MESURE DE LA TAILLE FONCTIONNELLE DES LOGICIELS TEMPS REEL

Jean-Marc Desharnais[1], Alain Abran[2], Marcela Maya[2], Denis St-Pierre[1]

[1]LMAGL
Laboratoire de Métriques Appliquées en Gestion du Logiciel
7415, rue Beaubien Est, bureau 509
Anjou (Québec), Canada
H1M 3R5

[2]Université du Québec à Montréal
Laboratoire de recherche en gestion des logiciels
C.P. 8888, succursale Centre-ville
Montréal (Québec), Canada
H3C 3P8

Résumé : La capacité de mesurer la taille d'un produit logiciel du point de vue de l'utilisateur, c'est-à-dire, avec une perspective fonctionnelle plutôt que technique, est un pré-requis à l'analyse et à l'évaluation de la productivité. La méthode de mesure des points de fonction est un exemple d'une mesure de taille fonctionnelle. De nos jours, cette méthode est largement utilisée dans le domaine des systèmes d'informatique de gestion où elle est devenue la norme «de facto» de l'industrie. Cependant, la méthode des points de fonction n'a pas obtenu le même degré d'acceptation dans d'autres domaines, par exemple dans le domaine des logiciels en temps réel. Cet article décrit les résultats d'un projet de recherche pour adapter la méthode des points de fonction aux caractéristiques fonctionnelles spécifiques aux logiciels temps réel. La méthode de mesure proposée, appelée Points de Fonction Étendus (PFE), est décrite ici et les résultats des bancs d'essais en industrie sont discutés.

1. Introduction

La méthode des points de fonction a été élaborée pour mesurer la taille fonctionnelle des logiciels du point de vue de leurs utilisateurs. Elle peut être appliquée à n'importe quelle stade d'un cycle de développement et les résultats du processus de mesure sont indépendants des techniques et des outils de développement et d'implantation utilisés. Appliquée au début du cycle de développement, elle peut être utilisée, par exemple, pour construire des modèles d'estimation de l'effort requis pour réaliser un projet informatique. Appliquée à la fin du cycle de développement, elle peut servir à bâtir des modèles de productivité ainsi que pour comparer la productivité d'un projet informatique à un autre.

La méthode des points de fonction est largement utilisée dans le domaine des systèmes d'informatique de gestion où elle est devenue la norme «de facto». Toutefois, elle n'a pas obtenu la même acceptation dans d'autres domaines d'application et une revue de la littérature sur son utilisation pour mesurer des logiciels temps réel montre qu'elle est très peu utilisée dans ce domaine spécifique d'application (Maya *et al.*, 1996). Plusieurs chercheurs soutiennent d'ailleurs que cette méthode n'est pas adéquate pour mesurer des logiciels qui effectuent un nombre très élevé de calculs internes et/ou de fonctions de contrôle, tels que les logiciels temps réel (Jones, 1991; Whitmire, 1992; Galea, 1995). Selon ces auteurs, les résultats de cette méthode de mesure ne reflètent pas de façon adéquate les perceptions des utilisateurs de la taille fonctionnelle de ce type de logiciel.

Cet article présente les résultats d'un projet de recherche réalisé conjointement par le Laboratoire de recherche en gestion des logiciels de l'Université du Québec à Montréal et par un de ses partenaires industriels, le Laboratoire de métriques appliquées en gestion du logiciel (LMAGL).

Le but du projet consistait à adapter cette méthode de mesure aux caractéristiques particulières des logiciels temps réel.

L'adaptation proposée, nommée *Points de Fonction Étendus (PFE)*, introduit de nouveaux types de fonction à mesurer. Ces nouveaux types ont été identifiés dans le but de prendre en considération deux caractéristiques importantes des logiciels en temps réel : (1) un grand nombre de données transitoires ou temporaires, et (2) un nombre élevé de traitements internes, c'est-à-dire, un nombre élevé de processus de traitement composés de plusieurs sous-processus. Ces deux caractéristiques étaient mal mesurées par la méthode traditionnelle des points de fonction

Il est important de souligner que la structure des PFE a été conçue afin de rencontrer des objectifs de qualité en tant que méthode de mesure, tels que :

- *Pertinence* : La méthode de mesure doit être perçue par les praticiens comme étant adéquate pour mesurer la taille fonctionnelle des logiciels propres à leur domaine.
- *Instrumentation* : L'instrumentation est un facteur essentiel pour pouvoir atteindre une des principales caractéristiques de qualité d'une bonne méthode de mesure, soit la répétitivité. Ceci signifie que différentes personnes, dans différents contextes et à différents moments dans le temps et en suivant les mêmes procédures de mesure, doivent obtenir des résultats qui sont semblables, qui ont été obtenus avec un minimum d'interprétation de leur part et qui peuvent être vérifiés. Pour la méthode des points de fonction, la base de l'instrumentation est le guide détaillé publié par l'International Function Point Users Group (IFPUG). Pour les PFE, un guide similaire a également été préparé et placé dans le domaine public[1].
- *Caractère pratique et applicable* : La méthode de mesure doit être pratique et applicable. Elle doit être basée sur les pratiques courantes d'ingénierie du logiciel en vigueur dans l'industrie.
- *Caractère public* : La méthode de mesure doit être du domaine public N'importe quelle personne ou organisation désirant l'utiliser doit pouvoir avoir accès aux concepts, aux définitions, aux procédures et aux règles de calcul.
- *Normalisation* : un comité internationale de normalisation doit voir à faire évoluer la méthode de mesure de façon ordonnée. L'IFPUG assume ce rôle dans le cas de la technique des points de fonction. Un tel comité a également été mis sur pied pour assurer l'évolution des PFE, la structure de ce comité ayant été calquée sur celle de l'IFPUG.

Afin de s'assurer que la méthode PFE rencontrait les objectifs fixés, des bancs d'essais ont été effectués dans quatre organisations différentes.

Dans cet article, la section 2 décrit brièvement la méthode des points de fonction pour situer les lecteurs qui ne sont pas familiers avec cette méthode de mesure. La section 3 présente un résumé des tentatives antérieures pour essayer de corriger cette méthode, puis introduit les concepts propres à la nouvelle méthode de mesure proposée. La section 4 présente les résultats des bancs d'essais réalisés en industrie. Finalement, dans la section 5, quelques observations et suggestions de recherche sont présentées.

2. La méthode des points de fonction

La méthode des points de fonction, mise au point par Allan Albrecht d'IBM, a été publiée pour la première fois en 1979 à la conférence IBM (SHARE Guide). Une autre version, plus élaborée, a été en publiée 1983 dans une revue arbitrée. À peu près à la même époque, un regroupement d'utilisateurs de cette méthode (International Function Point Users Group – IFPUG) a été créé dans le but de clarifier les règles de mesure, de fixer des normes et d'en favoriser l'utilisation et évolution. De 1987 à 1994, quatre versions successives des règles et des normes ont été publiées.

[1] Voir les sites Web suivants : http://www.lmagl.qc.ca ou http://www.info.uqam.ca/Labo_Recherche/lrgl.html.

Dans cet article, tous les renvois aux définitions et aux règles de la méthode des points de fonction sont fondés sur la publication la plus récente d'IFPUG, soit celle de 1994 (IFPUG, 1994a). Cette version à été traduite en français en 1994 (IFPUG, 1994b) conjointement par l'AFNOR, le FFPUG (groupe d'utilisateurs français des points de fonction) et le CIM (Centre d'intérêt sur les métriques – Canada).

Les principaux concepts de la méthode de mesure des points de fonction sont illustrés au schéma 1. On y trouve la frontière de l'application à mesurer, les deux types de groupes logiques de données (internes – GDI et externes – GDE) ainsi que les trois types de transactions (entrées, sorties et interrogations). Les principales étapes de mesure de même que l'ensemble des définitions spécifiques à chacun des types de fonction à mesurer sont présentées dans l'encart. Une description complète de cette méthode se trouve dans le «Guide de comptage des Points de Fonction» (IFPUG,1994b) et comprend les définitions, les procédures, les règles, les formules de calcul des points ainsi que des exemples.

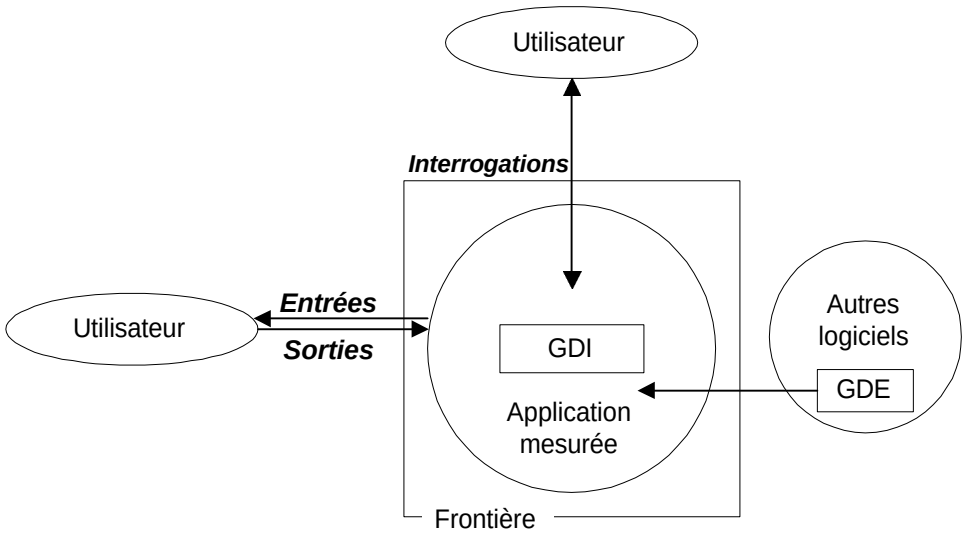

Schéma 1 – Types de fonction de la version IFPUG

MÉTHODE DE MESURE DES POINTS DE FONCTION IFPUG

ÉTAPES DE MESURE ET PRINCIPALES DÉFINITIONS

Les étapes suivantes sont nécessaires pour mesurer la taille fonctionnelle d'une application en utilisant la méthode de mesure des points de fonction (IFPUG, version 4,0):

1. La première étape consiste à déterminer le type de contexte dans lequel le logiciel sera mesuré. Il existe trois types de mesure : celle pour mesurer un projet de développement, celle pour mesurer un projet d'évolution fonctionnelle et celle pour mesurer une application existante.
2. La deuxième étape dans le calcul des points de fonction consiste à identifier la frontière du logiciel à mesurer, c'est-à-dire, la frontière délimitant le logiciel mesuré et les autres logiciels qui interagissent avec le premier. La frontière sert à déterminer les fonctions qui seront incluses dans le processus de mesure. Le schéma 1 illustre la frontière entre le logiciel mesuré et l'utilisateur.
3. L'étape suivante consiste à calculer le nombre de points de fonction brut (PFB), qui reflète le nombre de fonctionnalités spécifiques fournies aux utilisateurs par le logiciel. Le calcul des PFB commence par l'identification des cinq types de fonction du processus de traitement de l'information : deux types relatifs aux données et trois relatifs aux transactions. Les cinq types de fonction (IFPUG) sont :

   - Types relatifs aux données :
     - Groupe logique de Données Internes (GDI) : Groupe de données (ou de paramètres de traitement) identifiables par l'utilisateur comme logiquement liées, et mises à jour à l'intérieur des frontières de l'application.
     - Groupe logique de Données Externes (GDE) : Groupe de données (ou de paramètres de traitement) lu par l'application, identifiable par l'utilisateur comme logiquement lié, mais qui est mis à jour dans les frontières d'une autre application. Ceci signifie qu'un GDE lu pour une application doit être un GDI dans une autre application.
   - Composants relatifs aux transactions :
     - Entrées (ENT) : Une entrée traite des données ou paramètres de traitement qui viennent de l'extérieur des frontières de l'application mesurée. Les données traitées effectuent une mise à jour d'un ou plusieurs GDI. Les paramètres de traitement traités peuvent ou non mettre à jour un GDI.
     - Sorties (SOR) : Une sortie est un processus élémentaire qui produit des données ou des paramètres de contrôle envoyés en dehors des frontières de l'application.
     - Interrogations (INT) : Une interrogation est un processus élémentaire constitué d'une combinaison question/réponse qui a pour résultat l'extraction de données. La partie réponse ne contient aucune donnée calculée ou dérivée (c.-à-d., obtenue à partir d'autres données). Aucun GDI n'est mis à jour durant le traitement.

   Ces cinq types de fonction sont ensuite divisés en trois classes (faible, moyen ou élevé) en utilisant un ensemble de normes. Les PFB sont calculés en utilisant les poids prédéfinis pour chaque type et classe.
4. La quatrième étape du calcul des points de fonction consiste à évaluer les fonctionnalités globales du logiciel. Pour ce faire, l'impact de quatorze caractéristiques générales du logiciel est évalué sur une échelle de 0 à 5 en terme de leur effet global sur la fonctionnalité du logiciel. On obtient ainsi le Facteur d'ajustement (FA). Ce facteur peut modifier les PFB d'un maximum de ±35%.
5. La dernière étape du calcul des points de fonction consiste à calculer le nombre de points de fonction net (PFN) en utilisant les résultats de la quatrième étape (FA) et les résultats du calcul des PFB de la troisième étape.

3. Adaptation des points de fonction aux logiciels temps réel

3.1. Caractéristiques des logiciels temps réel

Pour mesurer la taille fonctionnelle d'un logiciel, la méthode IFPUG tient compte de deux types de composants : les composants relatifs aux transactions et les composants relatifs aux données. Les logiciels temps réel ont cependant des caractéristiques particulières concernant les transactions et les données, qui sont difficiles à capter avec la version actuelle de la méthode de mesure IFPUG.

Caractéristiques concernant les transactions

Dans la méthode IFPUG, un concept spécifique et unique à IFPUG a été défini pour circonscrire les composants relatifs aux transactions: un processus élémentaire. Un processus élémentaire est défini par IFPUG comme la plus petite activité qui soit significative au niveau de l'organisation pour l'utilisateur final. Ce processus élémentaire doit être indépendant et laisser le logiciel dans un état de cohérence fonctionnelle (IPFUG, 1994b). Cependant, le nombre et la nature des étapes ou des sous-processus nécessaires pour compléter un processus élémentaire n'est pas prise en considération. Selon des observations empiriques, les processus élémentaires du même type dans les logiciels d'informatique de gestion ont un nombre relativement constant de tels sous-processus.

Par exemple, un processus qui traite des données et envoie le résultat à l'utilisateur comporte, en général, quatre sous-processus : (1) recevoir la demande de l'utilisateur, (2) lire l'information nécessaire dans un fichier, (3) faire les calculs nécessaires, et (4) envoyer l'information à l'utilisateur. Dans un environnement d'informatique de gestion, le nombre de sous-processus n'ajoute donc, généralement, aucune information supplémentaire importante quant à la taille fonctionnelle de ce processus. Par contre, pour les logiciels en temps réel, le nombre de sous-processus associés à un seul processus peut varier significativement d'un processus à l'autre. Pour illustrer ce point, considérons les deux processus de contrôle suivants :

Exemple 1 - contrôle de température d'un moteur (processus avec quelques sous-processus). Posons que le but principal d'un processus est de démarrer, au besoin, le système de refroidissement d'un moteur. Un capteur envoie la température du moteur au logiciel (sous-processus 1). Cette température est comparée avec un seuil de surchauffe (sous-processus 2). Au besoin, un message de démarrage est envoyé au système de refroidissement (sous-processus 3). Le processus n'est pas dans un état de cohérence fonctionnelle si tous les sous-processus qui y sont attachés ne sont pas complétés. Il correspond donc à la définition d'un «processus élémentaire – IFPUG» tel que présenté plus haut.

D'une part, ce processus de contrôle comporte donc 3 sous-processus (Tableau 1). D'autre part, selon les règles IFPUG, un seul composant transactionnel serait identifié parce qu'il y a un seul *processus élémentaire*.[2]

[2] Son poids mesuré serait de 4 à 7 points selon la classe identifiée.

Processus	Description des sous-processus	# de sous-processus
Contrôle de température	Recevoir la température du moteur	1
	Lire seuil de surchauffe pour comparaison	1
	Envoyer message au système de refroidissement	1
	Total	**3**

Tableau 1 : Exemple 1

Exemple 2 - Diagnostic d'un moteur (processus avec plusieurs sous-processus). Le but principal de ce processus est de déclencher, au besoin, une alarme. Quinze (15) capteurs (tous de type différent), envoient des données au processus de diagnostic (15 sous-processus, un pour chaque type de capteur). Pour chaque capteur, l'ensemble des données externes reçues est comparé avec des seuils lus à partir des groupes de données internes : un groupe de données pour chacun des capteurs (15 autres sous-processus de lecture, un pour chaque capteur). En fonction d'un certain nombre de conditions, une alarme du tableau de bord peut être déclenchée (1 sous-processus). Ce processus de diagnostic comporte donc 31 sous-processus (Tableau 2).

Processus	Description des sous-processus	# de sous-processus
Diagnostic d'un moteur	Recevoir les données externes	15
	Lire les seuils pour comparaison	15
	Déclencher l'alarme	1
	Total	**31**

Tableau 2 - Exemple 2

Selon les règles d'IFPUG, dans ce second exemple un seul composant relatif aux transactions serait identifié, parce qu'il y a un seul *processus élémentaire*[3] : le processus n'est pas dans un état de cohérence fonctionnelle si tous les sous-processus qui y sont attachés ne sont pas complétés.

Or pour des praticiens, une méthode adéquate de mesure de la taille fonctionnelle d'un logiciel en temps réel devrait tenir compte du fait que certains processus ont seulement quelques sous-processus fonctionnels, tandis que d'autres processus ont un grand nombre de sous-processus fonctionnels. Pour ces praticiens les deux processus décrits dans les exemples ci-dessous n'ont pas la même taille fonctionnelle; conséquemment le résultat du processus de mesure de leur taille devrait différer de façon significative.

Caractéristiques concernant les données

Les fichiers logiques des logiciels d'informatique de gestion ont, en général, la structure de données suivante : occurrences multiples d'un enregistrement, chaque enregistrement ayant un ou plusieurs champs. Par exemple, un logiciel de contrôle d'un moteur pourrait comporter un groupe logique de données contenant de l'information sur chaque cylindre du moteur (identification du cylindre, synchronisation d'allumage, de pression, etc.). Il y aura donc un enregistrement pour chaque cylindre. Ce genre de groupes de données, appelé ici *«groupes de*

[3] Son poids mesuré serait de 4 à 7 points selon la classe identifiée.

données à occurrences multiples», a la structure typique d'un groupe de données internes ou externes (GDI ou GDE) dans la technique des points de fonction (IFPUG, 1994c).

Les logiciels en temps réel, par contre, se caractérisent aussi par l'existence d'un grand nombre de variables à une seule occurrence, c'est-à-dire qu'il n'y a qu'une et une seule occurrence de la variable mesurée (par exemple, l'état actuel d'un dispositif). Ces données sont utilisées typiquement pour contrôler, directement ou indirectement, le comportement d'un logiciel ou d'un dispositif mécanique. Le nombre de variables à une seule occurrence dans le logiciel temps réel peut être très important. Une extension des règles de mesure est donc nécessaire pour mesurer d'une façon adéquate ces données.

3.2. Revue de littérature

Quelques modifications à cette méthode ont été proposées pour mieux mesurer les logiciels temps réel, mais aucune de ces propositions ne semble avoir réussie à pénétrer dans la pratique courante au sein de l'industrie du logiciel temps réel.

Une revue de littérature a permis d'identifier six propositions visant à adapter la technique des points de fonction aux logiciels temps réel : Feature Points (Jones, 1991), Mark II (Symons, 1988), Asset-R (Reifer, 1990), 3D Function Points (Whitmire, 1992), Application Features (Mukhopadhyay et Kerke, 1992) et Case Study 4 d'IFPUG (IFPUG-98).

Une analyse de ces propositions permet d'observer que divers types de solutions ont été proposées : ajout de nouveaux types de fonction à être mesurés en plus des cinq déjà existants dans la méthode IFPUG (Feature Points, Asset-R, 3D Function Points); ajustement du résultat final des points de fonction (Asset-R); estimation du résultat final (Application Features); ajustement continu des matrices de complexité utilisées pour attribuer un poids à chaque type de fonction (Mark II) et, bien sûr, la proposition IFPUG-98 d'appliquer les règles telles quelles aux logiciels temps réel.

L'analyse de ces propositions révèlent certaines carences de ces propositions. D'une part, la version IFPUG98 consiste à appliquer les règles telles quelles sans se soucier de vérifier la pertinence des résultats et la méthode Mark II n'identifie pas de caractéristiques spécifiques au logiciel temps réel. D'autre part, les autres propositions laissent place à beaucoup d'interprétation dans leur utilisation car leurs auteurs n'ont pas développé de guide détaillé de règles et procédures de mesure permettant à différentes personnes d'obtenir des résultats semblables avec un minimum d'interprétation.

De plus, les publications sur l'utilisation en industrie de ces différentes propositions pour les logiciels temps réel sont rares, même si les premières adaptations mentionnées ont été proposées il y a près d'une dizaine d'années. En effet, pour la majorite des cas, les données empiriques présentées proviennent presque exclusivement du domaine de l'informatique de gestion et non pas du temps réel.

3.3. Approche proposée : Points de fonction étendus (PFE)

Pour mesurer de façon adéquate les caractéristiques fonctionnelles spécifiques aux logiciels en temps réel, il est nécessaire de considérer les sous-processus attachés aux divers processus de ce logiciel ainsi que le grand nombre de données à une seule occurrence. Pour ce faire, l'approche

préconisée, appelée Points de Fonction Étendus (PFE), introduit de nouveaux types de fonction : deux types relatifs aux données de contrôle et quatre types relatifs aux transactions.

Les données de contrôle sont définies comme les données utilisées par l'application pour contrôler directement ou indirectement le comportement d'une application ou d'une machine. Les nouveaux types de fonction sont alors :

- Types relatifs aux données de contrôle :

– Groupe de contrôle mis à jour (GCM) : c'est un groupe de données de contrôle qui est mis à jour par l'application comptée.

– Groupe de contrôle lu (GCL) : c'est un groupe de données de contrôle utilisé, mais pas mis à jour par l'application comptée.

- Types relatifs aux transactions :

– Entrée de contrôle externe (ECE) : c'est un sous-processus unique identifié selon une perspective fonctionnelle. Une ECE accepte des données qui viennent de l'extérieur des frontières de l'application mesurée.

– Sortie de contrôle externe (SCE) : c'est un sous-processus unique identifié selon une perspective fonctionnelle. Une SCE produit des données qui sortent à l'extérieur des frontières de l'application mesurée.

– Lecture de contrôle interne (LCI) : c'est un sous-processus unique identifié selon une perspective fonctionnelle et qui lit des données de contrôle.

– Écriture de contrôle interne (ECI) : c'est un sous-processus unique identifié selon une perspective fonctionnelle et qui crée ou met à jour des données de contrôle.

Ces nouveaux types de fonction sont utilisés uniquement pour mesurer des processus temps réel, appelés ici *processus de contrôle*. Les autres types de processus, appelées ici *processus de gestion*, seront mesurés avec les types définis dans la méthode IFPUG. Ainsi, pour mesurer un logiciel informatique de gestion, uniquement les 5 types de fonction de la méthode IFPUG doivent être identifiés (section 2). Pour mesurer un logiciel exclusivement temps réel, uniquement les six (6) types de fonction introduits par les PFE doivent être identifiés. Le schéma 2 illustre les différents types de fonction des PFE.

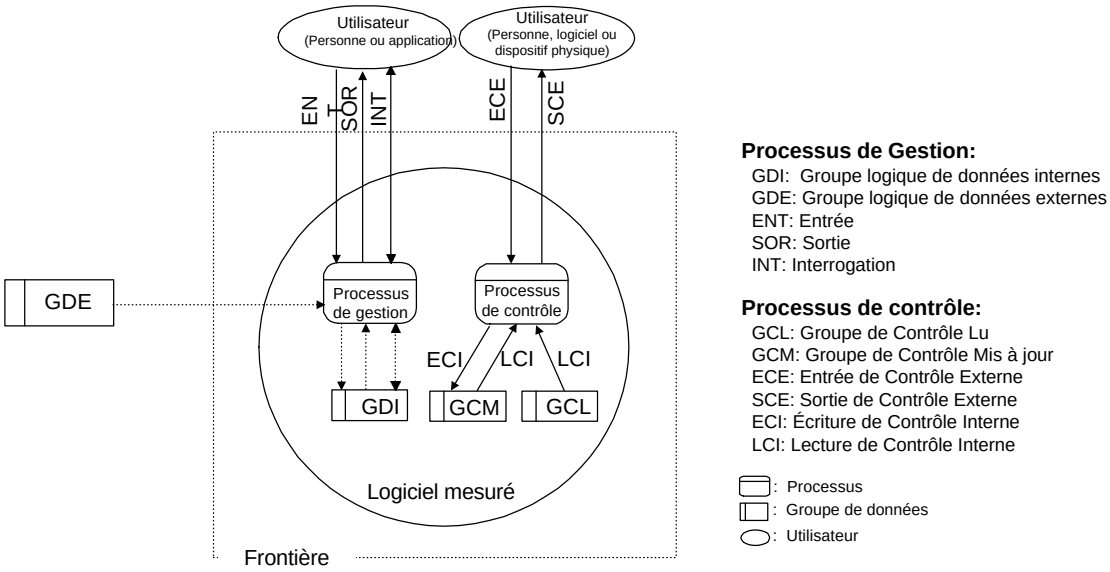

Schéma 2 – Types de fonction de la technique des PFE

Les principales étapes pour l'identification des différents type de fonction relatifs aux transactions des PFE sont les suivantes :

1. Identifier les différents processus de l'application selon une perspective fonctionnelle;
2. Déterminer si le processus est un processus de gestion ou un processus de contrôle en utilisant les définitions suivantes :
   - *Processus de gestion* : processus dont l'objectif est de soutenir l'utilisateur dans la gestion d'information, particulièrement les informations d'affaires et administratives.
   - *Processus de contrôle* : processus qui contrôle directement ou indirectement le comportement d'un logiciel ou d'un dispositif physique.
3. Pour chaque processus identifié, s'il s'agit d'un processus de contrôle, appliquer les règles et les définitions des quatre nouveaux types transactionnels. S'il s'agit d'un processus de gestion, appliquer la définition et les règles des types transactionnels de la méthode IFPUG : Entrées, Sorties et Interrogations (IFPUG, 1994a).

Pour chaque processus de contrôle identifié, il faut ensuite procéder aux étapes suivantes pour identifier les différents sous-processus du processus :

1. Sur la base de l'ordre d'exécution logique de l'ensemble des sous-processus au sein du processus, identifier le premier sous-processus qui entre.
2. Appliquer les règles pertinentes à chaque type de sous-processus (ECE, SCE, ECI ou LCI).
3. Déterminer le poids de chaque sous-processus selon sa nature (ECE, SCE, ECI ou LCI) en utilisant les normes appropriées.
4. Toujours selon l'ordre d'exécution, identifier le prochain sous-processus qui effectue l'entrée, la sortie, la lecture ou l'écriture d'un groupe de données de contrôle. Il peut y avoir plus d'un sous-processus répondant à ces critères (c.-à-d., branchement avec un ou plusieurs chemins logiques). Dans ce cas, tous les chemins logiques doivent être explorés afin de s'assurer qu'il ne reste plus de sous-processus à identifier.

5. Répéter les étapes 2 à 4 jusqu'à ce que tous les sous-processus d'un processus soient identifiés.
6. À la fin du cycle, enlever tous les sous-processus apparaissant plus d'une fois.

Une description complète des concepts des PFE ainsi que des procédures et règles de calcul se trouve dans (St-Pierre, 1997; Desharnais, 1998).

3.4. Comparaison entre les versions IFPUG et PFE

Pour illustrer les principales différences entre les types de fonction définis par IPFUG et les nouveaux types introduits par les PFE, prenons deux d'entre eux qui semblent à première vue similaires : une Entrée (ENT) tels que défini par IFPUG et une Entrée de Contrôle Externe (ECE) tels que défini par les PFE.

Une ECE est très simple et fait uniquement référence à un groupe de données de contrôle reçu de l'extérieur de la frontière du logiciel (Schéma 3a), c'est-à-dire, l'action (ou sous-processus) de recevoir un groupe de données de l'utilisateur. Selon la méthode PFE, les sous-processus de mise à jour et de lecture des fichiers logiques, ainsi que l'envoie d'information à l'utilisateur sont mesurés individuellement. Donc, tout en étant plus simples et par conséquent plus facile à mesurer, la méthode PFE tient compte ainsi d'un niveau de granularité plus fin, le niveau des sous-processus. Ce niveau de granularité plus fin prend toute son importance dans le cas des logiciels en temps réel, puisque leurs processus contiennent un nombre variable de sous-processus, chacun ayant une contribution individuelle à la mesure de leur taille fonctionnelle.

Une ENT de la version IFPUG, par contre, est beaucoup plus complexe puisqu'elle fait référence aux actions combinées de recevoir un groupe de données de l'utilisateur et de mettre à jour un fichier logique, et cela dans certains cas et selon le type de données. Une ENT, telle que définie par IFPUG, peut également lire des fichiers logiques internes et externes et envoyer de l'information à l'utilisateur (Schéma 3b). Il ne suffit donc pas de recevoir des données de l'utilisateur ; il faut également que les autres conditions soient satisfaites simultanément.

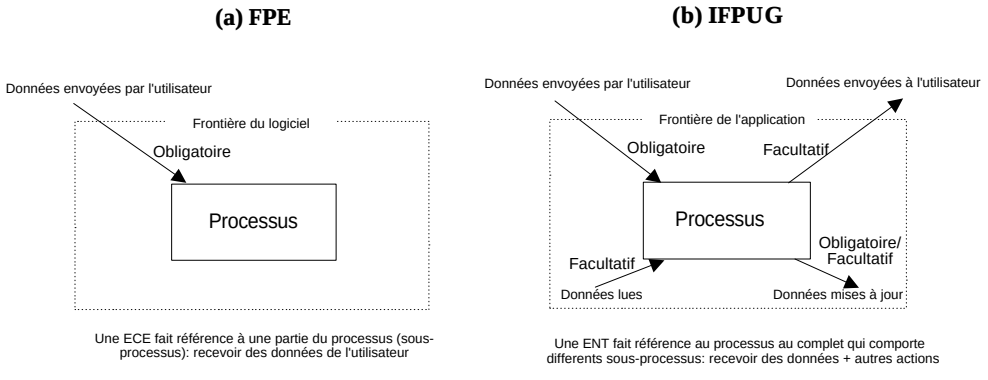

Schéma 3 – Entrées PFE et IFPUG

La version IFPUG des points de fonction examine le niveau du processus et il est requis qu'un ensemble de conditions soit satisfait en même temps; il existe alors un nombre de cas qu'il faut identifier comme des cas candidats, mais qui seront éventuellement rejetés puisqu'ils ne satisferont pas une des multiples conditions pour un des types de transactions selon les règles IFPUG. Il faut vérifier de plus que toutes les conditions de la définition spécifique à IFPUG sur les processus élémentaires soient également satisfaites.

4. Bancs d'essais en industrie

4.1. Stratégie de bancs d'essais

Pour évaluer de façon empirique l'applicabilité des concepts proposés pour la méthode de mesure PFE et pour vérifier si les objectifs de qualité décrits au début de l'article avaient été atteints, une stratégie de bancs d'essais en industrie a été choisie. Deux ensembles de bancs d'essais ont été effectués : un premier ensemble sous la direction de l'équipe de recherche, et un deuxième ensemble sans l'assistance de l'équipe de recherche.

4.2. Bancs d'essais par l'équipe de recherche

Le premier ensemble de bancs d'essais a été réalisé sous la direction de l'équipe de recherche et trois logiciels ont été mesurés en industrie.

4.2.1 Contexte

En raison des contraintes industrielles habituelles, telles que la rareté de disponibilité des experts des organisations, les logiciels choisis pour être mesurés devaient être d'une envergure moyenne ou constituer une partie autonome d'un logiciel de moyen ou grande envergure (±25,000 lignes de code). Les logiciels choisis ont été mesurés en utilisant les PFE ainsi qu'à l'aide de la méthode IFPUG (version 4,0) en vue de comparer les résultats des deux méthodes de mesure.

Au moins trois personnes étaient présente lors de la mesure de ces trois logiciels : un expert de l'organisation connaissant la nature fonctionnelle du logiciel mesuré et deux experts en mesure certifiés dans la méthode IFPUG (Certified Function Point Specialists). Ces derniers avaient participé à la conception et l'élaboration des PFE.

4.2.2 Analyse des résultats des bancs d'essais

Le Tableau 3 présente les résultats des bancs d'essais dirigés par l'équipe de recherche. Dans le Tableau 3, seul les résultats relatifs aux transactions sont présentés. Dans le but de comparer les résultats des PFE avec ceux de la version IFPUG, chacun des logiciels a été mesuré deux fois: d'abord, en utilisant les PFE (résultats dans la partie supérieure du tableau) et ensuite en utilisant la version IFPUG (résultats dans la partie inférieure du tableau). Les résultats obtenus avec les deux méthodes correspondent à la mesure des mêmes ensembles.

Il est à noter que les trois logiciels mesurés sont des logiciels exclusivement en temps réel, c'est-à-dire, qu'ils ne contiennent aucun processus de gestion. Par conséquent, aucun processus de gestion (Entrée, Sortie ou Interrogation) n'a été identifié dans ces bancs d'essai. Le Tableau 3 montre, pour chaque type de fonction, le nombre d'occurrences identifiées ainsi que le nombre de points correspondant.

Méthodes de mesure – Types de fonction	Logiciel A		Logiciel B		Logiciel C	
	Occurrences	**Points**	**Occurrences**	**Points**	**Occurrences**	**Points**
Points de Fonction Étendus - PFE						
- Entrées de Contrôle Externe (ECE)	123	123	10	10	67	69
- Sorties de Contrôle Externe (SCE)	93	97	8	10	136	139
- Lectures de Contrôle Interne (LCI)	395	403	14	18	100	103
- Écritures de Contrôle Interne (ECI)	142	154	8	8	165	168
Total	**753**	**777**	**40**	**46**	**468**	**479**
Points de fonction IFPUG 4,0						
- Entrées (ENT)	40	202	6	21	15	50
- Sorties (SOR)	2	14	2	11	17	73
- Interrogations (INT)	12	40	1	6	0	0
Total	**54**	**256**	**9**	**38**	**32**	**123**

Tableau 3 – Résultats des bancs d'essais dirigés par l'équipe de recherche

Lorsque les résultats obtenus à l'aide des deux méthodes sont comparés, on constate qu'en présence de sous-processus multiples à l'intérieur d'un même processus, la méthode PFE produit une taille fonctionnelle plus grande que la méthode IFPUG (par exemple, 479 points contre 123 pour le logiciel C). Ce constat réflète une amélioration significative apporté de la méthode IFPUG par la méthode PFE. Tel que comme mentionné dans la revue de littérature, la méthode IFPUG est souvent critiquée sur le fait qu'elle génère des mesures fonctionnelles trop petites dans un environnement en temps réel (Jones, 1991; Galea, 1995): c'est-à-dire, les résultats de mesure obtenus avec la méthode IFPUG ne semblent pas être cohérents avec la perception des tailles fonctionnelles relatives des logiciels mesurés.

La nouvelle méthode PFE tient compte, par ailleurs, des sous-processus fonctionnels à l'intérieur d'un processus de contrôle, en identifiant les différents groupes de données reçus, envoyés, lus et écrits, ce qui devrait produire plus des points que la version IFPUG 1994, comme c'est le cas ici au Tableau 3. En effet, les experts fonctionnels des logiciels mesurés dans le cadre des bancs d'essais étaient d'accord pour dire qu'une méthode de mesure ne tenant pas compte des sous-processus demandés par l'utilisateur, comme la méthode IFPUG, ne peut pas être considérée comme une mesure «adéquate» de la taille fonctionnelle. Par conséquent, les résultats des PFE constituent pour eux une mesure plus appropriée de la taille fonctionnelle de leurs logiciels.

Par ailleurs, il a été observé dans les bancs d'essais que le nombre de sous-processus par processus de contrôle varie beaucoup dans un environnement temps réel; par exemple, il y avait des processus avec seulement 3 sous-processus, alors que d'autres processus comptaient plus de 50 sous-processus.

Le Tableau 3 montre aussi que le nombre d'occurrences de chaque type de fonction est très similaire au nombre de points. Cela peut s'expliquer par le fait que la majorité des transactions sont situés dans le premier intervalle de la norme pour attribuer les points (voir St-Pierre et al., 1997). Ceci signifie également que la majorité des sous-processus manipulent moins de 20 données élémentaires.

4.3. Bancs d'essais indépendants

Un des partenaires industriels du projet a effectué un autre ensemble de bancs d'essais sans l'assistance des chercheurs associés au projet. Les résultats détaillés de ces bancs d'essais indépendants ont ensuite été analysés par les membres de l'équipe de recherche.

Seulement quelques erreurs mineures de mesure ont été identifiées lors d'une étape de vérification avec l'équipe de recherche. Ces erreurs provenaient principalement de quelques imprécisions dans la rédaction de la version préliminaire du guide des règles et procédures de la méthode PFE. Suite à cette vérification, le contenu du guide a été amélioré afin d'éviter que ce type d'erreur se reproduise.

Les commentaires obtenus auprès de ces partenaires industriels suite à une étape de retro-action sur ces bancs d'essais indépendants peuvent se résumer comme suit :

- Les concepts et les procédures présentées dans le guide des PFE sont relativement clairs et faciles à comprendre. Pour ces praticiens, il n'a pas été difficile de mesurer sans l'assistance d'un spécialiste.
- L'application des règles de mesure des PFE a permis d'identifier et de mesurer 79 des 81 processus documentés et tels que décrits dans les spécifications fonctionnelles du logiciel mesuré. Avec les nouvelles règles, seulement deux (2) processus contenant exclusivement des algorithmes internes n'ont pas été quantifiés au cours de l'exercice de mesure. Le taux de couverture de la méthode PFE s'établit donc à 97%, dans ce cas, ce qui constitue pour les besoins de cette organisation, une mesure «adéquate» de taille fonctionnelle.

4.4. Observation sur le processus de mesure

Compte tenu des commentaires reçus au cours des deux ensembles de bancs d'essais, les observations suivantes peuvent être :

Facilité de compréhension

On reproche souvent à la méthode IFPUG est fondée sur l'opinion que les différents concepts inhérents à cette technique ne sont faciles à comprendre et que les règles et procédures sont complexes à appliquer. La méthode IFPUG nécessite souvent la présence d'un expert en points de fonction pour s'assurer la précision et la conformité des résultats.

La nouvelle méthode proposée est plus simple de par ses règles moins complexes (c.-à-d., un arbre de décision plus simple et moins de sous-conditions). Par conséquent, lors des bancs d'essais, les membres du personnel informatique des organisations participantes n'ont pas rencontré de problème majeur pour identifier les fonctions à mesurer. Après l'assimilation de la définition des nouveaux types de fonction par ce personnel, il a été constaté qu'ils étaient en mesure d'appliquer la nouvelle méthode de mesure des PFE avec peu d'assistance de la part des chercheurs après une seule journée de pratique. Selon ces praticiens, ce constat est dû au fait qu'il est beaucoup plus facile d'identifier des sous-processus qui font référence à un seul type d'action, par exemple, recevoir des données que d'effectuer des regroupement de sous-processus qui font référence à potentiellement plus d'une action (certaines obligatoires, et d'autres facultatives, et avec les multiples combinaisons possibles) comme c'est le cas dans la technique IFPUG 1994. De plus, lorsque le logiciel mesuré est purement en temps réel (c.-à-d., sans processus de gestion), le personnel affecté au processus de mesure n'a pas besoin de connaître les règles plus complexes de la méthode IFPUG.

Par ailleurs, le personnel informatique qui a participé aux bancs d'essais a souligné que les définitions, procédures et règles de mesure sont suffisamment claires et détaillées pour permettre

à différentes personnes d'obtenir des résultats semblables. Ils soulignent également que les concepts, les procédures et les règles de mesure proposés sont fondés sur les pratiques courantes de documentation en vigueur dans l'industrie.

Effort pour le processus de mesure

L'effort requis pour mesurer un logiciel, avec la méthode PFE est similaire à celui de la méthode IFPUG : d'une part les sous-processus PFE sont plus faciles à identifier et à mesurer, même si, d'autre part, il y en a un plus grand nombre. De plus, les praticiens du domaine des logiciels en temps réel ont semblé capables de les mesurer avec moins d'assistance de la part des experts en points de fonction, ce qui confirme que l'identification d'un plus grand nombre de sous-processus n'accroît pas nécessairement l'effort du processus de mesure.

Assignation de points

Au cours des bancs d'essais, il a été constaté que le nombre total des points attribués à chaque type de sous-processus est presque le même que le nombre de leur occurrences. Sur cette base, il serait donc plausible de questionner l'utilité des normes quant à l'assignation des points. Ces normes peuvent nécessiter un ajustement, mais pour ce faire, plus de données empiriques sont requises pour vérifier la granularité des intervalles choisis pour l'assignation des points.

5. Conclusion

La proposition d'une extension de la méthode IFPUG 1994 pour la mesure de la taille fonctionnelle des logiciels temps réel constituait un défi. Une telle extension doit tenir compte non seulement des caractéristiques fonctionnelles spécifiques des logiciels en temps réel, mais aussi des caractéristiques de qualité propres à une méthode de mesure ainsi que des pratiques courantes de l'industrie en ce qui concerne la conception et la documentation de ce type de logiciel. La méthode de Points de Fonction Étendus (PFE) est le résultats d'un effort de recherche visant à relever ce défi.

De manière à vérifier si la méthode des Points de Fonction Étendus (PFE) rencontrait les objectifs fixés, des bancs d'essais en industrie ont été effectuées. Les résultats de ces bancs d'essais sont positifs : selon des praticiens des logiciels en temps réel, les PFE ont atteint l'objectif de mesurer adéquatement les exigences fonctionnelles de l'utilisateur. De plus, ceux-ci évaluent que la méthode proposée est objective, précise, contrôlable et vérifiable. Ils ont également souligné qu'avec les règles et les procédures de mesure définies, deux personnes différentes effectuant des comptages séparés du même logiciel pouvaient arriver à produire des résultats similaires. Ils ont également mentionné que ces règles et procédures de mesure sont fondées sur les pratiques en vigueur en ce qui concerne la conception des applications et le type de documentation disponible en industrie.

Cette nouvelle méthode de mesure de la taille fonctionnelle des logiciels en temps réel a été présentée publiquement pour la première fois à conférence IFPUG de l'automne 1997, puis à diverses associations nationales de mesure des logiciels au cours des mois suivants. Plusieurs rapports publics portant sur cette méthode sont maintenant disponibles en français, anglais et japonais. On trouve ces rapports sur les sites Web suivants : http://www.info.uqam.ca/Labo_Recherche/lrgl.html ou http://www.lmagl.qc.ca.

Depuis mai 1998, cette nouvelle méthode de mesure est reconnue comme une norme de collecte de données par l'International Software Benchmarking Standards Group (ISBSG), suite à un vote positif et unanime des associations nationales de mesure qui en constituent les membres.

Les concepts, procédures et règles de calculs dans le guide des PFE sont relativement clairs et faciles à comprendre. Ils sont fondés sur les pratiques industrielles quant à la documentation des logiciels temps réel. L'effort requis pour calculer les PFE est similaire à celui requis par la technique des points de fonction, malgré qu'un nombre plus élevé de sous-processus doit être mesuré.

Remerciements

Nous tenons à remercier les entreprises suivantes pour le financement du projet de recherche, pour l'accès aux données industrielles et pour les précieux commentaires reçus de la part des praticiens dans le domaine des logiciels en temps réel : Nortel, Bell Canada, Hydro Québec et JECS System Research (Japon).

Références bibliographiques

Albrecht, A.J. (1979), *Measuring Application Development Productivity*. IBM Applications Development Symposium, Monterey, CA.

Albretch, A.J. and Gaffney, J.E. (1983), *Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation,* IEEE Transactions on Software Engineering **SE-9**(6) : 639-648.

Desharnais, J.-M., St-Pierre, D., Maya, M. et Abran, A. (1997), *Points de fonction étendus: Guide de comptage - Procédures et règles de calcul*, Laboratoire de recherche en gestion des logiciels, Université du Québec à Montréal (UQAM), novembre 1998. http://www.info.uqam.ca/Labo_Recherche/Lrgl/publi/268.pdf

Galea, S. (1995), *The Boeing Company: 3D Function Point Extensions, V2.0, Release 1.0,* Boeing Information and Support Services, Research and Technology Software Engineering, June, 1995.

IEEE, (1990), *IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries,* IEEE Std 610-1990, The Institute of Electrical and Electronics Engineers, Inc., New York, NY, 1990.

IFPUG (1994a), *Function Point Counting Practices Manual, Release 4.0,* International Function Point Users Group - IFPUG, Westerville, Ohio, 1994.

IFPUG (1994b), *Guide de comptage des Points de Fonction, IFPUG Version 4.0, Traduction française Version 1.0,* International Function Point Users Group - IFPUG, Westerville, Ohio, 1994.

IFPUG (1994c)**,** *Function Point Counting Practices, Case Study 1, Release 1.0,* International Function Point Users Group, Westerville, OH, 1994.

IFPUG (1998), *IFPUG Case Study 4 (Draft),* International Function Point Users Group - IFPUG, Westerville, Ohio, 1998.

Illingworth, V. (1991) (editor), *Dictionary of Computing*, Oxford University Press, 3rd edition, 1991, 510 pages.

Jones, C. (1991), *Applied Software Measurement - Assuring Productivity and Quality*, McGraw-Hill, 1991, 493 pages.

Maya, M., Abran, A. et Bourque, P. (1996), *Mesure de la taille fonctionnelle des logiciels temps réel: Revue critique de la littérature,* Rapport de recherche, Laboratoire de recherche en gestion des logiciels, Université du Québec à Montréal (UQAM), août, 1996, 30 pages.

Mukhopadhyay, T. and Kekre, S. (1992), *Software Effort Models for Early Estimation of Process Control Applications,* IEEE Transactions on Software Engineering, Vol. 18, no. 10, October, 1992, pp. 915-924.

Reifer, D.J. (1990), *Asset-R: A Function Point Sizing Tool for Scientific and Real-Time Systems*, Journal of Systems and Software, Vol. 11, no. 3, March, 1990, pp. 159-171.

St-Pierre, D., Maya, M., Abran, A. and Desharnais, J.-M. (1997), *Full Function Points - Counting Practices Manual, Technical Report 1997-04*, Laboratoire de recherche en gestion des logiciels, Université du Québec à Montréal (UQAM), September, 1997, 49 pages. http://www.info.uqam.ca/Labo_Recherche/Lrgl/publi/treports/LRGL-1997-015.pdf

Symons, C.R. (1998), *Function Point Analysis : Difficulties and Improvements*, IEEE Transactions on Software Engineering, Vol. 4, no. 1, January 1988.

Whitmire, S.A. (1992*), 3D Function Points : Scientific and Real-Time Extensions to Function Points*, Proceedings of the 1992 Pacific Northwest Software Quality Conference, June 1, 1992.