CHAPITRE 2

LES MESURES EN GÉNIE LOGICIEL

«L'homme est la mesure de toute chose» (Protagoras, 485 av. J.-C.).

Ce chapitre propose une revue de la littérature sur les mesures en génie logiciel en présentant brièvement la place de la mesure dans le développement du logiciel et les besoins en mesure des différents acteurs dans le cycle de développement. Dans ce chapitre, l'emphase est mise sur la classification de la mesure en se basant sur les classifications proposées par plusieurs chercheurs du domaine. Le chapitre montre, en particulier, la classification des mesures des logiciels suggérées par Fenton. Un aperçu sur les échelles de mesure est présenté à la fin.

2.1 Introduction

La mesure est une partie intégrante de toute activité humaine : sociale, économique, industrielle, académique, environnementale, médicale, etc. Présente dans toutes nos activités quotidiennes pour avoir une vision objective sur la qualité, la mesure est devenue une pierre angulaire du développement industriel, scientifique et social. L'apport de la métrologie (e.g. la science de la mesure) dans le développement du commerce international et la réduction des barrières techniques aux échanges des biens ne sont plus à discuter.

2.2 Place de la mesure en génie logiciel

Il y a trente ans, la mesure des logiciels était une réflexion confinée à quelques chercheurs universitaires et organisations industrielles. Le premier article sur la mesure du logiciel est probablement celui de Rubey et al. (1968). Depuis la fin des années 1970,

la mesure en génie logiciel est recommandée pour planifier et pour contrôler les propriétés du développement des logiciels et des projets logiciels.

Par exemple, Tom DeMarco, un des pionniers de la mesure en génie logiciel, indique «You cannot control what you cannot measure» (T. DeMarco, 1982). Ainsi, dans le domaine du génie logiciel, la mesure de logiciels a été introduite pour contrôler les coûts, améliorer la qualité et livrer le produit logiciel à temps.

Depuis quelques décennies, plusieurs chercheurs tels que Fenton (1997), Pfleeger (1997) et Zuse (1998) voient la mesure de logiciel comme une mise en correspondance entre le système empirique et le système formel (voir figure 2 suivante).

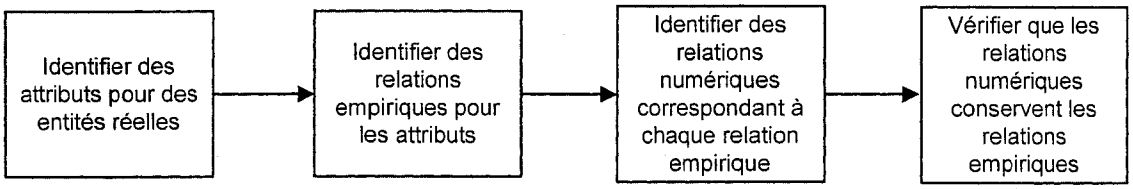

Figure 2 Étapes de la définition des mesures numériques (Fenton et al. 1997)

Pour Fenton, la mesure est une relation empirique obtenue par un consensus lorsqu'elle s'applique au monde réel :

«When the two people being compared are very close in height, we may find a difference of opinion; you may think that Jack is taller than Jill, while we are convinced that Jill is taller than Jack. Our empirical relations permit this difference by requiring only a consensus of opinion about relationships in the real world. A (binary) empirical relation is one for which there is a reasonable consensus about which pairs are in the relation» (N. Fenton et S.L. Pfleeger, 1997).

L'auteur définit le monde empirique de la taille du logiciel comme une composition d'artefacts, de livrables et de documents qui résultent du processus de construction du logiciel et qui proviennent des spécifications, des designs, des codifications et des tests.

Le monde formel est composé des conventions ou des règles de mesure du logiciel à travers ses artefacts, ses livrables et ses documents. C'est à partir de là que l'on peut définir la mesure de la taille du logiciel.

Selon Fenton et Pfleeger (1997) les objectifs principaux de la mesure de logiciels sont :

1. La compréhension : l'évaluation des attributs d'un logiciel permet de comprendre ce qui s'est passé durant le développement et la maintenance de ce logiciel. Cette compréhension permet d'éviter les erreurs commises antérieurement et de profiter des bénéfices de cette expérience dans un projet logiciel futur. Cela pourrait améliorer les processus de développement et de maintenance afin de produire des logiciels de qualité très élevée avec des coûts raisonnables.
2. Le contrôle : la prédiction des attributs d'un projet logiciel permet de contrôler son état d'avancement. Par exemple, les modèles d'estimation des coûts de développement de logiciels permettent de prédire le coût d'un logiciel à une étape assez précoce dans son cycle de vie et, ceci afin de bien gérer la répartition du coût estimé sur les différentes phases de développement.
3. L'amélioration : l'ajout de nouvelles révisions ou de nouveaux types de révision, basées sur les mesures du logiciels, pourrait améliorer le processus de développement du logiciel ou les produits.

Toujours selon Fenton et Pflegeer, une mesure concerne un attribut d'une entité logicielle. On distingue pour chaque entité :

1. des attributs internes : dépendent de l'entité seulement.
2. des attributs externes : dépendent de l'entité dans un environnement ou contexte donné.

Les attributs externes sont plus difficiles à mesurer cependant les attributs externes sont souvent mesurés indirectement. La taille est un exemple d'un attribut interne pour l'entité spécification du produit. La « maintenabilité » est un autre exemple d'un attribut externe pour l'entité spécification du produit.

Mais souvent dans la littérature ces propositions dites de « mesure » du logiciel se limitent à des concepts mathématiques avec représentations numériques sans faire appel

à un système de référence ni à une instrumentation de mesure ou aux protocoles de mesure.

Or, dans la plupart des disciplines de génie, c'est le domaine de la « métrologie » qui est la base du développement et du design des instruments et des processus de mesure. Selon le BIPM (Bureau international des poids et mesures, qui se trouve à Paris, a pour mission d'assurer l'uniformité mondiale des mesures et leur « traçabilité » au Système international d'unités - SI), « La métrologie est la science de la mesure; elle embrasse à la fois les déterminations expérimentales et théoriques à tous les niveaux d'incertitude et dans tous les domaines des sciences et de la technologie » (BIPM, 2005).

Avec l'introduction des concepts de la métrologie tels que des modèles théoriques et des instruments de mesure, la mesure des logiciels pourrait permettre d'obtenir des informations de plus en plus pertinentes nécessaires pour prendre les décisions influant la gestion et la performance des projets.

2.3 Le besoin en mesure dans le génie logiciel

D'après Zuse (1998), le concept de la mesure est évoqué d'une manière ou d'une autre dans la plupart des conférences en génie logiciel. Zuse mentionne que le besoin en mesure est reconnu dans la définition même du génie logiciel dans IEEE Std 610.12 (1990) : «The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software». L'auteur dit que la définition du mot « quantifiable » illustre qu'au moins une organisation professionnelle internationale a reconnu la mesure comme partie intégrale d'une approche bien conçue de logiciel, et non pas une simple adjonction.

La mesure des logiciels est essentielle dans plusieurs aspects comme le contrôle de la qualité et l'estimation du coût des logiciels. Giles (1995) résumait les avantages de la mesure en génie logiciel comme suit :

- permet le contrôle des produits et des processus;
- démontre la productivité, l'efficacité et la qualité du travail;
- attire le respect du client et la crédibilité en gestion;
- identifie où des améliorations peuvent être effectuées.

Quoiqu'il existe beaucoup de propositions de mesures en génie logiciel, il y a encore un grand nombre de questions sur l'utilité de ces propositions. Par exemple, il y a plus de vingt ans que Brown et al. (1981) ont signalé que le manque des principes de base dans le génie logiciel rend la modélisation des relations entre les variables et la spécification des systèmes de mesure difficiles.

De plus, l'absence d'étalons de mesures pour les logiciels rend la tâche des mesureurs de logiciels délicate en termes de précision et de « traçabilité » des résultats de mesure obtenus. Selon le Vocabulaire International de Métrologie - VIM (ISO, 1993) « un étalon est défini comme mesure matérialisée, appareil de mesure, matériau de référence destiné à définir, réaliser, conserver ou reproduire une unité ou une ou plusieurs valeurs d'une grandeur pour servir de référence ». La figure 3 suivante présente un exemple d'étalon.

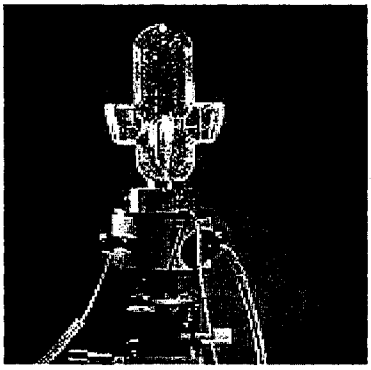

Figure 3 Lampe d'intensité lumineuse

« Cette lampe, d'apparence un peu désuète et pour le moins inhabituelle, matérialise et conserve l'une des sept unités de base du Système International : la candela. À l'heure du laser et du semi-conducteur, cet instrument reste associé à ces nouvelles techniques dans les mesures photométriques où sa fiabilité et sa simplicité en font un outil métrologique de référence » (ISO, 1993).

2.4 La classification des mesures en génie logiciel

Selon Zuse (1998), il y avait déjà en 1998 à peu près 1 500 différentes mesures des logiciels dont plus de 200 pour le langage de programmation orienté objet. La classification la plus utilisée pour ces mesures est celle de Fenton *et al.* (1995). Dans sa classification, Fenton distingue trois classes de mesures (voir figure 4 suivante) :

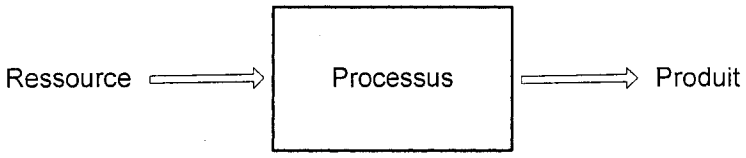

Figure 4 Classification de Fenton (1997)

1. Ressources : ce sont les objets qui contribuent au processus (exemple : expérience du personnel, degré de communication dans l'équipe, prix des progiciels, performance de l'équipement);
2. Processus : ce sont les activités liées au développement logiciel et comprenant normalement un facteur de temps (exemple : effort de construction du logiciel, nombre de changements pour une période donnée, durée de construction du logiciel);
3. Produits : ce sont les objets livrables qui sont le résultat du processus (exemple : taille de l'application, complexité des algorithmes, qualité du produit).

Dans la littérature, quand on parle de la mesure on évoque quelques chercheurs en mesure des logiciels, comme Zuse (1998), qui ont discuté du concept d'« Unité » dans le processus de mesure. ISO (1993) définit l'unité de mesure comme une « grandeur particulière, définie et adoptée par convention, à laquelle on compare les autres grandeurs de même nature pour les exprimer quantitativement par rapport à cette grandeur ». Dans cette définition, on indique qu'une grandeur particulière est définie et adoptée par convention. Ceci implique que dans toute activité de mesure, il y aura des règles à suivre.

Fenton écrit (1995) :

> « In any measurement activity, there are rules to be followed. The rules help us to be consistent in our measurement, as well as providing a basis for interpreting data. Measurement theory tells us the rules, laying the groundwork for developing and reasoning about all kinds of measurement. »

Le même auteur voit que ceci n'est pas unique au seul domaine du génie logiciel :

«This rule-based approach is common in many sciences. »

La plupart des mesures des logiciels manquent aussi d'une échelle d'interprétation comme par exemple la température de congélation de l'eau qui est 0 degré Celsius, et de son ébullition qui est 100 degrés Celsius. En fait, on ne peut pas évoquer la mesure sans le concept d'unité ou d'une échelle d'interprétation ou au moins d'un intervalle de

rapports. Pourtant, c'est ce qui se passe en génie logiciel où la définition de l'unité est rarement abordée.

Enfin, puisque cette recherche traite de la mesure, il est préconisé d'apporter ici un bref rappel sur les échelles de mesure. Les exemples de mesure les plus utilisés sont celui de Stevens (1946). Il a défini cinq types d'échelles : nominale, ordinale, intervalle, ratio et absolue.

L'échelle nominale place les éléments par catégorie, sans ordre. L'échelle ordinale est souvent utile pour ajouter à l'échelle nominale des informations sur l'ordre des catégories. L'échelle par intervalle donne plus d'informations pour les entités que les échelles nominale et ordinale. Cette échelle définit la taille de l'intervalle qui sépare les catégories. La caractéristique clé qui distingue l'échelle ratio des échelles nominale, ordinale et intervalle est l'existence de relations empiriques reflétant les ratios. Par l'échelle ratio, on peut dire que la période de développement d'un projet est deux fois plus longue qu'un autre. Enfin, l'échelle absolue est la plus restrictive. Par exemple, le comptage des employés d'une usine ne se fait que d'une seule façon. L'unicité de la mesure est une différence importante entre les échelles absolue et ratio.

Ces types d'échelles sont hiérarchiquement ordonnés de la moins granulaire à la plus granulaire. La détermination du type d'échelle est très importante car elle met en évidence les propriétés empiriques associées à l'attribut mesuré. Par exemple, on ne peut avoir une échelle de type Ratio que pour les attributs possédant des relations empiriques qui utilisent des opérations de rapport telles que n fois.

Le type d'échelle détermine les opérations statistiques applicables sur les valeurs de l'échelle. Par exemple, la moyenne arithmétique nécessite que l'échelle soit au minimum du type intervalle. Le tableau 1 (A. Abran, 1994) présente un résumé des types d'échelles de mesures.

Tableau I

Échelles de mesure (A. Abran, 1994)

Types d'échelle	Représen-tation	Transformations admissibles	Description des opérations	Exemples
Nominale	(R,=)	F est une correspondance de 1 à 1	Nommer, identifier	Couleurs, formes
Ordinale	(R,>=)	F fonction monotone strictement croissante	Ranger, ordonner	Préférence, dureté
Intervalle	(R,>=,+)	$F(x) = ax + b, a > 0$	Additionner	La date du calendrier, température Celsius
Ratio	(R,>=,+)	$F(x) = ax, a > 0$	Additionner, multiplier, diviser	Masse, distance, température Kelvin
Absolue	(R,>=,+)	$F(x) = x$	Additionner, multiplier, diviser	Compter les entités

En 1998, Zuse présentait plusieurs indicateurs qui exigent le concept d'unité dans le processus de mesure des logiciels, par exemple :

1. Lignes source
2. Personne mois
3. Pages par mois
4. Mois
5. Cas de test par jour
6. Lignes de code
7. Effort
8. Coût de la maintenance
9. Nombre de défauts
10. Nombre de boucles
11. Nombre de modules

Dans son livre A Framework for Software Measurement, Zuse présente une liste illustrative des propositions de mesure des logiciels qui sont :

1. Lignes de code (LOC);
2. Mesures de Halstead pour la prédiction de l'effort de maintenance;
3. Mesures de McCabe pour la complexité cyclomatique;
4. Mesures d'Oviedo pour le calcul des complexités du contrôle et des données;
5. Mesure Défaut–Densité (très utilisée en industrie). Les défauts sont ceux qui apparaissent après une période de livraison, par exemple six mois;
6. Mesures d'Henry et al. pour l'analyse de la complexité totale d'un système informatique;
7. Points de fonction pour la taille fonctionnelle.

2.5 Sommaire

Ce chapitre a présenté un aperçu sur la place de la mesure dans le développement du logiciel. La classification de la mesure est illustrée dans ce chapitre en se basant sur les classifications proposées par plusieurs chercheurs du domaine. À la fin, la classification des mesures des logiciels suggérées par Fenton a été exposée. Cette première revue de littérature a permis d'avoir une idée sur les mesures des logiciels et planifier les lectures à faire afin d'aborder le sujet de cette recherche.

Le prochain chapitre présente les propriétés des logiciels discutées par plusieurs chercheurs du génie logiciel pour les prendre en considération pendant la conception des étalons.