UNIVERSITÉ DU QUÉBEC À MONTRÉAL

APPLICATION DE LA MESURE FONCTIONNELLE COSMIC-FFP:

UNE APPROCHE COGNITIVE

PRÉSENTATION DU PROJET DE RECHERCHE

DIC 9410

PAR

JEAN-MARC DESHARNAIS

VERSION 1.05

8 AVRIL 2002

TABLE DES MATIÈRES

1. Introduction....6
2. Revue de littérature....7
   2.1 Domaine de la mesure....7
      2.1.1 Définition de la mesure....7
      2.1.2 La mesure fonctionnelle....8
      2.1.3 Processus de développement et mesure fonctionnelle....10
      2.1.4 Le processus de mesure et la taille du logiciel....11
      2.1.5 L'application de la méthode de mesure et sa validation....13
      2.1.6 Cheminement du mesureur et problématique de la mesure....16
      2.1.7 Problématique de la mesure fonctionnelle....19
   2.2 Domaine cognitif....20
      2.2.1 Nature et objectifs des systèmes experts....21
      2.2.2 Description de différentes technologies....21
      2.2.3 Problématique de la modélisation de la connaissance....25
      2.2.4 Approches mathématiques des moteurs d'inférence....27
      2.2.5 Problématique de la mesure et approches cognitives....31
      2.2.6 Conclusion....32
3. Démarche du mesureur....33
   3.1 Proposition d'une méthode diagnostique....33
      3.1.1 Connaissance du cycle de vie d'un logiciel....33
      3.1.2 Connaissance de la méthode de mesure COSMIC-FFP....34
   3.2 Modélisation de la connaissance du mesureur....36
      3.2.1 Tâches....37
      3.2.2 Résolution de problèmes....37
      3.2.3 Inférences....38
   3.3 Méthode diagnostique....40
      3.3.1 Les étapes et sous étapes de la méthode....40
4. Justification de l'Approche diagnostique....46
   4.1 Introduction....46
   4.2 Besoins et expertise....46
      4.2.1 Catégories d'utilisateurs....46
      4.2.2 Expertise disponible....47
   4.3 Justification de l'approche diagnostique....48

4.3.1 Qu'est-ce qu'un cas dans le cadre de notre problématique? 48
4.3.2 Est-ce que l'application des règles de mesure mène à la résolution d'un cas problème? 48
4.3.3 Comment peut-on les trouver? 48
4.3.4 Peuvent-ils servir à résoudre d'autres cas problèmes? 48
4.3.5 Information historique suffisante? 49
4.4 Justification des approches quantitatives 49
4.4.1 Est-il possible de formuler des questions quantifiables pour résoudre les cas problème? 49
4.4.2 Est-il possible d'utiliser des règles au niveau des cas problèmes? 49
4.4.3 Est-ce que les cas problèmes peuvent être catégorisés et indexés? 49
4.5 Originalité de l'approche choisie 49
5. Méthodologie de recherche 50
5.1 Validation du détail du design par des experts 50
5.1.1 Buts et étapes de la méthode 50
5.1.2 Activités du coordonnateur 50
5.1.3 Avantages et inconvénients de cette approche 50
5.1.4 La méthode Delphi et notre projet de thèse 51
5.2 Plan d'expérimentation de la méthode 51
5.2.1 Étapes de l'expérimentation 52
5.2.2 Choix de deux groupes de mesureurs 52
5.2.3 Composition des groupes 52
5.2.4 La formation 52
5.2.5 Mesure des logiciels 53
5.2.6 Caractéristiques des logiciels 53
5.2.7 Déroulement de la mesure 53
5.2.8 Analyse des résultats 53
6. Plan de réalisation 54
6.1 Principaux livrables 54
6.2 État des travaux 54
7. Conclusion 55
8. Bibliographie 56
Annexe A 59

Description de Help!CPR 59
1. Description fonctionnelle de Help!CPR (utilisateur) 60
2. Description fonctionnelle de Help!CPR (expert) 61
2.1 Les problèmes 61
2.2 Les questions 61
2.3 Les actions 62
2.4 Les mots 63
Annexe B 64
Exemples de "règles locales" 64

FIGURES

Figure 1 Les trois dimensions du logiciel [DEM82] 9
Figure 2 Modèle en cascade [BOE81] 10
Figure 3 Phases du processus de mesure [ABR98] 11
Figure 4 FUR : origine de la documentation (de haut en bas) [ABR01] 12
Figure 5 FUR : origine de la documentation (de bas en haut) [ABR01] 12
Figure 6 Cheminement du mesureur 16
Figure 7 Qualité de la documentation 17
Figure 8 Cheminement du mesureur et activités de la phase 2 18
Figure 9 Cycle du CBR (adaptation à partir de Watson) 23
Figure 10 Tâches de la méthode COSMIC-FFP 35
Figure 11 Types de connaissances 36
Figure 12 Procédure d'application 37
Figure 13 Ontologie COSMIC-FFP [ABR01a] 38
Figure 14 Étapes et sous étapes de la méthode diagnostique 40
Figure 15 Interface utilisateur de Help!CPR 60
Figure 16 Interface expert 61
Figure 17 Questions 62
Figure 18 Traitement d'une question par l'expert 62
Figure 19 Les mots 63
Figure 20 Les mots (suite) 63

TABLEAUX

Tableau 1 : Analyses de fiabilité de la mesure fonctionnelle [ABR94] 15
Tableau 2 : Différentes technologies et de leurs usages 24
Tableau 3 : Renforcement des croyances (règles) 29
Tableau 4 : Exemple d'application des règles 44
Tableau 5 : Thème 1 selon les faits 44
Tableau 6 : Thème 2 selon les faits 44
Tableau 7 : Selon 3 faits (positifs) 44
Tableau 8 : Selon 3 faits (positif et négatif) 45
Tableau 9 : Selon 1 fait (négatif) 45

1. INTRODUCTION

Cette thèse porte sur l'application de la mesure fonctionnelle COSMIC-FFP [ABR01][1]. Appliquer une mesure fonctionnelle à un logiciel veut dire mettre en correspondance les artefacts du logiciel à mesurer avec les règles de la méthode de mesure. Cette tâche est complexe et demande du mesureur une expertise qu'il acquiert, via des connaissances en génie logiciel et en mesure, et par la pratique de la mesure. Nous faisons l'hypothèse qu'un système à base de connaissances pourrait aider le personnel de la mesure à acquérir et à maintenir cette expertise.

La première partie de notre document est une revue de littérature de la mesure du logiciel, et plus particulièrement de la mesure de la taille fonctionnelle du logiciel et son application. Dans cette même partie, nous avons une deuxième revue de littérature sur les approches cognitives incluant les raisonnements à base de cas et les raisonnements à base de règles. Les autres chapitres traitent respectivement :

- de la modélisation de la démarche du mesureur
- de la justification (contribution) de notre approche et son originalité
- de la méthodologie et des expérimentations
- des livrables de notre thèse et de leur état d'avancement

L'originalité de notre recherche se trouve surtout dans le chapitre 3. La démarche du mesureur n'a jamais été systématisée et encore moins traitée dans le contexte d'une approche cognitive. Nous analyserons les résultats de nos expériences auprès des mesureurs par processus. Ce type d'analyse est quasi inexistant, à date, pour la mesure fonctionnelle des logiciels.

Nous avons aussi deux annexes. L'annexe A décrit sommairement le logiciel Help!CPR qui a inspiré dès le départ notre démarche cognitive. L'annexe B est un extrait d'un document d'information sur l'application des règles COSMIC-FFP.

[1] COSMIC (Common Software Measurement International Consortium). FFP (Full Function Point).

2. REVUE DE LITTÉRATURE

2.1 Domaine de la mesure

2.1.1 Définition de la mesure

2.1.1.1 Définition de l'unité de mesure

ISO 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 » [ISO93].

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 (conventions) à suivre. Fenton écrit ([FEN91], p. 24) :

> "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".

Ceci n'est pas commun au génie logiciel seulement. En effet, Fenton ajoute :

"This rule-based approach is common in many sciences" [FEN91].

Dans cette définition, il est aussi question de comparer des grandeurs de même nature. Pour Fenton, il s'agit d'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" [FEN91].

Le dernier élément de cette définition est que l'unité de mesure s'exprime "quantitativement". Pour ce faire, il faut partir des conventions et appliquer celles-ci à l'objet que l'on veut mesurer. Le mesurage (en anglais measurement) est l'activité qui permet d'exprimer quantitativement l'unité de mesure. Plus formellement, ISO définit mesurage de la façon suivante :

"Un ensemble d'opérations ayant pour but de déterminer une valeur d'une grandeur"[ISO93].

Comment se déroulent ces opérations? Fenton écrit « que la mesure est une mise en correspondance entre le monde empirique et le monde formel » [FEN91].

Le monde empirique de la taille du logiciel selon Fenton est composé d'artefacts, de livrables et de documents qui résultent du processus de construction du produit et qui proviennent des spécifications, des designs, des codifications et des tests.

Le monde formel est composé des conventions ou 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.

2.1.1.2 Les classes de mesure du logiciel

Il existe plusieurs classes de mesure du logiciel.

Zuse en a identifié quelques-unes ([ZUS98], p. 17):

- Le schème de classification de Fenton.
- La classification dérivée des modèles de logiciel (flowgraph, call graph, source code ou LOC)
- Les mesures de logiciel intra et inter modulaires. La mesure intra modulaire mesure les programmes, les graphes et les modules par exemple. La mesure extra modulaire mesure l'ensemble du logiciel.
- Les mesures traditionnelles de complexité du logiciel. Voici quelques exemples : les mesures de taille du logiciel en lignes de code, les mesures de complexité cyclomatiques de McCabe, les mesures sur les flux d'information d'Henry, les mesures de volume d'Halstead.
- La classification des mesures des programmes orientés objet.

Notre intérêt porte plus particulièrement sur la classification de Fenton [FEN91] déjà cité dans notre définition de l'unité de mesure.

2.1.1.3 Schème de classification de Fenton

Le schème de classification de Fenton est intéressant parce qu'il a établi son cadre de référence de mesure du logiciel en rapport avec les buts de la mesure du logiciel. Le cadre de référence est basé sur trois principes : classifier les entités qu'il faut étudier, déterminer les buts pertinents de la mesure, et identifier le niveau de maturité que l'organisation a atteint en utilisant ces mesures.

Il y a trois classes d'entités et d'attributs[2] à mesurer selon Fenton :

- les processus ou collections d'activités liées au logiciel
- les produits ou artefacts, livrables ou documents qui résultent du processus
- les ressources ou entités requises par un processus

Pour les fins de notre projet, nous considérons uniquement les mesures de produits qui résultent du processus. De plus, dans les mesures de produits, nous sommes intéressés uniquement par sa taille[3].

2.1.2 La mesure fonctionnelle

2.1.2.1 Définition de la mesure de la taille du logiciel

En se basant sur les classes d'entités et d'attributs de Fenton, on constate que la mesure de la taille du logiciel est une mesure de produit qui peut provenir de différentes entités du logiciel. Les entités des produits sont : les spécifications, les designs, les codifications et les tests. Le choix des entités à mesurer détermine largement la méthode de mesure.

La mesure des tests est souvent considérée comme une mesure de la taille de la complexité du logiciel. Halstead [HAL77] a introduit ce type de mesure en suggérant le calcul des opérateurs et des opérandes.

La mesure de la codification du logiciel utilisera la méthode de mesure des lignes de code. Boehm [BOE81] a fait la mesure de la taille de ses 63 projets en lignes de code (LOC). Il a

---

[2] Pour Fenton, l'entité est ce qui est à mesurer et l'attribut est une propriété de l'entité.

[3] Ceci se justifie par le fait que la mesure fonctionnelle COSMIC-FFP est une mesure de la taille du produit logiciel.

fourni une brève description de ce qu'est une ligne de code, description reprise par Conte et al. [CON86] d'une façon plus élaborée, 5 ans plus tard. Les mesures de la taille du logiciel en ligne de code, ou via les opérateurs et les opérandes, ont été beaucoup critiquées en raison de leur dépendance à la technique de programmation utilisée.

Les mesures indépendantes des techniques de programmation se basent sur les spécifications et le design. La première mesure indépendante des langages de programmation et, reconnue internationalement, est la méthode des points de fonction d'Allen Albrecht. L'analyse du processus de mesure des points de fonction d'Albrecht a fait l'objet d'une thèse de doctorat par le Dr Alain Abran [ABR94].

La norme ISO 14143 définit la mesure fonctionnelle de la façon suivante:

"A size of the software derived by quantifying the Functional User Requirements" [ISO97].

2.1.2.2 Dimensions du logiciel et méthodes de mesure fonctionnelle

Tom De Marco [DEM82] suggère que tous les logiciels tentent de solutionner des problèmes[4] qui ont trois dimensions:

- les données et l'interface utilisateur;
- les processus internes ou fonction;
- le comportement dynamique ou contrôle.

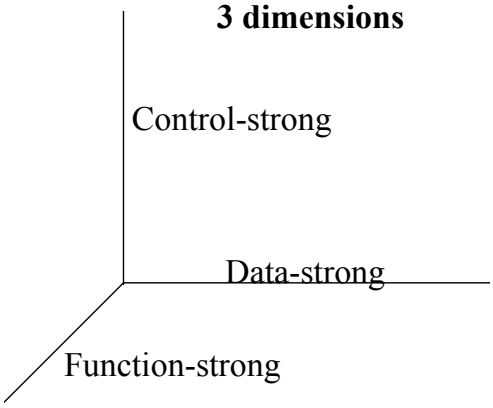

Figure 1 Les trois dimensions du logiciel [DEM82]

Sur cette base, on peut imaginer certains problèmes qui exigent des solutions plus liées aux données et aux interfaces tels les problèmes de comptabilité et d'inventaire (logiciels de gestion). D'autres problèmes sont plus axés sur les processus internes et le comportement dynamique des processus (logiciels en temps réel et d'infrastructure).

La mesure fonctionnelle d'Albrecht [ALB84] a été construite en prenant comme source les logiciels de gestion. Ces logiciels solutionnent des problèmes dont la principale dimension vise les données et l'interface utilisateur. La mesure fonctionnelle d'Albrecht a été reprise et définie de façon plus précise par un comité international[5] [IFPUG99].

---

[4] Pour plusieurs auteurs en génie logiciel, construire un logiciel c'est avant tout résoudre des problèmes. Budgen écrit: "Software is not the only field where design is involved. In the general sense, design can be seen as a form of problem-solving". [BUD94].

[5] IFPUG (International Function Point Users Group)

La méthode de mesure fonctionnelle COSMIC-FFP a été créée dans le but de combler les besoins d'une mesure fonctionnelle de taille pour les logiciels en temps réel[6]. Cette méthode a été présentée à un comité de normes ISO [ISO01]. Ce document devrait faire partie des normes officielles ISO avant la fin de 2002.

D'autres méthodes de mesure fonctionnelle sont utilisées par les entreprises, dont MARK II de Charles Symons [SYM91], utilisée surtout en Angleterre. Cette méthode, tout comme celle d'Albrecht, privilégie les données. Elle est plus explicite en ce qui concerne l'interface utilisateur[7].

2.1.3 Processus de développement et mesure fonctionnelle

La mesure est "une mise en correspondance entre le monde empirique et le monde formel » [FEN91]. Le monde empirique pour notre projet est décrit dans les méthodologies de construction du logiciel.

Barry Boehm [BOE81] a décrit le cycle de vie du logiciel avec une préoccupation liée directement à la mesure des coûts du logiciel. Le modèle décrit par Boehm est le modèle de développement en cascade (Waterfall Model) qui compte 8 phases pour la version simple de la méthode. Il reprend les mêmes phases pour la version incrémentale, mais elles peuvent se répéter plus d'une fois. Le tableau suivant présente les phases sur la base de la version simple.

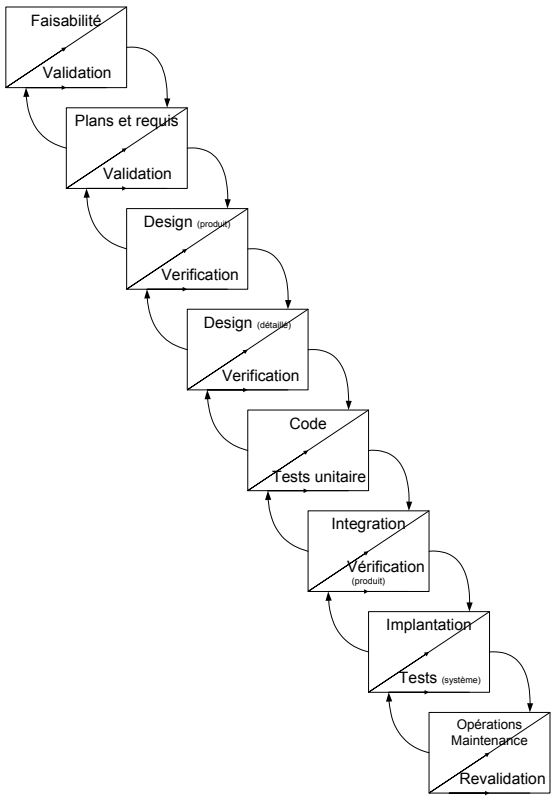

Figure 2 Modèle en cascade [BOE81]

[6] Pour plus d'information sur les critiques de la méthode IFPUG et sa comparaison avec COSMIC-FFP, voir [MOR98] et [DES98]

[7] Pour une critique détaillée de Symons concernant la méthode des points de fonction voir [SYM88]

Ce modèle a été repris autant par les chercheurs universitaires que par les entreprises ayant à construire du logiciel. Plusieurs des modèles modifiés proposés sont de nature commerciales (ex : DMR P+ methodology).

Plus récemment, des chercheurs de l'UQAM, en collaboration avec IEEE et plusieurs entreprises, ont défini un corpus de connaissances en génie logiciel et proposé cinq (5) phases pour le cycle de vie du logiciel (Requis, Design détaillé, Code, Test, Maintenance)[ABR01b]. Ils ont aussi défini cinq (5) phases transversales, c'est-à-dire des phases qui servent à toutes les phases du cycle de vie du logiciel (Gestion de configuration, Assurance qualité, Vérification et qualité, Amélioration des processus, Gestion des processus).

2.1.4 Le processus de mesure et la taille du logiciel

Le processus de mesure a été formalisé dans un article publié en 1998 [ABR98]. Selon Abran et al. [ABR98], il y a 4 phases[8] dans le processus de mesure. Ces phases sont présentées à la figure 3 ci-dessous:

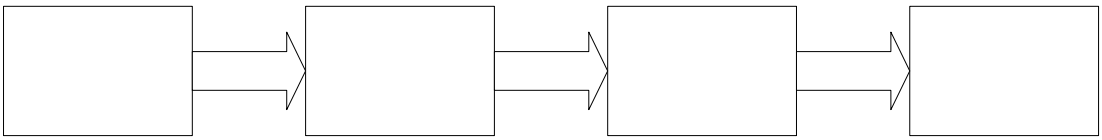

Figure 3 Phases du processus de mesure [ABR98]

La phase 4 constitue l'activité qui représente le plus d'intérêt pour les gestionnaires. C'est la phase finale d'un processus qui permet de construire, par exemple, les modèles de productivité et d'estimation nécessaires à la gestion. La phase 1, celle de la construction de la méthode de mesure, a été appliquée lors de la construction de la méthode de mesure fonctionnelle COSMIC-FFP. Les deux autres phases sont souvent ignorées par les chercheurs sur la mesure. Elles sont cependant essentielles à l'obtention de résultats de mesures cohérentes[9]. Un certain nombre de recherches ont été réalisées pour la phase 3. C'est à cette phase que l'on valide les résultats avant de les analyser dans le but de créer des modèles d'estimation et de productivité. Nous allons reprendre ce point plus loin.

La phase 2 nous intéresse ici plus particulièrement[10]. Cette phase comprend trois activités : la collecte des données, la modélisation du logiciel et l'application des règles d'assignation numérique.

2.1.4.1 La collecte des données

L'activité de collecte des données est spécifique à une organisation, et en général, elle n'est pas systématisée dans la documentation des méthodes de mesure. Par exemple, il y a des textes génériques sur le type de documentation que doit demander le « mesureur », mais dans la pratique, cela dépend de ce qui est disponible pour un projet spécifique dans une organisation, de l'expérience et des connaissances du « mesureur ».

---

[8] Nous remplaçons le mot étape utilisé dans l'article de Abran et al.[ABR98] par le mot phase pour éviter la confusion avec les étapes de la méthode COSMIC-FFP.

[9] Une mesure est cohérente, lorsque deux mesureurs, avec la même documentation (ou la même information) obtiennent les mêmes résultats de mesure.

[10] À notre connaissance il n'y a pas eu de recherches théoriques systématiques relatives à cette phase. Nishiyama et al. [NIS94] ont abordé cette question du point de vue de la qualité de la documentation.

2.1.4.2 La modélisation du logiciel

Le logiciel à mesurer est modélisé en utilisant les règles de la méthode. Plusieurs étapes sont nécessaires.

Ces étapes sont les suivantes dans la méthode COSMIC-FFP:

- l'identification des couches des logiciels
- L'identification de la frontière de chaque logiciel
- l'identification des processus

De plus, les informations peuvent provenir de plusieurs sources et sous divers angles.

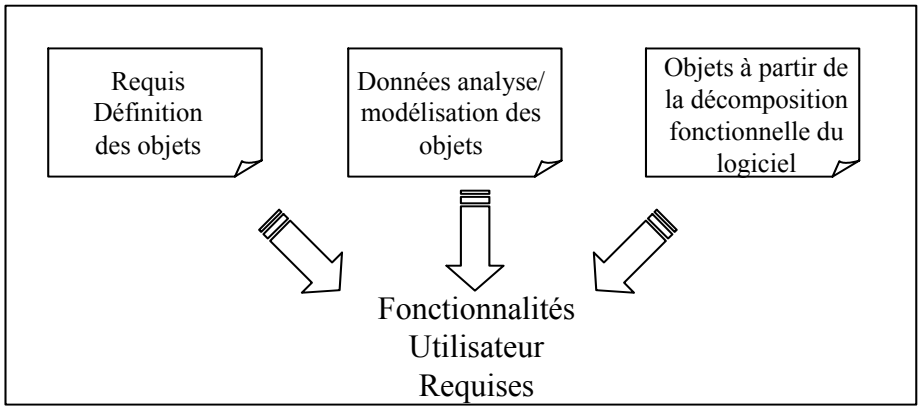

Figure 4 FUR : origine de la documentation (de haut en bas) [ABR01]

En effet, un logiciel peut être vu sous différents angles. Dans la perspective de la méthode de mesure COSMIC-FFP[11], la vision est celle des fonctionnalités livrées aux utilisateurs. Cette dernière est décrite via les Fonctionnalités Utilisateurs Requises (FUR) selon la définition de la norme ISO 14143-1 [ISO97]. En pratique, les FUR existent parfois sous la forme de documents spécifiques (spécifications des besoins requis par exemple), mais souvent les FUR doivent être dérivés des autres objets du génie logiciel. Tels qu'illustrés à la figure 4, les FUR peuvent être dérivés à partir de la documentation avant même l'existence du logiciel (typiquement des artefacts de l'architecture et de la conception). Ainsi, la taille fonctionnelle d'un logiciel peut être mesurée avant son implantation.

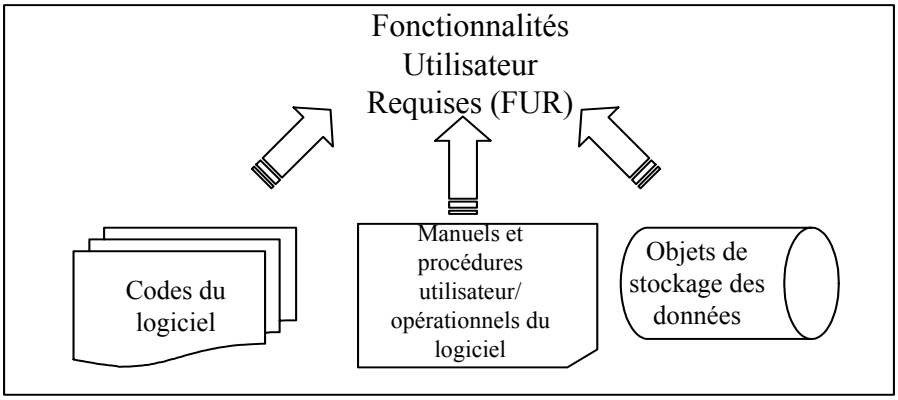

Figure 5 FUR : origine de la documentation (de bas en haut) [ABR01]

[11] Ces schémas sont tirés du Manuel de mesures COSMIC-FFP, version 2.1, octobre 2001, section 2.2, page 15 de la version française.

Dans d'autres circonstances, le logiciel peut être utilisé sans aucun ou peu d'objets provenant de l'architecture ou de la conception, et les FUR peuvent ne pas être documentées. Dans de telles circonstances, il est toujours possible de dériver les FUR du logiciel à partir des objets que l'on trouve à travers l'installation du logiciel dans l'ordinateur, tel qu'illustré à la figure 5.

2.1.4.3 L'application des règles d'assignation numérique.

Cette phase est dépendante de la première étape du processus de mesure et plus particulièrement de la définition des règles d'assignation numérique. Ces dernières sont appliquées par le mesureur à partir de l'identification des sous processus (qui se qualifient comme tels – c'est-à-dire 'mesurables' selon le modèle spécifique de la méthode de mesure) dans le cas de COSMIC-FFP.

Dans COSMIC-FFP, l'étalon de mesure est un mouvement de données élémentaire ou CFUT (**C**OSMIC-**F**FP **U**nité de **T**aille).

2.1.5 L'application de la méthode de mesure et sa validation

Dans cette section, nous abordons l'application de la méthode de mesure sous l'angle de son utilisation dans l'industrie et sous l'angle de la validation des résultats de la mesure.

2.1.5.1 Application de la méthode de mesure: approche industrielle

La méthode de mesure fonctionnelle est appliquée dans plusieurs grandes organisations à travers le monde. Il existe dans un certain nombre de pays des associations faisant la promotion, l'enseignement et l'interprétation des règles de mesures fonctionnelles. Ces associations répondent aussi aux questions des membres concernant l'application des méthodes de mesures fonctionnelles. Elles ont toutes des problèmes similaires de fiabilité des règles de mesures entre mesureurs[12], fiabilité liée à l'application des règles de mesure. Pour cette raison, nous avons consulté les organisations suivantes pour leur demander comment elles procédaient pour résoudre les problèmes d'application des mesures:

- ASMA en Australie (Pam Morris)
- GUFPI en Italie (Roberto Meli)
- DESMA en Allemagne (Robert Hürten, Günter Büren)
- NESMA en Hollande (Jolijn Onvlee)
- JFPUG au Japon (Mr.Kurasige)
- FFPUG (maintenant ASSEMI) en France (Christelle Fritz)
- CIM au Canada (Denis St-Pierre)

Ces associations ont, le plus souvent, un comité de pratiques composé de membres reconnus pour leur expertise dans le domaine de la mesure fonctionnelle de la taille des logiciels. Ces experts se réunissent à intervalles réguliers (mensuel ou trimestriel) pour discuter de l'interprétation et de l'application des règles de mesure. Pour un certain nombre d'associations, les interprétations sont conservées dans un document appelé incorrectement[13] "règles locales" en référence au fait que les règles sont appliquées dans le contexte des technologies locales.

---

[12] La section suivante sur la validation des résultats de la mesure traite de ce problème.

[13] Le terme "règles locales" peut laisser entendre qu'il s'agit de règles nouvelles appliquées localement. Il s'agit en fait d'une interprétation "locale" de règles existantes, c'est-à-dire les règles de COSMIC-FFP appliquées au contexte des technologies locales.

Les premiers résultats de notre enquête auprès des associations locales indiquent qu'aucune de ces associations n'a défini, malgré le fait qu'elles existent depuis plus d'une dizaine d'années pour certaines, une méthode pour faire la mise en correspondance des règles formelles avec les différents environnements de développement (monde empirique). Elles ont cependant pour la plupart des procédures administratives traitant de la nomination des membres du comité, des conditions d'acceptation d'une nouvelle règle d'application et autres règles administratives. La mise en correspondance repose principalement sur l'expertise du mesureur et aussi parfois avec, comme guide, des "règles locales"(annexe B).

2.1.5.2 Validation des résultats de la mesure

La principale conséquence d'une mauvaise application des règles de mesure est une variation des résultats de la mesure entre mesureurs. C'est ce dont nous allons traiter dans cette section.

Il y a trois types de validation des résultats selon Abran et al. [ABR98]. Il s'agit de la validation de la construction d'une mesure qui porte principalement sur l'étape 1, la validation de l'application d'une méthode de mesure et la validation (étape 3) des modèles analytiques (étape 4).

- validation de la construction d'une mesure: dans la littérature en génie logiciel, ce type de validation a principalement été abordé d'un point de vue théorique: une méthode de mesure est valide si elle vérifie le théorème de représentation (?). Cependant, la validation de toutes les sous étapes de l'étape de la construction d'une méthode de mesure n'a pas été complètement traitée.
- Validation des modèles analytiques: ce type de validation porte principalement sur l'étape 4 du processus de mesure. Les auteurs se limitent souvent à des propositions d'un seul type de validation de modèles analytiques, soit les modèles dits de type prédictif.
- Validation de l'application d'une méthode de mesure: le problème de la validation de l'application d'une méthode de mesure a rarement été abordé en génie logiciel, bien qu'il soit d'importance pour les praticiens.

Seule la validation de l'application d'une méthode de mesure est pertinente pour notre étude.

La validation de l'application d'une méthode de mesure consiste, entre autre, à se poser des questions du type [ABR94] :

- Comment savoir si l'application d'une méthode de mesure a été effectuée correctement ?
- Quel degré de confiance peut-on avoir dans le résultat obtenu après l'application d'une méthode de mesure ?

Le processus de validation abordé par Desharnais et Morris [DES93] [DES96] porte à la fois sur les étapes 2 et 3 du processus de mesure. Ce qu'ils appellent la validation a priori porte sur l'étape 2 et la validation a posteriori sur l'étape 3. Les études a priori portent principalement sur la structure de la mesure fonctionnelle. Les études de Desharnais et Morris ont porté sur la méthode de mesure fonctionnelle d'IFPUG. Elles sont donc moins pertinentes pour la méthode COSMIC-FFP puisque la structure de cette dernière est différente.

Pour la validation a priori, quelques études ont été réalisées depuis 1989. Ces études nous intéressent car elles portent sur les résultats de l'application de la mesure fonctionnelle entre mesureurs ayant une information de bonne ou de mauvaise qualité.

Dans sa thèse, Abran [ABR94] cite trois recherches sur la fiabilité des métriques dont la variation entre les mesureurs. Voici un tableau comparatif des trois recherches tiré sa thèse :

Caractéristiques	Rudolph 89 [RUD89]	Low et al. 90 [LOW90]	Kemerer 90]KEM90]
Sujets et niveau d'expérience	Analystes expérimentés	22 analystes et 7 organisations	De 2 a 4 analystes dans plusieurs organisations
Formation	3 jours de formation	20 individus formés par les chercheurs	Manuel d'instruction et guides de mesure
Nombre d'applications mesurées et mesures	9 applications et 373 mesures	2 programmes et 22 mesures chacun	27 applications et 90 mesures
Méthodes statistiques	Jugement moyen Moyenne, Écart-type et différence des moyennes	Jugement moyen Moyenne, Écart-type et différen-ce des moyennes	Tests statistiques H_0 et H_1, ERM
Variation inter-mesureurs	Albrecht 79: 30% IFPUG 88: 10%	30%	IFPUG 90: 10.78% IFPUG E/R: ERM = 17.30% Inter-méthode: ERM = 8.48%

Tableau 1 : Analyses de fiabilité de la mesure fonctionnelle [ABR94]

Ce tableau montre, entre autre, que la variation entre deux mesureurs utilisant la même méthode et les mêmes cas (programme ou logiciel), est de 10% à 30%, même s'ils ont reçu la même formation. Les causes de cette variation ne sont cependant pas analysées, sauf que l'on constate dans l'étude de Rudolph que la méthode utilisée a un impact (Albrecht 79 : 30% et IFPUG 88 : 10%). Nous avons constaté que la méthode Albrecht 79 était beaucoup moins documentée que celle de IFPUG 88.

Plus récemment Nishiyama [NIS99] a étudié les différences entre 2 mesureurs expérimentés. Dans cette étude, la différence relative moyenne entre les résultats des mesures des deux mesureurs est de 5% pour l'ensemble des projets non structurés et de 2.8% pour les projets avec des spécifications relativement bien structurées.

Cette recherche industrielle montre que la qualité des résultats est dépendante de l'expérience des mesureurs. Il faut noter qu'elle porte sur deux mesureurs avec seulement 5 projets, ce qui est un faible échantillon. L'intérêt de la recherche est lié au fait que ce type d'analyse entre mesureurs expérimentés est relativement rare dans la littérature.

L'étude de Nishiyama confirme donc une constatation de Low et Jeffrey à l'effet que "le niveau d'expertise dans la mesure des points de fonction est un facteur important" [LOW90]. Le commentaire de Nishiyama le plus intéressant pour notre recherche est le suivant: " il y a des différences importantes dans l'identification des fonctions" [NIS99]. Les différences auxquelles fait référence Nishiyama sont en rapport avec l'identification des fonctions d'entrées, de sorties et d'interrogations, que l'on retrouve dans la méthode de mesure d'IFPUG [IFPUG99]. Cette différence au niveau de chaque fonction (on dirait au niveau de chaque processus avec COSMIC-FFP) n'a jamais été analysée systématiquement à notre connaissance. Nishiyama semble le seul à l'avoir souligné. C'est à ce niveau (processus) que portera notre analyse pour l'expérimentation.

La qualité de la documentation joue aussi un rôle important selon une autre étude de Nishiyama [NIS94], mais surtout pour l'effort de mesure. Dans ses conclusions, Nishiyama indique qu'il faut jusqu'à 4 fois plus d'effort pour mesurer un logiciel dont la documentation est non structurée, par rapport à un logiciel dont la documentation est bien structurée. L'analyse des efforts en rapport avec la qualité des informations, bien qu'intéressante, ne fait pas partie de notre recherche.

2.1.6 Cheminement du mesureur et problématique de la mesure

Les activités 1 et 2 de la phase 2 (application de la mesure) ont pour but de permettre aux mesureurs d'interpréter de façon cohérente[14] et exacte les règles de la méthode de mesure en tenant compte de la qualité et de la disponibilité de la documentation pour le logiciel à mesurer. À notre avis, ces activités sont typiques de celles d'un expert dans le domaine de la mesure fonctionnelle, et par conséquent, elles peuvent être traitées comme des activités d'expertise. C'est pour cette raison qu'une modélisation appropriée des connaissances [DES01b] sera utile. Pour en arriver à la modélisation, il faut d'abord bien comprendre le cheminement du mesureur.

Du point de vue du mesureur, appliquer les règles de la mesure signifie qu'il doit solutionner le "problème" spécifique de mesure qui se présente à lui. Pour en arriver à la solution, le mesureur doit comprendre les artefacts du logiciel à mesurer, interpréter ce qu'il comprend afin d'identifier ce qui est à mesurer et enfin utiliser les règles de la méthode de mesure et les règles d'assignations numériques.

La figure suivante montre le cheminement du mesureur [DES01b]:

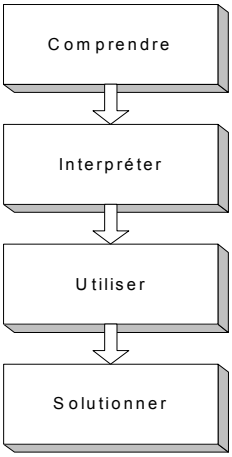

Figure 6 Cheminement du mesureur

En suivant son cheminement, il essaie d'abord:

- De comprendre quel est le problème, c'est-à-dire se faire un modèle mental de la requête dans ce cas précis. Cette requête est formulée par un utilisateur. Elle devrait se trouver dans un document de requis. La requête mentionne qu'il s'agit d'une interrogation.
- D'interpréter ce que signifie la demande de l'utilisateur, à savoir que dans les étapes de la méthodologie COSMIC-FFP, on en est à l'identification d'un processus qui implique aussi l'identification des groupes de données et de l'événement déclencheur.

[14] Notre interprétation est que la cohérence implique la précision, la répétitivité et la reproductibilité au sens ISO [ISO93]

- D'utiliser les règles de modélisation du logiciel. Pour ce cas-ci, il est requis d'utiliser les règles d'identification d'un processus et les règles d'identification d'un événement déclencheur. Il devra aussi utiliser les règles d'identification des groupes de données en lecture et écriture. Le mesureur devra se questionner sur chaque cas problème en rapport avec les règles de la mesure.
- De solutionner le problème, c'est-à-dire identifier qu'il s'agit bien d'un processus et d'un seul processus.

C'est par ce cheminement que le mesureur peut résoudre un "problème" spécifique d'application de la mesure, tout en mettant en relation sa compréhension du logiciel et les règles de mesure en vue de les appliquer. Par exemple, la documentation fournit un modèle entité/relation où le mesureur peut voir 10 entités et 2 relations. Il comprend que ce modèle (cet artefact) peut l'aider à déterminer le nombre de groupes de données dans le logiciel qu'il mesure. Il doit interpréter ce que relations et entités signifient dans le contexte des règles de mesure. Il doit ensuite utiliser les règles[15] relatives aux groupes de données. La solution pourrait être qu'il y a huit groupes de données.

Notre but est de permettre à différents mesureurs d'arriver à la même solution en utilisant la même information (cohérence de la mesure). La qualité de l'information peut être affectée par le manque de documentation ou par la difficulté à interpréter la documentation. Dans le cheminement du mesureur, la qualité de l'information joue un rôle important lors de la compréhension et de l'interprétation.

Dans notre pratique, nous avons identifié que l'application de la méthode de mesure fonctionnelle est difficile pour les raisons suivantes [DES01b]:

- il y a une grande diversité des sources d'information pour obtenir ce qui est nécessaire à la mesure;
- les sources ne sont pas homogènes puisque les façons de faire varient d'une organisation à une autre et même d'un projet à un autre;
- la qualité de la documentation est inégale;
- la documentation est souvent incomplète.

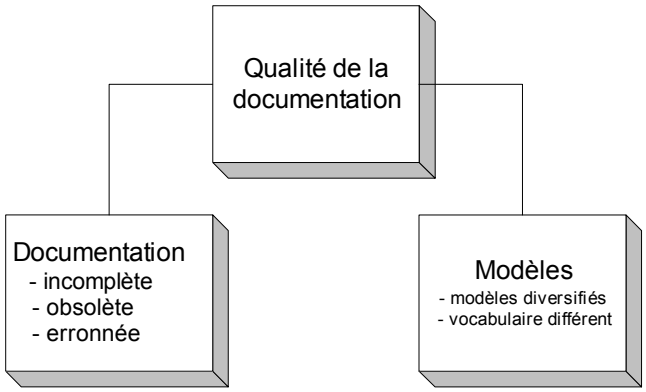

Figure 7 Qualité de la documentation

15 Utiliser les règles signifie ici utiliser tout ce qui peut être utile dans la méthode de mesure en rapport avec le problème à solutionner.

En pratique, la documentation des logiciels (qui s'exprime souvent via des modèles[16]) est souvent incomplète, obsolète et même parfois erronée (voir figure 7). Les problèmes de documentation, lors de la conception et de la mise en place des logiciels, affectent la compréhension des logiciels pour des fins de mesures. De plus, même avec une bonne documentation, il est difficile de comparer les modèles d'une organisation particulière à une autre; chaque organisation ayant tendance à utiliser sa méthodologie, ce qui veut aussi dire que les modes de représentation des modèles, ainsi que le vocabulaire, sont différents.

Dans ces situations, le mesureur ne peut se fier à la documentation et doit interroger les spécialistes qui ont réalisé le logiciel afin de combler ces lacunes. Le mesureur doit donc se fier à la documentation et/ou poser des questions aux spécialistes qui ont réalisé le logiciel pour appliquer les règles de la mesure.

Il est possible de comparer le cheminement du mesureur (figure 6) par rapport aux activités de la phase 2 du processus de mesure. C'est ce qui est présenté à la figure 8.

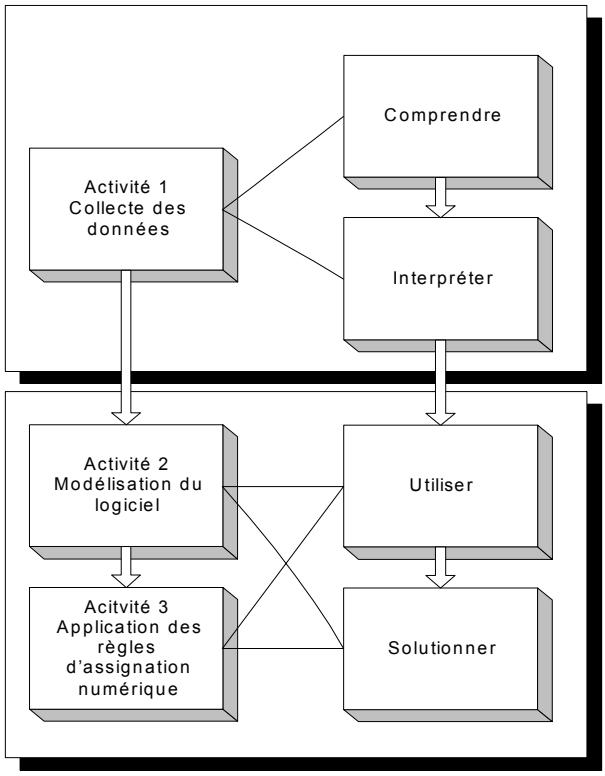

Figure 8 Cheminement du mesureur et activités de la phase 2

Voici un exemple d'information simple qu'un mesureur peut obtenir des requis.

L'utilisateur demande de pouvoir interroger une base de données fournisseur pour obtenir la liste des articles achetés dans le dernier mois pour un fournisseur précis. Le calcul de la taille de ce processus implique (à partir des données disponibles) une connaissance ou une compréhension:

- du mode de déclenchement du processus;

16 "Les systèmes d'information d'ores et déjà modélisés font appel à plusieurs modèles de représentation: modèles de données et de traitement, modèles de connaissances, modèles organisationnels et ergonomiques, modèles de communication"([PRI96], p. 86). Pour les fins de la mesure fonctionnelle, nous utilisons principalement les modèles de données et de traitement.

- des résultats du processus ainsi que du (ou des) groupe(s) de données lues;
- des validations et des résultats possibles des validations.

Avec cette information, qu'il peut obtenir par la documentation ou en interrogant le spécialiste du logiciel à mesurer, le mesureur pourra appliquer les règles de la mesure.

À partir d'autres requis, le mesureur peut aussi identifier un problème dont la démarche pour trouver une solution l'amènera à identifier un autre problème, et ainsi de suite. Ce processus répétitif s'arrête lorsque la solution d'un problème permet l'obtention d'un résultat. À cette fin, le mesureur doit bien connaître les relations qui peuvent s'établir entre les différents problèmes. Il faut prévoir une procédure assez souple pour répondre à ces différentes possibilités.

Ceci peut conduire à la construction d'un arbre de décision. Cependant, en essayant de vérifier toutes les possibilités, ces dernières peuvent augmenter de façon exponentielle. Dans ce contexte, l'utilisation d'heuristiques[17] est nécessaire.

Ainsi, il y a deux ensembles d'habiletés ou de connaissances qui sont nécessaires au mesureur:

- Les habiletés ou les connaissances pour comprendre le logiciel et le problème à résoudre. Le mesureur doit comprendre la documentation, les modèles et autres artefacts du logiciel[18] pour situer le problème et l'associer à des expériences passées.
- Les habiletés ou les connaissances pour appliquer la méthode de mesure au logiciel. Le mesureur qui utilise la méthode COSMIC-FFP doit, par ses connaissances de la méthode, pouvoir appliquer, c'est-à-dire mettre en correspondance les informations obtenues à partir des artefacts du logiciel et des règles de la méthode.

2.1.7 Problématique de la mesure fonctionnelle

Il ressort de notre analyse que:

La mesure est une mise en correspondance entre le monde empirique et le monde formel

La mesure fonctionnelle COSMIC-FFP est une mesure de produit et plus spécifiquement de la taille de logiciel, et de ce fait s'appuie sur les requis et le design du logiciel[19]

L'application de la méthode de mesure est une phase du processus de mesure dépendante de la qualité des données à collecter et de la capacité du mesureur à modéliser le logiciel à mesurer

Le monde empirique de la mesure fonctionnelle se trouve à travers le cycle de vie du logiciel, d'où nos considérations relatives à l'ontologie des méthodes de développement et de maintenance (ici l'ontologie SWEBOK)

Le monde formel de la mesure fonctionnelle se trouve à travers les règles de la mesure fonctionnelle, d'où la nécessité de prendre en considération l'ontologie d'une méthode de mesure fonctionnelle formelle (ici l'ontologie COSMIC-FFP)

---

[17] Le mot heuristique peut référer à un jugement de l'expert (Judgment Heuristics [MIT99] p 423-25) ou une recherche par ordinateur à partir de règles définies par un expert. Par exemple, la définition de Dehn et Schank est dans la deuxième catégorie : "ce sont des règles qui suggèrent une façon de tourner ou de revenir en arrière pour essayer quelque chose de différent" ([DEH82].

[18] Un développeur peut aider le mesureur à mieux comprendre les modèles de logiciel, mais pour les fins de la mesure, il reste que c'est la compréhension du mesureur qui prévaut, selon son interprétation.

[19] Strictement, il est aussi possible d'utiliser les autres entités du produit tels la codification et les tests, mais leur utilisation présente plus de difficultés que l'utilisation des requis et du design.

L'analyse des résultats des mesures fonctionnelles démontre qu'il y a un réel problème de fiabilité des résultats (10% à 30%)[20] de la mesure fonctionnelle. Ce problème serait moindre entre experts (5%) [NIS99]

On ne retrouve pas dans la littérature, connue à ce jour, d'analyses systématiques des variations entre mesureurs au niveau de chaque processus[21].

La problématique est la suivante:

Il n'existe pas de méthodes systématiques et reconnues pour réaliser la mise en correspondance des règles formelles avec les différents environnements de développement (monde empirique).

Les connaissances d'experts en mesure, connaissances acquises après plusieurs années de pratiques, reflètent deux ensembles d'habiletés ou de connaissances nécessaires au mesureur:

- La compréhension des artefacts du logiciel pour situer le problème (ou cas problèmes) à résoudre
- La compréhension des règles de mesure pour les appliquer aux cas problème à résoudre

Notre objectif de recherche est donc d'encapsuler et de structurer la démarche des experts pour permettre l'obtention de résultats cohérents (consistent) et exacts (accuracy) par processus, sans avoir recours à des experts de la mesure. La prochaine section fait une revue sommaire d'un certain nombre d'approches cognitives pouvant aider à résoudre cette problématique, et plus spécialement les raisonnements à base de cas (CBR) et les raisonnements à base de règles.

2.2 Domaine cognitif

Dans la problématique du domaine de la mesure, nous concluons qu'il faut une bonne expertise pour appliquer la méthode de mesure fonctionnelle COSMIC-FFP.

Qu'est-ce qu'un système expert? Quelle est la nature de l'expertise?

Jackson définit de la façon suivante un système expert :

"An expert system is a computer program that represents and reasons with knowledge of some specialist subject with a view to solving problem or giving advice" ([JAC98], p.2)

À partir de cette définition, on peut dire que :

- Le système expert possède un certain nombre de connaissances, que ce soit des algorithmes, des listes d'exemples, des questions et réponses, etc.
- Certaines connaissances appartiennent à un domaine spécifique. Une collection de connaissances provenant de différents domaines est rarement l'objet d'une expertise
- La connaissance doit permettre de résoudre des problèmes autres que des problèmes d'accès à une documentation en ligne (hyperliens).
- Enfin, le système expert peut aider à résoudre des problèmes ou donner des avis.

[20] Du moins lorsque les mesureurs ne sont pas expérimentés.

[21] En théorie on peut aussi réaliser cette analyse au niveau des sous processus. Nous visons dans notre recherche de réaliser au moins une analyse par processus.

2.2.1 Nature et objectifs des systèmes experts

Le terme "système expert" couvre un grand nombre de systèmes. Une classification de ces systèmes a été proposée par Hayes-Roth en 1983 et citée par Jackson [JAC98]. Il s'agit de systèmes :

- d'interprétation de situations décrites à partir d'observations ou à partir de senseurs
- de prédiction de conséquences possibles à partir d'une situation ou d'un événement.
- de diagnostic à partir des symptômes
- de design d'une configuration d'objets qui satisfait certaines contraintes
- de planification d'une séquence d'actions pour réaliser des buts identifiés
- de suivi des comportements d'un système dans le temps, en vue de l'empêcher de dévier de ses buts
- de "déboguage" ou d'identification des fautes de système
- de réparation des fautes de système
- d'instruction de diagnostique et traitement des incompréhensions d'étudiant(e)s pour un domaine de connaissance
- de contrôle de comportement des systèmes en anticipant des problèmes, en planifiant des solutions et en faisant le suivi des actions nécessaires.

2.2.2 Description de différentes technologies

Cette section examine les apports des différentes technologies à cette expertise (quand les utiliser ou ne pas les utiliser) et les différents types de systèmes experts.

2.2.2.1 Présentation des technologies

Pour Watson [WAT97], il est possible d'obtenir une expertise d'un domaine donné à partir de différents types de technologie. Ces technologies sont:

- Les bases de données
- La recherche d'informations textuelle
- Les statistiques
- Les apprentissages automatiques
- Les réseaux neuronaux
- Les systèmes à base de règles
- Les systèmes à base de cas (CBR)

Les bases de données sont très largement utilisées dans l'industrie. Elles requièrent une structuration de l'information d'où il sera possible d'extraire des informations utiles, le plus souvent à partir d'un programme codé pour les besoins des utilisateurs du logiciel. Pour ce faire, il faut prévoir la modélisation des données et des traitements. La modification des données entraîne aussi la modification des traitements et aussi, le plus souvent, l'intervention de spécialistes en analyse et programmation.

La recherche d'informations textuelles est utilisée, entre autre, sur le WEB à l'aide de moteurs de recherche. L'accès automatisé aux sources d'information est essentiel. Pour être efficace, avec un grand volume de données, les informaticiens créent des liens par l'utilisation de mots clefs (ou concepts) à des sources d'information textuelles. Les interrogations sont très flexibles et on peut trouver aussi bien des faits exacts que des faits inexacts. Règle générale, les moteurs d'inférence se limitent aux informations textuelles tout en donnant un poids à la valeur de l'information retrouvée.

La recherche statistique implique, tout comme pour les bases de données, des données relativement bien structurées (elles peuvent aussi être textuelles), mais les traitements reposent sur les fonctions statistiques prévues dans le programme statistique. L'efficacité de la recherche repose sur la bonne compréhension par l'utilisateur des différentes fonctions statistiques.

Les apprentissages automatiques impliquent l'analyse de cas passés pour dériver des règles qui s'appliquent à un ensemble de cas. Ces cas peuvent être appliqués pour résoudre de nouveaux problèmes. L'apprentissage automatique fait une distinction entre le processus d'apprentissage des règles et la résolution de nouveaux problèmes. La justification des réponses se fait à partir des règles utilisées et non à partir des cas précédents (comme dans les CBR).

Les réseaux de neurones sont inspirés de la structure biologique du cerveau humain. Le neurone est une unité de traitement des données (numériques, signaux, voix) reçues à travers des entrées. Un groupe d'entrées est composé d'informations non symboliques qui génèrent une sortie (un réseau élaboré peut avoir de multiples sorties avec différents groupes d'entrées). Il y a plus d'un type d'architecture de neurones, la plus connue est l'architecture en couches. Dans cette structure, chaque neurone est relié aux neurones de la couche suivante. Chaque connexion entre les neurones est affectée d'un poids. Le réseau génère une sortie en propageant les entrées à travers ses couches.

Le raisonnement à base de règles (SI…ALORS RÈGLE) permet entre autre l'abduction, c'est-à-dire que l'on peut partir d'un résultat pour en retrouver les causes. C'est une des plus anciennes techniques pour représenter le domaine de la connaissance dans un système expert. Il permet de traiter des problèmes précis et connus des experts. Les règles sont formalisées par des experts et peuvent servir à résoudre plus d'un type de problème.

Dans le raisonnement à base de règles la connaissance est présentée comme des faits sur le monde avec des règles pour manipuler ces faits. Il y a une complexité supplémentaire du fait qu'à n'importe quel moment, plus d'une règle peut s'appliquer et au moment où chaque règle s'applique, plusieurs autres peuvent aussi être applicables. Il faut donc prévoir une structure de contrôle qui décide l'ordre d'application des règles et leur enchaînement.

Un CBR "permet de résoudre un nouveau problème en se souvenant d'un problème passé similaire et en réutilisant l'information et la connaissance de cette situation"[AAM95].

Il est parfois difficile de réduire à de simples règles la mémoire des choses et les expériences. On peut alors imaginer une bibliothèque de cas passés significatifs pour un problème donné. Cette approche est l'essence même du raisonnement à base de cas. Un autre aspect important du CBR, c'est qu'il n'est pas nécessaire de restructurer la base de connaissance en ajoutant un nouveau cas.

L'utilisation d'un CBR repose sur un certain nombre d'hypothèses selon Kolodner [KOL93]:

- la régularité: les mêmes actions réalisées sous les mêmes conditions devraient donner les mêmes résultats ou des résultats similaires
- la répétition (caractéristiques types): les événements ont tendance à se répéter.
- la cohérence: des changements mineurs dans la réalité requièrent des changements mineurs dans notre raisonnement, ce qui veut dire qu'il y aura des changements mineurs dans la solution

Qu'est-ce qu'un cas? Pour Alterman (89), David (91) et Kolodner (93) un cas est un « morceau » de connaissance "contextualisé" et représentant une expérience.

Un cas typique comprend:

- un problème qui décrit l'état de la situation lorsque le cas arrive
- des questions qui, à partir de la situation, fournissent une solution au problème
- un résultat qui décrit l'état de la situation

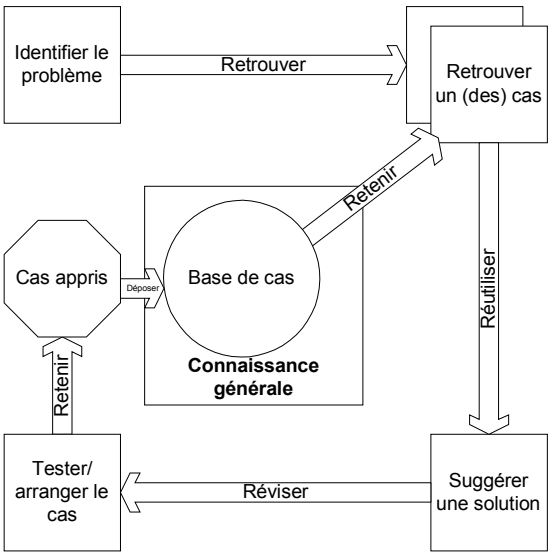

Figure 9 Cycle du CBR (adaptation à partir de Watson)

Voici un tableau résumé, adapté de Watson [WAT97], des différentes technologies et de leurs usages :

Type de technologie	**Quand est-il préférable de les utiliser?**	**Quand est-il préférable de ne pas les utiliser ?**
Bases de données	Données structurées, normalisées, avec des interrogations précises. Les problèmes sont bien modélisés	Problèmes complexes, peu de structures dans les données et interrogations complexes (floue)
Recherche d'informations textuelles	Grand volume de données textuelles	Possibilités d'accès à l'information source. Données structurées et nécessité de savoir le pourquoi[22]
Statistiques	Beaucoup de données, bien comprises, hypothèses bien formulées	Analyse exploratoire des données avec des variables dépendantes et possiblement textuelles.
Apprentissages automatiques	Des règles généralisées sont nécessaires pour une formation de masse et une justification des règles.	Règles non requises, pas plus que leur justification. Note: il n'est pas nécessaire d'inférer de nouvelles règles.
Réseaux neuronaux	Données de nature numériques avec beaucoup de "bruits" ne permettant pas de trouver facilement un modèle de reconnaissance des données ou un processus régulier.	Données textuelles avec justification requise.

22 Des approches du type Explanation Based Learning qui favorisent la déduction plutôt que l'induction peuvent aussi utiliser des données moins structurées. Les interprétations de ce tableau peuvent évoluer dans le temps et doivent être prises à titre indicatif seulement.

Systèmes à base de règles	Problèmes bien compris, stables, dans un créneau très stable et une justification des règles.	Problèmes imprécis pouvant changer constamment
Système à base de cas (CBR)	Structure de données complexes (ou non modélisables) pour résoudre des problèmes peu définis au départ et qui peuvent changer dans le temps. Des réponses peu précises sont possibles. Des cas peuvent être ajoutés en tout temps.	Peu de données disponibles, adaptation très complexe et des réponses précises sont nécessaires.

Tableau 2 : Différentes technologies et de leurs usages

2.2.2.2 Commentaires généraux

La description des problèmes que nous voulons résoudre peut se faire du point de vue de l'accès aux données et de nos objectifs.

Accès aux données:

- Règle générale, nous n'avons pas accès aux textes sous un format électronique structuré, ce qui exclut les approches à base de données et les approches statistiques
- Nous voulons aider le mesureur à effectuer un raisonnement impliquant une manipulation de symboles
- les données évoluent, en ce sens que de nouveaux cas peuvent s'ajouter dans le temps. En raison de la diversité des méthodes de développement dans les organisations et de leur constante évolution, il n'est souhaitable de se limiter aux cas connus, ce qui rend intéressant les CBR ;
- les nouveaux cas problème peuvent avoir des solutions existantes ou similaires. Il est possible qu'une question déjà utilisée pour un problème, puisse l'être pour un autre problème, ce qui rend intéressant les technologies à base de règles;
- il est très souhaitable de justifier les solutions associées à chaque problème. Cette exigence n'est pas absolument nécessaire, mais obtenir la justification des solutions peut aider le mesureur à apprendre plus rapidement le pourquoi du raisonnement de l'expert, ce qui exclut les technologies d'apprentissages automatiques;

Notre analyse privilégie les systèmes à base de cas et à base de règles. Nous faut-il choisir l'une ou l'autre technologie ? Des auteurs qui ont publié récemment [JAC98] [LUG00] et [WAT97] croient que les chercheurs peuvent trouver intéressante la construction de systèmes hybrides. On suggère même dans [JAC98] que MYCIN était un système hybride (à base de règles et de cas) avant qu'il ne soit question de systèmes à base de cas.

Notre problématique suggère que les deux approches peuvent nous aider. La technique du raisonnement à base de règles est souvent utilisée pour résoudre des problèmes similaires aux CBR, par exemple le diagnostic. La combinaison des raisonnements à base de règles et à base de cas (CBR) est aussi utilisée pour créer un system hybride.

Watson écrit à ce propos:

"Interestingly, an increasing number of hybrid systems are being developed that use rule-based techniques for areas that are well understood and CBR to handle less well-understood problems"([WAT97] p. 48).

2.2.3 Problématique de la modélisation de la connaissance

Le choix de ces approches cognitives nous amène à la problématique de la modélisation de la connaissance. En génie logiciel, il existe des méthodologies de résolution de problèmes dont nous nous servons pour la mesure. Est-ce qu'il existe des méthodologies de modélisation de la connaissance? Quelles sont les difficultés? Avant même de répondre à ces deux questions, il y a une question plus fondamentale à laquelle nous devons répondre: qu'est-ce que la résolution de problème?

2.2.3.1 Résolution de problème

Pour Laurière, un problème[23] est ce qui est ressentit par une personne « dès qu'elle se trouve dans une situation où elle désire quelque chose alors qu'elle ne voit pas immédiatement la suite d'actions à accomplir pour obtenir ce quelque chose »([LAUR87] p.9).

Voici les étapes de la démarche de Laurière et des extraits des questions qu'il suggère à chaque étape:

A – Comprendre le problème

Quelles sont les inconnues? Quelles sont les données? Quelle est la condition? Est-il possible de satisfaire à la condition? La condition est-elle suffisante pour déterminer l'inconnue? Est-elle insuffisante? Redondante? Contradictoire?

B – Concevoir un plan

Trouvez le rapport entre les données et l'inconnue. Vous pouvez être obligé de considérer des problèmes auxiliaires si vous ne trouvez pas un rapport immédiat. Trouvez le plan de la solution. Avez-vous déjà rencontré ce problème? Ou bien avez-vous rencontré le même problème sous une forme légèrement différente? Connaissez-vous un problème qui s'y rattache? Connaissez-vous un théorie qui puisse être utile?

C – Mette un plan à exécution

En mettant votre plan à exécution, vérifiez-en chaque détail l'un après l'autre. Pouvez-vous voir clairement si ce détail est correct? Pouvez-vous démontrer qu'il est correct?

D- Examiner la solution obtenue

Pouvez-vous valider le résultat? Pouvez-vous vérifier le raisonnement? Pouvez-vous obtenir le résultat différemment? Pouvez-vous le voir d'un coup d'œil? Pouvez-vous vous servir du résultat ou de la méthode pour quelque autre problème?

2.2.3.2 Transfert et modélisation de la connaissance

Buchanan[24] écrit que l'acquisition de la connaissance dans les systèmes experts "is the transfer and transformation of potential problem-solving expertise from some knowledge source to a program".

Le transfert de la connaissance est généralement accompli par une série d'interviews entre l'ingénieur de la connaissance et l'expert du domaine en autant qu'il puisse articuler son expertise pour les fins du système de connaissance. C'est à ce point que se situe la problématique de la modélisation de la connaissance.

---

[23] Cette définition de problème est similaire à celle de Budgen en génie logiciel ([BUD94].

[24] Voir aussi ([JAC98], p. 4).

van Heijst écrit:

“Different knowledge elements play different roles in reasoning… Therefore, knowledge elements must be type according to their role in problem solving”[VAN97].

van Heijst [VAN97] distingue 5 différents types de connaissance et suggère la construction d'un méta-modèle des connaissances[25] :

- les tâches: correspondent aux buts qui doivent être réalisés lors de la résolution du problème
- les méthodes de résolution de problèmes: ce sont les façons de réaliser les buts dans les tâches. Dans certains cadres de référence de la modélisation de la connaissance, les méthodes de résolution de problèmes définissent des sous tâches pour lesquelles d'autres méthodes de résolution de problèmes peuvent être appliquées
- les inférences: décrivent les étapes du raisonnement dans le processus de résolution de problèmes. Les inférences sont aussi appelées des mécanismes d'inférences. Les modèles d'inférences peuvent être aussi appelés "structure d'inférence"
- les ontologies: décrivent la structure et le vocabulaire du domaine de la connaissance statique
- la connaissance du domaine: ensemble des énoncés tenus pour vrai sur le domaine.

Ces types de connaissance sont directement inspirés de la méthodologie CommonKADS. Schreiber et al. définissent trois types de connaissance dans la méthodologie CommonKADS [SCH99], p. 89): connaissance des tâches, connaissance des inférences et connaissance du domaine. Il apparaît que la connaissance des tâches regroupe les tâches et les méthodes de résolution de problèmes, alors que la connaissance du domaine regroupe les ontologies et la connaissance du domaine.

Nous reprenons les différents types de connaissances suggérés par van Heijst dans le chapitre suivant en les adaptant aux domaines du génie logiciel (SWEBOK), de la mesure fonctionnelle COSMIC-FFP et de la démarche du mesureur.

2.2.3.3 Remarques sur les difficultés de la modélisation de la connaissance

Selon Jackson ([JAC98], p. 4), la productivité d’un tel transfert est habituellement relativement faible. Il est intéressant de noter les raisons de ce faible transfert car elles précisent en même temps la nature de l’expertise :

- le spécialiste a son propre jargon et il est difficile pour lui de le communiquer dans le langage de tous les jours ;
- les faits et principes sous-jacents ne peuvent être caractérisés précisément, c'est-à-dire en utilisant des théories mathématiques ou des modèles déterministes par exemple ;
- les experts ont besoin de savoir plus qu’un certain nombre de principes ou des faits particuliers d’un domaine pour résoudre un problème. Par exemple, ils savent généralement quelles sortes d’informations sont pertinentes pour quels sortes de jugements, la fiabilité des sources d’information, comment diviser un problème difficile en sous problèmes pouvant avoir des solutions plus ou moins indépendantes. Obtenir ce type de connaissances, qui est normalement basé sur l’expérience

[25] Les buts des tâches à réaliser pour van Heijst seraient les tâches telles que définies par Hayes-Roth (diagnostic, réparation, planification, etc.) alors que pour nous, ce sont les tâches du mesureur (diagnostiquer un processus, diagnostiquer une frontière, etc.). van Heijst envisage l'utilisation d'un grand nombre d'ontologies, alors que nous nous limitons au génie logiciel et à la mesure fonctionnelle COSMIC-FFP. La vision est donc très différente en terme de niveau.

personnelle plus que sur la formation formelle, est beaucoup plus difficile que d'obtenir des faits particuliers ou des principes généraux ;

- enfin, l'expertise humaine, même relative à un domaine très étroit, est souvent liée à un contexte beaucoup plus large impliquant une bonne connaissance d'autres domaines. Par exemple, l'application d'une méthode de mesure fonctionnelle pour obtenir la taille d'un logiciel est un domaine où l'expertise requise est très pointue. Cependant, cette tâche implique que le mesureur doit avoir non seulement une bonne connaissance des processus de développement en génie logiciel, mais aussi avoir la capacité d'acquérir très rapidement une certaine connaissance des fonctions du logiciel qu'il mesure du point de vue de son utilisateur (par exemple acquérir des connaissances sur le fonctionnement d'une banque, pour des systèmes bancaires, etc.).

2.2.4 Approches mathématiques des moteurs d'inférence

Les systèmes à base de règles utilisent des formules mathématiques pour résoudre les problèmes. Nous avons examiné trois approches mathématiques : la théorie baysienne, la théorie de la certitude et la logique floue. Nous allons décrire chacune brièvement et faire ressortir leurs avantages et inconvénients.

2.2.4.1 Théorie baysienne

La théorie baysienne est une approche par raisonnement inexact. Traditionnellement, le raisonnement inexact en est un de probabilité mathématique. La théorie des probabilités repose sur des essais répétitifs et effectués au hasard permettant de déterminer un nombre P(E) appelé probabilité de l'événement. L'ensemble des résultats possibles pour une expérience est appelé l'espace d'un échantillon qui est noté par (S) ([DUR93], p. 306). Si les événements sont indépendants et que l'on désire connaître la probabilité de plus d'un événement, on a une probabilité composée. Si les événements sont dépendants, on utilise la probabilité conditionnelle. La question est: quelle est la probabilité que l'événement A survienne compte tenu que l'événement B est déjà survenu? ([DUR93], p. 309)

Dans plusieurs problèmes, on est préoccupé par la situation inverse. Quelle est la probabilité d'un événement antérieur compte tenu d'un événement postérieur à celui-ci? C'est une probabilité a posteriori. Pour la recherche des causes d'un problème (diagnostique), c'est l'approche qui est préconisée. Au 18^{ième} siècle, Bayes a formulé la question de la façon suivante:

Quelle est la probabilité de la véracité d'une hypothèse (H), compte tenu d'un événement (E) ou encore P(H/E)?

Avec le théorème de Bayes, il faut connaître les probabilités d'un événement antérieur pour interpréter la situation présente.

Il y a des variations au théorème de Bayes. On peut essayer de mesurer le degré de support à une hypothèse (H) compte tenu de la présence de l'événement (E). C'est ce qu'on appelle le support (likelihood) suffisant (LS). On peut aussi faire l'inverse et essayer de mesurer le discrédit (LN) pour une hypothèse (H) si l'événement (E) est manquant.

En utilisant les mesures de support (LS) et de discrédit (LN), il est possible de développer des règles qui calculent leurs valeurs pour des conclusions compte tenu des faits.

SI événement (E) ALORS hypothèse (H) (mesure de support, LS; mesure de discrédit,LN)

Cette règle statue que l'événement (E) suggère l'hypothèse (H) avec un degré (de support ou de discrédit) fournit par les mesures de support (LS) et de discrédit (LN).

Ces deux facteurs (LS et LN) sont fournis par l'expert[26] et sont utilisés (avec un certain nombre de contraintes mathématiques[27]) pour calculer les probabilités postérieures d'une hypothèse (H).

L'approche baysienne a été utilisée par le système expert PROSPECTOR pour propager les probabilités à travers son réseau d'inférence. De plus, dans PROSPECTOR, les probabilités de chaque hypothèse sont mises à jour à chaque fois qu'un nouvel événement est soumis par l'utilisateur.

La théorie des probabilités est une technique qui permet de travailler avec de l'information inexacte ou au hasard. Il y a cependant un certain nombre de contraintes:

- Cette théorie implique que l'échantillon est bien défini et que les probabilités de chaque événement peuvent être obtenues à partir d'un ensemble de données historiques
- Lorsque les informations changent, il faut recalculer tous les facteurs de probabilités, ce qui implique un effort de maintenance supplémentaire
- La somme des probabilités pour et contre une hypothèse, sur la base d'un événement, doit être égale à 1. Dans les faits, les experts peuvent donner une valeur positive à un événement (ex: .7), mais hésitent à donner une valeur de contrepartie (ex: .3) qui permettrait d'avoir une somme de probabilité égale à 1 pour une hypothèse
- On assume l'indépendance conditionnelle des événements, ce qui n'est pas toujours le cas.

Les théories alternatives pour traiter les problèmes avec des informations inexactes et pour lesquelles les informations a priori sont manquantes sont: la théorie de la certitude et la logique floue.

2.2.4.2 Théorie de la certitude

La théorie de la certitude a été développée par Shortcliffe et Buchanan lors des travaux de construction du système expert MYCIN. Elle repose sur une mesure de croyance plutôt que sur les probabilités strictes ([DUR92], p. 333).

Pour Durkin [DUR93], il y a trois types d'incertitude à considérer :

- l'incertitude des faits
- l'incertitude des règles
- l'incertitude des inférences

L'incertitude des faits oblige l'expert à prendre des décisions en composant avec des informations incomplètes et incertaines. Il dira: « il semble que… », « nous croyons que…. ». Ce sont alors des croyances exprimées par l'expert et non des certitudes. Le facteur de certitude ou Certainty Factor (CF) représente alors une croyance. CF peut avoir une valeur entre -1 (totalement faux) et +1 (totalement vrai). Une valeur positive représente un degré de croyance, et une valeur négative

---

[26] Par exemple, l'expert peut dire que le fait est important (LS >1), mais aussi que l'absence d'évidence est peu importante (LN=1) ([DUR93], p. 313).

[27] Ces contraintes mathématiques se formulent de la façon suivante:

SI LS>1 ALORS LN<1

SI LS<1 ALORS LN>1
SI LS=1 ALORS LN=1

un degré de non croyance. Le jugement d'un utilisateur sur la qualité de l'information obtenue (complète, sommaire, assumée ou extrapolée) est un exemple de l'incertitude des faits.

L'incertitude des règles oblige l'expert à dériver une inférence inexacte à partir de l'information disponible. Autrement dit, l'expert, à partir de faits certains, peut croire seulement partiellement à la conclusion. Dans ce cas, le groupe MYCIN a décidé d'attribuer une valeur (entre -1 et +1) pour le facteur de certitude de chaque règle. Par exemple, le CF aura une valeur de .7 pour signifier que la croyance à la règle n'est pas absolue, mais seulement à 70%.

L'incertitude de la conclusion (provenant de l'inférence) est liée au fait qu'il y a un lien entre l'évènement et la conclusion avec les règles. Si la croyance avec l'événement diminue, la croyance à la conclusion diminuera.

Si l'expert reçoit des confirmations concernant ses hypothèses de différentes sources, sa confiance dans son hypothèse sera plus grande. Comment évaluer cette confiance?

Voici un tableau avec un exemple:

Règle 1	Règle 2
SI A	SI C
ET B	ET D
ALORS Z	ALORS Z
CF = 0.8	CF = 0.7

Tableau 3 : Renforcement des croyances (règles)

Croyance nette

L'équipe a aussi constaté que l'expert balance ses croyances entre différentes hypothèses (H) lorsqu'il est confronté à une évidence (E) positive et une autre négative. Pour tenir compte de cette situation, l'équipe de MYCIN a ajouté le concept de croyance nette aux règles.

On utilise la croyance nette avec une simple soustraction de la façon suivante: MB = mesure de la croyance et MD = mesure de la non croyance. Alors, la croyance nette est la différence entre ces deux valeurs. Par la suite, compte tenu que la croyance peut provenir de plusieurs faits, et non d'un seul, la formule a été modifiée pour tenir compte de la pondération des croyances.

Du point de vue de la probabilité, comment peut-on comprendre la théorie de la certitude?

La théorie de la certitude suggère que la « probabilité » à priori de l'hypothèse P(H) devrait être la croyance de l'expert en l'hypothèse avant la considération des faits. D'autre part, sa non croyance a priori en l'hypothèse est P(~H).

Pour la théorie classique de la probabilité, la somme de P(H) et P(~H) est égale à 1.

Ceci semble logique, mais pour un expert sa croyance peut être de .7 pour un événement, mais sa non croyance pour cet événement n'est pas nécessairement de .3. Elle pourrait être de .5.

Comment combiner ces deux croyances? Comment combiner un probable avec un peut-être?

Plutôt que d'essayer de déduire mathématiquement des formules, l'équipe MYCYN a décidé de demander à l'expert ce qu'il aimerait obtenir de cette combinaison. Plutôt que d'avoir une formule exacte pour évaluer l'effet de la combinaison, on s'est préoccupé de trouver les propriétés de la formule. Deux propriétés ont été retenues :

- La propriété commutative visant à prévenir la dépendance des règles selon l'ordre de recherche des règles. Que la règle soit considérée en premier ou en dernier, ça n'affecte pas le niveau de connaissance global
- La croissance asymptotique de la confiance pour permettre une incrémentation partielle de la croyance globale sans qu'elle n'atteigne jamais 100%.

Le point central de cette théorie est que :

- la croyance de l'expert dans une hypothèse augmente ou diminue avec la prise de connaissance des faits qui supportent ou infirment l'hypothèse;
- la mesure de la croyance d'un expert ne correspond pas nécessairement à celle inférée à partir de la théorie de la probabilité c'est-à-dire $MB(H) \neq P(H)$.

Utilité de la théorie de la certitude

L'équipe de développement de MYCIN a introduit dans son système expert des techniques de raisonnement "inexact". Ces techniques sont utiles lorsque l'approche probabiliste est peu appropriée et qu'il n'est pas possible d'obtenir des informations statistiques fiables sur le problème.

2.2.4.3 Logique floue

Le terme "fuzzy" en mathématique a été utilisé au début du siècle passé par Lukasiewicz [LEJ67] lorsqu'il a voulu mettre en relation des termes tels que "grand", "vieux", "haut". Ces termes ont une nature binaire (vraie ou fausse) qui semble simple: l'appartenance des objets ou des personnes à un de ces termes n'est pas facile à déterminer. Il a ajouté un "possible" à "vrai" et à "faux". Ses recherches ont conduit à une technique formelle de raisonnement inexact appelée théorie des possibilités.

Durkin définit la logique floue de la façon suivante: "une branche de la logique qui utilise des degrés d'appartenance à un ensemble, plutôt qu'à une appartenance stricte vraie ou fausse"[DUR93].

Les théories traditionnelles voient le monde en noir ou en blanc c'est-à-dire qu'un objet appartient ou pas à un ensemble. La logique floue permet de relativiser cette approche: ainsi, l'élément "personne de 5 ans" pourrait appartenir à l'ensemble "enfant" à 90% alors que l'élément "personne de 13 ans" appartiendrait à ce même ensemble à 10%. On peut alors construire un échantillon dont les éléments peuvent appartenir en partie à celui-ci, sans qu'on ait à choisir de rejeter ou non un élément. Voici les principales caractéristiques de la logique floue:

- Permet de représenter et de traiter des termes ambigus avec un ordinateur
- Utilise le terme variable linguistique pour décrire un concept vague ou ayant une valeur vague
- Représente les valeurs des variables floues dans un ensemble flou. Cet ensemble met en correspondance les éléments avec un degré de croyance qu'un tel élément appartient à l'ensemble flou
- Permet, avec un adverbe, d'avoir un impact flou sur ce qui est déjà flou. Par exemple, si l'on dit "très grand" le mot "très" est une restriction (hedge) supplémentaire à un mot déjà flou. Des opérations mathématiques peuvent être réalisées pour tenir compte de la nouvelle variable floue

- Comprend plusieurs techniques d'inférence pour la construction d'un système flou dont les plus populaires sont: max-min et max-product[28]
- Est un processus itératif qui demande des ajustements lors de sa construction
- Permet des ajustements aux règles existantes ou aux ensembles flous dans le système expert existant

Est-ce que ces caractéristiques sont nécessaires pour notre problème? Les documents lus par les mesureurs présentent certainement des "variables linguistiques" floues. Elles sont cependant traitées par un mesureur humain et non par un ordinateur. Lorsque les problèmes sont cernés, le mesureur doit appliquer des règles dont les mots ont un sens défini par les règles de la méthode de mesure. Il ne devrait pas y avoir de variables linguistiques floues du moins pour l'expert. Des études plus poussées pourraient nous permettre éventuellement de trouver une application de la logique floue.

2.2.4.4 Conclusion sur les approches mathématiques

La théorie de la certitude semble appropriée pour l'expert de la mesure dont la problématique de résolution de problème est très semblable. L'expert a un grand nombre de décisions à prendre pour mesurer un seul logiciel, et il ne peut se fier qu'à ses connaissances du domaine et à son expérience pour prendre ces décisions.

Voici les raisons qui nous font croire que la théorie de la certitude répond à notre problématique:

- Il est possible de donner un poids, non seulement aux faits, mais aussi à chaque thème[29] du point de vue de l'expert. Il est aussi possible pour le mesureur de donner un poids à la qualité de la documentation
- Il est possible de combiner les thèmes pour évaluer un cas problème
- Il est possible de tenir compte des valeurs positives et négatives des poids en combinant les thèmes.

2.2.5 Problématique de la mesure et approches cognitives

Notre revue de littérature dans le domaine cognitif nous amène à conclure que les raisonnements à base de cas et à base de règles peuvent nous aider à résoudre notre problématique de la mesure. Les raisonnements à base de règles requièrent une approche mathématique dont la plus appropriée semble la théorie de la certitude. La méthodologie suggérée par van Heijst (et tirée de CommonKADS) nous semble prometteuse pour la modélisation de la connaissance du mesureur.

2.2.5.1 Raisonnement à base de cas

Le mesureur doit faire une correspondance entre le monde empirique et le monde formel. Puisque le monde empirique de la mesure fonctionnelle se trouve à travers le processus de construction du logiciel, il doit d'abord comprendre son environnement pour retrouver les cas problème adaptés à la situation. Pour réaliser cette tâche, l'expert de la mesure fonctionnelle doit référer à des expériences passées et retrouver les cas problème associés à ces expériences. C'est ce qu'il pourra faire s'il trouve un moyen d'établir les liens entre l'ontologie du génie logiciel et l'ontologie de la mesure fonctionnelle COSMIC-FFP. L'ontologie du génie logiciel utilisée est celle de SWEBOK (déjà mentionnée au point 2.1.3). Des concepts de l'ontologie sont définis au point 3.1.1. L'ontologie de la mesure fonctionnelle COSMIC-FFP se trouve au point 3.2.3.1.

---

[28] La présentation de ces techniques se trouve aux pages 376 à 380 de Luger [LUG93].

[29] Le thème correspond à l'hypothèse dans la théorie de la certitude.

Ceci rejoint notre autre remarque au point 2.1.5.1: malgré le fait que plusieurs méthodes de mesure fonctionnelle soient appliquées depuis plusieurs années à travers le monde, il n'existe pas de méthode systématique et reconnue pour réaliser la mise en correspondance des règles formelles avec les différents environnements de développement (monde empirique). On se fie surtout à l'expertise des mesureurs

2.2.5.2 Raisonnement à base de règles

Le mode formel de la mesure fonctionnelle se retrouve à travers les règles de la mesure fonctionnelle, d'où la nécessité de prendre en considération une méthode de mesure fonctionnelle formelle. L'analyse du cheminement du mesureur indique que deux ensembles d'habiletés ou de connaissances sont nécessaires au mesureur dont la compréhension des règles de mesure pour les appliquer aux cas à résoudre. Le raisonnement à base de règles devra être construit de façon à ce qu'il s'applique aux différents cas qui seront contenus dans la base de cas. Le premier ensemble d'habiletés est la connaissance du génie logiciel et le deuxième ensemble est la connaissance de la mesure et plus spécifiquement des règles de mesure COSMIC-FFP.

2.2.5.3 Théorie de la certitude

La théorie de la certitude est celle qui semble le mieux correspondre à nos besoins pour les inférences. Elle permet, non seulement de combiner plusieurs thèmes, mais aussi de les traiter différemment si nécessaire. En cela, elle répond à trois principaux critères:

- Si un thème est négatif, la conclusion peut-être rejetée sans tenir compte des autres thèmes. Par exemple, pour identifier un processus (conclusion), il faut que tous les critères ou thèmes soient positifs, sinon ce n'est pas un processus).
- L'acceptation ou le refus de la conclusion vient du poids de chaque thème et des faits associés à chaque thème (l'identification d'un événement déclencheur vient d'un ensemble de thèmes positifs et négatifs, même si un thème est faux, la conclusion n'est pas nécessairement fausse)
- Si un thème est vrai, la conclusion est vraie, même si tous les autres thèmes sont faux (l'identification du sous processus écriture vient que le mesureur à identifié au moins un groupe de données écrit (fait). Même si tous les autres groupes de données ne sont pas écrits, ça reste un sous processus écriture).

Nous n'avons voulu ici que montrer le lien entre la théorie et nos besoins. D'autres expérimentations seront nécessaires pour vérifier tous nos besoins.

2.2.5.4 Modélisation de la connaissance

Nous avons aussi constaté que la modélisation de la connaissance était une tâche importante. Nous avons trouvé dans les différents types de connaissance de van Heijst une piste de solution pour la modélisation de la connaissance du mesureur. Nous appliquons son approche dans le chapitre suivant.

2.2.6 Conclusion

L'approche cognitive devrait nous permettre de modéliser les connaissances des experts en mesure COSMIC-FFP et d'appliquer les méthodes (ex: CBR) et algorithmes qui permettront de résoudre la problématique identifiée:

- Il n'existe pas de méthodes systématiques et reconnues pour réaliser la mise en correspondance des règles formelles avec les différents environnements de développement (monde empirique).

- Les connaissances d'experts en mesure, connaissances acquises après plusieurs années de pratiques, reflètent deux ensembles d'habiletés ou de connaissances nécessaires au mesureur:
    - La compréhension des artefacts du logiciel pour situer le problème (ou cas problème) à résoudre
    - La compréhension des règles de mesure pour les appliquer aux cas problème à résoudre

3. DÉMARCHE DU MESUREUR

3.1 Proposition d'une méthode diagnostique

La méthode de diagnostique repose sur la modélisation des deux types de connaissances nécessaires au mesureur[30], soit la connaissance du génie logiciel et la connaissance de la méthode de mesure COSMIC-FFP.

3.1.1 Connaissance du cycle de vie d'un logiciel

Nous savons que les méthodes et modèles[31] de construction et de maintenance du logiciel varient, le plus souvent, d'une organisation à une autre, mais aussi à l'intérieur d'une même organisation. Il est donc difficile de se raccrocher à une seule méthode ou à un seul modèle. Il existe cependant des modèles bien connus, tel que le modèle en cascade décrit par Boehm ([BOE81], pp 36-37) déjà mentionné dans notre revue de littérature. Nous avons aussi mentionné dans notre revue de littérature les phases de SWEBOK [ABR01b].

Les principales phases du cycle de vie du logiciel de SWEBOK recoupent celles du modèle en cascade. Par exemple, les requis de SWEBOK couvrent les deux premières phases du modèle en cascade de Boehm (faisabilité, plan et requis), etc.

Notre intérêt pour SWEBOK se justifie, entre autre, par une meilleure définition du contenu ou livrables de chaque phase, particulièrement pour les livrables utiles à la mesure.

Voici 4 exemples de définition de "livrables"[32] au niveau des requis.

3.1.1.1 Modèle du processus:

Le modèle du processus fournit une compréhension des requis du processus d'ingénierie :

- ce n'est pas un livrable qui se termine avec la première phase du cycle de vie, mais plutôt le résultat d'une activité qui est initiée au début du projet et que l'on continue de raffiner tout au long du cycle de vie du logiciel;

---

[30] Il faut distinguer ici les "types de connaissance" du mesureur des "types de connaissance" du modélisateur de la connaissance du mesureur tel que proposé par van Heijst.

[31] Un modèle est un ensemble de caractéristiques servant à classer des faits ou des objets, alors qu'une méthode est un ensemble de règles permettant l'apprentissage d'une science ou d'une technique. Ce que nous présentons à partir de Boehm est un modèle, alors que dans SWEBOK nous allons aussi utiliser la méthode pour créer l'ontologie.

[32] La description des « livrables » se fait à un niveau conceptuel dans SWEBOK. Le but de SWEBOK n'est pas de décrire des techniques particulières, mais de montrer leur utilité à différentes phases.

- le livrable doit identifier la configuration des items des requis et les gérer comme tout autre livrable du processus de développement
- il doit être adapté au contexte de l'organisation et du projet

En résumé, le modèle du processus vise à démontrer,dès le départ, comment les activités d'analyse, de spécifications, de validation et de gestion sont configurées.

3.1.1.2 Acteurs du processus:

Ce livrable introduit les rôles des acteurs qui participent au processus d'ingénierie. C'est un processus interdisciplinaire et l'ingénieur doit faire le lien entre le domaine de l'utilisateur et l'ingénierie du logiciel. Il peut y avoir, en plus des ingénieurs, plusieurs utilisateurs et clients impliqués dans ce processus. L'identification des personnes ayant des intérêts dans le logiciel, et la nature de ceux-ci, constituent un pré requis nécessaire, voire même indispensable.

3.1.1.3 Requis du système (SRS):

Ce livrable est un document (connu aussi sous le nom de document des requis) qui prend note des requis du logiciel. Il définit les requis à un haut niveau, selon la perspective du domaine et écrit en collaboration avec les utilisateurs et clients du logiciel. Il doit indiquer les requis du logiciel avec l'information sous-jacente aux objectifs du logiciel, l'environnement visé, ses contraintes et ses hypothèses. Il peut inclure des modèles conceptuels pour illustrer le contexte du logiciel, des scénarios d'utilisation, les entités du domaine principal, les données et les flux d'information.

3.1.1.4 Modèles conceptuels

Le développement des modèles du problème à solutionner est fondamental pour les analyses des requis. On vise ici à comprendre le problème plus qu'à le solutionner. Les modèles conceptuels comprennent les modèles entités/relations du domaine afin de refléter le plus possible les relations et les dépendances rencontrées dans le monde réel. Plusieurs sortes de modèles peuvent être développés : flux de données et de contrôle, modèles d'état, traces des événements, interactions entre les utilisateurs, les modèles objets et autres.

Les définitions de ces livrables sont un guide pour la recherche des cas problème (via la documentation) aux fins de la mesure du logiciel. À partir de ces livrables, il est possible de situer l'environnement du problème. En terme CBR, on dirait qu'ils permettent de retrouver des cas similaires.

3.1.2 Connaissance de la méthode de mesure COSMIC-FFP

À partir de la méthode COSMIC-FFP (2), il est possible de décrire les étapes et les sous étapes qui seront à la base des principales tâches du mesureur. Le cheminement du mesureur peut s'appliquer à chacune des ces étapes et sous étapes.

3.1.2.1 Mise en correspondance

Étape 1a: Sur la base des requis et des spécifications sur l'interaction entre l'équipement et le logiciel, le mesureur doit détecter s'il y a plus d'une couche et les identifier.

Étape 1b: Sur la base des requis et des spécifications sur l'interaction entre l'équipement et le logiciel, le mesureur doit identifier les utilisateurs.

Étape 2: Sur la base des requis et des spécifications sur l'interaction entre l'équipement et le logiciel, le mesureur doit identifier la frontière du logiciel.

Étape 3a: À partir des requis, le mesureur doit identifier tous les processus fonctionnels du logiciel

Étape 3b: À partir des requis, le mesureur doit identifier tous les événements déclencheurs[33] du logiciel.

Étape 3c: À partir des requis, le mesureur doit identifier tous les groupes de données fonctionnels du logiciel.

3.1.2.2 Mesurage

Étape 4: Les sous processus (4a: Entrée, 4b: Sortie, 4c: Lecture, 4d: Écriture) doivent être identifiés pour chaque processus du logiciel.

Étape 5: La taille fonctionnelle du logiciel résulte de l'agrégation des résultats de la mesure, c'est-à-dire de l'addition de tous les sous processus. Cette opération étant strictement arithmétique, elle ne requiert pas d'expertise particulière; il n'est donc pas nécessaire de la considérer pour notre expérimentation.

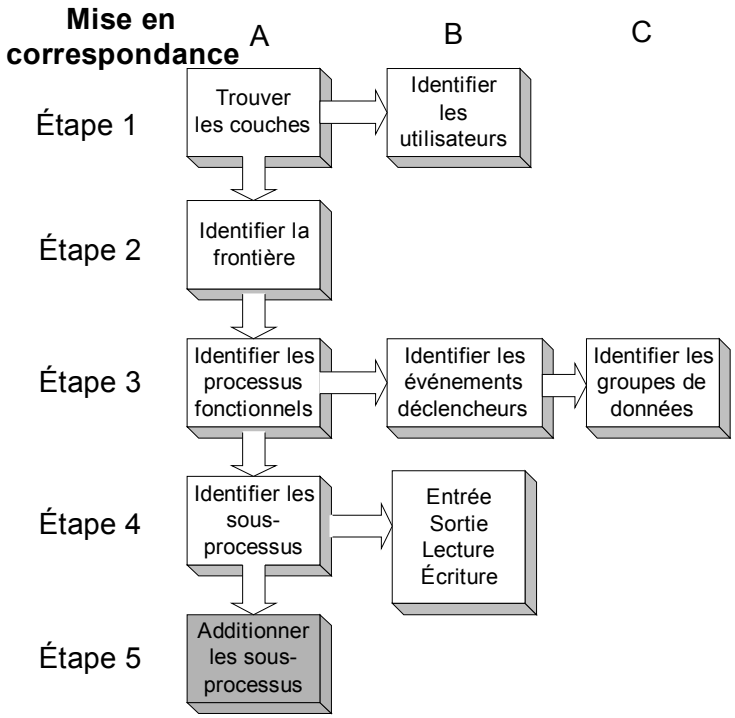

Figure 10 Tâches de la méthode COSMIC-FFP

À partir de ces étapes, il est possible de retrouver les principaux cas problème rencontrés par le mesureur (figure 10). Chaque bloc peut être identifié à un problème que le mesureur cherche à solutionner. Il faut noter aussi qu'il y a une hiérarchie dans les tâches de la figure 10. Une tâche appartenant à l'étape 1 doit être réalisée avant la tâche de l'étape 2, et ainsi de suite. Par exemple, à l'étape 2, le mesureur doit d'abord identifier la frontière du logiciel s'il veut pouvoir identifier les processus appartenant à ce logiciel.

---

[33] L'événement déclencheur pourrait aussi être étudié au niveau des sous processus, puisque de même niveau. Nous avons opté pour l'étudier au niveau du processus puisque l'événement déclencheur sert surtout à identifier le processus.

Ces tâches, mises en relation avec la démarche du mesureur et les livrables de SWEBOK, nous permettent de proposer une procédure visant à permettre au mesureur de solutionner les différents problèmes de mesure de façon cohérente. Ils sont aussi notre base pour modéliser la connaissance du mesureur. En effet, si nous prenons l'exemple des processus, il n'est pas possible pour le mesureur de les identifier sans avoir des informations sur ces processus. Ces informations se trouvent dans la documentation (livrable) proposée par la méthode de développement et de maintenance du logiciel. On peut trouver dans SWEBOK une description (non nécessairement exhaustive) de cette documentation selon les phases de développement et de maintenance du logiciel. Le travail du mesureur consiste à identifier les documents pertinents à l'identification des « éléments » décrits dans la méthode de mesure fonctionnelle COSMIC-FFP. Nous utilisons SWEBOK pour aider le mesureur à s'orienter dans la documentation des logiciels. C'est en quelque sorte une classification de la documentation du logiciel. Par exemple, cette classification peut aider à répondre à la question suivante : dans quel type de documentation peut-on trouver les informations relatives à l'identification des couches?

3.2 Modélisation de la connaissance du mesureur

Nous reprenons ici les 5 types de connaissance (modélisation) selon van Heijst et al. [VAN97], et les adaptons pour nous aider à modéliser la connaissance du mesureur[34] (figure 11).

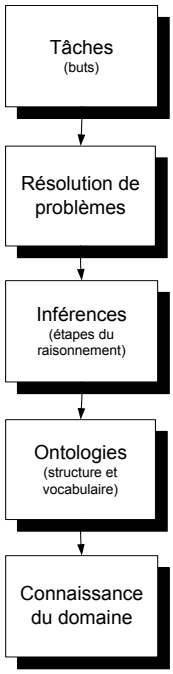

Figure 11 Types de connaissances

Il faut noter que nous allons décomposer la connaissance à un niveau plus fin que les exemples de van Heijst. Par exemple, pour la tâche, nous savons déjà qu'il s'agit de diagnostic. Nous voulons savoir comment le mesureur fera ce diagnostic. Voici une brève définition de chaque type de connaissance et un exemple avec la mesure COSMIC-FFP.

[34] On suppose ici que le mesureur possède la connaissance d'un expert en mesure fonctionnelle.

3.2.1 Tâches

La tâche permet de décrire les buts du mesureur et les stratégies employées pour réaliser ces buts. Pour le mesureur, le but est d'identifier différents concepts de la méthode de mesure COSMIC-FFP à partir des artefacts de la documentation du logiciel (ou des commentaires des développeurs).

La procédure de haut niveau (stratégie) est la suivante :

- Le mesureur doit identifier le problème et le situer dans la tâche à accomplir
- Le mesureur pourrait aussi devoir situer chaque problème par rapport aux autres s'il y a plus d'un problème potentiel
- Pour chaque problème, le mesureur doit se poser les questions appropriées pour bien comprendre et interpréter le problème
- Les réponses aux questions conduisent à la solution cherchée
- La solution peut conduire à un autre problème ou à une information pertinente qui aidera à la solution. Elle peut aussi conduire à un nouveau cas potentiel.

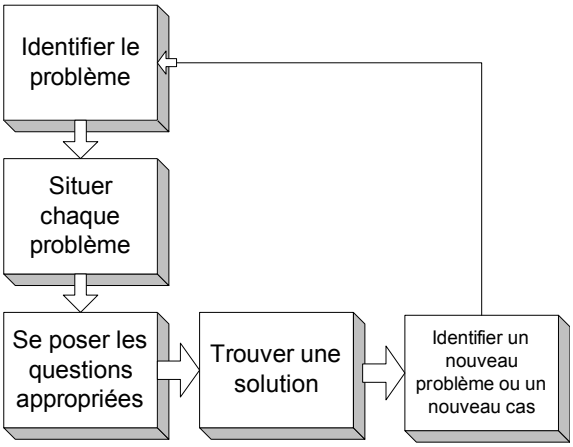

Figure 12 Procédure d'application

Les tâches du mesureur dans la procédure d'application (figure 12) et les étapes de la méthode diagnostique (figure 14) de résolution de problèmes sont très semblables. Ce qui s'explique par le fait que la méthode diagnostique de résolution de cas problème vise à offrir une solution à la procédure d'application du mesureur.

3.2.2 Résolution de problèmes

Les méthodes de résolution de problèmes sont des approches pour identifier, pour les fins de la mesure, un concept de la mesure fonctionnelle COSMIC-FFP (but) décrit dans les tâches. Les principales approches sont les suivantes:

- La recherche par mots clefs. À partir du vocabulaire du cycle de vie du logiciel et de la mesure fonctionnelle COSMIC-FFP, des mots clefs seront retenus et utilisés pour permettre au mesureur d'identifier l'environnement du problème.
- La déduction. Par exemple, à partir de cas problème précis, le logiciel déduira quelles sont les questions qu'il faut poser pour aider à le résoudre.
- L'induction. Par exemple, à partir des réponses aux questions, le logiciel induira si le concept identifié par le mesureur est le bon
- Le chaînage arrière. C'est une conséquence de l'induction.

3.2.3 Inférences

Le terme inférence est généralement utilisé pour signifier qu'un processus permet d'obtenir des conclusions ([RUS95], p. 163). Pour Schreiber et al. ([SCH99], p. 105), dans le cadre du modèle de connaissance, c'est le plus bas niveau de décomposition fonctionnelle. D'autre part, nous utilisons le terme « règles d'inférences » pour désigner le processus permettant d'obtenir des conclusions. C'est la « boîte noire » ou le processus interne qui permet d'obtenir l'inférence. Les règles d'inférences décrivent les étapes du raisonnement permettant de résoudre un problème. Les règles d'inférences utilisent les approches de la résolution de problèmes. Le mesureur recherche deux types de conclusions ou d'inférences:

- des recherches permettant de prioriser les cas problèmes à résoudre à partir de mots clefs ou d'autres informations.
- des recherches permettant de solutionner les cas problèmes ou de recommander d'autres tâches pour mieux circonscrire les cas problèmes à partir de calculs basés sur la théorie de la certitude.

3.2.3.1 Ontologies (vocabulaire et structure)

Pour van Heijst [VAN97], une ontologie est un niveau de connaissances explicite spécifiant des concepts, c'est-à-dire un ensemble de distinctions qui sont utiles pour un agent (le mesureur par exemple). La conceptualisation (et ainsi l'ontologie) peut être affectée par un domaine particulier et une tâche particulière.

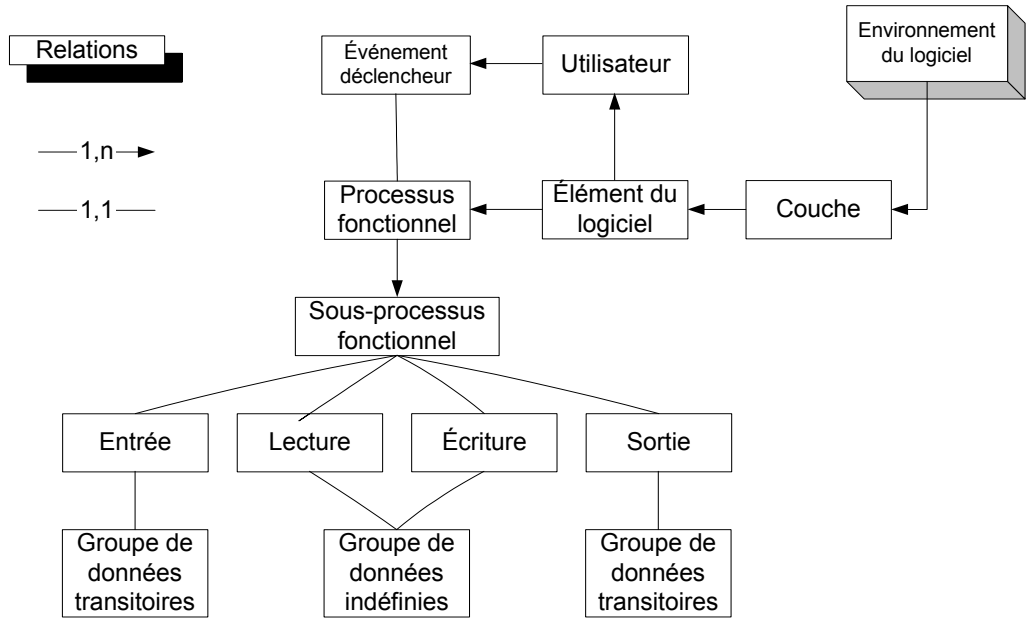

Figure 13 Ontologie COSMIC-FFP [ABR01a]

L'ontologie décrit le vocabulaire et la structure du domaine de la connaissance statique (en opposition avec les tâches qui sont de nature dynamique). Pour le mesureur, le vocabulaire et la structure nécessaires se trouvent dans le cycle de vie du logiciel et dans la méthode de mesure COSMIC-FFP. À partir de cette l'ontologie COSMIC-FFP (figure 13), on peut trouver son vocabulaire (couches, frontières, processus, groupes de données, etc.). L'ontologie n'est pas aussi formelle pour le génie logiciel à travers SWEBOK. Dans ce domaine, il reste encore des avancées importantes à réaliser pour que l'état de l'art et de la pratique permettent une ontologie précise. Cet état de fait est en lien avec les difficultés liées à la qualité des informations, comme nous l'avons souligné dans notre problématique sur la mesure.

3.2.3.2 Connaissance du domaine

La connaissance du domaine réfère à un ensemble d'énoncés sur le domaine. Ce sont les énoncés que l'expert du domaine de la mesure fonctionnelle doit faire pour réaliser la mise en correspondance entre les artefacts du logiciel à mesurer et les règles de la méthode de mesure. Dans les étapes de la méthode diagnostique que nous exposons à la section suivante, on constate que ces énoncés se situent sur plus d'un niveau. Par exemple, le mesureur fera l'énoncé suivant pour mettre en correspondance le concept de processus dans COSMIC-FFP et le modèle de processus dans SWEBOK: il est possible de trouver des processus dans le modèle du processus. Cet énoncé identifie, entre autre, un concept topologique servant à identifier les processus. À un autre niveau, il fera un énoncé établissant le lien entre un concept topologique et les cas problèmes, puis un lien entre les cas problèmes et les questions au niveau le plus bas.

La méthode diagnostique permet de construire une base de connaissances comprenant les règles utilisées par l'expert lui permettant ultimement de réaliser ses tâches et d'en arriver à la mesure du logiciel. Elle est essentiellement basée sur le diagnostic.

Dans la prochaine section, nous allons décrire les étapes et les sous étapes de la méthode diagnostique.

3.3 Méthode diagnostique

3.3.1 Les étapes et sous étapes de la méthode

Les étapes de la méthode diagnostique sont représentées dans la figure qui suit :

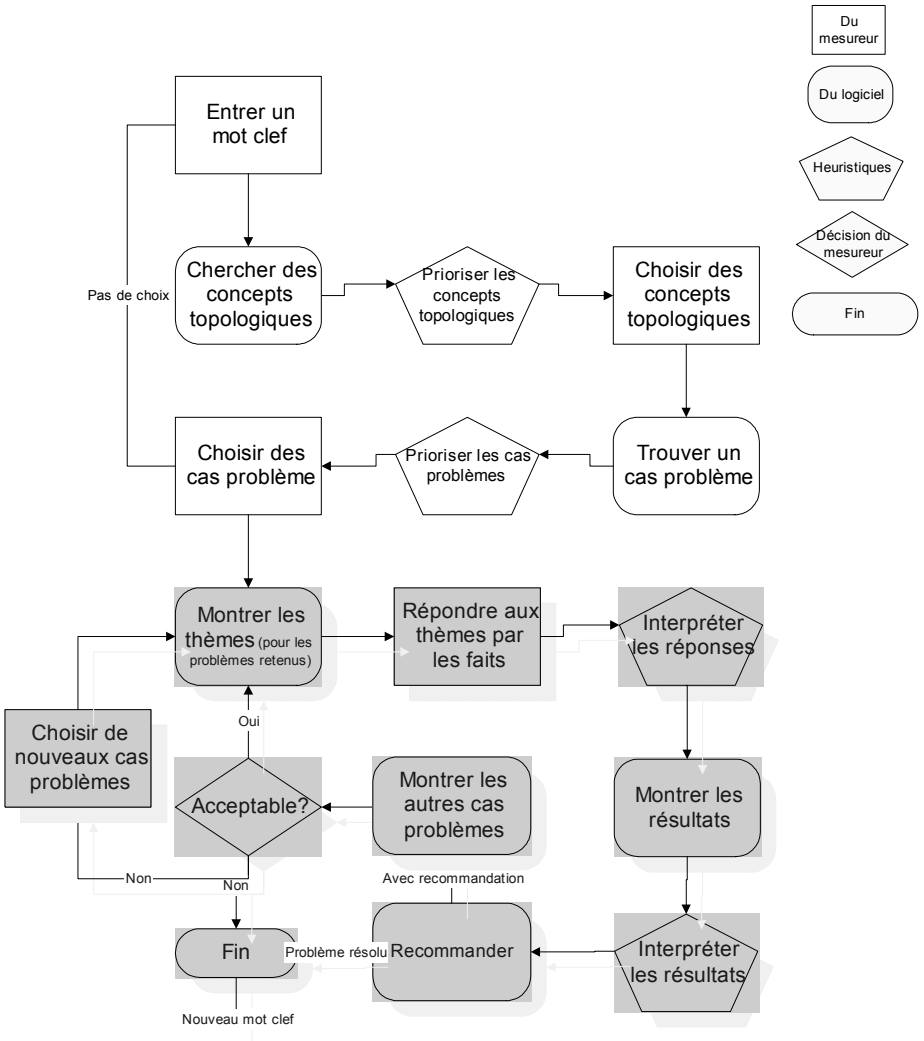

Figure 14 Étapes et sous étapes de la méthode diagnostique

3.3.1.1 Description des étapes de la méthode diagnostique

La méthode diagnostique comprend plusieurs étapes et sous étapes. Voici une brève description de ces dernières.

Étape 1: Identifier le problème

Sous étape 1a: Entrer un mot clef[35]

Le mesureur doit entrer un mot clef lui permettant de situer le problème qu'il doit résoudre. Ce mot clef peut être un concept dans le vocabulaire de la méthode de mesure COSMIC-FFP, tels "processus", "lecture", "écriture", "couche", etc. Il peut aussi être un mot trouvé dans la

[35] Nous employons le terme au singulier, mais il pourrait y avoir plus d'un mot clef.

documentation des logiciels mesurés, tels "fichier", "table", "événement", etc. Ça peut aussi être un concept trouvé dans le vocabulaire de SWEBOK tel "design", "maintenance", "requis", etc. Ces mots clefs devront avoir été prévus par l'expert dans la méthode diagnostique.

Sous étape 1b: Trouver un concept topologique

Cette sous étape est réalisée par l'outil de diagnostic[36] ou prototype de diagnostic. L'outil, à partir du mot clef, fait une recherche des concepts topologiques associés à ce mot clef. Le concept topologique est le point d'intersection entre un concept du processus de développement et une partie de la tâche du mesureur. Autrement dit, c'est la mise en correspondance des concepts du processus de développement (dans le cas présent ceux de SWEBOK) et les concepts de la méthode de résultat qui lui apporte la solution ou le conduit à un autre problème.

Sous étape 1c : Prioriser les concepts topologiques

Le prototype est conçu pour permettre à l'expert de prioriser les concepts topologiques sur la base des mots clefs. À chaque mot clef sera associé un pourcentage représentant la croyance d'appartenance des mots clefs aux différents concepts topologiques. Un mot clef peut trouver plusieurs concepts topologiques. Un concept topologique peut être trouvé à partir de plus d'un mot clef. De plus, pour permettre de déterminer la priorité d'un concept topologique par rapport à un mot clef, il faut lui associer un pourcentage. Les pourcentages d'association entre les mots clefs et les concepts topologiques devraient varier pour permettre de déterminer la priorité des concepts topologiques par rapport à un mot clef.

Sous étape 1d: Choisir un concept topologique

Le mesureur peut faire un ou plusieurs choix entre plusieurs concepts topologiques pour la suite des opérations. Notre hypothèse est que le mesureur peut faire ce choix mieux que le prototype puisque c'est lui qui a accès à la documentation du logiciel qu'il mesure. Les options du mesureur sont :

- de faire un choix à partir des suggestions du prototype;
- de faire plus d'un choix à partir des suggestions du prototype;
- de ne pas faire de choix et prendre toutes les suggestions;
- de refuser tous les choix et de revenir à l'entrée d'un nouveau mot clef

Étape 2: Situer le problème

Sous étape 2a : trouver les cas problème

À partir du (ou des) choix du mesureur (concept topologique), le prototype va trouver les cas problème associés à chaque choix en leur donnant une priorité.

Sous étape 2b : prioriser les cas problème

Tout comme pour le concept topologique, la priorité des cas problème a été donnée par l'expert en terme de pourcentage.

Sous étape 2c : choisir les cas problème

Tout comme pour le concept topologique, le mesureur peut faire un ou plusieurs choix entre plusieurs cas problème.

---

[36] La méthode diagnostique se concrétisera dans un outil de diagnostic ou prototype de diagnostiqc. Lors de la description des étapes et sous étapes de la méthode diagnostique nous allons indiquer si elles exigent l'intervention du mesureur ou du prototype. Nous utiliserons le mot prototype pour signifier prototype de diagnostic, sauf indication contraire.

Étape 3: établir les thèmes appropriés

Sous étape 3a : montrer les thèmes (pour tous les cas problème retenus)

Un certain nombre de thèmes sont associées à chaque cas problème. Il est possible qu'un thème soit associé à plus d'un cas problème apparaissant dans la liste des cas problème retenus. Le même thème ne devrait par apparaître plus d'une fois.

Sous étape 3b : établir les faits pour chaque thème

Le mesureur établit les faits pour chaque thème associé à un (ou des) cas problème. Il peut y avoir un des trois types de faits suivants pour chaque thème : des choix multiples, oui/non ou un nombre.

Sous étape 3c : évaluer les résultats

Les thèmes seront évalués sur la base des faits par le prototype. Vous trouverez à la section Inférences une explication des formules qui sont utilisées pour évaluer les thèmes à partir des faits.

Étape 4: Trouver une solution

Sous étape 4a : montrer les résultats

Les résultats sont montrés, en terme de pourcentage, au mesureur. Ce dernier peut se faire une idée sur le cas problème à partir de ces résultats bruts.

Sous étape 4b : évaluer les résultats

Le prototype fait une évaluation des résultats sur la base de paramètres d'interprétation fournis par l'expert. Celui-ci pourra, à partir des pourcentages, fournir des seuils qui indiqueront si la solution est trouvée ou pas.

Sous étape 4c : recommander une action

Une recommandation quant à la suite des actions à prendre sera faite par le prototype. Cette recommandation pourra être :

- d'indiquer que la solution est trouvée (ex : le mesureur a clairement identifié un processus) et qu'il n'est pas nécessaire de faire plus de recherches ou de recommander d'autres cas problèmes

Étape 5: Nouveaux cas

Le mesureur répète les étapes à partir de l'étape 2.

3.3.1.2 Commentaires

Le prototype peut aussi donner une explication sur la recommandation si le mesureur le désire et si l'explication est disponible. Il n'y a pas nécessairement une explication pour chaque recommandation.

La méthode diagnostique sera utilisée par le mesureur. Une autre question se pose: comment l'expert devrait-il s'y prendre pour créer les cas problèmes à partir de son expérience? Nous y répondrons après avoir donné un exemple d'un logiciel de diagnostic.

3.3.1.3 Help!CPR et la méthode diagnostique

Il y a un parallèle à faire entre la méthode diagnostique que nous venons de décrire et le logiciel de diagnostic Help!CPR.

Pour une première expérimentation, nous avons choisi un outil de type CBR vendu par la firme Haley Enterprise (Help!CPR). Son interface utilisateur, relativement facile à utiliser, a des ressemblances avec la méthode diagnostique que nous venons de décrire. Nous croyons important de présenter ce logiciel afin que le lecteur puisse se faire une idée de ce qui a inspiré notre démarche cognitive au départ. Une description sommaire de ce logiciel se trouve à l'annexe A.

3.3.1.4 Exemple d'application

Voici un exemple (tableau 4) d'application des règles de mesure fonctionnelle sur la base de la théorie de la certitude.

Dans cet exemple, nous présentons ce que le mesureur peut voir lorsqu'il doit résoudre différent cas problèmes[37]. Un cas problème comprend deux thèmes (nous présentons deux thèmes pour simplifier l'exemple). Les faits comprennent des réponses multiples. Nous avons sept (7) colonnes qui se comprennent comme suit:

Colonne 1: identification du cas problème qui est ici l'identification d'un processus ou notre conclusion.

Colonne 2: identification des thèmes qui sont ici liés uniquement au nombre d'événements déclencheurs et à la nature de l'événement déclencheur.

Colonne 3: les différentes possibilités de faits quant à la nature des thèmes.

Colonne 4: le pourcentage attribué à un fait spécifique en rapport avec la nature du thème de la colonne (2). Le pourcentage indique la confiance de l'expert qu'il y a un lien positif entre un fait spécifique lié au thème et l'existence d'un processus (notre cas problème ou conclusion). La confiance de l'expert en l'existence d'au moins un événement indiquant qu'il y a effectivement un processus, est à 80% pour le fait a.

Colonne 5: le pourcentage indique la confiance de l'expert qu'il y a un lien négatif entre le thème et l'existence d'un processus (notre cas problème). La confiance de l'expert que la non existence d'au moins un événement déclencheur puisse indiquer qu'il n'y a pas de processus, est à 60% pour le fait c.

Colonne 6: indique le degré de confiance de l'expert que le thème est pertinent au cas problème, ou encore que le thème permet de résoudre effectivement le cas problème. L'expert a confiance à 80% que le nombre d'événements déclencheurs est un thème qui permet d'identifier un processus.

Colonne 7: indique le degré de confiance du mesureur qu'il a toute l'information pour répondre correctement à la question. La valeur par défaut est 100%.

(1) Cas problème (H)	(2) Thèmes (Q)	(3) Faits	(4) CFR	(5) CFR	(6) CFR	(7) CFQ
Identification d'un processus	Nombre d'événements déclencheurs	a) Au moins un événement déclencheur	0.8		0.8	1.0
		b) Il y a un événement déclencheur	0.6			
		c) Il n'y a pas d'événement déclencheur		0.6		

[37] Le cas problème est aussi une conclusion dans notre exemple.

	Nature de l'événement déclencheur	a) Action d'un utilisateur humain	0.8		0.6	1.0
		b) Action d'un autre logiciel	0.6			
		c) Données provenant d'une horloge	0.4			
		d) Données d'une autre couche		0.2		

Tableau 4 : Exemple d'application des règles

Nous utilisons la formule CF(H,E1) = CF(E1) * CF(RULE 1), ou CF(RULE 1) correspond à la multiplication des colonnes (4) ou (5) à la colonne (6) et CF(E1) à la colonne (7).

CF(H,E1) = CF(E1) * CF(RULE 1) conjonctive	
Choix de a) E 1	0.64
Choix de b) E 1	0.48
Choix de c) E 1	-0.48
D n'influence pas	

Tableau 5 : Thème 1 selon les faits

CF(H,E2) = CF(E2) * CF(RULE 1) conjonctive	
Choix de a) E 2	0.48
Choix de b) E 2	0.36
Choix de c) E 2	0.24
Choix de d) E 2	-0.12

Tableau 6 : Thème 2 selon les faits

Les deux tableaux précédents (5 et 6) montrent le résultat de la confiance que l'on peut avoir en la conclusion considérant son poids. Le fait a) avec le thème 1 donne une confiance de 0.64, alors que le fait c) pour le même thème donne une confiance de -0.48. Pour obtenir ces résultats, on multiplie le poids donné au fait (ou confiance par rapport au choix) et le poids donné au thème par rapport au problème.

CFcombine(CF1,CF2) = CF1 + (CF2 * (1 - CF1)) conjonctive		Les deux > 0
Choix de a) E 1 et a) E 2	0.81	
Choix de b) E 1 et b) E 2	0.67	
Choix de b) E 1 et c) E 2	0.60	

Tableau 7 : Selon 3 faits (positifs)

Le tableau 7 donne différentes solutions selon différents choix de faits aux thèmes 1 et 2. On constate que la combinaison d'un fait positif à un autre fait positif donne un résultat positif encore

plus élevé. Ceci confirme l'influence de deux faits positifs, selon deux sources, comme étant plus positive.

CFcombine(CF1,CF2) = (CF1 + CF2) / (1 - min (CF1, CF2))		Un < 0
Choix de c) E 1 et a) E 2	0.00	
Choix de b) E 1 et d) E 2	0.41	
Choix de a) E 1 et d) E 2	0.59	

Tableau 8 : Selon 3 faits (positif et négatif)

Pour le tableau 8, nous avons pris des réponses avec des thèmes ambigus, soit la combinaison d'un fait positif à un fait négatif. Dans tous les cas, les résultats sont mitigés, passant de 0.05 à 0.28. Il n'est pas certain que l'on ait pu identifier un événement déclencheur.

CFcombine(CF1,CF2) = CF1 +(CF2 * (1 + CF1))		Les deux < 0
Choix de c) E 1 et d) E 2	-0.54	

Tableau 9 : Selon 1 fait (négatif)

Le tableau 9 combine deux pourcentages négatifs. Le résultat est non ambigu: il ne s'agit pas d'un événement déclencheur.

4. JUSTIFICATION DE L'APPROCHE DIAGNOSTIQUE

4.1 Introduction

En 1977, Faigenbaum, alors conférencier invité à l'International Joint Conference on Artificial Intelligence, faisait cette observation sur les systèmes experts :

"La puissance d'un système expert provient plutôt des connaissances qu'il possède que des formalismes particuliers ou des schémas d'inférence qu'il utilise"[PAQ89]

Par analogie, les formalismes seraient aux systèmes experts, ce que les langages de programmation sont aux systèmes d'information. Dans ce contexte, la justification de la construction d'un système expert viendrait :

- de l'existence d'utilisateurs pour le système que l'on veut développer. Est-ce qu'il y a un besoin? Est-ce que le système sera utile? Est-ce que les mesureurs utilisent implicitement ou explicitement des expertises du passé pour résoudre les problèmes d'application de la mesure?
- de la capacité d'aller chercher les connaissances nécessaires pour répondre aux utilisateurs. Existe-t-il un ou des experts qui pourront alimenter le système expert? Est-ce que ces experts sont suffisamment disponibles?

Dans un deuxième temps, il faut justifier l'approche particulière choisie. Pour notre recherche, nous croyons que l'approche diagnostique est celle qui convient le mieux. Puisque l'approche diagnostique est basée sur les cas[38], il faut se demander :

- ce qu'est un cas dans le cadre de notre problématique (application de la mesure fonctionnelle COSMIC-FFP)
- si les interprétations des règles de mesure pour l'application de la mesure mènent à la résolution d'un cas
    - de quelles façons les cas peuvent être trouvés
    - si les cas peuvent être généralisés et servir à résoudre d'autres cas
    - si le nombre d'informations est suffisant historiquement

Avec notre approche diagnostique, nous utiliserons des approches quantitatives (théorie de la certitude) pour aider le mesureur à appliquer la mesure fonctionnelle. On peut alors se demander :

- s'il est possible de formuler des questions quantifiables pour résoudre la plupart des cas
- s'il est possible d'utiliser des règles au niveau des cas
- si les cas peuvent être catégorisés et indexés

4.2 Besoins et expertise

4.2.1 Catégories d'utilisateurs

Il existe des utilisateurs pour un système expert qui aidera à appliquer les règles de mesures fonctionnelles.

[38] Il faut se rappeler que ce terme a une signification particulière dans notre projet de recherche.

Les catégories d'utilisateurs sont les suivantes :

- les entreprises qui appliquent la mesure fonctionnelle pour la mesure des logiciels;
- les firmes conseils qui offrent des services en mesure fonctionnelle auprès des entreprises;
- les associations nationales et internationales de mesures, et plus spécifiquement celles qui maintiennent et diffusent des informations quant à l'interprétation et à l'application des mesures fonctionnelles;
- les experts eux-mêmes qui désirent échanger d'une façon formelle des informations quant à l'interprétation des règles de mesures fonctionnelles;
- toutes les personnes qui doivent appliquer la mesure fonctionnelles COSMIC-FFP à un moment ou à un autre du cycle de vie du logiciel.

Toutes ces catégories d'utilisateurs utilisent des expertises du passé pour interpréter les règles de la méthode de mesure fonctionnelle. Les règles fondamentales dans une méthode de mesure fonctionnelle sont relativement peu nombreuses, mais elles doivent se répéter dans des environnements variés. Par exemple, l'identification d'un processus repose sur la définition d'un processus, définition qui tient en quelques lignes. Cependant, il existe une grande variété de processus qu'il est possible de reconnaître à la condition d'avoir réalisé cet exercice avec un nombre suffisant d'environnements. À partir de là, il est non seulement possible de les reconnaître, mais aussi de structurer les processus compte tenu de l'environnement, ce qui est le rôle de l'expert.

4.2.2 Expertise disponible

Nous devons enfin nous interroger sur la capacité d'aller chercher les connaissances des experts et sur leur disponibilité

Il existe actuellement, à travers le monde, un certain nombre de documents appelés (incorrectement) "règles locales"(annexe B) ou "règles industrielles" qui aident le mesureur à interpréter les règles de la mesure fonctionnelle dans différents environnements et pour différents types de logiciels. Ces documents comprennent un certain nombre de "cas" qui sont interprétés par des "experts" du domaine en vue d'aider les mesureurs à appliquer de façon "cohérente" les règles de la mesure fonctionnelle. Un extrait d'un document de ce type se trouve à l'annexe B (en anglais). Ces documents peuvent appartenir à une organisation en particulier, à une firme conseil, à une association nationale ou internationale. Il peut aussi arriver qu'un expert en particulier construise ses propres "règles locales". Pour les fins de notre recherche, nous aurons accès à plusieurs de ces documents, ayant dans certains cas, participé à la construction de ces "règles locales".

De nos jours, le terme "expert" est utilisé assez largement et souvent à mauvais escient, surtout dans le domaine du conseil. Un certain nombre de critères devraient être retenus avant d'affirmer d'une personne qu'elle est un "expert" dans un domaine particulier:

- sa compréhension de la théorie relative à son domaine d'expertise et les domaines connexes (par des publications et des conférences)
- sa pratique (au moins 5 ans à plein temps)
- sa capacité à composer avec diverses situations et à appliquer les théories de son domaine (nombre d'organisations où la personne est intervenue)
- sa capacité à faire évoluer la théorie pour mieux répondre aux différentes situations (publications et conférences)
- la reconnaissance par les pairs (participation à des comités de pratique)

4.3 Justification de l'approche diagnostique

4.3.1 Qu'est-ce qu'un cas dans le cadre de notre problématique?

Dans le cadre de notre problématique il y a une distinction à faire entre un cas et un cas problème. La définition de cas et cas problème se retrouve dans la démarche.

Cas : plusieurs cas problèmes ayant une logique fonctionnelle du point de vue d'un utilisateur d'un logiciel. Un cas n'est pas synonyme de concept topologique.

Cas problème : une situation dans laquelle le mesureur doit utiliser un processus de diagnostic pour identifier correctement un concept de l'ontologie COSMIC-FFP (ex: groupe de données, processus).

Concept topologique: c'est le point d'intersection entre un concept du processus de développement et une partie de la tâche du mesureur. Autrement dit, c'est la mise en correspondance des concepts du processus de développement (dans le cas présent ceux de SWEBOK) et les concepts de la méthode de mesure fonctionnelle COSMIC-FFP. Un ensemble de cas problèmes est identifié pour chaque concept topologique.

Dans ce contexte, la recherche d'un "cas similaire" se trouve au niveau de la recherche du concept topologique approprié, mais nous sommes intéressés, à régler les cas problèmes. Nous allons donc utiliser le terme "cas problème".

4.3.2 Est-ce que l'application des règles de mesure mène à la résolution d'un cas problème?

Pour appliquer les règles de mesure, nous nous sommes basés sur la démarche du mesureur et nous avons établi les étapes de la méthode diagnostique :

- situer son problème par rapport à la méthode de mesure et le cycle de développement;
- trouver les cas problèmes qui sont les plus appropriés à la situation
- résoudre le cas problème
- appliquer la solution ou aller vers un autre cas problème.

Le mesureur commence donc à trouver les cas problèmes via la recherche du concept topologique approprié, puis utilise les autres étapes de la démarche pour les résoudre.

4.3.3 Comment peut-on les trouver?

Pour les experts, les connaissances proviennent de leur expérience, d'échanges avec d'autres experts (rencontres informelles ou rencontres formelles, tels les comités de pratiques), de la lecture des « règles locales » ou via des échanges par Internet (il existe un listserv sur Internet géré par un québécois via le site du CRIM).

4.3.4 Peuvent-ils servir à résoudre d'autres cas problèmes?

L'application de la mesure fonctionnelle est un acte répétitif, en ce sens, que le nombre de règles à appliquer est relativement restreint. Nous avons constaté, de plus, que la construction des logiciels est un acte aussi relativement répétitif à partir d'un certain nombre de concepts qu'il est possible et nécessaire de délimiter pour des fins d'apprentissage.

4.3.5 Information historique suffisante?

À partir de la documentation des "règles locales", il est possible de construire plusieurs centaines de cas problème. Cette documentation n'est pas la seule source, puisqu'il est aussi possible d'utiliser la documentation des logiciels pour construire de nouveaux cas problème.

4.4 Justification des approches quantitatives

4.4.1 Est-il possible de formuler des questions quantifiables pour résoudre les cas problème?

L'exemple d'application dans le chapitre précédent démontre cette possibilité.

4.4.2 Est-il possible d'utiliser des règles au niveau des cas problèmes?

L'exemple du chapitre précédent utilise des règles.

4.4.3 Est-ce que les cas problèmes peuvent être catégorisés et indexés?

Il est possible de catégoriser et indexer les cas problèmes. Dans l'explication de la démarche, nous utilisons le terme "concept topologique" pour catégoriser les cas problèmes. Pour chaque concept topologique on a :

- une définition
- sa relation avec un certain nombre de mots clefs
- sa relation avec un certain nombre de cas problèmes
- une explication de la raison du choix de ce concept.

Il y a donc une indexation basée sur le concept topologique qui est lié à des mots clefs qui conduisent aux cas problèmes.

4.5 Originalité de l'approche choisie

L'originalité de notre approche se trouve principalement dans la création d'une ontologie de l'application de la mesure (en créant les concepts topologiques) à partir du cycle de vie d'un logiciel et de la méthode de mesure fonctionnelle COSMIC-FFP. La création de cas problème associés aux concepts topologiques et la création de règles de résolution de problèmes, en font aussi partie. Autrement dit, notre originalité est au niveau de l'utilisation d'une approche cognitive hybride pour permettre aux mesureurs l'application des règles de mesures COSMIC-FFP. Elle se retrouve aussi dans l'analyse des résultats de nos expériences auprès des mesureurs. Il n'existe pas, à date, d'analyse par processus des résultats de la mesure fonctionnelle des logiciels.

5. MÉTHODOLOGIE DE RECHERCHE

5.1 Validation du détail du design par des experts

Nous avons défini dans le chapitre 3 la démarche du mesureur. Nous avons aussi fourni un exemple de règle d'inférences. Le détail du design constitue l'ensemble des règles d'inférences qui seront inscrites dans la démarche du mesureur. La méthode Delphi nous semble la plus appropriée pour la tâche de validation du détail du design par des experts c'est-à-dire du contenu.

5.1.1 Buts et étapes de la méthode

L'idée à la base de la méthode Delphi est que l'on peut obtenir, avec un jugement le moins biaisé possible, des opinions sur une méthode, des règles et une expertise. Pour ce faire, il est nécessaire de suive une procédure gérée par un coordonnateur.

5.1.2 Activités du coordonnateur

Les activités sont:

- Sélectionner un certain nombre d'experts
- Transmettre un certain nombre de règles d'inférences aux experts
- Inscrire les résultats de façon factuelle
- Retourner des résultats aux experts sans mentionner qui a répondu quoi
- Inscrire à nouveau les résultats de façon factuelle
- Continuer jusqu'à ce qu'il y ait un consensus

5.1.3 Avantages et inconvénients de cette approche

Avantages:

- L'élimination des rencontres en groupe
- Le participant peut fonctionner à son rythme
- Les biais inhérents à ces rencontres en groupe[39] peuvent être évités
- Les participants peuvent changer d'opinion sans « perdre la face »
- Les coûts sont peu élevés

Inconvénients :

- On peut prendre beaucoup de temps
- On peut perdre des participants en cours de route
- Il peut être plus difficile d'expliquer les buts visés

[39] Position dominante de certains participants, magnétisme personnel, avantage de la langue, expertise non justifiée, etc.

5.1.4 La méthode Delphi et notre projet de thèse

La méthode Delphi peut être utilisée pour vérifier auprès des experts de la mesure fonctionnelle les règles d'inférences appliquées à la base de connaissances du prototype que nous allons construire. Nous aurons déjà, en tant qu'expert, réalisé un premier design du prototype.

Les experts devront vérifier :

- si le niveau de confiance « d'avoir le cas problème X si on a identifié un concept topologique Y » est acceptable
- si le niveau de confiance de la résolution d'un cas problème X est pertinent avec les questions a, b, c
- si les pourcentages de "croyance" associés aux questions a, b, c permettent la résolution du cas problème X selon ces experts
- si le niveau de confiance des recommandations W par rapport aux résultats e, f et g est pertinent
- si les pourcentages de « croyance » aux recommandations h, i et j sont acceptables.

La collecte des informations pourra se faire progressivement. Nous commencerons, par exemple, par les problèmes associés aux concepts topologiques, pour poursuivre avec les deux autres étapes.

Nous croyons que le processus de consultation devrait prendre tout au plus trois mois pour les trois étapes.

Nous aurons de la difficulté à trouver plus de 5 experts que nous jugeons en mesure de faire cet exercice (c'est-à-dire avec assez d'expérience et de connaissances) et qui pourront en plus prendre le temps de le faire correctement. Ces experts, en plus d'être dispersés à travers le monde, sont généralement très occupés. Nous bénéficions ainsi de l'avantage d'éliminer les rencontres de groupe. De plus, le nombre d'experts étant limité, nous diminuerons les risques liés à la compréhension des buts de l'exercice.

Les règles d'inférence et la base de connaissances devraient évoluer dans le temps. Nous ne prétendons pas que toutes les connaissances des experts en mesure fonctionnelle auront été accumulées dans la base de connaissances. Nous voulons avant tout fournir une base suffisante pour les cas prévus dans notre expérimentation.

5.2 Plan d'expérimentation de la méthode

Nous faisons l'hypothèse que les mesureurs qui utiliseront notre méthode pour mesurer un logiciel à partir de sa documentation, sur la base de la méthode de mesure fonctionnelle COSMIC-FFP, auront de meilleurs résultats[40] que ceux qui utilisent seulement le guide de mesure COSMIC-FFP. Bien que cet objectif soit important, ce n'est pas le seul. Nous croyons aussi qu'un prototype peut être notre outil d'implantation de la méthode[41]. Il peut servir à suivre le raisonnement des novices et ainsi améliorer la clarté du contenu de la base de connaissances, entre autre, les cas problèmes et les questions. Ce prototype est aussi un compendium de la pensée des experts qui appliquent régulièrement la méthode de mesure fonctionnelle COSMIC-FFP.

---

[40] La référence est le résultat fourni pas les experts pour les cas prévus.

[41] Nous avons utilisé un logiciel déjà sur le marché pour une première expérimentation. Ce logiciel est décrit à l'annexe A.

5.2.1 Étapes de l'expérimentation

Voici les étapes de l'expérimentation

- choix de deux groupes de mesureurs, un groupe qui utilisera le prototype et un groupe qui n'aura que le guide de mesure COSMIC-FFP
- formation sommaire à la mesure COSMIC-FFP pour les deux groupes
- mesure de deux logiciels (ou partie de logiciels) par les mesureurs
- analyse des résultats

Nous prévoyons réaliser cette expérimentation deux fois. Une fois avant la vérification des règles auprès des experts (voir méthode Delphi) et une autre fois après cette vérification. La première expérimentation nous renseignera sur l'utilité du prototype pour un novice et le mode de fonctionnement d'un novice lors de la mesure. La deuxième expérimentation devrait nous permettre de vérifier s'il y a une amélioration des résultats avec une interface améliorée et des règles vérifiées par les experts.

5.2.2 Choix de deux groupes de mesureurs

Les mesureurs choisis devront avoir les caractéristiques suivantes :

- ne pas avoir d'expérience (ou très peu) dans la mesure fonctionnelle, aussi bien celle de COSMIC-FFP que les autres mesures fonctionnelles sur le marché
- si possible, ne pas avoir déjà eu de formation COSMIC-FFP avant leur choix pour l'expérience
- avoir une bonne connaissance du génie logiciel, au moins théorique

5.2.3 Composition des groupes

Les groupes devraient avoir les caractéristiques suivantes :

- le nombre idéal pour chaque groupe devrait être entre 5 et 10 mesureurs
- les mesureurs ne sauront à quel groupe ils appartiennent avant le début de la mesure
- la méthode aléatoire pour le choix des mesureurs ne sera connue qu'au début de la séance de mesure (ex : choix par ordre alphabétique des noms des personnes)
- un questionnaire sur la scolarité et l'expérience des mesureurs sera aussi rempli.

5.2.4 La formation

Caractéristiques :

- la formation sera dispensée aux deux groupes en même temps
- sa durée sera d'environ trois heures
- le but de la formation est de permettre aux participants de comprendre : la démarche, le vocabulaire et les principes COSMIC-FFP
- nous assumons que les mesureurs connaissent déjà la démarche et le vocabulaire du processus de développement du logiciel (liés à la bonne connaissance du génie logiciel)
- la formation sera dispensée par un expert de la mesure fonctionnelle
- cette formation a déjà été dispensée auprès d'un certain nombre d'entreprises, c'est donc une formation déjà rôdée
- il y aura une formation sur le logiciel (FFPXpert[42]) aux deux groupes

[42] C'est un logiciel que nous allons construire.

5.2.5 Mesure des logiciels

Les mesureurs n'auront pas à mesurer un logiciel complet, mais seulement un certain nombre de fonctionnalités.

5.2.6 Caractéristiques des logiciels

Voici les caractéristiques :

- les logiciels choisis ont des fonctionnalités qui peuvent être comprises facilement par tous
- les logiciels seront fournis par une source indépendante et non connus avant le déroulement de la mesure
- des informations floues sur les fonctionnalités du logiciel pourront être fournies, c'est-à-dire que le mesureur devra interpréter l'application des règles

5.2.7 Déroulement de la mesure

Voici le déroulement du processus de mesure

- la mesure devra se dérouler dans un délai de moins de 15 jours après la formation
- tous les mesureurs auront les mêmes mesures à réaliser
- la durée sera la même (le but n'est pas de demander de réaliser le tout de la façon la plus rapide possible, mais le mieux possible)
- nous pensons à une durée de trois heures, mais ceci peut être réévalué selon les besoins
- chaque mesureur faisant partie du groupe ayant accès au prototype aura le Guide et un ordinateur à sa disposition
- chaque mesureur, faisant partie du groupe ayant accès seulement au Guide du mesureur, aura un document papier du Guide de mesure COSMIC-FFP. Ce document aura été distribué lors de la formation. Par contre, il aura été remis au professeur à la fin de la formation.

5.2.8 Analyse des résultats

Le but de l'analyse est de déterminer si les résultats sont meilleurs pour le groupe ayant accès à l'utilisation du prototype que pour le groupe n'ayant que le guide du mesureur. Nous pourrons aussi analyser le cheminement des mesureurs pour le groupe ayant accès au prototype.

Les meilleurs résultats seront ceux qui seront le plus prêt des résultats des experts. Pour un des cas, plusieurs experts se sont prononcés sur la façon de le mesurer et sont arrivés à un consensus. Les analyses se feront par processus en comparant les mesureurs individuellement et par groupe.

Des analyses complémentaires pourront être faites sur la base de la scolarité et de l'expérience des mesureurs.

Enfin, nous ferons des analyses sur le cheminement des mesureurs ayant utilisé le prototype pour solutionner les cas problèmes. Ces analyses se feront à partir des informations fournies par la base de données du prototype.

Selon les disponibilités des mesureurs, il pourrait aussi être intéressant d'obtenir des commentaires sur les difficultés rencontrées.

6. PLAN DE RÉALISATION

6.1 Principaux livrables

Les principaux livrables sont:

- Les ontologies du cycle de vie du logiciel et de la mesure fonctionnelle COSMIC-FFP
- Les concepts topologiques, les cas problèmes et les règles de résolution des cas problème
- La consultation auprès des experts de la mesure fonctionnelle
- La création des concepts topologiques, cas problèmes et règles d'inférences pour deux cas d'étude connus (à venir)
- Les prototypes de diagnostic (préliminaire et accepté par les experts)
- L'expérimentation auprès de mesureurs novices
- La rédaction du rapport de thèse

Il y a aussi un cours de formation COSMIC-FFP qui sera utilisé pour l'expérimentation auprès des mesureurs. Pour des raisons de droit d'auteur, ce cours ne fera pas partie des livrables de la thèse.

6.2 État des travaux

L'état des travaux est le suivant:

- Ontologies: les ontologies (ainsi qu'une définition de tous les concepts) ont été réalisées. Il reste à valider ces définitions et s'assurer qu'elles sont toutes pertinentes.
- Concepts topologiques: les concepts topologiques ont été identifiés et un certain nombre de concepts topologiques ont été définis. Leur validation se fera à la suite de la validation des ontologies.
- Cas problèmes: un certain nombre de cas problèmes ont été identifiés en lien avec les concepts topologiques. La majeure partie du travail d'identification reste à faire.
- Règles de résolution des cas problème: pour les cas problèmes identifiés nous avons vérifié la faisabilité et la pertinence des règles COSMIC-FFP.
- Consultation des experts: nous avons identifié cinq experts et déjà demandé à trois d'entre eux leur collaboration. Il est possible que d'autres experts se joignent au groupe. Ce sont tous des bénévoles.
- Prototype de diagnostic: l'analyse et le design du prototype ont été réalisés et la programmation est en cours.
- Expérimentation auprès des mesureurs novices: nous avons identifié les mesureurs novices cibles et une première expérimentation pourra se faire dès qu'une bonne partie des travaux précédents seront complétés. Pour une première expérimentation, nous ne croyons pas nécessaire d'avoir terminé la consultation des experts.
- Rédaction de la thèse: les revues de littérature (mesure et cognitif) sont en bonne partie complétées (ce qui n'exclut pas des ajouts dont entre autre des études de fiabilité des systèmes experts) et la démarche est définie. Elle pourra être finalisée lorsque les travaux précédents seront complétés.

7. CONCLUSION

Nous avons réalisé une revue de littérature de la mesure fonctionnelle et avons conclu qu'il n'existait pas de méthodes systématiques et reconnues pour réaliser la mise en correspondance des règles formelles avec les différents environnements de développement (monde empirique). De plus, les connaissances des experts en mesure, connaissances acquises après plusieurs années de pratique, reflètent deux ensembles d'habiletés ou de connaissances nécessaires au mesureur:

- La compréhension des artefacts du logiciel pour situer le problème (ou cas problèmes) à résoudre
- La compréhension des règles de mesure pour les appliquer aux cas problèmes à résoudre

Notre objectif de recherche est donc d'encapsuler et de structurer la démarche des experts pour permettre l'obtention de résultats cohérents (consistent) et exacts (accuracy) sans avoir recours à des experts.

À cette fin, nous avons réalisé une revue de littérature en informatique cognitive où nous avons conclu que l'approche cognitive devrait nous permettre de modéliser les connaissances des experts en mesure COSMIC-FFP et d'appliquer les méthodes (ex: CBR) et les algorithmes qui permettront de résoudre notre problématique.

À partir de la démarche du mesureur, nous avons fait une proposition d'une approche diagnostique qui utilisera les raisonnements à base de cas et les raisonnements à base de règles. Nous avons justifié cette approche dans le chapitre 4. Les deux derniers chapitres traitent de notre méthodologie de recherche et du plan de réalisation de la recherche.

8. BIBLIOGRAPHIE

1. [AAM94] Aamodt, A., and Plaza, E. (1994), Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches, AI Communications, 7.
2. [ABR01] Abran, A., Desharnais J.-M., Oligny S., St-Pierre D., Symons C., Measurement Manual, Version 2.1, April 2001.
3. [ABR01a] Abran, A., Symons C., COSMIC, The 2nd generation of Functional size measurement methods, ESCOM, London, April, 2001
4. [ABR01b] Abran, A., Moore, J.W., Bourque, P., Dupuis, R., Tripp, L., Guide for the Software Engineering Body of Knowledge, A project of the IEEE Computer Society, Software Engineering Coordination Committee. Adresse URL: http://www.swebok.org.
5. [ABR98] Abran, A., Jacquet J.-P., Dupuis R., Une analyse structurée des méthodes de validation des métriques, Laboratoire de recherche en gestion des logiciels, Département d'informatique UQAM, 1998.
6. [ABR94] Abran, A., 1994, Analyse du processus de mesure des points de fonction, thèse de doctorat, Montréal, École polytechnique, 324 p.
7. [ALB84] Albrecht, A. J., IBM CIS&A Guideline 313, AD/M Productivity Measurement and Estimate Validation, IBM, 1984.
8. [BOE81] Boehm, B., Software Engineering Economics, Prentice Hall, 1981, p. 767.
9. [BUC83] Buchanan, B.G., Barstow, D., Bechtel, R., Bennet, J., Clancey, W., Kulikowski, C., Mitchell, T. M., Waterman, D.A. (1983) Construction an Expert System, in Building Expert Systems, Chapter 5, Reading, MA: Addison-Wesley.
10. [BUD94] Budgen, D., *Software Design,* Addison-Wesley, 1994.
11. [DEH82] Dehn, N. and Schank, R., Artificial and Human Intelligence, Handbook on Human Intelligence, Robert J. Stenberg, 1982, pp. 352-391.
12. [DEM82] DeMarco, T., Controlling Software Projects: Management, Measurement, and Estimation, Yourdon Press, New York, 1982.
13. [DES01a] Desharnais J.-M., Westwood P., Rollo T., The Strengths and Weaknesses of Functional Measure as a Good Negotiation Tool for New Development Projects, ESCOM Conference, April 2001.
14. [DES01b] Desharnais, J.-M., Abran A., Applying a Functional Measurement Method: Cognitive Issues, International Workshop on Software Measurement, Montreal, August 2001.
15. [DES00] Desharnais, J.-M., Description sommaire de Help!CPR et Help FFP, document produit à l'occasion du cours DIC 9200 pour Alain Abran, été 2000.
16. [DES98] Desharnais, J.-M.; Morris, P., Comparison between FPA and FFP: a field experience, in 8th International Workshop on Software Measurement, Magdeburg, Germany, September 17-18, 1998.
17. [DES96] Desharnais, J-M., Morris P., Post Measurement Validation Procedure for Function Point Counts*,* Position Paper Forum on Software Engineering Standards Issues, October 1996.

18. [DES93] Desharnais, J.M., Validation Process for Industry Benchmarking Data, Software Engineering Laboratory in Applied Metrics, Invited Paper, IEEE Conference on Software Maintenance, sept.-oct 1993, Montréal.
19. [DUR93] Durkin, John, Expert system: Design and Development, Macmillan Publishing Company, 1993
20. [FEN91] Fenton, N., Software Metrics: a rigorous approach, Englewood, 1991.
21. [IFPUG99] Function Point Counting Practices Manual, International Function Point Users Group, version 4.1, 1999
22. [ISO93] ISO, Vocabulaire International des Termes Fondamentaux et Généraux de Métrologie, Deuxième édition, 1993, ISBN 92-67-01075-l
23. [ISO97] ISO/IEC 14143-1:1997 - Information technology - Software measurement - Functional size measurement - Definition of concepts.
24. ISO/IEC 14143-3:1997 – Software engineering – Software measurement – Functional size measurement – Part 3: Verification of functional size measurement methods.
25. [ISO01] ISO TC JTC1/SC SC7/WG 12, Software Engineering — COSMIC-FFP Functional size measurement method, Document type: International Standard, 2001.
26. [JAC98] Jackson Peter, Introduction to Expert Systems, Addison Wesley, Third Edition, 1998, 542 pages.
27. [KEM90] Kemerer, C.F., Function Point Measurement Reliability: A Field Experiment, IFPUG Fall Conference, San Antonio, Texas, October 1990.
28. [KOL93] Kolodner, J.L., Case-Based Reasoning, San Francisco: Morgan Kaufmann Publishers, 1993.
29. [LAU87¨Laurière, J.-L., Intelligence artificielle, résolution de problèmes par l'homme et la machine, Eyrolles, 1987.
30. [LEJ67] Lejewski, C. "Jan Lukasiewicz," Encyclopedia of Philosophy, Vol. 5, MacMillan, NY: 1967, pp. 104-107.
31. [LOW90] Low, G.C. et Jefferey D.R., Function Points in the Estimation and Evaluation of the Software Process, IEEE Transaction on Software Engineering, Vol. 16, no 1, Jan. 1990, 64-71.
32. [LUG00] Luger,G. F., Artificial Intelligence, Structures and Strategies for Complex Problem Solving, Fourth Edition, Addison-Wesley, Fourth edition published 2002
33. [MOR98] Morris, P., Desharnais, J.-M., Measuring ALL the Software not just what the Business Uses, IFPUG, 1998.
34. [NIS99] Nishiyama, Shigeru, On Precision on Function Point Analysis, Research and Development Center, Nippon Telegraph and Telephone, Japan, 1999.
35. [NIS94] Nishiyama, S. , Furuyama, T., The validity and applicability of function point analysis, EOQ-SC'94, Basel, Switzerland.
36. [PAQ89] Paquette, Gilbert, L'intelligence artificielle, comprendre et prolonger l'intelligence humaine, textes écrits et recueillis par Gilbert Paquette avec la collaboration de Anne Bergeron, Télé-université, 1989, 718p.
37. [PRI96] Prince, Violaine, Vers une informatique cognitive dans les entreprises: Le rôle central du langage, Masson, Paris, 1996, 190 p.

38. Rudolph, E.E., Precision of Function Point Counts, IFPUG Spring Conference, San Diego, CA, April 1989.

39. [RUS95] Russell S., Norvig P. (1995): *Artificial Intelligence: A Modern Approach*. Prentice-Hall.

40. [SCH99] Schreiber, G., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., Van de Velde, W., Wielinga, B., Knowledge Engineering and Management, The CommonKADS Methodology, The MIT Press, 1999.

41. [SYM91] Symons Charles R., Software Sizing and Estimating Mark II FPA, Wiley Series in Software Engineering Practice, 1991.

42. [SYM88], Symons C. R.,"Function Point Analysis: Difficulties and Improvements", IEEE Transactions on Software Engineering, Vol. 14, no 1, Janvier 1988.

43. [VAN97] van Heijst,G., Schreiber, A.Th., Wielinga, B.J., Using Explicit Ontologies in KBS Development, University of Amsterdam, Department of Social Science Informatics, Roetersstraat 15, NL-1018 WB, Amsterdam, 1997.

44. [WAT97] Watson, I., Applying Case-Based Reasoning: Technique for Enterprise Systems, AI-CBR, University of Salford, UK, 1997, 289 p.

45. [MIT99] The MIT Encyclopedia of the Cognitive Sciences, edited by Robert A. Wilson and Frank C. Keil, The MIT Press, Cambridge, Massachusetts, 1999, 964 p.

Référence sur le WEB

Grand Dictionnaire Terminologique de l'Office de la Langue Française, http://www.granddictionnaire.com/_fs_global_01.

Annexe A

Description de Help!CPR

1. DESCRIPTION FONCTIONNELLE DE HELP!CPR (UTILISATEUR)

Help!CPR a un menu permettant d'ouvrir et de sauvegarder plusieurs bases de données en format "Microsoft Access". Help!CPR organise l'information sur les cas en trois objets distincts: les problèmes, les questions et les actions. Un quatrième objet pourrait être le "query" ou les mots clefs. Les cas sont créés en interconnectant ces objets ensembles. Il n'est pas nécessaire d'avoir l'objet "action" pour la résolution de cas, par contre les autres "objets" sont essentiels.

Il est possible d'assigner (via l'interface de l'expert[43]) une information au niveau des "problèmes" (objet "problems") ou des actions (objet "actions"), mais pas au niveau des questions (objet "questions"). Cette assignation est l'équivalent d'une référence hypertexte.

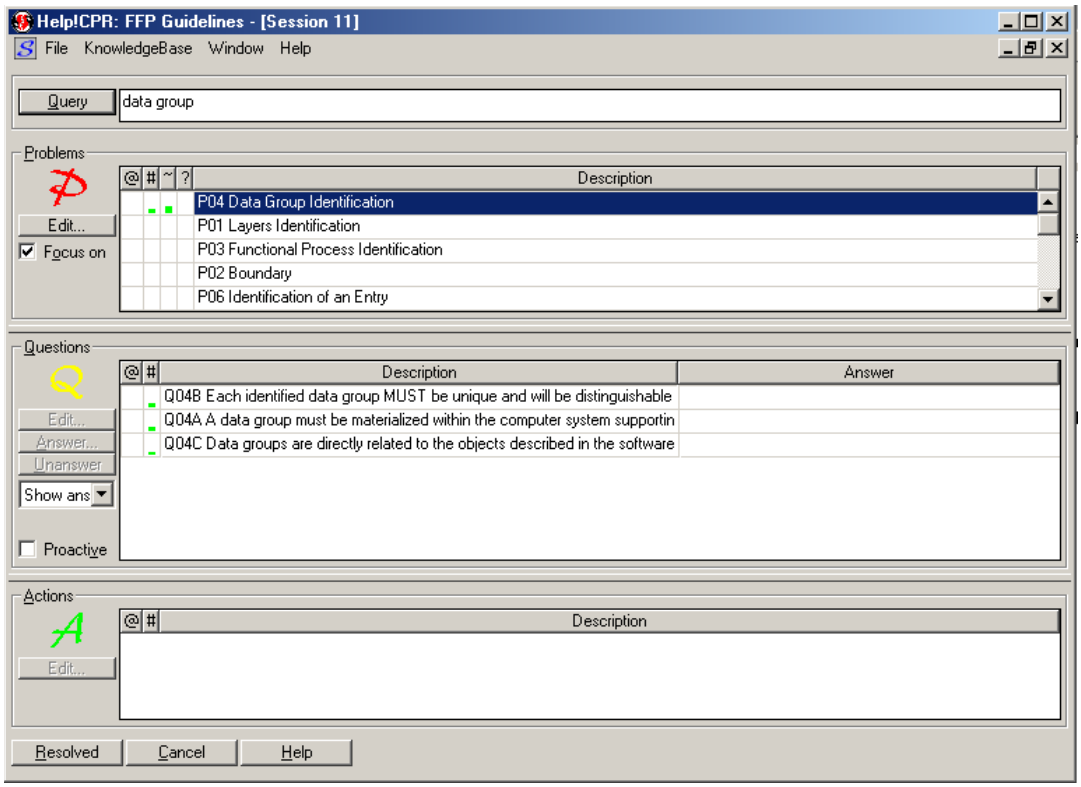

Figure 15 Interface utilisateur de Help!CPR

Il est possible de choisir un problème particulier en utilisant la fonction ou l'objet « Query » (mot clef). Help!CPR met en relation le problème identifié et les cas se trouvant dans la base de cas. Il suggère alors un ou des problèmes à solutionner. Par exemple, en entrant le terme « data group » dans l'objet "Query" on retrouve dans l'objet "problems", le problème relatif au « data group ».

Automatiquement, dans l'objet "questions", les questions relatives à ce problème apparaissent. Le novice doit alors entrer une réponse. Ces réponses donnent une solution possible au problème. La couleur verte à gauche indique une réponse positive, alors que la couleur rouge indique une réponse négative. Il est aussi possible de suggérer une action selon la nature des réponses aux questions. Ici l'action pourrait être

[43] Nous ne présentons pas les interfaces de l'expert de Help!CPR, ce qui demanderait une trop longue explication non nécessaire pour les fins de ce document.

d'aller vers d'autres « mots clefs » pour compléter la connaissance. À noter aussi le numéro de session. Chaque session est numérotée et suivie.

2. DESCRIPTION FONCTIONNELLE DE HELP!CPR (EXPERT)

Ce logiciel a inspiré notre démarche cognitive au départ.

2.1 Les problèmes

La figure 15 montre l'interface expert qui permet de poser des questions et aussi l'interface des questions. Le problème peut apparaître sous forme d'énoncés ou de questions. À chaque problème est associé un ensemble de questions (voir questions plus bas). Il est aussi possible d'introduire des commentaires à chaque problème en éditant un problème particulier (voir EDIT).

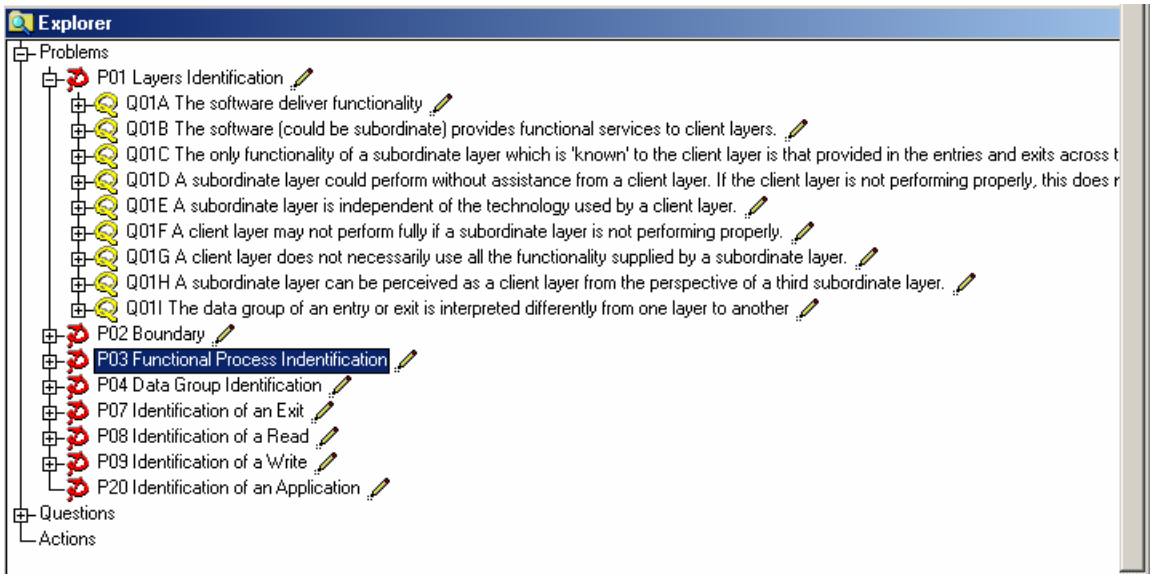

Figure 16 Interface expert

La figure 16 montre les questions liées au problème d'identification du processus unique.

Il est aussi possible d'entrer des commentaires.

2.2 Les questions

La figure 17 montre la section "questions" d'une session. À une réponse à une question est associé un pourcentage. Ici, à la question : "Do we have batch and on line function?", la réponse "Yes" est associée à un pourcentage d'influence de la réponse à la résolution du problème selon l'expert. Ces influences sont ensuite additionnées. Il est aussi possible d'introduire des commentaires à chaque question en éditant une question particulière (voir EDIT).

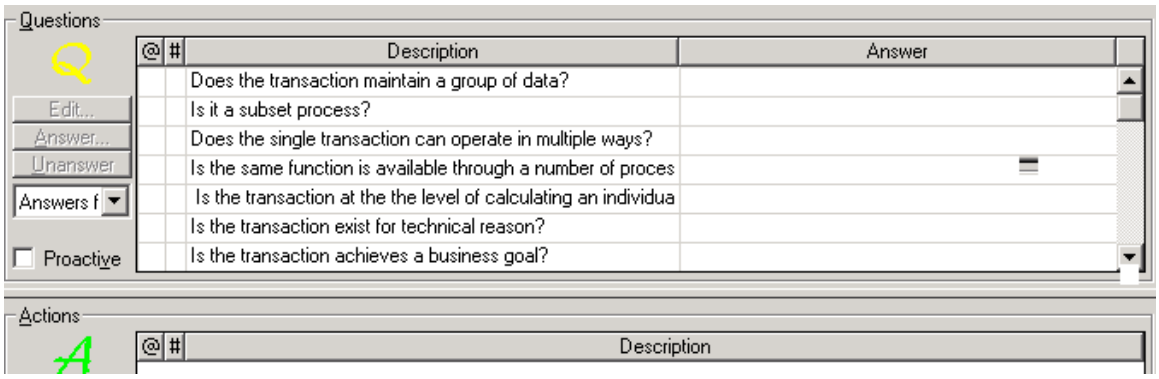

Figure 17 Questions

La figure 18 montre comment est traitée une question par l'expert.

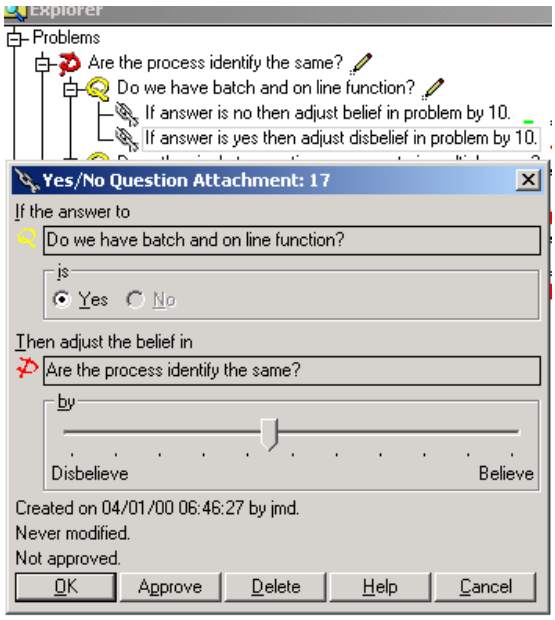

Figure 18 Traitement d'une question par l'expert

Il est possible à l'expert de noter un pourcentage sur la valeur de la réponse, ici appelée croyance positive ou négative. Help!CPR permet des réponses aux questions avec des choix multiples et avec des nombres. Ces options ne sont pas montrées à l'écran. Dans le contexte de COSMIC-FFP ces options ne sont pas utilisées, sur la base des expériences que nous avons réalisées à date. Nous ne rejetons pas cependant définitivement ces options, mais il nous faudra des expérimentations plus poussées pour les valider.

2.3 Les actions

La section action est reliée aux problèmes. Dans le modèle de diagnostic, nous utiliserons l'action pour inciter un novice à regarder d'autres problèmes apparaissant normalement avec le type de problème posé. Les exemples d'action de Help!CPR ne sont pas de cette nature : lors du diagnostic d'une imprimante, on suggérera comme action de vérifier la connexion par exemple. Notre intention est de pouvoir associer d'autres problèmes au problème du novice (les mots clefs permettent les réponses aux questions à d'autres problèmes). Ce serait une forme d'apprentissage guidé en tenant compte du contexte des réponses aux questions.

2.4 Les mots

Il est aussi possible, pour un utilisateur, de voir la liste des mots clés qu'il peut utiliser pour résoudre un problème. Il y a trois catégories de mots (figure 19) : les mots clefs, les mots qu'il faut ignorer et les mots non classifiés. La classification des mots devrait se faire par l'expert dans le mode sélection. Il suffit de sélectionner un mot dans la liste et d'indiquer s'il s'agit :

- d'un mot clef ou
- d'un mot à ignorer

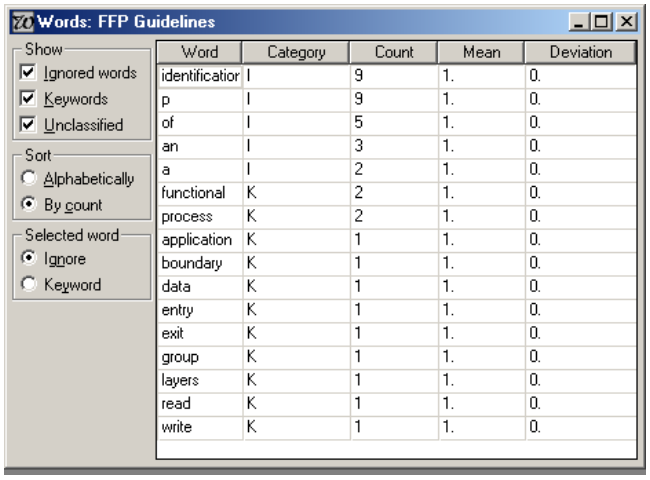

Figure 19 Les mots

Les mots peuvent apparaître par ordre alphabétique ou selon leur nombre d'occurrences. Il est aussi possible de choisir s'il faut montrer une (ou des) catégorie(s) de mots.

La somme des mots indique combien de fois un mot apparaît au moins une fois dans la description d'un problème ou dans la documentation. La moyenne fait référence au nombre moyen de fois qu'un mot apparaît au moins une fois dans la chaîne de caractères de la description d'un problème ou dans la documentation. La déviation réfère à la déviation à partir de la moyenne.

La table des mots (figure 20) est extraite des descriptions des problèmes et de leur documentation. Il est possible de réindexer cette table à tout moment.

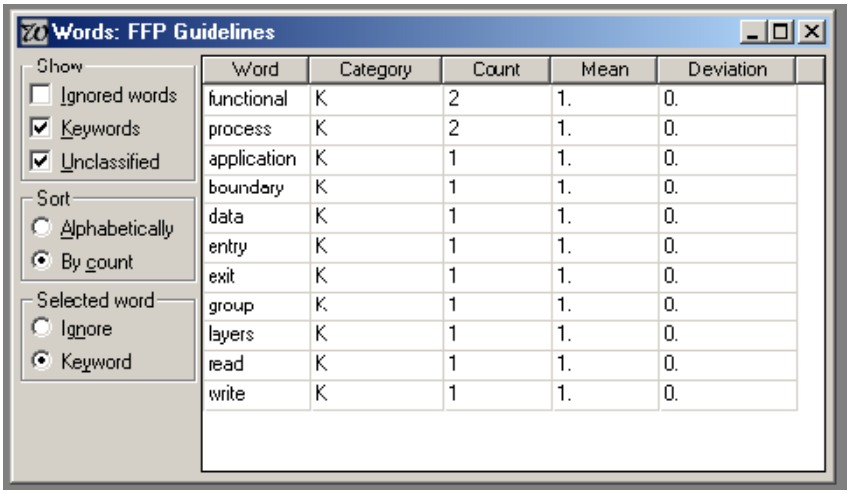

Figure 20 Les mots (suite)

Annexe B

Exemples de "règles locales[44]"

[44] Nous avons laissé les règles locales en anglais puisque c'est la langue d'échange entre les experts de la mesure fonctionnelle.

1. Introduction

1.1. Objective of this document

Present a cohesive set of local rules on how to apply COSMIC 2.1.

1.2. Reasoning behind the Local Rules

FFP is a measure standardized by the international organization COSMIC. While most counting rules are described in the COSMIC FFP measurement manual, some rules must be clarified to determine how to apply them. In some cases, local rules make the counting process more efficient.

1.3. Target Audience

This document is targeted to measurement specialists that already have some training and COSMIC FFP experience. It is structured as a reference book, so it is intended to be used as such rather than a training material. Use the table of contents and the index to locate a specific counting issue.

1.4. How to find your way in this document

This document presents measurement issues grouped by topics (e.g. boundaries, layers, group of data). The table of contents and the index can be used to find a specific detailed topic. if you read this document on screen rather than on paper, you can search for a topic using the "Find" function of Microsoft Word. If you do this be aware that specific topic titles are followed by "&" in hidden (e.g. trigger &) characters in order to facilitate searching. Also, navigation can be facilitated by using the hypertext links.

In this document, extracts from the COSMIC FFP manual are identified in italic characters.

2. General Rules

2.1 Purpose of the count

The purpose of the count is the reason why the count is done. For example, it can be to estimate the cost of a project, or to size an existing application. These are the two main count purposes: Estimation and Application. You will find in this document different ways of applying FFP local rules for Estimation counts and Application counts. For example, section 2.6 is about Boundaries and Scope of Estimation counts and section 2.7 is about Boundaries and Scope of Application counts.

2.2 Boundary

Boundary definition:

"The boundary of a piece of software is the conceptual frontier between this piece and the environment in which it operates, as it is perceived externally from the perspective of its users. The boundary allows the measurer to distinguish, without ambiguity, what is included inside the measured software from what is part of the measured software's operating environment." COSMIC FFP v2.1 Measurement Manual.

3. COSMIC-FFP Processes

Here is the definition of a COSMIC-FFP Process:

"A functional process is a unique and ordered set of data movements (entry, exit, read, write) implementing a cohesive set of Functional User Requirements. It is triggered by an event and, once performed, must leave the software in a coherent state with respect to the triggering event." COSMIC FFP v2.1 Measurement Manual

Here are additional guidelines to identify FFP process:

On completion of the FFP process the user can exit the application without corrupting their data or leaving their business process incomplete.

The transaction achieves a "business goal" and often have an equivalent in a manual system.

The transaction does not exist for technical or implementation reason.

The transaction completes a single unit of work.

The transaction can be triggered and complete its processing independently, that is, it does not have a direct synchronous link to other functions.

Subset processes are not process on their own. Here are examples of subset processes:

Report which can optionally print or hide fields.

A single transaction which can operate in multiple ways (e.g.: depending on the contents of fields, other fields become optional or mandatory).

An inquiry with multiple selection criteria.

Hint: Sequences of events which end with an "OK" button to save or output data (not "OK" to move to another windows only) are often FFP processes.

Note on FFP processes: take care not to decompose the functionality beyond the lowest level. For example, the level of calculating individual field, is too low. On the other hand, take care not to decompose enough. For example, Manage New Employee must be decomposed if there are different types of activity: Create a new employee, Change an employee, Delete an employee.

Trigger &: (note: italic text is text extracted from the official FFP Measurement Manual)

Triggering event (-type): A triggering event occurs outside the boundary of the measured software and initiates one or more functional processes. Clock and timing events can be triggering events. Since each identified layer is separated by a boundary, triggering events can occur in one layer and initiate functional processes belonging to another layer.

Functional process (-type) (Synonym 'Transaction-type'): A functional process is a unique set of data movements (entry, exit, read, write) implementing a cohesive and logically indivisible set of Functional User Requirements. It is **triggered directly, or indirectly via** an 'actor', by an Event (-type) and is complete when it has executed all that is required to be done in response to the triggering Event (-type).

Functional process principles

a) *A functional process is derived from at least one identifiable Functional User Requirement,*
b) *A functional process is performed when an identifiable **triggering event** occurs,*
c) *A functional process contains at least two data movements, an entry and an exit or a write,*
d) *A functional process contains no more than one self-induced wait state (which may occur when it is completed),*
e) *A functional process belongs to one, and only one, layer.*

Boundary rule

Start by identifying triggering events, then identify the functional processes enabled by those events. The boundary lies between the triggering events and those functions.

Functional process rules

Subsets of triggering events are not considered different triggering events.

> *For instance, if a specific event occurrence triggers the entry of a data group comprising data attributes A, B and C, and then another occurrence of the same event-type, triggers an entry of a data group which has values for attributes A and B only, this is not considered a different triggering event-type. It is considered to be the same for the purpose of identifying COSMIC-FFP functional processes. Consequently, only one entry and one functional process are identified, manipulating data attributes A, B and C.*
> *In the context of real-time software, a functional process is also triggered by an event. It terminates when a point of asynchronous timing is reached. A point of asynchronous timing is reached when, in a sequence of data movements, a given data movement is not synchronized with the one preceding it. A point of asynchronous timing is equivalent to a self induced wait state.*

Entry rules

Clock-triggered events are considered external. Therefore, an event occurring every 3 seconds is associated with an ENTRY moving one data attribute, for instance. However, the functional process that generates the event periodically is ignored since it occurs, by definition, outside of the software boundary.

Minor variations do not justify different FFP processes, even if they have different external triggers. As a variation of the previous example, if there are separate menu items for creating Permanent Employees and Temporary Employees, then there is still only one process.

Polling

In some circumstances we can contrive that the events of interest in the external world directly generate messages which form the input side of a logical transaction. In other circumstances, a software application must periodically inspect (c'est-à-dire "poll") the status of the external world to determine whether an event of interest has occurred, generating an input message to document a positive result. In either case, the resulting message is regarded as Entry (the detection mechanism is purely implementation detail). The polling mechanism is, of course, triggered by an event in the external world c'est-à-dire the passage of a specified amount of time