il faudrait que le lecteur ait des idées trés précises sur les caractéristiques de qualité et le marquage et certaines connaissances dans le domaine mathématique et géométrique. Ces derniers ne sont pas assez expliqués dans les publications concernant QEST. Des recherches doivent être faites pour éclaircir certains concepts et permettre l'application du modèle.

Les étapes du modèle QEST telles que définies par Buglione et al. ne sont pas simples non plus. Comme nous avons vu dans la section précédente, les tableaux ne sont pas faciles à manipuler. En résumer, tout d'abord, il faut commencer par choisir les différents ratios en utilisant GQM(R) (voir chapitre 3) et collecter les différents points de vues des participants, selon les domaines économique, social et technique, grâce à des formulaires prêts à être remplis. Par la suite, les six tableaux doivent être remplis comme indiqué dans la section 2. Chaque tableau nécessite de multiples entrées et des calculs intermédiaires à faire pour obtenir les résultats nécessaires. À la fin, il faut calculer des coordonnées, équations des plans, surfaces et volumes dans le tétraèdre pour arriver à la valeur numérique de la performance du système à mesurer.

Un autre inconvénient de cette méthode est la perte considérable de temps qu'un utilisateur aura à consacrer lors de chaque utilisation du modèle, car les étapes de ce dernier sont faites manuellement. Des erreurs peuvent s'introduire facilement dans les tableaux comme dans les calculs. Nous ne pouvons pas détecter les erreurs dans cette méthode à moins de refaire plusieurs fois les étapes depuis le début. Il faut donc être très vigilant en manipulant ce modèle QEST.

Tous ces inconvénients laisse le modèle à désirer et décourage le lecteur à continuer sa lecture et à essayer d'utiliser QEST. Jusqu'à présent, seuls les auteurs de ce modèle comptent parmi les utilisateurs c'est-à-dire environ cinq personnes dans le monde qui sachent manipuler QEST.

2.3. OBJECTIFS

Vu les désavantages du modèle QEST décrits dans la section 3, nous proposons dans ce travail de concevoir un outil logiciel sur web pour faciliter l'utilisation du modèle. Pour ce faire, nous automatisons entre autre les fonctions manuelles. Ainsi, nous évitons, aux utilisateurs, la manipulation des tableaux et les traitements des

données, en cachant la complexité du modèle et en concevant une interface conviviale pour les utilisateurs.

Notre but est de réduire le nombre de tableaux à utiliser à seulement deux qui demandent peu d'effort pour les compléter, et aucun calcul à faire par la suite pour obtenir le résultat. En résumer, pour utiliser ce nouveau modèle QEST, l'utilisateur n'aura qu'à collecter les données concernant le projet c'est-à-dire les mesures et leurs valeurs en utilisant la méthode GQM(R) et les points de vues des participants sur les caractéristiques de qualité. Ces deux étapes ont été décrites en détail dans la section 2 et la section 3 du troisième chapitre. Ainsi, l'utilisateur n'aura plus besoin de connaissances mathématiques au préalable et n'aura pas, à chaque fois, à préciser les entrées pour chaque tableau. La figure 2.4 résume les entrées et sorties de notre nouveau prototype.

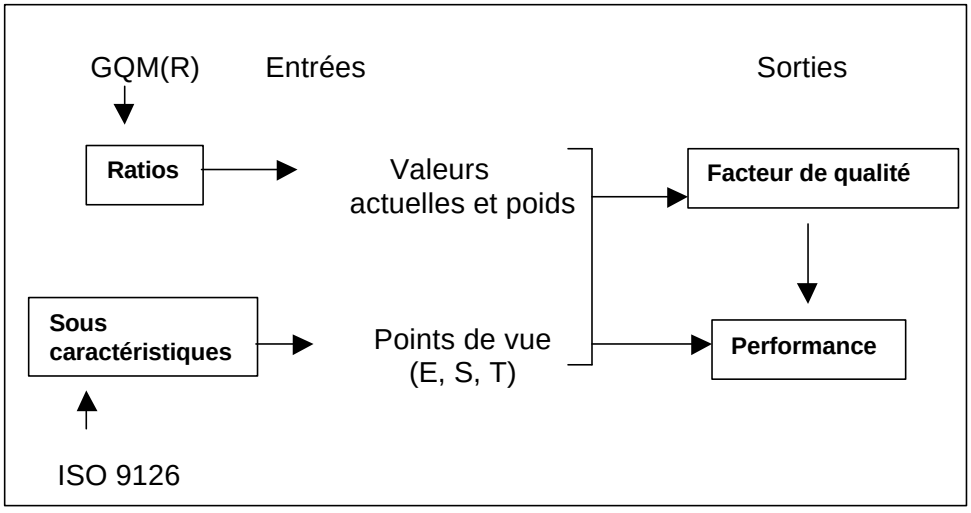

Figure 2.4 Entrées et sorties du nouveau QEST

De plus, nous voulons que ce modèle soit accessible à tout le monde. Nous avons donc cherché à simplifier les termes techniques utilisés dans la première version de

QEST et à présenter ce dernier dans un environnement auquel nous pouvons avoir accès facilement.

2.4. CONCLUSION

La modèle QEST, tel que présenté par Buglione et al., n'est pas facile à comprendre et à manipuler. Il présente plusieurs inconvénients de compréhension et d'utilisation. La nouvelle version de QEST que nous présentons, permet de faire gagner du temps aux consommateurs et leur fournit un outil pour déterminer la performance d'un système avec un résultat sans erreur dès la première application.

En conclusion, ce travail permet de cacher les parties qui exigent des efforts d'attention et de calcul considérables de la part de l'utilisateur afin de mieux comprendre la manipulation de QEST. Tout ce qui concerne l'utilisation des tableaux et les calculs intermédiaires pour avoir le résultat final, ainsi que les traitements mathématiques compliqués : calculs d'équation de plan et de volume dans un tétraèdre ne font plus partie des étapes de notre nouveau prototype QEST. De cette manière, nous pouvons espérer augmenter considérablement le potentiel d'utilisateurs pour une utilisation courante dans les groupes de qualité.

CHAPITRE III

LES CONCEPTS UTILISÉS

Depuis plusieurs années, les ingénieurs utilisent des modèles pour déterminer la qualité et la performance des logiciels et systèmes. La plus grande majorité des entreprises suivent des méthodes standards qui les guident à organiser le mesurage et analyser les résultats pour évaluer leurs produits. Mais les méthodes sont tellement variées, que le fait d'en choisir une pose un problème.

Pour l'évaluation d'un projet, nous avons besoin d'identifier les personnes impliquées dans ce processus, puis d'identifier les différentes mesures que nous avons à collecter et la démarche à suivre dans le calcul pour obtenir le résultat de notre évaluation. Afin de rendre le modèle QEST plus performant et applicable à tous les projets, il est important d'utiliser des outils standards telle que les normes ISO appropriées.

3.1. IDENTIFICATION DES MESURES

Pour mesurer la performance d'un produit, dans les organisations modernes, les groupes d'assurance de qualité appliquent une méthode qui associe des quantités mesurables internes avec des caractéristiques externes de la qualité. Plusieurs exemples de ces mesures et leurs interprétations se trouvent dans la littérature des mesures de logiciels. Par exemple : les points de fonction sont utilisés pour estimer le coût d'un produit et la complexité cyclomatique est utilisée dans le but d'estimer la maintenabilité et la complexité d'un système, etc.

Pour notre projet, nous utilisons l'approche GQM(R) (Goals, Questions, Measures and Ratios). Cette extension nous est très utile pour préciser notre mesurage. GQM(R) est utilisé pour définir et interpréter les mesures des systèmes. Ces mesures doivent être basées sur des buts et des modèles [Basili, 1993]. Il existe un grand nombre de buts pour les logiciels et systèmes et il y a plusieurs méthodes pour définir ces buts mesurables. Les approches les plus connues sont : Quality Function Deployment Approach (QFD), Goal/Question/Metric Paradigm (GQM) et Software Quality Metrics Approach (SQM) [Basili, 1993].

Comparativement aux approches QFD et SQM qui ne tiennent compte que des points de vues des utilisateurs et clients, le paradigme GQM(R) donne les moyens nécessaires pour cerner les mesures selon les opinions des programmeurs, gestionnaires et consommateurs.

3.1.1. Paradigme GQM

Un bon processus de mesurage nous aide à réussir notre évaluation de performance. Il nous fournit des informations sur lesquelles nous pouvons agir, ce qui signifie que le mesurage doit nous procurer des informations pertinentes pour atteindre nos objectifs.

Le paradigme GQM est un mécanisme pour définir et évaluer, en utilisant les mesures, un ensemble de buts opérationnels. Il représente une approche systématique pour mettre en évidence et intégrer les buts avec les modèles du système et les perspectives de qualité, basés sur les besoins spécifiques du produit et de l'organisation [Basili, 1993].

Dans ce qui suit le mot métrique selon Basili est équivalent au mot mesure.

Les buts sont définis par un ensemble de questions quantifiables qui sont utilisées par la suite pour extraire les informations appropriées des modèles. Les questions et les modèles, à leur tour, définissent un ensemble de métriques et de données spécifiques. Comme le montre le graphe de la figure 3.5, pour passer des buts aux métriques, nous avons à déterminer les questions et à y répondre. Dans cette figure nous avons n buts, chacun d'entre eux génère un ensemble de questions quantifiables qui aident à définir et à quantifier un but précis. De chaque question émerge un ensemble de métriques. Les mêmes questions peuvent être utilisées pour définir plusieurs buts et les métriques pour répondre à plus d'une question.

Malgré qu'il peut y avoir plusieurs buts et même plusieurs questions, le nombre de métriques n'est pas proportionnel à ces derniers. Ainsi un ensemble de métriques peut être collecté pour caractériser un produit qui nous permettra de répondre à plusieurs questions qui sont générées par différents buts [Basili, 1993].

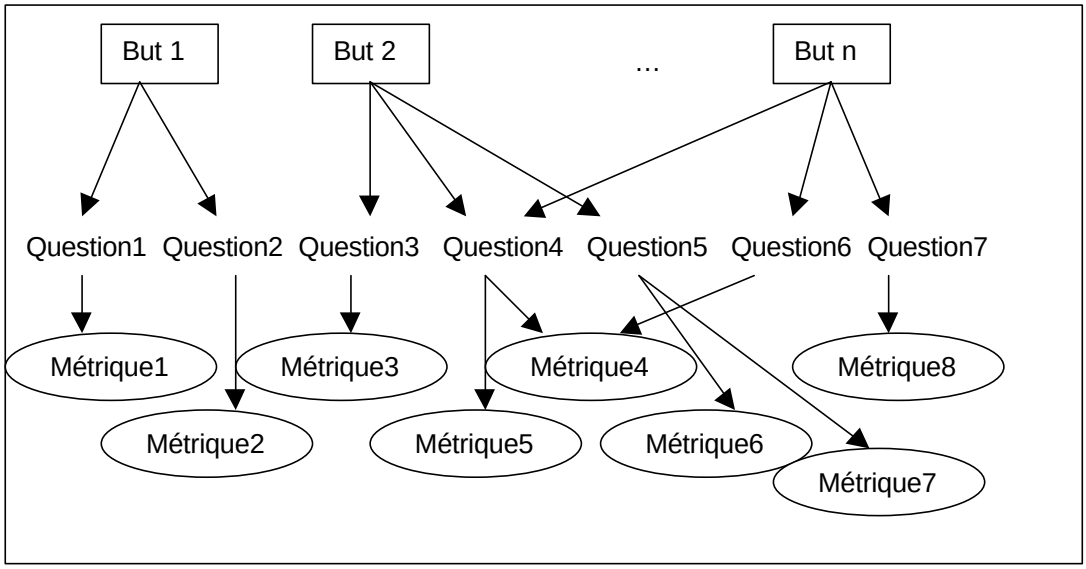

Figure 3.5 Graphe du paradigme GQM

L’application de GQM nous permet, en premier lieu, de développer un ensemble de buts pour le système concernant la qualité et la productivité, par exemple la satisfaction du client. En deuxième lieu, GQM nous aide à améliorer la qualité et à générer des questions qui définissent et détaillent ces buts d’une manière quantifiable. Enfin, cette approche nous offre la possibilité de spécifier les mesures nécessaires à collecter pour répondre aux questions tout en respectant nos buts.

Le processus de définir les buts et de les modeler en questions quantifiables est complexe et requiert de l’expérience. Basili propose de suivre un certain modèle pour déduire ces informations. Ce modèle se compose de trois paramètres : objectif, perspective et environnement.

L’objectif permet de définir le ou les sujets de l’étude (processus, mesure des produits…) pour chaque but, déterminer ce que nous allons faire et pourquoi (caractérisation, évaluation, prédiction, motivation, amélioration…). Il nous aide à analyser l’objet et son objectif. Il faut à tout prix diviser et éviter les objectifs compliqués et les remplacer par plusieurs buts plus simples. La perspective définit selon quel point de vue (utilisateur, gestionnaire, développeur…) en respectant quel modèle (coût, exactitude, changements…) allons nous atteindre notre évaluation. Il est recommandé d’utiliser plus d’un modèle et plus d’un point de vue. Et le paramètre environnement permet de fixer le contexte à suivre dans notre évaluation en définissant tous les aspects du projet (facteurs problèmes, facteurs personnes, facteurs ressources, facteurs processus, méthodes, outils, contraintes…).

Le choix des mesures est déterminé par les questions quantifiables qui sont basées selon les modèles utilisés. Ces premières peuvent être objectives, c’est-à-dire une mesure absolue, comme par exemple le nombre de lignes de code et le temps pour le développement, ou subjectives, c’est-à-dire une mesure non exacte tels que le degré d’utilisation d’une méthode et l’expérience des programmeurs. Indépendamment du type de mesures, nous choisissons les méthodes appropriées pour collecter et valider les données.

En conclusion, le paradigme GQM est un mécanisme pour définir et interpréter des buts opérationnels et mesurables d'un système. Il fournit une approche systématique pour déterminer les buts d'un projet qui, à leur tour, sont raffinés en questions auxquelles on peut répondre, d'une manière quantifiable en utilisant des métriques.

3.1.2. Paradigme GQM(R)

De nos jours, GQM est utilisé par de nombreuses organisations, comme la NASA, Capability Maturity Model et Siemens [Lamprecht, Weber]. Le choix de ce paradigme dans le modèle QEST est dù au fait qu'il est le plus utilisé et représente le meilleur moyen pour fixer les métriques d'un système selon les points de vues économiques, sociaux et techniques. Par contre, le fait d'avoir un ensemble de mesures simples, c'est-à-dire à une variable à la fois, ne rend pas possible la connaissance des différentes relations qui existent entre ces mesures [Buglione et al., 1999]. Pour cette raison, une nouvelle étape a était ajoutée au paradigme GQM qui permet d'établir un lien entre les mesures des trois domaines. Cette étape consiste à concevoir des ratios à partir des mesures car il est plus significatif d'analyser des ratios que des mesures simples, d'où le nom de Goal Question Measure and Ratio (GQM(R)). La cadence de livraison de projet (Project Delivery Rate = Points de fonction / effort du travail) peut être un exemple de ratio pour le domaine économique, le ratio de stabilité (Stability Ratio = nombre de changements/ Points de fonction) peut être un ratio pour le domaine social et le ratio d'erreurs (Defect Ratio = No. d'erreurs / points de fonction) peut être un ratio pour le domaine technique.

La figure 3.6 représente l'arbre du paradigme GQM(R).

Après la détermination des buts et des questions pour chacun des domaines économique, social et technique, et la spécification des mesures et ratios, il faut

installer une liste pour ces deux dernières (mesures et ratios) avec leurs unités de mesures. Comme par exemple, la mesure Effort de travail est calculée en heures.

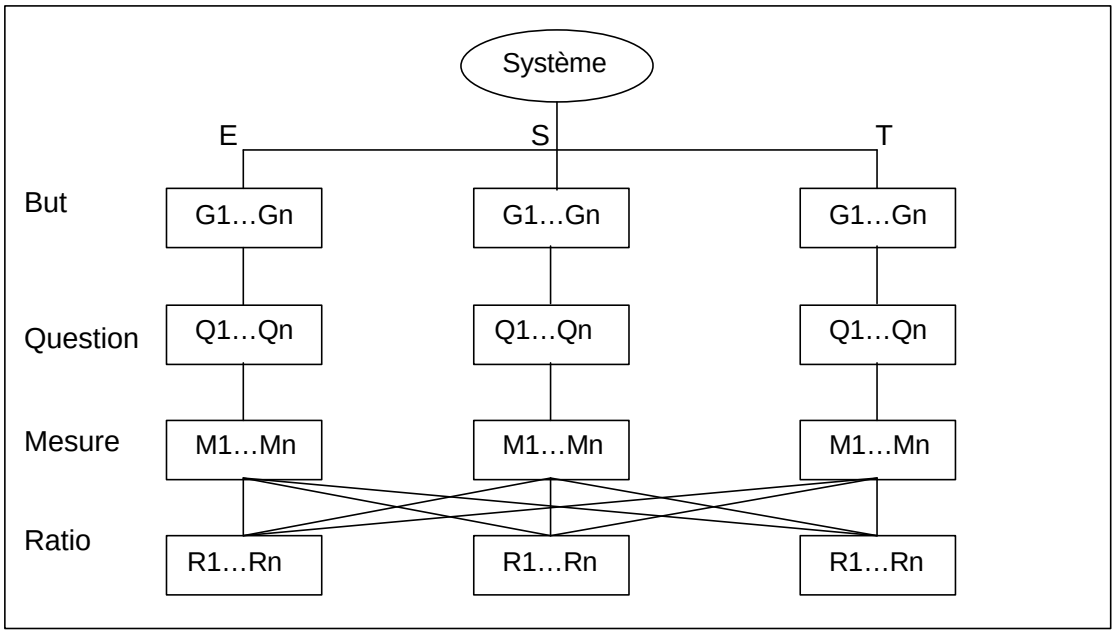

Figure 3.6 Représentation du paradigme GQM(R)

3.1.3. Exemple d'un modèle GQM(R)

La figure 3.7 montre un exemple sur l'utilisation du paradigme GQM(R) pour trouver les ratios nécessaires à l'évaluation de la qualité de productivité d'un logiciel. Selon les mesures obtenues, nous pouvons par la suite déterminer ce qu'il faut rectifier pour avoir une meilleure productivité. D'abord, nous commençons par fixer notre but qui est l'amélioration de la productivité du logiciel. Ensuite, nous cernons les questions pour nous aider à mieux comprendre notre but. Pour cet exemple, la productivité se résume par l'influence de la taille et de l'équipe et par l'état actuel de cette dernière. Après, nous dégageons les mesures qui nous permettent de répondre à ces questions. L'exemple nous en définit cinq : le nombre de lignes du code, les points de fonction, l'effort du programmeur, l'expérience du programmeur

et le nombre de personne dans l'équipe. Finalement, à partir de ces mesures, retrouver les ratios à utiliser devient beaucoup plus facile. Selon l'exemple, il y a trois ratios : la productivité du programmeur, la cadence de livraison et la compétence de l'équipe.

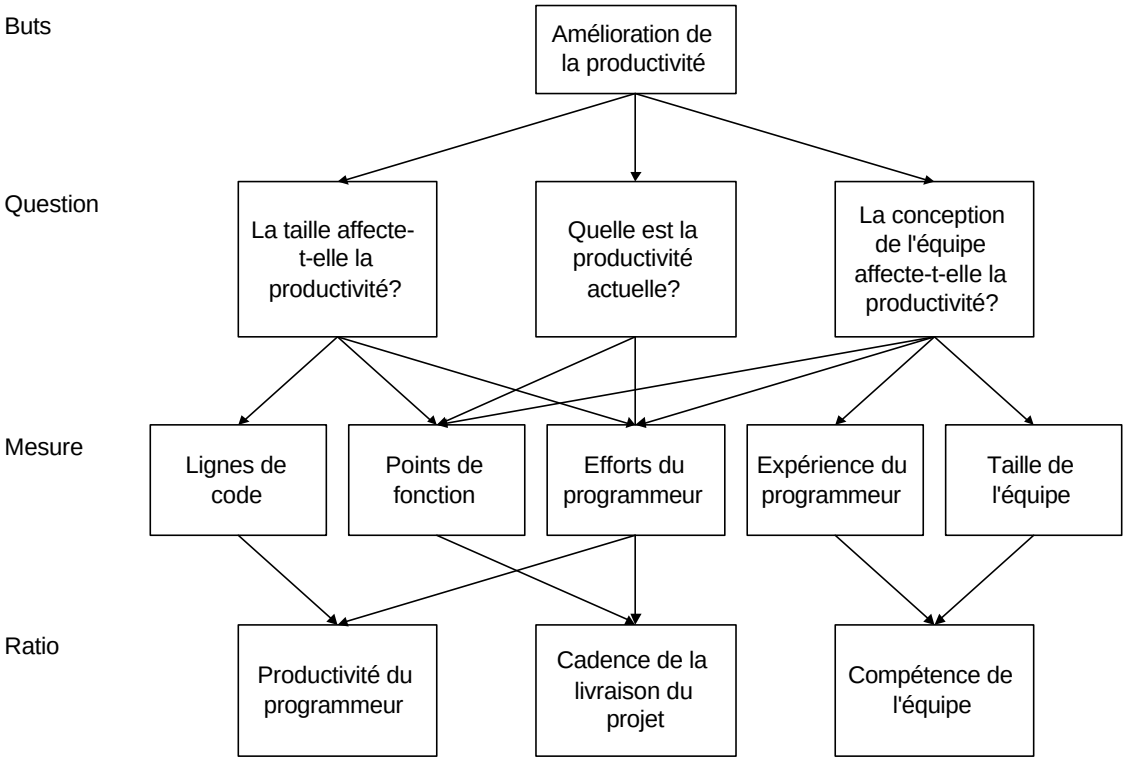

Figure 3.7 Exemple d'un modèle utilisant GQM(R)

3.2. PERCEPTION DE LA PERFORMANCE

Dans la plupart des modèles de mesure de performance, les domaines technique et économique sont les seuls à être considérés. On ne tient pas compte de la perception des clients et des utilisateurs vis-à-vis du système à mesurer. Négliger le fait que les utilisateurs à qui est destiné le produit, ne sont pas impliqués pour donner leurs points de vues sur ce dernier, peut entraîner une fausse évaluation de la performance.

Comme les avis diffèrent dans une même entreprise et dans une même équipe, que dire lorsque ceux-ci ne sont pas d'une même spécialité. Il est donc préférable d'intégrer dans notre modèle de mesure de qualité, des points de vues multidimensionnels. La qualité est ainsi évaluée en tenant compte concurremment des trois opinions économique, sociale et technique. L'équipe d'implantation du programme de mesure doit être représentée par les groupes suivants: ingénieurs programmeurs, clients ou utilisateurs et gestionnaires. Si un des groupes est absent pour l'évaluation de la performance d'un système, nous courons un risque de fausser nos résultats et de donner au produit une qualité qu'il ne mérite pas.

3.2.1. Perception des ingénieurs programmeurs

L'ingénieur programmeur est responsable de la conception du système pour satisfaire les spécifications fonctionnelles. Il s'intéresse au succès technique du produit et donc il considère surtout la qualité technique. Voici un exemple de questions que se posent les programmeurs:

- Combien d'erreurs se trouvent dans la totalité du système?
- Quel est l'effort nécessaire pour tester le système?
- Combien de temps faut-il pour résoudre les erreurs?

3.2.2. Perception des clients et consommateurs

Les clients et les consommateurs sont les acheteurs du système. Ce sont eux qui se préoccupent de satisfaire des contraintes budgétaires, de l'exactitude technique du produit et de la livraison à temps. Ils sont les utilisateurs finaux qui s'intéressent à la démarche à suivre pour s'assurer que le produit répond à leurs besoins.

Le client joue un rôle important dans la qualité d'un système. Les standards internationaux tels que ISO 9000 et IEEE mettent l'accent sur la perception du client sur la qualité et s'attendent à ce que la satisfaction de client soit fortement liée à

toutes les fonctions du système. Le fait de rencontrer la perception de la qualité des utilisateurs finaux du système représente un facteur majeur de succès. Malgré les informations qui dérivent d'un mesurage interne, les clients sont les juges finaux de la qualité d'un produit [Xenos, 1997]. Il est donc indispensable d'avoir leurs opinions.

Voici un exemple de questions que les utilisateurs se posent pour évaluer un système :

- Est-ce que les fonctions dont il a besoin existent dans le système?
- Est-ce que le système est facile à utiliser?
- Comment est la fiabilité du système?
- Quelle est l'efficacité du système?
- Quelle est la difficulté de transférer le système d'un environnement à un autre?

L'utilisateur évalue le système sans connaître ses aspects internes ou comment il a été développé [ISO 9126, 1991]. Les points de vues des utilisateurs peuvent être considérés comme des mesures subjectives qui varient selon le jugement des clients.

3.2.3. Perception des gestionnaires

Le groupe de gestionnaires se compose des gestionnaires du projet et du programme, leurs points de vues concernent le succès technique d'un produit pour satisfaire les besoins du consommateur en respectant les contraintes budgétaires et commerciales. Le gestionnaire s'intéresse plus à la qualité totale du système plutôt qu'à une caractéristique bien précise vu qu'il essaie d'optimiser la qualité en tenant compte des contraintes (compensations - tradeoff) de limites de coût, des ressources humaines et du temps. Le gestionnaire se pose souvent des questions telles que :

- Est-ce que le budget est dépassé?
- Est-ce que le temps de livraison est respecté?

3.3. CHOIX DES CARACTÉRISTIQUES DE QUALITÉ

Les caractéristiques de qualité peuvent différer d'une compagnie à une autre. IEEE, PSM (Appendice A), ISO et d'autres standards proposent divers ensembles de caractéristiques. Bien que jusqu'à présent aucun ensemble de caractéristiques de qualité n'a pu être fixé et considéré comme un standard universel, les caractéristiques de qualité les plus reconnues et utilisées à travers le monde sont celles de la norme ISO-9126. L'approche standard du modèle de qualité utilisé par ISO-9126, réalisée en 1991, représente la qualité comme un ensemble de caractéristiques. Ce standard international en identifie six pour l'évaluation de logiciels. Les caractéristiques reliées au comportement du système dans son environnement sont appelées les caractéristiques de qualité interne : elles incluent, par exemple, la facilité d'utilisation (usability) et la fiabilité (reliability). Les caractéristiques concernant le développement du système sont appelées les caractéristiques de qualité externe : elles incluent, par exemple, la complexité structurale, la taille, le taux d'erreurs et les tests.

Les caractéristiques seules ne sont pas suffisantes pour décrire exactement la qualité, des critères devraient être utilisés. Ainsi, chacune des six caractéristiques est décomposée en un certain nombre de sous-caractéristiques : il y en a vingt et une en tout. Le tableau 3.7 présente les différentes caractéristiques et sous-caractéristiques définies par ISO 9126.

Les caractéristiques et les sous-caractéristiques de qualité n'ont pas toutes la même évaluation pour chaque domaine. Il est nécessaire de déterminer les différents points de vues pour chaque utilisateur et domaine.

Tableau 3.7 Les caractéristiques et sous-caractéristiques de ISO 9126

CARACTÉRISTIQUES	SOUS-CARACTÉRISTIQUES
Fonctionnalité La capacité d'un logiciel de fournir les fonctions qui répondent aux besoins indiqués quand le logiciel est utilisé dans des conditions bien spécifiques.	**Aptitude** Concerne la présence et la convenance d'un ensemble de fonctions pour des tâches précises.
	Exactitude Concerne l'exactitude ou la convenance des résultats ou d'effets.
	Interopérabilité La capacité du système d'agir avec d'autres systèmes.
	Conformité réglementaire Le respect du système aux normes ou aux conventions ou aux réglementations de droit et d'autres prescriptions.
	Sécurité La capacité de prévenir les accès non autorisés aux programmes et données.
Fiabilité La capacité d'un système de maintenir son niveau de performance quand il est utilisé dans des condition spécifiques.	**Maturité** La convenance de la fréquence d'échec.
	Tolérance aux fautes La capacité de maintenir un niveau de performance dans le cas d'erreurs.
	Possibilité de récupération La capacité de rétablir son niveau de performance et récupérer directement, en un temps et effort déterminés, les données affectées en cas d'échec.
Facilité d'utilisation La capacité d'un système à être compris, appris, utilisé et être aimé par l'utilisateur, quand il est appliqué dans des conditions spécifiques.	**Facilité de compréhension** Les efforts que fournissent les utilisateurs pour connaître le concept du système et ses possibilités.
	Facilité d'apprentissage Les efforts que fournissent les utilisateurs pour apprendre l'utilisation du système (opérations de contrôle, entrées, sorties).
	Facilité d'exploitation Les efforts que fournissent les utilisateurs pour les opérations et les opérations de contrôle.
Rendement La capacité d'un système de fournir la performance requise, relative aux ressources utilisées, dans des conditions indiquées.	**Comportement vis-à-vis du temps** La convenance du temps de réponse et de traitement.
	Comportement vis-à-vis des ressources La convenance de la quantité de ressources utilisées et la durée d'une telle utilisation.
Maintenabilité La capacité d'un système à être modifié. La modification peut inclure les corrections, l'amélioration ou l'adaptation du système aux changements de l'environnement.	**Facilité d'analyse** Les efforts fournis pour diagnostiquer les déficiences ou les causes d'une panne ou pour identifier des parties à modifier.
	Facilité de modification Les efforts fournis pour modifier ou éliminer les erreurs ou pour des changements environnementaux.
	Stabilité Les risques d'effets inattendus dus aux modifications.
	Facilité de test Les efforts fournis pour valider le système modifié.

Portabilité La capacité du système à être transféré d'un environnement à un autre.	**Adaptabilité** L'opportunité de son adaptation à différents environnements sans utiliser d'autres moyens que ceux fournis pour le système considéré.
	Facilité à l'installation Les efforts fournis pour l'installation du système.
	Conformité La convenance que le système a pour adhérer aux standards ou conventions reliés à la portabilité.
	Interchangeabilité L'occasion et l'effort de son utilisation à la place d'un autre système.

3.4. MARQUAGE DES ÉLÉMENTS DE QUALITÉ

La quantification des mesures, à elle seule, ne peut pas nous montrer le niveau de satisfaction des utilisateurs dans les différents domaines. Il est donc impératif de trouver une méthode qui permet de remédier à ce problème.

Selon les besoins spécifiques de chaque utilisateur, la priorité et le niveau de détails des différents aspects de qualité varient. Les opinions permettent de décider quelles sont les éléments de qualité qui ont été appliqués dans le système et de déterminer si la qualité exigée a été réalisée selon la perception des différents utilisateurs de chaque domaine. Les entreprises ont besoin d'une méthode pour leur permettre de mesurer la perception de tous ceux qui sont impliqués dans l'évaluation du système vis-à-vis de la qualité.

Chaque personne, de chaque groupe cités auparavant, qui va participer au mesurage de la qualité et performance d'un système, doit assigner son point de vue sur les différents éléments de qualité de celui-ci. Cette assignation se résume à l'affectation, pour chaque élément de qualité, d'une des quatre classes que propose la norme ISO 9126. Chaque classe indique un niveau de classement. Le tableau 3.8

décrit, pour chaque niveau de classement, le taux correspondant. Pour quantifier ces classes, chacune d'elle est représentée par un chiffre entre 0 et 3.

Cette méthode de classement est subjective et elle est destinée à une détermination rapide des objectifs prioritaires [ISO9126, 1992].

Tableau 3.8 Échelle de classement selon ISO 9126

Nombre	Classement	Classement général
3	Excellent	Satisfaisant
2	Bon	Satisfaisant
1	Faible	Satisfaisant
0	Absent	Insatisfaisant

Pour que l'application de cette méthode soit plus pratique, une définition du niveau d'estimation est requise pour chaque caractéristique. Un exemple des définitions des échelles de classement pour la caractéristique « Fonctionnalité » est expliqué dans le tableau 3.9 [ISO 9126, 1992]. Des exemples pour les cinq autres caractéristiques se trouvent dans l'appendice B.

Tableau 3.9 Définition des échelles de classement pour la fonctionnalité

ÉCHELLE DE CLASSEMENT	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Ces limitations peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

D’après ISO 9126, l’échelle de classement est une plage de valeurs sur une échelle permettant de classer un logiciel conformément aux besoins indiqués ou implicites. Cette méthode de marquage est adéquate pour nous rendre possible l’évaluation des points de vues des utilisateurs en général, pour chaque sous-caractéristique de qualité et donc sur l’ensemble de la qualité du système.

3.5. FACTEUR DE QUALITÉ

Le facteur de qualité (QF – Quality Factor) est un moyen de donner une valeur à la perception de la qualité d’un projet en considérant les trois points de vues des trois domaines : économique, social et technique. QF est un système de mesure ouvert : il ne nous impose pas des mesures à utiliser. Ces dernières sont choisies selon le système à mesurer en utilisant un questionnaire auprès des clients, gestionnaires et programmeurs. Le questionnaire est structuré suivant les principes de la Méthodologie de Recherches Sociales et Statistiques, il permet aux trois groupes d’exprimer leurs opinions sur la qualité. Par la suite, et en fonction des réponses collectées, il faut déterminer les mesures et ratios primordiaux à notre évaluation. Et enfin, les informations sur les mesures sont traitées pour obtenir la valeur de la qualité finale. La figure 3.8 présente le flux de procédures du facteur de qualité.

D’après Buglione et al., quatre tableaux (A, B, C et D) sont aussi nécessaires pour pouvoir enregistrer les différents calculs et faciliter la démarche pour trouver la valeur du QF :

1- Tableau A contient la liste de contrôle pour la sélection des sous-caractéristiques de qualité les plus appropriées et le rapport des résultats du questionnaire;

2- Tableau B représente le tableau de calcul de QF utilisé en jonction avec le tableau C;

3- Tableau C est utilisé pour le calcul des poids de pondération des caractéristiques;

4- Tableau D énumère les poids génériques pour chaque priorité.

3.5.1. Procédure de calcul

Une fois les questionnaires collectés et les ratios dégagés, il faut procéder au calcul du facteur de qualité (QF) qui donne la valeur de la qualité du projet à évaluer. Il y a six étapes à suivre.

3.5.1.1. Détermination des sous-caractéristiques de qualité les plus importantes

À partir des questionnaires collectés des trois groupes (économique, social et technique), il faut déterminer une série de caractéristiques et de sous-caractéristiques conformément aux points de vues des utilisateurs et selon les échelles de classement de ISO 9126. D'après ces informations, il faut établir la valeur moyenne de chaque sous-caractéristique par rapport au nombre de personnes qui l'ont choisie. Cette moyenne permet de sélectionner les sous-caractéristiques les plus importantes. Pour chaque sous-caractéristique triée on assigne un poids.

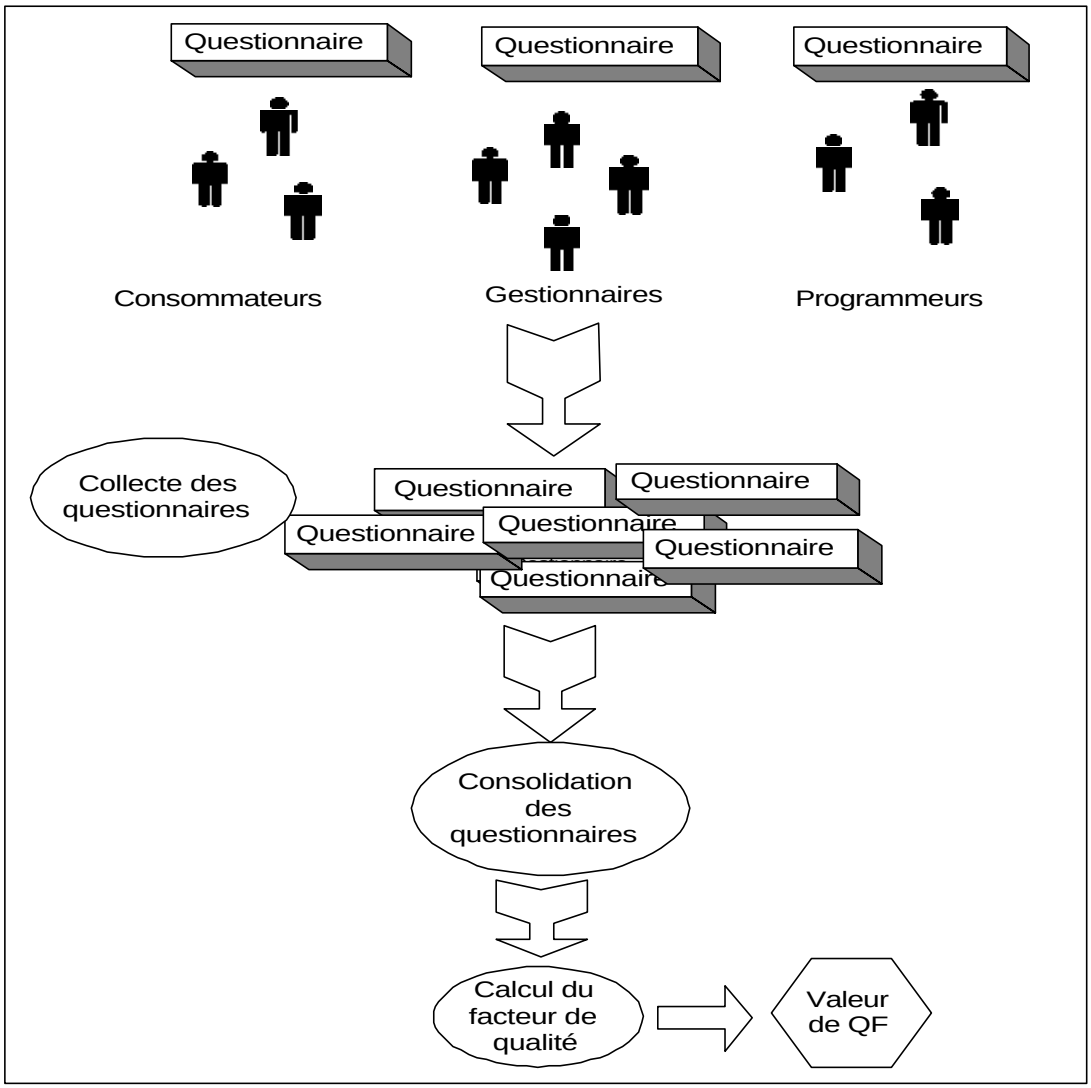

Figure 3.8 Flux des procédures de QF

3.5.1.2. Détermination des priorités des caractéristiques

La priorité d'une caractéristique est fixée en fonction du nombre de ses sous-caractéristiques. Plus il y a de sous-caractéristiques, plus elle est prioritaire et plus grand est le nombre à attribuer à la caractéristique. Le nombre de la plus grande priorité est 1 et le nombre minimum qu'on peut assigner correspond au nombre des

caractéristiques à utiliser. Les règles suivantes doivent être respectées quand deux caractéristiques ou plus ont la même priorité :

a – Le nombre de sous-caractéristiques choisi est égal au nombre des sous-caractéristiques pour deux ou plusieurs caractéristiques : Dans ce cas, on assigne le même rang de priorité pour ces caractéristiques. Le prochain rang de priorité est égal à la valeur de la priorité utilisée à laquelle nous ajoutons le nombre de caractéristiques de ce rang.

Exemple : lignes 3 et 5 et lignes 6 et 7 du tableau 3.10.

b – Le nombre des sous-caractéristiques choisies est différent de celui des sous-caractéristiques pour deux caractéristiques : On utilise le pourcentage respectif de chaque caractéristique. La caractéristique qui a la valeur du nombre de sous-caractéristiques choisies divisé par le nombre des sous-caractéristiques, plus grand, est considérée la plus prioritaire.

Exemple : Lignes 2 et 6 et lignes 2 et 4 du tableau 10.

c – Le nombre des sous-caractéristiques choisies est différent pour quelques caractéristiques : Dans ce cas, il faut considérer, à tour de rôle, la règle (a) puis la règle (b). Pour la prochaine caractéristique, le nombre des caractéristiques qui ont été traitées est ajouté.

Exemple : Lignes 2, 6 et 7.

Tableau 3.10 Exemple de calcul des priorités

Caractéristique	Nombre de sous-caractéristiques	Nombre de sous-caractéristiques choisies	Nombre de sous-caractéristiques choisies / Nombre de sous-caractéristiques	Priorité
Fonctionnalité	5	3	0.6	6
Fiabilité	3	3	1	1
Facilité d'utilisation	3	2	0.666	5
Rendement	2	2	1	1
Maintenabilité	4	3	0.75	3
Portabilité	4	3	0.75	3

3.5.1.3. Assignation des poids aux caractéristiques

Cette étape permet de déterminer un poids pour chaque rang de priorité. Pour commencer on suppose que tous les utilisateurs ont sélectionné toutes les sous-caractéristiques, avec la plus grande échelle de classement, pour chaque caractéristique. Il faut calculer par la suite, et pour chaque caractéristique, la somme des valeurs des sous-caractéristiques ($SCV_{max}(i)$).

Pour chaque caractéristique i, on assigne une pondération de priorité p_i, avec $p_{max} = p_1$ et $p_{min} = p_{nombre\text{-}des\text{-}caractéristiques} > 0$.

Si un nombre de n (n >= 2) caractéristiques ayant la même priorité, leur pondération est égale à la somme des p_i de ces dernières divisée par n.

La relation qui existe entre ces pondérations est la suivante :

$$p_i = p_{min} (\text{nombre des caractéristiques} - i + 1) \qquad (1)$$

Ensuite, il faut calculer la valeur maximale des caractéristiques (CV_{max}) et la valeur totale maximale des caractéristiques (TCV_{max}) d'après les formules suivantes :

$$CV_{max}(i) = p_i * SCV_{max}(i) \qquad (2)$$

$$TCV_{max} = \text{Somme des } CV_{max}(i), \qquad 0 <= TCV_{max} <= 100 \qquad (3)$$

En remplaçant p_i de l'équation (2) par l'équation (1) et selon TCV_{max}, qui doit être spécifié par l'utilisateur, on peut obtenir p_{min}. Et enfin, il faut calculer les priorités pour chaque rang en remplaçant p_{min} dans l'équation (1).

3.5.1.4. Somme des valeurs des sous-caractéristiques

Cette étape permet de calculer la somme des valeurs des sous-caractéristiques pour chaque caractéristique, mais cette fois-ci on ne considère que les valeurs qui ont été collectées des questionnaires.

SSV = Somme SCV (i); SSV : Sum of subcharacteristics values

3.5.1.5. Calcul de la valeur des caractéristiques

On calcule la valeur de chaque caractéristique comme suit :

CV = Pondération des priorités * SSV CV : Characteristic value

3.5.1.6. Calcul de la valeur totale des caractéristiques

L'avant dernière étape pour le calcul du facteur de qualité, consiste à calculer la valeur caractéristique totale :

TCV = Somme des CV; TCV : Total characteristic value

3.5.2. Détermination du facteur de qualité

Enfin, la valeur du facteur de qualité doit être calculée selon l'équation suivante :

$$QF = TCV / TCV_{max}$$

Dans d'autres documents, nous trouvons aussi que :

$$QF = (TCV – MQL) * 0.01$$

avec MQL qui représente la subjectivité des poids des sous-caractéristiques.

Cette méthode pour calculer le facteur de qualité selon la perception de la qualité de groupes de domaines différents permet de donner une idée globale de la qualité d'un logiciel. Mais le QF n'est pas suffisant pour prendre des décisions concernant un logiciel tout au long de son cycle de vie.

3.6. CONCLUSION

Dans ce chapitre, nous avons défini l'utilisation des concepts que nous avons adoptés pour la réalisation de notre projet. Nous avons modifié certains de ces concepts pour mieux les adapter dans le contexte de notre travail. Il se trouve que ces derniers ne sont pas bien expliqués dans les articles concernent QEST. Il est nécessaire de bien les comprendre pour pouvoir, par la suite, mieux assimiler l'utilisation du modèle QEST. Le détail de l'utilisation de ces méthodes dans le modèle QEST se trouve dans le chapitre 4 suivant.

CHAPITRE IV

LE PROTOTYPE QEST

Le modèle QEST est un modèle qui nous permet d'avoir une vision globale sur la performance d'un logiciel. Il prend en considération les opinions des consommateurs, des programmeurs et des gestionnaires sur la qualité du produit. Le plus grand avantage de ce modèle est qu'il résume la perception qualitative et quantitative d'un projet en une seule valeur numérique.

Étant donné les inconvénients de compréhension et de traitement de calcul que présente QEST tels que défini par Buglione et al., le fait de faciliter l'utilisation en automatisant les fonctions manuelles du modèle est d'un grand secours pour les utilisateurs. Notre but est de réduire les efforts des utilisateurs de ce modèle en leur proposant seulement deux tableaux à manipuler et aucun calcul à traiter. Toute la partie complexe de ce modèle est traitée par le prototype QEST lui-même.

4.1. CONDITIONS POUR RÉALISER LE PROTOTYPE

Avant de commencer à programmer notre prototype, il nous a fallu du temps pour maîtriser le modèle théorique QEST. La compréhension de ce dernier tel que présenté dans les articles de Buglione et al. n'est pas facile à assimiler. Tout d'abord, il nous a fallu acquérir certaines connaissances dans le domaine de l'informatique et plus précisément en ce qui concerne les systèmes de qualité. De plus, pour mieux comprendre l'utilisation de certaines formules mathématiques, nous

avons eu recours à des explications de M. Buglione (Appendice C) qui nous a aidé à mieux comprendre le déroulement des étapes du modèle.

Ensuite, nous avons fait une recherche sur les concepts qui étaient moins bien expliqués dans les articles, comme par exemple le concept de marquage et GQM.

Par la suite, nous avons essayé de comparer la méthode QEST à la méthode TQM (Total Quality Model) [Kataoka et al., 1998] et nous avons développé une synthèse sur les avantages de ces modèles et de ces exigences (Appendice D).

À partir de toutes ces informations, nous avons pu déterminer la stratégie appropriée pour concevoir notre prototype et définir les entrées et les étapes nécessaires pour arriver à notre valeur de performance tout en tenant compte de nos objectifs.

Après avoir réalisé une première version du prototype QEST, nous l'avons envoyé au professeur Luigi Buglione pour avoir son avis et ses commentaires sur l'interface présentée et l'exactitude des résultats (Appendice E). Ces commentaires nous ont permis de mieux présenté le prototype.

4.2. ENTRÉES ET SORTIES DU SYSTÈME QEST

Lors de la manipulation du prototype QEST proposé, nous pouvons obtenir des définitions de tous les mots utilisés. A chaque fois que nous rencontrons un mot clé dont nous ne savons pas exactement la signification, nous pouvons cliquer dessus pour avoir la définition précise dans le contexte de notre modèle, ce qui va nous permettre de bien l'utiliser (Appendice F).

Afin d'arriver à de bons résultats corrects de calcul de performance en utilisant QEST, un ensemble d'entrées doit être minutieusement choisi. Ces entrées se

divisent en deux classes : la première classe regroupe les entrées quantitatives (ratios) que nous pouvons mesurer dans notre système et, la deuxième, concerne les entrées qualitatives et subjectives qui reflètent les opinions des participants.

4.2.1. Sélection des ratios et assignation des poids

Le premier ensemble d'entrées dont nous avons besoin pour l'utilisation de QEST est celui des ratios. Comme nous l'avons décrit à la section 2 du chapitre 3 concernant l'identification des mesures, la méthode GQM(R) (Goals Questions Measures and Ratios) est une méthode qui nous facilite la fixation des mesures et ratios pour définir la qualité d'un système. Ces mesures sont définies selon les besoins des utilisateurs dans les domaines économique, social et technique.

En suivant GQM(R), nous commençons par définir les objectifs que nous voulons atteindre lors de la conception et l'utilisation de notre système. Pour bien préciser ces buts, nous les classons en trois catégories : économique, social et technique. Par la suite, nous essayons de fixer les questions appropriées qui nous aident à mieux cerner nos buts. Une fois ces questions établies, nous devons leur trouver des réponses sous formes de mesures, c'est-à-dire d'éléments à une variable qui vont nous permettre de quantifier nos besoins. Enfin, à partir de ces mesures, et afin de déterminer les relations qui existent entre les premières et les différents domaines, nous établissons un ensemble de ratios qui nous permet une analyse plus significative par rapport à l'usage de mesures simples. Pour chaque ratio, nous donnons la valeur optimale que nous voulons atteindre dans notre projet et nous la plaçons dans la deuxième colonne concernant les valeurs cibles.

Lors de l'utilisation de notre nouveau modèle QEST, et après avoir inscrit le nom du projet à évaluer et le nom du chef du projet qui accomplit cette tâche, nous

remplissons la première colonne du premier tableau (Tableau 4.11) par le nom des ratios que nous avons dégagés. Nous pouvons mettre jusqu'à vingt et un ratios.

L'étape suivante consiste à évaluer les ratios en commençant par donner les limites de chacun c'est-à-dire de définir la valeur minimale et la valeur maximale que peut avoir un ratio. Puis, nous observons la valeur du ratio dans notre projet. Toutes ces informations sont nécessaires pour rendre possible la normalisation des valeurs par notre modèle QEST vu que nous travaillons toujours par rapport à un tétraèdre régulier.

Tableau 4.11 Tableau des ratios

Ratios	Cible	Val. Pro.	Val. Min.	Poids			Val. réel.	Valeurs finales		
				E	S	T		E	S	T
1	0	0	0	0.00	0.00	0.00	0			
2	0	0	0	0.00	0.00	0.00	0			
3	0	0	0	0.00	0.00	0.00	0			
4	0	0	0	0.00	0.00	0.00	0			
5	0	0	0	0.00	0.00	0.00	0			
6	0	0	0	0.00	0.00	0.00	0			
7	0	0	0	0.00	0.00	0.00	0			
8	0	0	0	0.00	0.00	0.00	0			
9	0	0	0	0.00	0.00	0.00	0			
10	0	0	0	0.00	0.00	0.00	0			
11	0	0	0	0.00	0.00	0.00	0			
12	0	0	0	0.00	0.00	0.00	0			
13	0	0	0	0.00	0.00	0.00	0			
14	0	0	0	0.00	0.00	0.00	0			
15	0	0	0	0.00	0.00	0.00	0			
16	0	0	0	0.00	0.00	0.00	0			
17	0	0	0	0.00	0.00	0.00	0			
18	0	0	0	0.00	0.00	0.00	0			
19	0	0	0	0.00	0.00	0.00	0			
20	0	0	0	0.00	0.00	0.00	0			
21	0	0	0	0.00	0.00	0.00	0			

Tous les ratios n'ont pas la même participation dans le calcul de la qualité du projet et dans tous les domaines, ce qui nous mène à donner un poids spécifique pour chaque ratio que ce soit dans le domaine économique, social ou technique. Avant de continuer à remplir notre premier tableau, nous avons à définir la contribution de ces poids à assigner, par rapport aux informations objectives, dans le calcul de la qualité du projet. La somme de cette contribution dans un même domaine est désignée par Buglione et al. par MQL (Mean Quality Level).

$$0 < \text{MQL} \leq 50 \qquad (1)$$

Il est donc impératif de prendre en considération que la somme des poids des ratios (pr) dans un domaine doit être inférieure ou égale à la valeur de MQL.

$$\sum_{i=1}^{n} pr_i \leq \text{MQL} \qquad (2)$$

Une fois que nous avons fixé le MQL, il nous est possible de définir le poids de chaque ratio selon chaque domaine et ainsi nous complétons la manipulation de notre premier tableau et nous passons au tableau suivant. Les colonnes de Val. réel. et Valeurs finales seront utilisées pour afficher les résultats de normalisation des valeurs et poids des ratios.

4.2.2. Fixation des points de vues

La deuxième catégorie des entrées de notre modèle QEST concerne les opinions des participants sur la qualité du projet à évaluer. Ces opinions sont nécessairement subjectives. Dans notre modèle, nous utilisons les caractéristiques et sous-caractéristiques de qualité de la norme ISO 9126. Le deuxième tableau (Tableau 4.12) contient ces dernières et il nous permet de noter les points de vues de cinq participants : deux clients, deux gestionnaires et un technicien programmeur.

Chaque participant doit donner son avis sur les sous-caractéristiques dont il estime la présence importante dans le projet à évaluer. Son avis se résume par l'assignation d'une valeur entre 0 et 3 pour les sous-caractéristiques qui se trouvent au deuxième tableau, pour tolérer par la suite, le traitement mathématiques de ces points de vues. Cette assignation doit refléter leurs perceptions du degré de satisfaction de chaque sous-caractéristique dans le système à évaluer. Plus la sous-caractéristique est satisfaisante, plus grand est le chiffre qui lui est attribué.

Tableau 4.12 Tableau des points de vues

Caractéristiques	Sous-caractéristiques	Util1	Util2	Gest1	Gest2	Tech1	SCV
Fonctionnalité	Aptitude	0	0	0	0	0	0
	Exactitude	0	0	0	0	0	0
	Interopérabilité	0	0	0	0	0	0
	Conformité réglementaire	0	0	0	0	0	0
	Sécurité	0	0	0	0	0	0
Fiabilité	Maturité	0	0	0	0	0	0
	Tolérance aux fautes	0	0	0	0	0	0
	Possibilité de récupération	0	0	0	0	0	0
Facilité d'utilisation	Facilité de compréhension	0	0	0	0	0	0
	Facilité d'apprentissage	0	0	0	0	0	0
	Facilité d'exploitation	0	0	0	0	0	0
Rendement	Comportement vis-à vis du temps	0	0	0	0	0	0
	Comportement vis-à-vis des ressources	0	0	0	0	0	0
Maintenabilité	Facilité d'analyse	0	0	0	0	0	0
	Facilité de modification	0	0	0	0	0	0
	Stabilité	0	0	0	0	0	0
	Facilité de test	0	0	0	0	0	0
Portabilité	Adaptabilité	0	0	0	0	0	0
	Facilité à l'installation	0	0	0	0	0	0
	Conformité	0	0	0	0	0	0
	Interchangeabilité	0	0	0	0	0	0

Plus de détails sur le classement des points de vues se trouvent dans la section 3.4 du troisième chapitre.

4.2.3. Résultat et interprétation

Le modèle QEST que nous présentons ne se contente pas de nous donner la valeur de la performance d'un système, il nous donne en quelques secondes une foule d'informations qui peuvent nous donner une idée encore plus précise sur la qualité et la productivité du projet.

La représentation des résultats par le modèle QEST se trouve à la figure 4.9.

Après avoir rempli les deux tableaux avec les informations nécessaires, l'utilisateur n'a qu'à appuyer sur le bouton « OK » pour voir ses valeurs des ratios normalisées (val réel) et interprétées selon les poids affectés (Valeurs finales E, S et T) (Tableau 4.11). Nous obtenons aussi les valeurs de productivité de chaque domaine (productivité économique (e), productivité sociale (s) et productivité technique (t)), la moyenne des points de vues des participants concernant les sous-caractéristiques (SCV), la somme des valeurs des sous-caractéristiques pour chaque caractéristique (SSV) et le nombre des sous-caractéristiques choisies pour chaque caractéristique et dans chaque domaine (Nb de sscaract E, Nb de sscaract S, Nb de sscaract T) (Voir tableau de la figure 4.9).

Le bouton QF nous permet d'avoir la valeur du facteur de qualité (facteur de qualité), tout en donnant la valeur intermédiaire pour le calcul de ce dernier, qui est la valeur totale des caractéristiques qui représente le niveau de qualité estimé du projet (TCV). Le facteur de qualité représente l'évaluation numérique de la qualité selon les différentes opinions des participants. En plus, le modèle détermine les valeurs de productivité et de qualité pour chaque domaine (e', s', t').

Pour avoir la valeur de la performance du système à évaluer, nous appuyons sur le bouton « Performance ». Une fois le traitement effectué, nous obtenons les informations suivantes :

- la priorité des caractéristiques selon les points de vues des clients (P(U)), des gestionnaires (P(G)) et des techniciens programmeurs (P(T));

- les coordonnées des extrémités des plans (Q_e, Q_s, Q_t) et ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) :

  Vu que QEST utilise un tétraèdre (3 dimensions) pour le calcul de performance dont la base est formée de trois vecteurs E, S,T (pour chaque domaine) et le sommet (P) présente la performance maximale d'un projet, le plan (Q_e, Q_s, Q_t) est défini par la position de chaque valeur finale d'un domaine (e, s, t) sur le vecteur correspondant EP, SP ou TP. Ce plan détermine une évaluation quantitative du projet.

  Le plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) est le plan défini par les valeurs e', s', t' dans le tétraèdre.

- l'équation du plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$). Ce plan détermine une évaluation qualitative et quantitative du projet;

- la hauteur qui représente la valeur de la distance entre le sommet du tétraèdre et l'hyperplan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$). Moins cette distance est importante, meilleure est la performance;

- l'aire du plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) : plus la valeur de cette aire se rapproche de zéro, plus notre système est performant;

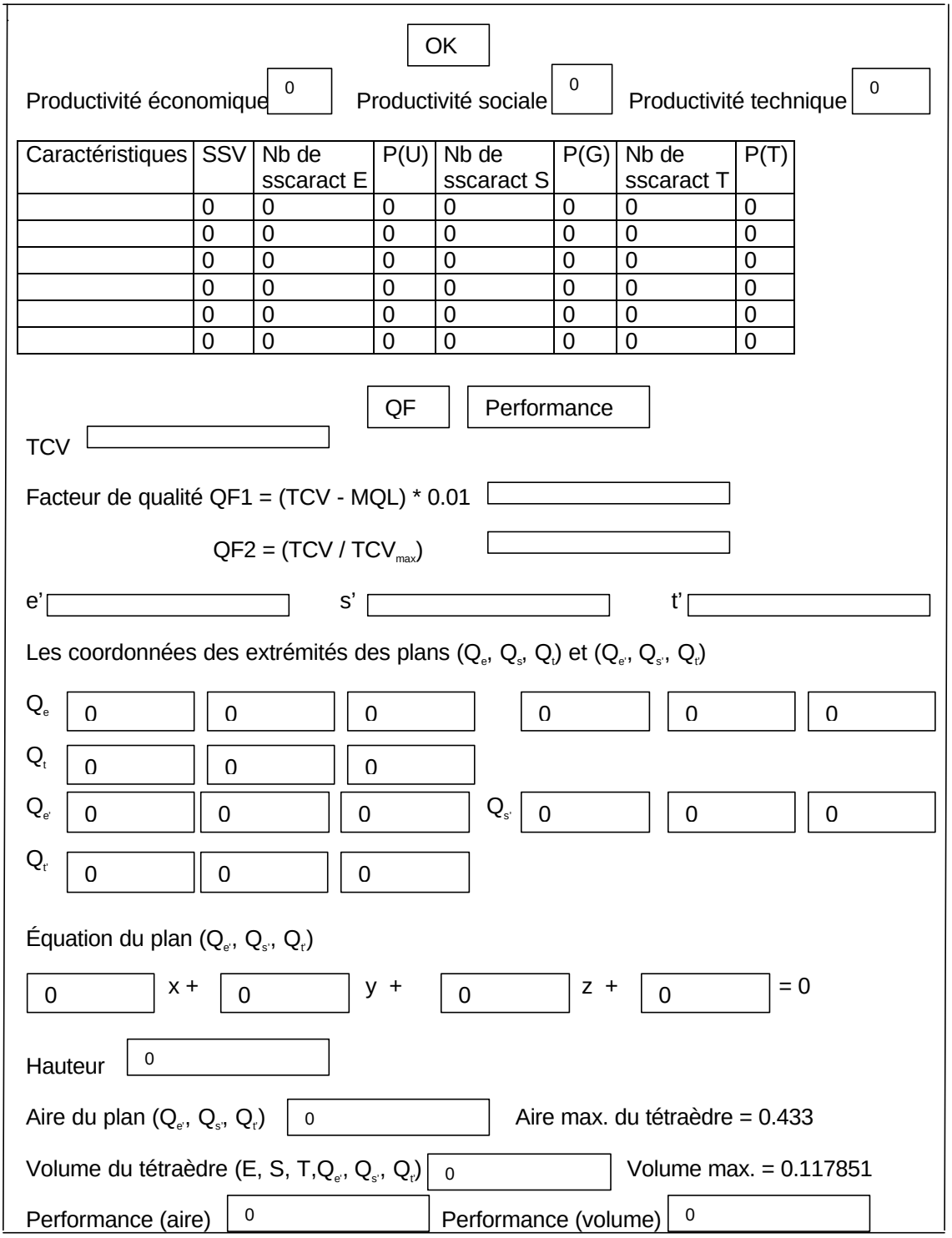

Figure 4.9 Résultats

- le volume du tétraèdre tronqué (E, S, T, $Q_{e'}$, $Q_{s'}$, $Q_{t'}$) qui représente le volume comblé pour avoir la performance du projet. La valeur 0,117851 représente le volume maximum du tétraèdre, c'est-à-dire plus la valeur du volume est proche de cette dernière, plus la performance est optimale;

- la valeur de la performance selon l'aire (Performance(aire)) et la valeur de la performance (Performance(volume)) selon le volume qui est la plus significative.

Dans les deux cas, que ce soit pour la performance selon l'aire ou selon le volume, la meilleure valeur de performance que nous pouvons avoir est de 1. Plus la valeur de performance est proche de 1, plus notre système est performant et répond aux exigences économique, sociale et technique.

Nous remarquons bien la simplicité de ce modèle QEST par rapport à celui présenté par Buglione et al. : moins de données à manipuler et aucun calcul à faire.

4.3. IMPLANTATION DU PROTOTYPE

Afin d'automatiser le modèle QEST, et pour qu'il soit accessible par tous, rien de tel que de réaliser ce modèle en utilisant l'outil le plus en vogue dans le domaine de l'informatique qui est l'Internet.

Notre implantation se présente en deux étapes : la première concerne la réalisation de la page Web en utilisant l'outil Dreamweaver et la deuxième concerne l'implantation du traitement mathématique avec JavaScript.

4.3.1. Utilisation de Dreamweaver

Pour la création de la page Web, l'outil HTML est nécessaire. Le logiciel Dreamweaver permet de faciliter l'utilisation du code HTML grâce à ses fonctions prêtes à être implantées. Par exemple, il suffit de cliquer sur « Table » du menu « Insert » et de choisir le nombre de lignes et de colonnes de notre tableau pour avoir ce dernier sur notre page.

L'espace de travail de Dreamweaver est flexible, et offre différents modèles pour travailler différents niveaux d'expertise. Il nous permet de collecter les informations des utilisateurs grâce aux Forms dans lesquelles nous pouvons inclure des objets standards, par exemple des champs texte, des boutons, des tableaux, etc. Pour traiter les informations se trouvant dans notre Form, nous devons utiliser un éditeur de texte pour écrire le script ou l'application à utiliser. Plusieurs langages peuvent être utilisés en Dreamweaver tels que JavaScript et Perl.

Les deux principaux outils de Dreamweaver que nous avons utilisés, à part les tableaux, sont les champs texte et les liens.

Pour insérer un champ texte, nous n'avons qu'à utiliser « Text Field » de la rubrique « Form Object » du menu « Insert ». Nous pouvons déterminer la longueur de notre champ et sa valeur initiale s'il y a lieu. Nous avons aussi inséré des champs texte dans nos tableaux.

À chaque champ texte, nous devons associer un identificateur qui nous permet, par la suite, de saisir la valeur de ce champ donnée par l'utilisateur, pour être traitée par le programme. La saisie de cette valeur en utilisant JavaScript est comme suit :

```
eval(form.id_champ_texte.value)
```

Pour chaque terme technique utilisé dans notre interface, nous avons créé un lien qui nous donne une définition précise de ce terme. Pour ce faire, nous avons utilisé l'instruction suivante :

javascript:alert(`` Définition du terme technique ``)

À l'exécution, ceci nous présente une boîte de dialogue dans laquelle nous pouvons lire la définition.

Ces deux outils nous ont permis d'établir une interface facile à manipuler par les utilisateurs.

4.3.2. Calculs par JavaScript

La phase d'implantation des calculs se divise en deux parties : une partie de contrôle numérique et une autre pour le traitement des informations qui nous permet d'aboutir à nos résultats voulus.

4.3.2.1. Contrôle numérique

Pour éviter de se tromper et de donner de fausses valeurs, nous avons établi des instructions de contrôle qui permettent de vérifier si les valeurs correspondent bien aux attentes de QEST sinon un message indiquant le type d'erreur s'affiche lors de l'exécution. Comme par exemple donner une valeur minimale de ratio qui soit plus grande que celle de la valeur maximale, ou bien de ne pas dépasser la valeur de MQL dans la somme des poids de nos ratios,

Un exemple de ces instructions est le suivant :

```
if ( eval ( form.MQL.value ) > 50){
    javascript : alert (" Erreur: MQL doit être inférieur ou égal à 50. ")     }
```

Cette instruction nous informe que nous avons dépassé la limite de 50 pour la valeur de notre MQL, il faut donc revoir cette valeur.

Ce contrôle nous aide à bien choisir nos valeurs tout en respectant les contraintes exigées par le modèle QEST.

4.3.2.2. Calculs

Dans le programme, nous avons utilisé pour chaque bouton une fonction. En tout, nous avons trois grandes fonctions :

1. fonction de normalisation;
2. fonction de calcul du facteur de qualité;
3. fonction de calcul de la performance.

Dans ces trois fonctions, nous avons utilisé les deux structures d'itération : IF THEN ELSE et FOR. Des tableaux étaient aussi nécessaires pour notre travail pour la fonction calcul de facteur de qualité. Vu que dans les tableaux de Dreamweaver nous ne pouvons pas faire directement référence aux champs des éléments des tableaux, nous avons décidé d'en créer. Ces derniers nous permettent de stocker des informations qui seront utiles pour d'autres instructions. De plus, il est plus facile d'utiliser un tableau pour chercher, par la suite, nos valeurs par une simple indexation. Ensuite, nous pouvons manipuler nos informations en utilisant de simples boucles For. Sans ces tableaux, nous pourrions nous perdre dans les calculs et compliquer notre code de programme.

Un exemple d'utilisation de tableau dans notre programme se trouve à la figure 4.10.

```
E = new Array(5)
 for (i=0; i<6 ; i++) {
    E[i] = new Array(5);
    E[0] = eval(form.E1.value);
    E[1] = eval(form.E2.value);
    E[2] = eval(form.E3.value);
    E[3] = eval(form.E4.value);
    E[4] = eval(form.E5.value);
    E[5] = eval(form.E6.value);
  }
 maxE = new Array(5)
 for (i=0; i<6; i++) {
    maxE[i] = new Array(5);
    maxE[0] = 5;
    maxE[1] = 3;
    maxE[2] = 3;
    maxE[3] = 2;
    maxE[4] = 4;
    maxE[5] = 4;
  }
 PE = new Array(5)
 for(k=0; k<6; k++){
     PE[i] = new Array(5);
     if (E[k]!= 0) {
       PE[k] = 1}
         else{ PE[k] = 1000}
  }
 for(i=0; i<5; i++) {
     for (j=i+1; j<6 ;j++) {
          if (E[i] > E[j]) {
              PE[j] = PE[j] + 1 }
          if (E[i] < E[j]) {

              PE[i] = PE[i] + 1;
           }
          if  (E[i] == E[j]) {
             if  (maxE[i] > maxE[j]){
                PE[i] = PE[i] + 1 }
               else {
                   if (maxE[i] < maxE[j]){
                      PE[j] = PE[j] + 1}
                     else {
                         PE[j] = PE[i]   }
                  }
             }
        } }
     form.Pu1.value = PE[0];
     form.Pu2.value = PE[1];
     form.Pu3.value = PE[2];
     form.Pu4.value = PE[3];
     form.Pu5.value = PE[4];
     form.Pu6.value = PE[5];
```

Figure 4.10 Programme concernant les tableaux

Cette partie du programme nous donne la priorité des caractéristiques par rapport au nombre de ses sous-caractéristiques choisies par les participants du domaine économique. Les résultats de cette partie du code se trouve dans le tableau des résultats et elle est désignée par la colonne P(U).

- E est un tableau de six lignes et une colonne. E[i] contient le nombre de sous-caractéristiques choisies par les participants pour la caractéristique i.
- maxE est aussi un tableau de six lignes et une colonne. maxE[i] désigne le nombre maximum de sous-caractéristiques pour la caractéristique i.
- PE est un tableau de six lignes qui nous permet de sauvegarder la dernière valeur pour chaque caractéristique à la fin de chaque itération.
- Pu1 jusqu'à Pu6 sont les champs textes pour mettre les résultats. Nous ne pouvons utiliser Pu[i] en Dreamweaver.

Comme nous pouvons le constater, les tableaux nous ont été d'un grand secours. Plus d'une centaine de lignes auraient dû être utilisées pour atteindre notre résultat si nous n'avions pas ce système d'indexation.

4.3.3. Problèmes rencontrés

Lors de notre implantation, nous avons rencontré trois problèmes : le premier concerne les tableaux dynamiques, le second l'indexation des éléments d'un tableau de Dreamweaver et le troisième, les valeurs initiales des champs texte.

Nous avons bien voulu que notre tableau de ratios soit un tableau dynamique, c'est-à-dire qu'à chaque fois que l'utilisateur voudrait ajouter un ratio, une nouvelle ligne se créerait automatiquement. Ceci aurait pu être utilisé aussi pour ne pas limiter le nombre de personnes qui donneront leur avis sur les sous-caractéristiques. Hélas, Dreamweaver ne nous permet pas de telles fonctions et nous n'avons pas pu trouver

un autre moyen pour arriver à ce but. À la fin, nous avons opté pour offrir des tableaux statiques, en attendant de trouver une solution à ce problème.

Le deuxième problème est le fait que nous ne pouvons pointer directement sur les éléments d'un tableau de Dreamweaver. À chaque fois, il fallait évaluer le contenu du champ et donc vérifier l'identificateur de chacun avant. Ceci nous a pris un temps considérable car il nous a fallu traiter chaque élément du tableau seul. Une méthode d'indexation des tableaux aurait facilité la tâche et contribué à diminuer la taille du code.

Le troisième problème auquel nous avons fait face, est celui de l'initialisation des champs texte qui nous sont utiles pour le traitement mathématique. Un champ utilisé pour le calcul qui ne contient pas un 0 pour dire que cette valeur est nulle donnera une erreur dans les calculs. C'est pour cela que nos champs sont initialisés par 0 et que l'utilisateur doit effacer à chaque fois ce 0 pour le remplacer par la valeur appropriée selon les mesures de son système ou les points de vues des participants.

Heureusement pour nous que ces problèmes n'étaient pas sans solution et que nous avons pu réaliser notre prototype.

4.4. Tests

Une fois notre prototype réalisé, nous pouvons dire que nous avons atteint nos objectifs :

- Accessibilité à un nombre plus grand d'utilisateurs vu qu'aucun pré-requis dans le domaine mathématique ni celui du génie logiciel n'est exigé;
- Facilité d'utilisation : moins de données à manipuler;

- Utilisation de deux tableaux par rapport aux six proposés par Buglione et Abran : un premier tableau pour les ratios et un autre pour les points de vues;

- Définition de chaque mot technique employé afin de faciliter l'utilisation du prototype et éviter toutes ambiguïtés;

- Aucun traitement mathématique à effectuer : le prototype se charge de tous les calculs nécessaires pour donner des résultats justes et précis;

- Gain de temps et des résultats sans fautes vu que les traitements sont automatiques et non manuelles.

Pour déterminer que notre prototype s'adapte à différents navigateurs, nous l'avons manipulé en utilisant les navigateurs Internet Explorer 5 et Netscape 4.7. Dans les deux cas, notre prototype a bien passé le test.

Afin de prouver le bon fonctionnement de notre prototype QEST, nous avons demandé à deux étudiants de niveau maîtrise en informatique de l'utiliser. Vu que ces étudiants n'avaient pas un vrai système à mesurer, nous leur avons fourni les informations qui ont été données dans l'exemple que proposent Buglione et al. [Buglione et al., 1999]. Ils ont apprécié la facilité d'utilisation du prototype, les définitions qui accompagnent les termes techniques et le contrôle numérique qui se fait automatiquement. Leurs résultats étaient identiques aux résultats trouvés dans l'exemple de Buglione. L'appendice G montre comment est utilisé l'exemple de Buglione avec notre prototype QEST.

4.5. CONCLUSION

Si nous comparons le modèle QEST de Buglione et al. avec le nôtre, nous voyons bien que ce dernier est beaucoup plus simple. Les deux tableaux pour nos entrées sont de loin comparables aux six tableaux à manipuler avec l'ancien modèle. De plus, nous avons caché tout ce qui pose des problèmes dans les procédures d'utilisation de QEST. Ceci rend notre nouveau modèle QEST facile à comprendre et évite de rentrer dans des détails de traitement mathématique qui peuvent induire en erreur.

CONCLUSION ET PERSPECTIVES

Avoir un outil qui nous permet de mesurer le degré de performance des projets, nous aide à évaluer ces derniers et à les améliorer.

Le modèle QEST est un modèle qui détermine quantitativement la performance d'un système, utilisant des valeurs de ratios et des points de vues de participants appartenant à différents domaines : économique, social et technique. Ce modèle, présenté par Buglione et al., est difficile à comprendre et à utiliser. Au niveau pratique, ce modèle présente certains inconvénients.

Ce modèle très complexe ne pouvait donc être utilisé que par quelques rares experts du domaine au niveau international. Une utilisation plus courante en industrie, demandait donc que toutes les parties complexes du modèle QEST soient identifiées et même cachées en arrière plan de façon à créer une interface beaucoup plus conviviale pour les utilisateurs de façon à leur permettre de focuser leur attention sur la collecte de données, somme toute relativement simple chacune, mais dont la manipulation subséquente était hautement complexe, et par conséquent très problématique. L'utilisateur se trouve mêler entre plusieurs concepts, mots techniques et différents tableaux. De plus, des pré-requis sont nécessaires pour arriver à le manipuler. Des connaissances en mathématique et qualité de systèmes sont nécessaires pour comprendre le fonctionnement du modèle.

Notre travail consistait à rendre QEST accessible à tout le monde. Les solutions que nous avons trouvées, étaient d'éclaircir certains concepts, en présentant dans ce mémoire l'utilisation de marquage, la façon d'établir la liste des ratios avec GQM(R) et d'éliminer les étapes qui nécessitent des traitements mathématiques et de gestions de tableaux, en les automatisant. Ces étapes exigent pas mal d'effort de la

part de l'utilisateur, vu que ce sont des fonctions manuelles et répétitives. Pour mieux utiliser le prototype QEST, et pour que l'utilisateur ne s'embrouille pas avec les différents mots techniques employés, nous avons donné l'accès aux définitions de chaque terme pour que celui-ci peut être compris par les utilisateurs du modèle.

A la fin, nous sommes arrivés à réaliser notre prototype qui consiste à concrétiser ce modèle en une application sur Internet qui permet de faciliter la compréhension et l'utilisation de QEST et d'automatiser le traitement des fonctions manuelles. L'utilisateur n'a qu'à remplir deux tableaux et à appuyer sur un bouton pour voir ses résultats s'afficher.

Les trois principaux avantages de notre prototype sont : la simplicité d'utilisation, l'utilisation de concepts standards et la rapidité de traitement. L'utilisateur n'aura plus à se soucier des calculs et de toutes les étapes de la première version du modèle QEST.

Le modèle de mesure de qualité et de performance QEST continue à évoluer et se doit d'être maintenu [Buglione et al., 1999]. Voici quelques suggestions pour des applications futures.

Pour plus de flexibilité du prototype QEST, nous pouvons laisser le choix à l'utilisateur d'intégrer son propre modèle de qualité en fixant ses caractéristiques et sous-caractéristiques de qualité au lieu d'utiliser juste les caractéristiques de la norme ISO 9126. De plus, notre prototype serait plus flexible si nous donnions à l'utilisateur la possibilité d'avoir les points de vues de plus de cinq participants. Pour ces deux perspectives, il faudra trouver le moyen d'utiliser des tableaux dynamiques qui permettront d'ajouter des lignes et des colonnes selon le désir des utilisateurs.

Une autre éventualité qui pourrait être intégrée dans le prototype, est l'utilisation de valeurs cibles pour les ratios, c'est-à-dire des valeurs définies durant la phase de spécification de la qualité, et des valeurs estimées qui peuvent être recueillies à

partir des données passées. Pour ces dernières, les valeurs des expériences passées de mesure de performance des projets devront être sauvegardées de façon à permettre la consultation de ces valeurs et la comparaison de celles-ci avec celles des nouveaux systèmes à mesurer. En plus, ces expériences seront utiles pour l'assignation des différentes valeurs et le choix des ratios.

Pour améliorer encore plus QEST, il pourrait être fait en sorte que chaque participant donne, non seulement son point de vue sur les différentes sous-caractéristiques de qualité mais aussi les valeurs minimales, maximales et cibles du projet à mesurer, selon son opinion, car il peut y avoir une certaine divergence avec les valeurs établies par le chef de projet. Ceci, permettrait de mieux voir les perceptions économiques, social et technique quantitativement et qualitativement. De plus, avoir comme résultat un graphique en trois dimensions qui représenterait le tétraèdre avec la performance évaluée du système à mesurer nous donnera une idée globale sur les différents résultats obtenus.

Il est aussi intéressant de pouvoir utiliser le prototype QEST tout au long du cycle de vie du système. Ainsi, nous pourrons, à la fin de chaque phase, déterminer ce qui est possible d'améliorer avant de passer à l'autre. Et par la suite, nous aurions un système dont la performance serait meilleure.

Nous remarquons que ce prototype QEST est riche et que beaucoup d'options peuvent être ajoutées dans le futur pour le rendre encore plus flexible et utilisable par un plus grand nombre de gestionnaires de qualité et de gestionnaires de projet.

APPENNDICE A

TABLE DES CARACTÉRISTIQUES, SOUS-CARACTÉRISTIQUES ET MESURES SELON PSM

CARACTÉRISTIQUE	SOUS-CARACTÉRISTIQUE	MESURE
Shedule and Progress	Milestone Performance	Milestone Data
	Work Unit Progress	Component Status Requirement Status Test Case Status Path Tested Problem report Status Reviews Completed Chang request Status
	Incremental Capability	Build Content - Component Build Content - Function
Resources and Cost	Personnel	Effort Staff Experience Staff Turnover
	Financial Performance	Earned value Cost
	Environment Availability	Resource Availability Dates Resource Utilization
Growth and Stability	Product Size and Stability	Lines of Code Components Words of Memory Database Size
	Functional Size and Stability	Requirements Function Points Change Request Workload
Product Quality	Defects	Problem Reports Defect Density Failure Interval
	Complexity	Cyclomatic Complexity
	Rework	Rework Size Rework Effort
Development Performance	Process Maturity	Capability Maturity Model Level
	Productivity	Product Size/Effort Ratio Functional Size/Effort Ratio

Technical Adequacy	Target Computer Resource Utilization	CPU Utilization CPU Throughput I/O Utilization I/O Throughput Memory Utilization Storage Utilization Response time
	Technical Performance	Achieved Accuracy in Requirements (Concurrent Tasking, Data Handling, or Signal Processing)
	Technology impact	Quantitative impact of new technology (NDI Utilization, Size by Origin, or Cycle Time)

APPENDICE B

LES DÉFINITIONS DE NIVEAUX D'ESTIMATION SELON LES CARACTÉRISTIQUES DE ISO

Tableau B.1

NIVEAU D'ESTMATION	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Cette limitation peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

Tableau B.2

NIVEAU D'ESTMATION	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Cette limitation peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

Tableau B.3

NIVEAU D'ESTMATION	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Cette limitation peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

Tableau B.4

NIVEAU D'ESTMATION	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Cette limitation peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

Tableau B.5

NIVEAU D'ESTMATION	DEGRÉ DE SATISFACTION
Excellent	Exécute les fonctions dont l'utilisateur a besoin sans limite.
Bon	Exécute les fonctions dont l'utilisateur a besoin avec quelques limites mineures dans certaines.
Faible	Exécute les fonctions dont l'utilisateur a besoin avec certaines limites. Cette limitation peuvent être surmontées par des procédures manuelles.
Absent	Les fonctions fournies ne répondent pas aux besoins de l'utilisateur.

APPENDICE C

COMMENTAIRES DU PROFESSEUR LUIGI BUGLIONE À PROPOS DE LA PREMIÈRE VERSION DU PROTOTYPE QEST

Bilbao, February 3rd, 2000

Subject: *comments on Myriam's QEST-based web prototype*

Bonjour Myriam,
merci encore pour ton intérêt dans le modèle QEST: here in the following there are some observations and suggestions after analysing the web prototype you sent me last week. A premise: I refer to Table 1, 2 and 3 with respect to the tables inserted in the web page in the order they appear.

1. **Subdivide the actual web page into several ones**. In order to give user a higher visibility and readability of the contents of the page, it could be better to split the content into distinct pages. For instance:
   - **Page 1**: insert project data (System name, Project Manager name, project characteristics…)
   - **Page 2**: * insert quantitative data (MQL and TCVmax level, insert data in Table 1)
     * execute the quantitative evaluation
     * showing the quantitative result (Qe, Qs, Qt)
   - **Page 3**: * insert qualitative data (Number of Managers, Technicians and Users – in this order to respect the acronym E-S-T, insert data into Table 2 and table 3
     * execute the qualitative evaluation
     * show the qualitative results (TCV, QF1, QF2)
   - **Page 4**: calculation of *p* value, showing (Qe', Qs', Qt' and Equation of the plan, height, area, volume and the p value)

The link between different steps could be done, as in set-up programs, with buttons at the end of the page:

On the last page an " Finish" button will be added and the "Forward" button will be deleted.

2. the number of sub-characteristics is set to 21; there is no need to ask for this number
3. the total number of Managers, Technicians and Users is variable and not set to 5, as in table 3
4. Set every numeric field to NULL: users can save time if they have not to delete every time the "0" value when inserting a number.
5. Have all numerical controls be done? I.e. verify that in table 1 users can not go over the (100-MQL) percentage for each perspective in the quantitative analysis? Or that MQL is half TCVmax?
6. MQL and TCVmax value should be chosen by users and not set to 25 and 50 at the start of the program.
7. Any printing option is offered to the user: it could be useful the possibility in Page 4 to print the results of the calculation, as well as a graphical representation of the *p* value (have you already imagined something about this point?).
8. Since these data should be recorded in a web database, you could give user the possibility to re-start the procedure with data from a new project and viewing-editing projects already inserted in the database, giving to each project an ID-code, as in QEST filling forms

Waiting to receive your feedback on my feedback,

À la prochaine,

Luigi

APPENDICE D

COMPARAISON ENTRE LA MÉTHODE QEST ET LA MÉTHODE TQM ET LES COMMENTAIRES DU PROFESSEUR LUIGI BUGLIONE SUR CE SUJET

Méthodes QEST et TQE

Dans ce qui suit une comparaison des deux méthodes QEST et TQE est présentée, et comment peut-on bénéficier de la méthode TQE pour implanter le model QEST.

Introduction

La méthode QEST permet d'exprimer la performance d'un logiciel à partir de mesures spécifiques et non prédéterminées par le modèle même tout en tenant compte des différents points de vue des utilisateurs, des gestionnaires et des techniciens. QEST donne une idée globale sur la qualité d'un logiciel. Ce modèle est basé sur les étapes du cycle PMAI.
La méthode Total Quality Evaluation (TQE) est utilisée pour évaluer des systèmes tels que les systèmes de vidéoconférence basés sur des réseaux ATM. Cette méthode permet de choisir entre l'utilisation d'une conférence face à face traditionnelle ou d'une vidéoconférence. Ce choix est déterminé selon la comparaison des coûts pour les deux types de conférences tout en regardant le facteur qualité de la communication. L'évaluation de la qualité est basée sur MOS (Mean Opinion Score) des éléments de la qualité.

1. Concept de la méthode TQE

La méthode Total Quality Evaluation est déterminée comme suit :
Tout d'abord on définit les différents éléments de service, puis les différents éléments de qualité de chaque service. Ensuite, on détermine l'index de qualité totale qui représente une vue globale de la qualité de tout le système c'est à dire la combinaison des index de différents types, dans le cas des conférences il y a deux groupes : l'index des éléments de qualité concernent la communication et ceux concernent le coût. Ainsi l'index de qualité totale est obtenu en multipliant l'indice de facteur de qualité de la communication qui exprime la dégradation de la densité de communication, et l'index représentant la réduction des coûts, dans l'utilisation de la vidéoconférence.

Le coût est mesuré par le ratio coûts d'une conférence face à face par rapport aux coûts de vidéoconférence.

$$Q_r = (C_f / C_r) * Q_m$$

avec Q_r est l'index de qualité totale pour vidéoconférence, C_f le coût d'une conférence face à face, C_r le coût de vidéoconférence) et Q_m le facteur de qualité de communication.

Si Q_r est supérieur à 1 alors on choisit d'utiliser la vidéoconférence au lieu de la conférence face à face.

Les facteurs de qualité de communication sont les éléments de service. Pour chaque élément, on doit assigné un poids. Les poids des éléments différents selon le type de la conférence c'est à dire on peut avoir des conférences de type générales dont l'image vidéo a une grande importance et d'autres dont l'image n'est pas prioritaire.
Pour obtenir les poids, des points d'évaluation d'élément sont adoptés. Pour l'élément le plus important on assigne la plus grande valeur de point d'évaluation pour les éléments, pour ceux d'une importance moindre on donne des valeurs inférieures…

Le poids est par la suite calculé :

W_e = point d'évaluation e / Somme des points d'évaluation des éléments

Le facteur de qualité pour chaque élément est égal au poids multiplier par le facteur de qualité. Ainsi le facteur de qualité de communication est égal à la somme des facteurs de qualité de chaque élément.

$$Q_m = \text{Somme } (Q_e * W_e) \text{ de tous les éléments}$$

2. QEST vs TQE

Les méthodes QEST et TQE permettent de ``juger`` la qualité d'un système à partir d'opinions des participants. Le premier modèle aide à prendre des décisions pour l'amélioration de la qualité d'un logiciel, quant au deuxième il indique le système ayant la meilleure qualité par rapport à un autre. Ces deux méthodes ont certains points en communs et d'autres en différences.

2.1. Similitudes

2.1.1. Décomposition de la qualité et assignation des poids
Pour les deux méthodes, on essaie de décomposer la qualité du système en plusieurs facteurs de qualités (caractéristiques vs éléments de services), auxquels on va assigner des marques (points d'évaluation) et des poids selon les informations

données par les participants. L'évaluation de la qualité est de ce fait déterminée en tenant compte des points de vue des usagers du système.

2.1.2. Valeurs de qualité

Les deux méthodes permettent de donner une évaluation quantitative de la qualité d'un système en fonction des poids des caractéristiques. La valeur finale de la qualité est la somme de qualité des différentes caractéristiques qui est le facteur de qualité QF (en fonction de la somme des valeurs totales des caractéristiques pour QEST et somme des facteurs de qualité pour TQE).

2.1.3. Insuffisance de QF

Dans les deux méthodes, le facteur de qualité seul ne permet pas de prendre des décisions sur le système, il est accompagné d'autres calculs.

2.2. Différences :

TQE est une méthode plus simple vu que les calculs ne sont pas si compliqués comparer aux calculs de la méthode QEST.

2.2.1. Performance et qualité totale

La qualité totale est exprimée en fonction des coûts et de la qualité pour le modèle TQE par contre l'évaluation de la qualité est exprimée par la performance du logiciel qui est l'ensemble des facteurs de qualité des différents domaines dans la méthode QEST.

Performance = Qualité et Productivité

TQE = Qualité et Coût

2.2.2. Collecte des données

- Les facteurs de qualité d'un système sont prédéfinis par le système même, pour l'utilisation de TQE, contrairement à QEST, les sous caractéristiques doivent être déterminées à partir des questionnaires des participants des trois domaines social, technique et gestion, ce qui n'est pas une mince affaire.

- Pour la décomposition de la qualité, la méthode QEST propose d'identifier tous les sous caractéristiques en premier puis de les regrouper en caractéristiques. Par contre TQE effectue les étapes dans l'autre sens, on dégage les différents facteurs de qualité (équivalent aux caractéristiques) et par la suite les éléments de qualités correspondants affectés à chaque facteur.

2.2.3. Priorité

La priorité se résume dans le cas de TQE par l'assignation des points d'évaluation pour chaque élément de service selon le type du système, vu que pour chaque type de système on affecte des priorités différentes pour les mêmes éléments. Pour le cas de la méthode QEST, ceci n'est pas aussi simple. En plus d'assigner des

marques aux sous caractéristiques, un traitement est effectué pour donner les priorités aux caractéristiques puis déterminer les poids des priorités.

Conclusion
Je ne vois pas comment peut-on utiliser la collecte de données de la méthode TQE vu que les caractéristiques de qualité sont prédéfinis et qu'on a juste à assigner des priorités et donner la qualité selon les points de vue des participants.

Date: 03-12-99
From: Luigi Buglione
To: Myriam Ben Ezzeddine
Cc: Alain Abran
Subject: **Comment on "Méthodes QEST et TQE"**

As correctly said in the introduction, QEST and TQE have different objectives: the first one intends to evaluate the performance level of a software, trying to put in evidence improvement processes through the analysis of related product metrics, while the second one intends to evaluate the convenience in using a face-to-face conference in place of a videoconference.

Another difference is given by the different focus: QEST is project-focused, while TQE is oriented to support a business decision.

Again, QEST is a multidimensional model, taking into account three viewpoints that must be integrated in order to generate a performance index, while TQE analyses the economic and technical viewpoints (I'm not so sure it is possible to intend some of the factors in the Communication Quality as "Social" factors and furthermore, this taxonomy is not very clear in the text. The only point in the procedure where Users are involved is in the filling of the questionnaires for determining the MOS).

It seems more correct in Section 3 to refer to a direct comparison between QF and Qm and not to the whole TQI (Total Quality Index, introduced in the Japanese paper on page 1710). In the hypothesis, TQI can be compared to the whole *p* performance value in QEST, even if to calculate the rough performance (RP) value in QEST several steps must be followed, while the Conference Cost factor in TQE (Cf/Cr) is defined in a simpler way.

It is normal that implementing QEST is not a "*mince affaire*", being an open model while TQE predefines all elements (the only thing to do are gathering cost data and the MOS from questionnaires).

So, the following table summarises basic characteristics of the models, stressing commonalities and differencies:

Models Characteristics	**QEST**	**TQE**
Main Objective	Measuring the performance level of a software product, but in a more generic way it can be used as a performance measurement system for any kind of artifact. Furthermore, QEST nD extends this possibility to a greater number of concurrent viewpoints (>3).	Determine the most convenient option between a face-to-face conference or a videoconference system
Basic elements in the calculus	Productivity and Quality	Cost and Quality
Model nature	Open	Closed
Multidimensionality	Declared – 3 viewpoints (E, S, T)	Implicit ? (E, T) – In section 3.2 it is just said that approximately 10 persons has filled up the questionnaire, without specifying their nature and status.
Quality - Characteristics	QF - Those of ISO/IEC 9126:1991	Qm - Predefined
Second element – considerations	Productivity – list of *n* possible metrics from the three viewpoints	Cost – two predefined factor to take into account (Cf, Cr)
Weighting system	Flexible – weights are determined by the results of the questionnaire	Fixed – Table 4 in the TQE paper gives a predefined list to apply in the formula (2)

About the basic question (how TQE procedure can help in implementing QEST), it is also my opinion that there is a similarity in the procedures, but the inner difference of objectives and hypothesis make not possible any joint utilisation.

In order to demonstrate this sentence, it is possible to look at the following table, comparing the PMAI Cycle steps and the correspondant actions in TQE. It seems that the only phase in common is the "Measure" one, without any particular difference between the two.

QEST – PMAI Steps	**TQE**
Plan	
1. determination of measurement guidelines	No
2. selection of representative measures for each dimension	Predefined
3. determination of relative importance between productity and quality	Predefined
4. determination of ratio weights	Predefined
5. establishment of acceptability threshold values	1 is the only fixed threshold value in order to take a decision between the application of a face-to-face or videoconference. In QEST, every indicator has its own threshold established by the Metric Working Group (MWG).
Measure	
6. data gathering	OK – collect Cf, Cr, Qt, Qv, Qd
7. application of numerical assignment rules	OK
8. normalisation of ratios	OK
9. calculation of QF, V and p	OK – it is the calculation of Qm and TQI
Analysis	
10. presentation of measurement results	OK
11. analysis of observed results	OK
Improve	
12. process improvement	no

APPENDICE E

RÉPONSE DU PROFESSEUR LUIGI BUGLIONE CONCERNANT LE CALCUL DE FACTEUR DE QUALITÉ ET LES EXCEPTIONS DU MODÈLE QEST

Quality Factor calculation

About the QF calculation, **249** is the maximum value that TCV can assume using the structure of the ISO/IEC 9126:1991 standard. This value is derived from the table in section 4 of the paper entitled "*A Quality Factor for Software*" you can download from the LRGL website. The basic hypothesis is that the six weights (p1, …, p6) are equally spaced and that all the respondant to the questionnaire will select all the 21 quality sub-characteristics with the maximum score possible (3).

So, Functionality, for instance, has 5 sub-characteristics and it is the one with the highest priority, since it has the higher number of sub-characteristics in the ISO/IEC 9126:1991 scheme. In this case, SCV (Sub-Characteristic Value) is equal to 15 given by 3 (number of viewpoints that have chosen that sub-characteristic) by 5 (the number of sub-characteristics chosen for that characteristics). The CV (Characteristic Value) will be given by the SCV by the Priority weight of column 5 and so on for all the other characteristics.

Next step is to express weights with just one of them. The common proxy is p6, since is the minor one. Substituting all weights with p6 (Equation 1), it is possible to determine Equation 7. In order to have a clearer view of all the procedure, you can download all calculation forms at the address:

http://www.geocities.com/lbu_measure/qf/qf.htm

Hoping to have given you some help. Good work!

Best regards,

Luigi Buglione

PS: a later comment on your document dated 14-10-99 entitled "**Calculus du volume et de la performance**" with the basic description of the QEST model. On page 4 you wrote "quelque exceptions sont prises en compte telle que si e'+ t' = s' = 1…." is not correct.

The list of exception points for QEST is defined in the following table:

Values	Geometrical Shapes expressed
[e'=1 or s'=1 or t'=1]	*(portion of the) face of the tetrahedron*
[e'=s'=1 or s'=t'=1 or e'=t'=1]	Segment
[e'=s'=t'=1]	*Point*

APPENDICE F

DÉFINITION DES MOTS TECHNIQUES UTILISÉS DANS QEST SELON LEUR APPARITION DANS LE PROTOTYPE

MQL (Mean Quality Level) : Taux moyen de qualité :
La contribution des poids assignés, par rapport aux informations objectives, dans le calcul de la qualité du système .
Le taux de subjectivité des mesures.
0 <= MQL <= 50.

Ratios :
La liste des différentes mesures et les relations entre ces dernières qui découlent de la décomposition du système selon la méthode GQM.
Les ratios ce sont les réponses fournies pour répondre aux attentes à propos des buts spécifiques de chaque domaine (E, S ,T) d'un projet. Un ratio peut être affecté à plus d'un domaine.

Valeur cible : La valeur idéale qu'on veut atteindre pour chaque ratio du système.

Valeur projet : La valeur atteinte de chaque ratio dans notre projet.

Valeur min. : Valeur minimale que peut avoir le ratio correspondant.

Valeur max. : Valeur maximale que peut avoir le ratio correspondant.

Poids :
La contribution, relative à chaque ratio, dans l'évaluation de la performance du projet.
L'importance de chaque ratio pour chaque domaine : économique, social et technique.
La somme des poids d'un domaine (E, S, T) doit être inférieure ou égale à
1 –(MQL * 0.01) .

Valeur réelle :
La valeur normalisée d'un ratio, c'est-à-dire comprise entre 0 et 1.
C'est la valeur du projet transformée en pourcentage.

Valeur finale :
C'est la valeur d'un ratio par rapport à son domaine, exprimé en poids * valeur réelle.

Marquage :
Attribution d'un chiffre entre 0 et 3 pour les sous-caractéristiques du système selon le point de vue de chaque personne:

3 ➔ excellent
2 ➔ bon
1 ➔ passable
0 ➔ faible / absent.

Caractéristique :
Selon ISO les mesures sont groupées en six catégories, chacune d'elle définit un but précis. Une caractéristique est une catégorie.

Fonctionnalité (Functionality)
La capacité d'un logiciel pour fournir les fonctions qui répondent aux besoins indiqués quand le logiciel est utilisé dans des conditions bien spécifiques.

Fiabilité (Reliability)
La capacité d'un système de maintenir son niveau de performance quand il est utilisé dans des condition spécifiques.

Facilité d'utilisation (Usability)
La capacité d'un système à être compris, appris, utilisé et être aimé par l'utilisateur, quand il est appliqué dans des conditions spécifiques.

Rendement (Efficiency)
La capacité d'un système de fournir la performance requise, relative aux ressources utilisées, dans des conditions indiquées.

Maintenabilité (Maintainability)
La capacité d'un système à être modifié. La modification peut inclure les corrections, l'amélioration ou l'adaptation du système aux changements de l'environnement.

Portabilité (Portability)
La capacité du système à être transféré d'un environnement à un autre.

Sous-caractéristique :
Chaque caractéristique se divise en sous-caractéristiques. Dans la norme ISO-9126 il y en a 21.

Aptitude (Suitability)
Concerne la présence et la convenance d'un ensemble de fonctions pour des tâches précises.

Exactitude (Accuracy)
Concerne l'exactitude ou la convenance des résultats ou d'effets.

Interopérabilité (Interoperability)
La capacité du système d'agir avec d'autres systèmes.

Conformité réglementaire (Compliance)
Le respect du système aux normes ou aux conventions ou aux réglementations de droit et d'autres prescriptions.

Sécurité (Security)
La capacité de prévenir les accès non autorisés aux programmes et données.

Maturité (Maturity)
La convenance de la fréquence d'échec.

Tolérance aux fautes (Fault tollerance)
La capacité de maintenir un niveau de performance dans le cas d'erreurs.

Possibilité de récupération (Recoverability)
La capacité de rétablir son niveau de performance et récupérer directement, en un temps et effort déterminés, les données affectées en cas d'échec.

Facilité de compréhension (Understability)
Les efforts que fournissent les utilisateurs pour connaître le concept du système et ses possibilités.

Facilité d'apprentissage (Learnability)
Les efforts que fournissent les utilisateurs pour apprendre l'utilisation du système (opérations de contrôle, entrées, sorties).

Facilité d'exploitation(Operability)
Les efforts que fournissent les utilisateurs pour les opérations et les opérations de contrôle.

Comportement vis-à-vis du temps (Time behavior)
La convenance du temps de réponse et de traitement .

Comportement vis-à-vis des ressources (Resource behavior)
La convenance de la quantité de ressources utilisées et la durée d'une telle utilisation.

Facilité d'analyse (Analysability)
Les efforts fournis pour diagnostiquer les déficiences ou les causes d'une panne, ou identifier des parties à modifier.

Facilité de modification (Changeability)
Les efforts fournis pour modifier, éliminer les erreurs ou pour des changements environnementaux.

Stabilité (Stability)
Les risques d'effets inattendus dues aux modifications .

Facilité de test (Testability)
Les efforts fournis pour valider le système modifié.

Adaptabilité (Adaptability)
L'opportunité de son adaptation à différents environnements sans utiliser d'autres moyens que ceux fournis pour le système considéré.

Facilité à l'installation (Installability)
Les efforts fournis pour l'installation du système.

Conformité (Conformance)
La convenance que le système a pour adhérer aux standards ou conventions reliés à la portabilité.

Interchangeabilité (Replaceability)
L'occasion et l'effort de son utilisation à la place d'un autre système.

Utilisateur :
Acheteur et utilisateur qui est disponible à donner son point de vue sur l'efficacité et l'exactitude du système à mesurer selon ses besoins.
Il donne des informations sur la qualité en cours d'utilisation.

Gestionnaire :
Celui qui s'occupe de la qualité générale et donne son opinion sur le coût et cédule.

Technicien :
Programmeur / Personnel technique…
Il donne des informations sur la qualité technique.

SCV :
Moyenne des marques pour chaque sous-caractéristique.
Détermine la moyenne des points de vues des personnes.
Détermine la valeur des sous-caractéristiques selon les opinions des personnes.

Valeur finale économique / sociale / technique (e / s /t)
La valeur de productivité économique / sociale / technique.
Somme des valeurs finales pour le domaine économique / sociale / technique.

SSV (Sum of subcharacteristics Value):
Somme des valeurs des sous-caractéristiques pour chaque caractéristique.

P(U) / P(G) / P(T) :
Priorité des caractéristiques selon les points de vues des utilisateurs / gestionnaires / techniciens.
La priorité d'une caractéristique par rapport aux autres. Elle est déterminée à partir du nombre des sous-caractéristiques choisies par les personnes.
La valeur des priorités est un chiffre entre 1 et 6.

1 ➔ Plus prioritaire et 6 ➔ moins prioritaire

Note : pour éviter les divisions par zéro, par défaut une valeur de 1000 ou plus montre que la caractéristique n'est pas considérée dans le domaine correspondant.

TCV (Total Characteristics Value) : Valeur totale des caractéristiques :
TCV représente le niveau de qualité estimé du système.
La somme des valeurs des caractéristiques de qualité de ISO selon les opinions des personnes concernant le système.
TCVmax. = MQL * 2

- La somme des valeurs des six caractéristiques de qualité d'ISO dans le cas où toutes les sous-caractéristiques ont des priorités maximales.

0 <= TCVmax. <= 100

QF (Quality Factor) :
La valeur de la qualité du projet par rapport aux différents points de vues dans les trois domaines.
Évaluation de la qualité selon les différentes opinions.

e' / s' / t' :
La valeur de productivité économique / sociale / technique et de qualité c'est-à-dire valeur finale + facteur de qualité.

Le plan (Q_e, Q_s, Q_t) :
QEST utilise un tétraèdre (3 dimensions) pour le calcul de performance, dont la base est formée de trois vecteurs E, S, T (pour chaque domaine) et le sommet (P) présente la performance maximale d'un système.
Le plan (Q_e, Q_s, Q_t) est défini par la position de chaque valeur finale d'un domaine (e ,s, t) sur le vecteur correspondant EP, SP ou TP.
Ce plan détermine une évaluation quantitative du système.

Le plan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$) :
C'est le plan défini par les valeurs e', s', t' dans le tétraèdre.
Ce plan détermine une évaluation qualitative et quantitative du système.

Hauteur :
La Distance entre le sommet du tétraèdre et le hyperplan ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$). Moins cette distance est importante meilleure est la performance.

Aire max. du tétraèdre : Aire de la base du tétraèdre.

Volume ($Q_{e'}$, $Q_{s'}$, $Q_{t'}$, P) : Le volume non comblé pour avoir une performance maximale.

Volume max. du tétraèdre: Volume total de tout le tétraèdre.

Performance :
Le degré que possède le système à mesurer d'accomplir ses fonctions désignées selon les contraintes données.

APPENDICE G

EXEMPLE UTILISANT LE PROTOTYPE QEST

RÉFÉRENCES

Basili, V. R. 1993. Applying the Goal/Question/Metric paradigm in the experience factory. *Institute for advanced computer studies department of computer science*, University of Maryland.

Bevan, N. 1999. Quality in use : Meeting user needs for quality. *The journal of systems and software,* Vol. 49, p. 89-96.

Boregh, J. 1995. Evaluation modules : The link between theory and practice. IEEE. DELTA Danish Electronics, Light & Acoustics, Venlighedsvej 4, DK-2970 Horsholm, Denmark, p. 170-174.

Boregh, J., S. Depanfilis, B. Kitchenham, et A. Pasquini. 1999. A methode for software quality planning, control, and evaluation. *IEEE Software*, Vol. 16, no 2 (March-April), p. 69-77.

Buglione, L. et A. Abran. 1999. A quality factor of software, in *"QALITA99 Proceedings", Third international multidisciplinary congress in quality and reliability*, ENSAM-RUFEREQ, Paris, France, p. 335-344.

Buglione, L. et A. Abran. 1999. Multidimensional software performance measurement models : A tetrahedron-based design. Rapport de recherche No.99-01*. Département d'informatique*, Université du Québec à Montréal, Canada.

Buglione, L. et A. Abran. 1999. Geometrical and statistical foundations of a three-dimensional model of software performance, A*dvances in engineering software* including computing systems in engineering, Vol. 30, no 12, p. 913-919.

Buglione, L. et A. Abran. 1999. Analyse of a three-dimensional model of software performance. Rapport technique. *Département d'informatique*, Université du Québec à Montréal, Canada, (à publier).

Buglione, L. et A. Abran. 1999. Implementation of a three-dimensional model of software performance measurement model, (à publier).

Buglione, L. et A. Abran. 1999. LIME : A three-dimensional measurement model for life cycle project management. *International workshop on software measurement (IWSM'99)*, Lac Supérieur, Québec, Septembre.

Desharnais, J.-M. 1994. Implantation d'un programme de mesures en Génie Logiciel, *CRIM*.

Donaldson, S.E. et S.G. Siegel. 1997. Cultivating successful software development : A practitioner's View, Prentice Hall.

Gonzalez, R.R. 1995. A unified metric of software complexity : Measuring productivity, quality, and value. *The journal of Systems and software*, Vol. 29, no. 1, p. 17-37.

Gray, A. et S.G. MacDonel. GQM++ a full life cycle framework for the development end implementation of software metrics programs. *Software Metrics Research Laboratory, Departement of Information Science*, University of Otago, Dunedin, New Zeland.

Hatfield, M.A. 1995. Managing to the corner cube : Three-dimensional management in a three-dimensional world. *IEEE engineering management review*, Vol. 23, no. 3, p. 63-68.

IEEE. 1990. Standard Glossery of Software Engineering Terminology, IEEE std 610.12.

ISO/IEC. 1991. International standard 9126. Information technology – Software product evaluation – Quality characteristics and guidelines for their use.

ISO/IEC 9126. 1992. Guide for measurement, ratingand assessment 2 : Byer 's guide (GMRA 2). JTC1/SC 7/WG 6.

Jedrzejowicz, P. 1998. Setting-up software reliability measurement program. Présenté à *FESMA 98*. Buisiness Improvement Through Software Measurement , Antwerpen. Belgium, p. 359-367.

Kataoka, N., H. Koizumi, H. Doi, K. Kitagawa, et N. Siratori. 1998. A proposal of a method of Total quality Evaluation in remote conference systems based on ATM networks. *IEICE Transaction of Communication*, Vol.E81-B, no 9.

Lamprecht, W. et C. Weber*. Benefits from user-focused measurement based on GQM*. Siemens, Germany, p. 157-164

Miluk, G. *The role of measurement in software process improvement, Measurement with care and logic*. The Denver Metrics Group. Inc, p. 191-197.

Morisio, M. 1999. Measurement processes are software, too. *The journal of Systems and software,* Vol. 49, p. 17-31.

Punter, T. et G. Lami. 198. Factors of software quality evaluation – Results of two european surveys, présenté au *European Software Control and Metrics conference*, Rome, Italy.

NAVAL Undersea Warfare Center. 1996. Practical software measurement. A guide to objective program inisight, Department of the NAVY, NAVAL Undersea Center Division, Newport, RI, Guide Version 2.1.

Sears, K. et E.M. Aspinwall. 1998. Viewpoint quality model, présenté à *European Software Control and Metrics conference*, Rome, Italy.

Van Latum, F., R. Van Solingin, M. Oivo, B. Hoisl, D. Rombach et G. Ruhe. 1998. Adopting GQM-based measurement in an industrial environment. *IEEE Software*, Vol. 15, no 1, January-February, p. 78-86.

Xenos, M. et D. Christodoulaski. 1997. Measuring perceived software quality. *Information and software technology,* Vol. 39*, Departement of computer engineering and informatics*, university of Patras, Greece, p. 417-424.