UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE
APPLIQUÉES

PAR
ALIOUNE BADARA GUEYE

COHÉSION DES CLASSES DANS LES SYSTÈMES ORIENTÉS OBJET : ÉTUDE EXPÉRIMENTALE ET VALIDATION

TROIS-RIVIÈRES, JUILLET 2007

COHÉSION DES CLASSES DANS LES SYSTÈMES ORIENTÉS OBJET : ÉTUDE EXPÉRIMENTALE ET VALIDATION

Alioune Badara GUEYE

SOMMAIRE

Un large nombre de métriques ont été proposées pour mesurer les propriétés des systèmes orientés objet comme la taille, le couplage ou la cohésion. Par ailleurs, la validation de la changeabilité des systèmes logiciels est un point majeur de plusieurs recherches en génie logiciel. Une manière d'approcher le problème est d'investiguer la dépendance entre la changeabilité des systèmes logiciels et leur architecture avec pour but de trouver les métriques caractérisant les propriétés architecturales des systèmes qui peuvent être utilisées comme indicateurs de changeabilité.

Dans le domaine des systèmes orientés objet, plusieurs expérimentations ont été menées montrant que la taille et le couplage entre classes, par exemple, représentaient de bons indicateurs de changements contrairement aux métriques de cohésion utilisées.

Dans cette recherche, nous tenterons d'explorer si la cohésion, à travers plusieurs métriques, est corrélée avec la changeabilité. Il s'agira également de déterminer, si l'hypothèse est vérifiée, parmi les métriques de cohésion utilisées laquelle représente le meilleur indicateur. Nous avons retenu plusieurs métriques de cohésion parmi celles proposées dans la littérature. Nous nous sommes également intéressés à étendre, dans un but de raffinement, certaines d'entre elles. Nous avons également considéré, dans le cadre de notre recherche, le couplage et la taille des systèmes logiciels orientés objet. Nous avons mené une étude expérimentale d'envergure, sur plusieurs systèmes. Les différentes étapes de l'expérimentation ainsi que les résultats obtenus sont discutés dans ce mémoire.

CLASS COHESION IN OBJECT-ORIENTED SYSTEMS: EXPERIMENTAL STUDY AND VALIDATION

Alioune Badara GUEYE

ABSTRACT

A large number of metrics have been proposed in the literature to assess quality attributes of object-oriented systems such as size, complexity, coupling and cohesion. Moreover, the assessment of software changeability is a major issue and has been the subject of several researches in software engineering domain. Several metrics have been proposed in this context. A way of approaching the problem of validation of these metrics is to investigate, for example, dependences between software changeability and some metrics characterizing its architecture which can be used as indicators of changeability. In the field of object-oriented systems, many experiments were carried out showing that size and coupling metrics are good indicators of changeability.

In this research, we explored whether cohesion, using several metrics, is correlated with the changeability. The goal consisted also of determining, if the case where the hypothesis is confirmed, among the cohesion metrics which one represents the best indicator of changeability. We retained in our study several cohesion metrics among those proposed in literature. We were also interested to extend, with an aim of refinement, some of them. We also considered, within the framework of our research, coupling and size metrics. We conducted an experimental study on several complex and large-scale Java systems. The various steps of the experimentation as well as the results obtained are discussed in this dissertation.

REMERCIEMENTS

Je tiens à exprimer toute ma gratitude à mes directeurs de recherche, Les professeurs Linda Badri et Mourad Badri, pour leur soutien, leurs conseils et leurs encouragements, sans lesquels ce travail n'aurait pu aboutir.

Je tiens également à remercier tous ceux sans lesquels ce travail n'aurait pas été possible, ainsi que toutes les personnes du département de mathématiques et d'informatique.

Je voudrais aussi remercier toutes les personnes qui me sont chères, qui m'ont aidé par leurs encouragements, leurs conseils ou tout simplement leur affection, et tout particulièrement :

Mes frères et sœurs : Abdourahmane, Mohamed, Maïmouna, Mariéme et Nafissatou.
Ma Tante et son mari : Aminata Lountandi Gueye et Moustapha Diop.
Une personne particulière : Mariame Diaw, pour ton amour, ta patience et ton soutien constant.
Mes amis : je ne peux vous citer tous, mais vous vous reconnaitrez.

Ce mémoire n'aurait pu voir le jour sans l'affection, les sacrifices, la confiance de mes parents Birahim Gueye et Mame Fatou Diop, qui m'ont permis d'aller au bout de mes études et de réaliser ce mémoire. Qu'ils trouvent ici l'expression de toute mon affection et de ma reconnaissance. Je vous le dédie, avec tout mon amour.

TABLE DES MATIÈRES

Page

LISTE DES TABLEAUX

Page

LISTE DES FIGURES

Page

LISTE DES ABRÉVIATIONS ET SIGLES

OO : Orienté Objet

JavaCC : Java Compiler Compiler

CHAPITRE 1

INTRODUCTION

1.1 Contexte

Le paradigme objet est le paradigme de développement actuellement dominant pour les systèmes logiciels [Kab 01, Dag 03, Bad 04]. Avec l'augmentation de la complexité et de la taille des systèmes orientés objet, la capacité de raisonner au sujet des attributs de qualité, basés sur des mesures automatiquement calculables, est devenue de plus en plus importante [Dag 03]. Ainsi, plusieurs attributs de qualité logicielle tels que la fonctionnalité, la portabilité, la rentabilité, l'efficacité, et la maintenance ont été définis par divers chercheurs et entités de standardisation [Dag 03].

La maintenance est un attribut particulièrement intéressant de la qualité. Il a été identifié que les activités de maintenance expliquent, aujourd'hui, le plus grand coût dans le développement logiciel [Som 04, Dag 03, Li 96].

1.2 Problématique

Après plusieurs hypothèses émises pour essayer d'expliquer le coût croissant de la maintenance, un consensus s'est établi. La maintenance d'un système est liée à sa conception [Rum 98]. Les acheteurs industriels de logiciels veulent être sûrs de la qualité du produit qu'ils acquièrent. Pour ceci, ils ont besoin des mesures OO pour évaluer le logiciel qu'ils souhaitent acheter [Kab 01]. Les métriques logicielles sont devenues essentielles dans plusieurs disciplines du génie logiciel [Pre 01]. Dans le domaine de la qualité logicielle, les métriques sont utilisées pour évaluer plusieurs attributs (complexité, couplage, cohésion, etc.). Ils fournissent, par conséquent, une importante assistance aux développeurs et managers pour évaluer et améliorer la qualité

du logiciel durant le processus de développement. Les propriétés architecturales, en orienté objet, les plus connues et les plus utilisées sont le couplage et la cohésion. Dans le domaine des systèmes orientés objet, plusieurs expérimentations ont été conduites. Elles montrent que le couplage des classes constitue un indicateur de changeabilité important. Chaumun et al. [Cha 00A] ont observé une forte corrélation entre la changeabilité et quelques métriques de couplage, à travers différents systèmes industriels et différents types de changements. Cependant, mesurer le couplage est difficile et consomme du temps du fait qu'il constitue une propriété interclasses [Kab 01]. En effet, pour la mesurer, la connaissance de l'ensemble du système et de tous les liens entre les classes doit être maitrisée. La cohésion de classe est une propriété intra-classe. Pour la mesurer, nous avons juste besoin de considérer la classe étudiée. Aussi, une large croyance soutient qu'une cohésion élevée est reliée à un faible couplage. C'est un principe largement reconnu dans la communauté génie logiciel, même si à notre connaissance, très peu de travaux se sont intéressés à la validation formelle de cette relation. Plusieurs études expérimentales ont révélé le fait que certaines caractéristiques des classes ne sont pas reflétées dans les métriques de cohésion existantes et ont appelé à leur raffinement [Cha 00, Kab 01].

1.3 Objectifs

L'objectif de ce travail consiste donc, à raffiner, dans un premier temps, les métriques de cohésion existantes en rajoutant de nouveaux critères et procéder à leur validation. Dans un second temps, étudier la relation entre la cohésion et le couplage, du fait qu'une large croyance soutient qu'une cohésion élevée est reliée à un faible couplage et vice versa, afin d'apporter des preuves palpables à cette croyance. Cette supposition, entre le couplage et la cohésion, nous amène à vérifier si la cohésion est un indicateur de changeabilité car plusieurs études ont montré que le couplage des classes constitue un indicateur de changeabilité [Cha 00A, Kab 01].

1.4 Cheminement du mémoire

Les objectifs ont guidé ce mémoire dans une direction bien précise. Ainsi, dans le chapitre 2, nous proposons une revue sommaire de la littérature et les principaux travaux réalisés dans le domaine de l'analyse de l'impact de changement. Le chapitre 3 présente les principales propriétés architecturales des systèmes orientés objets plus particulièrement celles reliées à la cohésion. Le fait que plusieurs études appellent à un raffinement des métriques de cohésion orientées objet [Cha 00, Kab 01], le chapitre 4 présente la définition d'un nouveau critère de cohésion. Dans le chapitre 5, nous présentons l'étude réalisée sur la relation entre le couplage et la cohésion. L'impact de changement est traité au chapitre 6. Le chapitre 7 présente les environnements des différentes expérimentations réalisées. Le chapitre 8 porte sur l'étude de la cohésion en tant qu'indicateur de changeabilité. Enfin, une synthèse de nos travaux ainsi que certaines perspectives de recherche sont présentées dans la conclusion.

CHAPITRE 2

REVUE DE LA LITTÉRATURE

Plusieurs études ont été entreprises pour valider les métriques, orientées objet entre autres, et pour les relier à quelques propriétés de maintenance. Li et Henry [Li 93] ont pris cinq métriques de Chidamber et Kemerer [Chi 94] en plus de leurs propres métriques qui sont au nombre de trois pour prouver qu'il y a un rapport fort entre ces métriques et l'effort de maintenance, exprimé en nombre de lignes de code changées.

Dans [Bri 98A], les auteurs ont prouvé que le choix des architectures, dans les premières étapes de conception des systèmes logiciels, ont un impact important sur un certain nombre de facteurs de qualité, par exemple, la maintenance, l'efficacité, et la réutilisabilité. Lounis et al. [Lou 97] ont proposé une succession de 24 métriques de code pour produire des modèles prédictifs liés à la propagation de défauts. En conclusion, dans [Bri 01], les auteurs ont également étudié des rapports entre la majeure partie des métriques de couplage, de cohésion, et d'héritage d'un coté, et propagation de défauts des classes de l'autre côté. De cette étude, il en découle que la qualité ne suit pas des lois universelles et des modèles de qualité doivent être développés localement, partout où nécessaire bien que des principes et techniques puissent être réutilisés.

Plusieurs travaux ont été conduits sur l'impact de changement. Han [Han 97] a développé une approche pour le calcul d'impact de changement au niveau conception et sur des documents d'implémentation. Son approche ne considère pas les dépendances d'invocation. En outre, les impacts ne sont pas définis d'une manière formelle. Dans [Ant 99], les auteurs ont prévu l'évolution de la taille des systèmes orientés objet à partir de l'analyse des classes affectées par une demande de changement. Ils ont prévu le changement de la taille en termes d'ajout/modification de lignes de code. Kung et al.

[Kun 95], intéressés par le test de régression, ont développé un modèle d'impact de changement basé sur trois liens : l'héritage, l'association, et l'agrégation. Ils ont également défini des algorithmes formels pour calculer toutes classes affectées incluant le ripple effect c'est-à-dire la propagation de défauts. Li et Offutt ont examiné, dans [Li 96], les effets de l'encapsulation, de l'héritage, et du polymorphisme sur l'impact de changement autrement dit sur la conséquence d'un changement. Ils ont également proposé des algorithmes pour calculer l'impact complet des changements effectués dans une classe donnée. Cependant, quelques changements, impliquant par exemple l'héritage et l'agrégation, n'ont pas été complètement couverts par leurs algorithmes.

Briand et al. [Bri 99] ont essayé de voir si les mesures de couplage, capturant toutes sortes de collaboration entre les classes, peuvent aider à analyser l'impact de changement. La stratégie adoptée dans cette étude est différente des autres stratégies puisqu'elle est purement empirique. Cette étude a montré que la métrique de couplage, liée à l'agrégation et à l'invocation, est reliée au ripple effect d'une part, et permet l'exécution de l'analyse de la dépendance et la réduction de l'effort d'analyse d'impact d'autre part. Dans [Cha 98 A] et [Kab 01], un modèle d'impact de changement a été défini à un niveau abstrait, pour étudier la changeabilité des systèmes orientés objet. Pour prévoir la changeabilité, l'approche adoptée emploie les propriétés caractéristiques de conception des systèmes orientés objet quantifiées par les métriques.

En bref, les études effectuées dans [Bri 99] et [Wil 98] sont des exemples d'approches purement empiriques. Les travaux de [Kun 95], [Li 96], [Can 01], et [Lee 98] exploitent des approches basées principalement sur des modèles comme, les graphes de dépendance, enrichis par quelques formalismes. Les études de [Cha 98A] et [Kab 01] proposent une approche différente. Un modèle de changement et d'impact de changement conçu au cours du projet SPOOL, a été défini. Ce modèle s'avère être plus complet et plus systématique que les précédents. Abdi et al. [Abd 06] ont noté qu'il y a plus de travaux basés sur des graphes de dépendance que des travaux basés sur d'autres

abstractions ou des travaux purement empiriques. Généralement, l'impact n'est pas calculé d'une manière systématique d'une part et la plupart des expériences faites étaient sur des systèmes réduits ne permettant pas la généralisation des résultats obtenus (règles, rapports, lois, etc.) d'autre part.

Kabaili et al. [Kab 01] ont montré que les métriques de couplage, à travers l'invocation de méthodes et l'accès aux variables, constituent de bons indicateurs de changeabilité. Selon l'hypothèse « une cohésion faible est corrélée à un couplage fort », la relation entre métriques de cohésion et changeabilité a été aussi étudiée et les résultats de cette expérimentation ont été négatifs. Dans ce mémoire nous nous intéresserons à ce dernier point. Nous étudierons, grâce aux raffinements des métriques dans le chapitre 4, la relation entre la cohésion et le changement.

CHAPITRE 3

QUELQUES MÉTRIQUES ORIENTÉES OBJET

3.1 Introduction

Définitions de métrique (selon IEEE)**:**

- Mesure quantitative du degré auquel un élément tend vers un facteur de qualité.
- Une fonction dont les entrées sont les données du logiciel et dont la sortie est une valeur numérique qui peut être interprétée comme le degré auquel un élément tend vers un facteur de qualité.

Selon ISO/IEC 9126 [2003], une métrique de qualité du logiciel est définie comme une échelle quantitative et une méthode pouvant être employées pour déterminer la valeur que prend un attribut d'une entité d'un produit logiciel spécifique.

Depuis leur apparition dans les années 70, les métriques automatiquement calculables sont devenues un outil important pour évaluer des attributs de logiciel et des activités logicielles connexes [Dag 03].

Fenton a classé les métriques logicielles par catégorie :

- Les métriques de processus, pour mesurer les caractéristiques des processus de développement de logiciel.
- Les métriques de produit, pour évaluer les produits logiciels comme les composantes, les procédures et les programmes, etc.

- Les métriques de ressource, pour mesurer les caractéristiques des ressources reliées au logiciel telles que le matériel et le personnel [Dag 03, Fen 99].
Fenton fait, en outre, la distinction entre les métriques internes (examinant seulement le logiciel) et externes (examinant le logiciel dans un environnement).

Dans le contexte de notre travail, on ne considère que les métriques internes et plus exactement les métriques de produit pour les logiciels orientés objet. Rocacher [Roc 88] fût l'un des premiers à remarquer que les métriques traditionnelles ne pouvaient pas évaluer de façon efficace les systèmes OO. Les métriques existantes ne prennent pas en compte certains concepts importants véhiculés par le paradigme OO tels que l'encapsulation, l'héritage, le polymorphisme, etc. [Dag 03, Roc 88].

Les métriques traditionnelles mesurent les structures de conception et/ou les structures de données indépendamment. En revanche, la métrique OO doit prendre en considération à la fois la combinaison de la fonction et des données en termes d'objets intégrés [Dag 03]. Dans le domaine des métriques, il y a eu plusieurs travaux pertinents. Parmi les plus connus nous pouvons citer, entre autres, ceux de Chidamber et al. [Chi 94], Li et al. [Li 93, Li 95], Chen et al. [Che 93], Abreu et al. [Abr 92, Abr 96], Hitz et al. [Hit 95], Badri et al. [Bad 95], Bieman et al. [Bie 95], Abounader et al. [Abo 97] ont réalisé une synthèse afin de classifier ces métriques.

Les métriques existantes proposent en règle générale de quantifier les aspects suivants :

– Les relations de couplage entre classes ou entre méthodes. Ces relations essaient d'évaluer dans quelle proportion une entité utilise des entités qui lui sont externes. Dans cette optique, nous pouvons citer comme exemple les métriques CBO (Coupling Between Objects) de Chidamber et Kemerer [Chi 94], MPC (Message-Passing Coupling) et DAC (Data Abstraction Coupling) de Li et Henry [Li 93, Li 95] ou encore

OLC (Object Level Coupling) et CLC (Class Level Coupling) de Hitz et Montazeri [Hit 95].

– La cohésion des classes permet d'évaluer dans quelle proportion une entité s'utilise elle-même. Dans cette optique, nous pouvons citer comme exemple les métriques LCOM (Lack of Cohesion in Method) de Chidamber et Kemerer [Chi 94] ainsi que LCOM1 et LCOM2 de Etzkorn et al. [Wei 98].

– La complexité permet d'évaluer dans quelle proportion une entité est complexe à comprendre ou à utiliser. Dans cette optique, nous pouvons citer comme exemple les métriques WMC (Weighted Methods per Class) de Chidamber et Kemerer [Chi 94] ou encore OpCom (Operation Complexity), AOC (Operation Argument Complexity) et AC (Attribute Complexity) de Chen et Lu [Che 93].

– La taille permet d'évaluer les dimensions d'un programme. Dans cette optique, nous pouvons citer comme exemple les métriques IS (Interface Size) de Abbot, Korson et McGregor [Abb 94], NOM (Number Of Method) et SIZE2 de Li et Henry [Li 93, Li 95], ou encore NCL (Number Of Class) de Shetz, Teegarden et Monarchi [Abr 91].

– Certaines relations d'héritage comme la suite MOOD [Abr 92, Abr 96] qui s'intéresse à mesurer l'encapsulation des propriétés sans mettre en avant une sémantique derrière cette notion d'encapsulation. Il existe également DIT (Depth of Inheritance Tree) de Chidamber et Kemerer [Chi 94].

– Le voisinage direct permet d'évaluer dans quelle proportion une entité du système est directement reliée à une autre. Dans cette optique, nous pouvons citer la métrique NOC (Number Of Children) de Chidamber et Kemerer [Chi 94].

– Les flux permettent de mesurer dans quelle proportion une entité peut recevoir ou donner de l'information à une autre entité. Dans cette optique, nous pouvons citer la métrique MIP (Method Input Parameter) de Shetz, Teegarden et Monarchi [Abr 91].

3.2 Les métriques de taille

La taille est un attribut interne permettant de mesurer un produit, un processus ou un projet [Pre 01]. La taille contribue à la prédiction de certaines propriétés des systèmes, à l'estimation de l'effort et à l'élaboration d'un planning de projet. La taille permet aussi de normaliser d'autres mesures de produit, comme les erreurs qui ont tendance à croître avec la taille des applications [Kab 01]. Cette normalisation permet aussi de comparer les produits et les processus. Le nombre d'erreurs dans un produit est corrélé positivement avec la taille du produit [Kab 01]. Ceci a d'ailleurs été confirmé par plusieurs études. Un programme a 100 erreurs et un autre a seulement une erreur n'est pas une information pertinente. Mais, sachant que le premier programme possède 10 000 lignes de code et que le second en a juste 10, cela nous renseigne plus sur le problème. En divisant le nombre d'erreurs par le nombre de lignes de code, nous obtenons le taux d'erreurs ou la densité d'erreurs. Ce type d'information permet de comparer des systèmes et de les ordonner en termes de densité d'erreurs.

La métrique de taille est l'une des métriques les plus utilisées dans la pratique. Plusieurs études empiriques soutiennent que la taille d'un système logiciel est corrélée avec sa propagation d'erreur [Dag 03, El 99, El 01]. La métrique la plus simple et la plus utilisée, en général, est LOC (ligne de code) [Suz 98]. Plusieurs centaines d'articles scientifiques mentionnent cette métrique. Elle a fait l'objet de nombreuses critiques [Fen 99]. L'inconvénient d'utiliser LOC est qu'elle ne prend pas en compte le concept d'encapsulation. LOC dépend, par ailleurs, énormément du style de codage du développeur.

3.3 Les métriques de couplage

Le couplage a été introduit, d'abord, pour les systèmes procéduraux. Stevens et al. [Ste 74] définissent le couplage comme "la mesure de la force de l'association établie par une connexion d'un module avec un autre". D'après Pressman [Pre 01], "le couplage est une mesure d'interconnexion parmi les modules formant la structure du logiciel". Un module présentant un fort couplage est un module complexe. Cette complexité se traduit par la difficulté à comprendre le module, à détecter les erreurs, à les corriger et à le changer. L'interaction d'un module avec plusieurs modules du système le rend plus difficile à maintenir puisque chaque changement subi par le module peut affecter les modules auxquels ce dernier est associé. La complexité d'un système peut être réduite par la conception de modules interagissant faiblement les uns avec les autres.

Les meilleures pratiques en matière de génie logiciel favorisent un couplage faible entre les composants afin de diminuer les interdépendances et faciliter l'évolution [Pre 01]. Dans la littérature, plusieurs études démontrent que les métriques de couplage sont de bons facteurs prédictifs pour la maintenabilité des systèmes OO [Kab 01]. Il existe différents types de couplage, tels que le couplage d'héritage versus le couplage de non-héritage, le couplage d'importation versus le couplage d'exportation, le couplage direct versus le couplage indirect, etc. [Dag 03, Ben 97]. Cependant, il existe des insuffisances empiriques clarifiant leur signification pour la prévision dans la maintenabilité [Dag 03].

Le tableau ci-dessous, tiré de [Ben 97], illustre les différents types de couplage et leur correspondance au niveau de la conception orientée objet.

Tableau I :

Analogie entre les relations mesures de couplage et conception OO

Coupling measure	OO Design Concept
Static- Import -Class- Attribute Import coupling interactions where the involved object are class attributes whose values are directly accessed	*Aggregation*
Static-Import-(Class, Method), Method Import coupling interactions where the involved objects are class methods parameters whose values are directly manipulated	*Usage*
Dynamic-Import-Class-Attribute Import coupling interactions where the involved objects are class attributes whose values are manipulated by pointers	*Association*
Dynamic-Import-(Class, Method), Method Import coupling interactions where the involved objects are class methods parameters whose values are manipulated via pointers	*Message passing*
Static-Export-Class-Attribute Export coupling interactions where the involved objects are class attributes whose values are directly accessed	*Embedding*
Static-Export-(Class, Method), Method Export coupling interactions where the involved objects are class methods parameters whose values are directly manipulated	*Usage*
Dynamic-Export- Class – Attribute Export coupling interactions where the involved objects are class attributes whose values are manipulated by pointers	
Dynamic-Export--(Class, Method), Method Export coupling interactions where the involved objects are class methods parameters whose values are manipulated via pointers	*Message passing*

Dans la littérature orientée objet, on note essentiellement 23 mesures de couplage [Bri 97].

3.4 Les métriques de cohésion

La cohésion a été définie pour les systèmes procéduraux comme le degré de liaison des éléments appartenant à un même module [Ste 74]. Elle a été reconnue comme un des critères de qualité les plus importants de la conception. Les modules présentant une forte cohésion sont faciles à maintenir, et représentent des candidats potentiels pour la réutilisation. Selon le critère de fonctionnalité, un module est dit fortement cohésif s'il accomplit une seule tâche particulière et que tous ses éléments y contribuent. Les éléments d'un module peuvent être des instructions, des données, des sous-fonctions ou des modules.

La cohésion de classe est considérée comme étant l'un des attributs les plus importants des systèmes orientés objet. Elle fait référence au degré de liaison des membres dans un composant. Une cohésion élevée est une propriété souhaitable des composants d'un logiciel. Il est largement reconnu que les composants fortement cohésifs tendent à avoir une grande maintenabilité et réutilisabilité [Bie 95, Cha 00].

Yourdon et Constantine [You 79] présentent la cohésion dans les applications traditionnelles comme la mesure de l'ampleur des rapports fonctionnels des éléments dans un module. Ils ont décrit la cohésion comme critère pour l'évaluation de la qualité de conception.

Selon G. Booch [Boo 94] la cohésion fonctionnelle est élevée quand les éléments d'un composant (classe) travaillent ensemble pour fournir un bon comportement (délimité).

Dans le paradigme d'objet, une classe est cohésive quand ses parties sont fortement corrélées. Il devrait être difficile de décomposer une classe cohésive. Une classe avec une faible cohésion possède des membres disparates et non-connexes. La cohésion peut être utilisée pour identifier les classes mal conçues. La cohésion est un but fondamental à considérer continuellement durant le processus de conception [Lar 02].

Plusieurs métriques ont été proposées pour mesurer la cohésion d'une classe dans les systèmes orientés objet. Les principales métriques existantes sont présentées en détail et sont classées par catégorie dans [Bri 98].

Elles sont basées sur l'utilisation de variables d'instance ou le partage des variables d'instance. Ces métriques capturent la cohésion de classe en termes de liaisons parmi les membres d'une classe. Ces métriques ont été expérimentées et largement discutées dans la littérature [Bas 96, Cha 98, El 99, Hen 96]. Plusieurs études ont noté que ces métriques échouent dans beaucoup de situations pour refléter correctement la cohésion des classes [Kab 01, Cha 00]. Selon plusieurs auteurs, elles ne tiennent pas compte de certaines caractéristiques des classes, par exemple, de la taille des parties cohésives [Ama 02] et de la connectivité des membres [Cha 00, Cha 04].

Badri et al. [Bad 04] ont proposé une nouvelle approche. Elle tient, dans ses considérations, non seulement compte de l'utilisation de variables d'instance ou le partage des variables d'instance mais aussi du partage de méthodes même si ces dernières n'utilisent aucune variable en commun de façon directe ou indirecte [Bad 03].

Tableau II

Principales métriques de cohésion existantes [BAD 03]

Metric	Description
LCOM1	The number of pairs of methods in a class using no attribute in common.
LCOM2	Let P be the pairs of methods without shared instance variables, and Q be the pairs of methods with shared instance variables. Then LCOM2 = $\|P\| - \|Q\|$, if $\|P\| > \|Q\|$. If this difference is negative, LCOM2 is set to zero.
LCOM3	The Li and Henry definition of LCOM. Consider an undirected graph G, where the vertices are the methods of a class, and there is an edge between two vertices if the corresponding methods share at least one instance variable. Then LCOM3 = |connected components of G|
LCOM4	Like LCOM3, where graph G additionally has an edge between vertices representing methods M_i and M_j, if M_i invokes M_j or vice versa.
Co	Connectivity. Let V be the vertices of graph G from LCOM4, and E its edges. Then $Co = 2 . \frac{\|E\| - (\|V\| - 1)}{(\|V\| - 1) . (\|V\| - 2)}$
LCOM5	Consider a set of methods $\{M_i\}$ $(i = 1, \ldots, m)$ accessing a set of instance variables $\{A_j\}$ $(j = 1, \ldots, a)$. Let $\mu(A_j)$ be the number of methods that reference A_j. Then $LCOM5 = \frac{(1/a)\,\Sigma_{1 \le j \le a}\,\mu(Aj) - m}{1 - m}$
Coh	Cohesiveness is a variation on LCOM5. $Coh = \frac{\Sigma_{1 \le j \le a}\,\mu(Aj)}{m \,.\, a}$
TCC	Tight Class Cohesion. Consider a class with N public methods. Let NP be the maximum number of public method pairs: NP = [N*(N - 1)]/2. Let NDC be the number of direct connections between public methods. Then TCC is defined as the relative number of directly connected public methods. Then, TCC = NDC / NP.
LCC	Loose Class Cohesion. Let NIC be the number of direct or indirect connections between public methods. Then LCC is defined as the relative number of directly or indirectly connected public methods. LCC=NIC/NP.
DCD	Degree of Cohesion (direct) is like TCC, but taking into account Methods Invocation Criterion as well. DCD gives the percentage of methods pairs, which are directly related.
DCI	Degree of Cohesion (indirect) is like LCC, but taking into account Methods Invocation Criterion as well

Dans ce mémoire, nous nous sommes intéressés surtout à la cohésion. Plusieurs études expérimentales ont révélé le fait que certaines caractéristiques des classes ne sont pas reflétées dans les métriques de cohésion existantes et ont appelé à leur raffinement [Cha 00, Kab 01]. C'est dans cette optique, que dans le chapitre suivant, nous avons introduit un nouveau critère de cohésion que nous avons validé à travers une étude empirique.

CHAPITRE 4

PROPOSITION D'UN NOUVEAU CRITERE DE COHÈSION

4.1 Définition d'un nouveau critère pour les métriques de cohésion

Dans notre étude, le nouveau critère que nous avons défini étend la définition des deux métriques DCd et DCi définies dans [Bad 03]. Ce critère se définit comme suit :

Deux méthodes d'une classe peuvent partager un même type objet passé en paramètre.

Dans le but de démontrer l'efficacité du critère rajouté, nous avons procédé à une étude empirique sur plusieurs systèmes. DCd et DCi sont la conséquence d'une révision de la définition initiale de la cohésion proposée par Badri et al. dans [Bad 95]. Cette définition est essentiellement basée sur la connexion des méthodes publiques d'une classe. Elle est définie en termes de nombre relatif de méthodes publiques reliées dans une classe. Les autres méthodes (privées et protégées) sont incluses indirectement à travers les méthodes publiques. Cette approche est comparable à celle adoptée par Bieman et Kang dans [Bie 95]. Badri et al, dans [Bad 03], redéfinissent la première définition de la cohésion proposée dans [Bad 95] en étendant le critère d'invocation de méthodes d'un coté, et en introduisant le concept d'usage indirect des attributs défini par Bieman et Kang dans [Bie 95] d'un autre coté. Ce dernier concept a été étendu au critère d'invocation de méthodes. DCd et DCi sont définis respectivement par les relations directe et indirecte entre les méthodes.

4.1.1 Relation directe entre les méthodes

Deux méthodes publiques M_i et M_j peuvent être directement connectées de plusieurs manières : elles partagent au moins une variable d'instance en commun (UA relation [Bad 03]), ou interagissent avec au moins une autre méthode de la classe (IM relation [Bad 03]), ou les deux. Ce qui signifie : $UA_{M_i} \cap UA_{M_j} \neq \varnothing$ ou $IM_{M_i} \cap IM_{M_j} \neq \varnothing$.

Considérons une classe C avec PUM = {M_1, M_2, ..., Mn} l'ensemble des méthodes publiques. Le nombre maximum de paires de méthodes publiques, comme mentionné dans [Bad 95, Bie 95], est n*(n-1)/2.

Considérons un graphe non orienté G_D, ou les sommets sont les méthodes publiques de la classe C, et il y'a un arc entre deux sommets si les méthodes correspondantes sont directement reliées. Soit E_D, le nombre d'arcs dans le graphe. Le degré de cohésion dans la classe basé sur la relation directe entre ses méthodes publiques est défini comme suit : $DC_D = |E_D| / [n*(n-1)/2] \in [0,1]$. DC_D donne le pourcentage de paires de méthodes publiques, qui sont directement reliées. La métrique LCC_D (Lack of Cohesion in the Class) de la classe C est ainsi donnée par : $LCC_D = 1 - DC_D \in [0,1]$.

4.1.2 Relation indirecte entre les méthodes

Deux méthodes publiques M_i et M_j peuvent être indirectement connectées si elles sont directement ou indirectement reliées à une méthode M_k. La relation indirecte introduite par Bieman et Kang dans [Bie 95], est la fermeture transitive de la relation directe. Badri et al. [Bad 03] utilisent ce concept pour identifier les méthodes reliées indirectement. Ainsi, une méthode M_1 est indirectement connectée à une méthode M_k s'il y'a une séquence de méthodes M_1, M_2, M_3..., M_K telle que M_i est directement connectée à M_{i+1} (i=1, k-1).

Considérons maintenant un graphe indirect G_I, où les sommets sont les méthodes publiques de la classe C. Il y'a un arc entre deux sommets si les méthodes correspondantes sont directement ou indirectement reliées (fermeture transitive du graphe G_D). Soit E_I le nombre d'arcs dans le graphe. Le degré de cohésion dans la classe C basé sur les relations directes ou indirectes entre les méthodes publiques est défini comme : $DC_I = |E_I| / [n*(n-1)/2] \in [0,1]$. DC_I donne le pourcentage de paires de méthodes publiques, qui sont directement ou indirectement reliées. La métrique LCC_I (Lack of Cohesion in the Class) de la classe C est ainsi donnée par : $LCC_I = 1- DC_I \in [0,1]$.

4.1.3 Nouvelle définition de la cohésion

4.1.3.1 Relation directe entre les méthodes

Deux méthodes publiques M_i et M_j peuvent être directement connectées de plusieurs manières : elles partagent au moins une variable d'instance en commun (UA relation [Bad 03]), ou interagissent avec au moins une autre méthode de la classe (IM relation [Bad 03]), ou leurs arguments partagent un même type la classe (UC relation), ou les trois. Ce qui signifie : $UA_{M_i} \cap UA_{M_j} \neq \varnothing$ ou $IM_{M_i} \cap IM_{M_j} \neq \varnothing$ ou $UC_{M_i} \cap UC_{M_j} \neq \varnothing$.

Considérons une classe C avec PUM = {M_1, M_2, ..., Mn} l'ensemble des méthodes publiques. Le nombre maximum de paires de méthodes publiques, comme mentionné dans [Bad 95, Bie 95], est n*(n-1)/2.

Considérons un graphe indirect G_D, ou les sommets sont les méthodes publiques de la classe C, et il y'a un arc entre deux sommets si les méthodes correspondantes sont directement reliées. Soit E_D, le nombre d'arcs dans le graphe. Le degré de cohésion dans la classe basé sur la relation directe entre ses méthodes publiques est défini comme suit : $DC_{DE} = |E_D| / [n*(n-1)/2] \in [0,1]$. DC_{DE} donne le pourcentage de paires de méthodes

publiques, qui sont directement reliées. La métrique LCC_D (Lack of Cohesion in the Class) de la classe C est ainsi donnée par : $LCC_{DE} = 1 - DC_{DE} \in [0,1]$.

4.1.3.2 Relation indirecte entre les méthodes

Deux méthodes publiques M_i et M_j peuvent être indirectement connectées si elles sont directement ou indirectement reliées à une méthode M_k. La relation indirecte introduite par Bieman et Kang dans [Bie 95], est la fermeture transitive de la relation directe. Badri et al. [Bad 03] utilisent ce concept pour identifier les méthodes reliées indirectement.

Ainsi, une méthode M_1 est indirectement connectée à une méthode M_k s'il y'a une séquence de méthodes M_1, M_2, M_3..., M_K telle que M_i est directement connectée à M_{i+1} (i=1, k-1).

Considérons maintenant un graphe indirect G_I, ou les sommets sont les méthodes publiques de la classe C. Il y'a un arc entre deux sommets si les méthodes correspondantes sont directement ou indirectement reliées (fermeture transitive du graphe G_D). Soit E_I le nombre d'arcs dans le graphe. Le degré de cohésion dans la classe C basé sur les relations directes ou indirectes entre les méthodes publiques est défini comme : $DC_{IE} = |E_I| / [n*(n-1)/2] \in [0,1]$. DC_{IE} donne le pourcentage de paires de méthodes publiques, qui sont directement ou indirectement reliées. La métrique LCC_{IE} (Lack of Cohesion in the Class) de la classe C est ainsi donnée par : $LCC_{IE} = 1 - DC_{IE} \in [0,1]$.

4.2 Sélection des systèmes

La première étape de notre expérimentation consiste à sélectionner plusieurs systèmes Java pour évaluer les différentes métriques de cohésion proposées et démontrer la validité du nouveau critère. Notre but est d'analyser le maximum de classes Java de plusieurs applications. Le but étant de nous assurer dans un premier temps si le critère rajouté est statistiquement significatif avant de procéder à une validation plus poussée.

Systeme1: *JIU0.10* (Java Imaging Utilities) est une bibliothèque en Java pour le chargement, l'édition, l'analyse et la sauvegarde de pixel de fichiers images (http://sourceforge.net/projects/jiu).Ce système contient 180 classes.

Systeme2: *JIU0.11* (Java Imaging Utilities) est une bibliothèque en Java pour le chargement, l'édition, l'analyse et la sauvegarde de pixel de fichiers images (http://sourceforge.net/projects/jiu). Ce système contient 191 classes.

Systeme3: *FujabaUML* est un outil de développement logiciel qui permet d'étendre facilement UML et le développement Java par l'ajout de plug-ins (http://www.fujaba.de). Ce système contient 186 classes.

Systeme4 : *Wbemservices* est une implémentation open source Java de Web Based Enterprise Management (WBEM) pour des applications commerciales et non commerciales. C'est un projet composé d'APIs, de serveurs, d'applications clientes et d'outils (http://wbemservices.sourceforge.net/). Ce système contient 463 classes.

Nous avons donc plus de 800 classes au total appartenant à des systèmes divers.

4.3 Environnement

L'environnement développé pour le calcul des métriques est composé de plusieurs outils. Un parseur Java (JavaCC), que nous avons étendu, analyse le code source des systèmes testés. Les informations collectées contiennent toutes les données nécessaires (attributs, méthodes, paramètres, attributs utilisés, etc.). Ces données sont sauvegardées dans une base de données spéciale.

Ces informations sont ensuite traitées par un outil de calcul de métriques que nous avons développé. Nous collectons les valeurs des métriques pour chacun des systèmes sélectionnés. Pour chaque métrique, nous calculons quelques statistiques descriptives (moyenne, déviation standard).

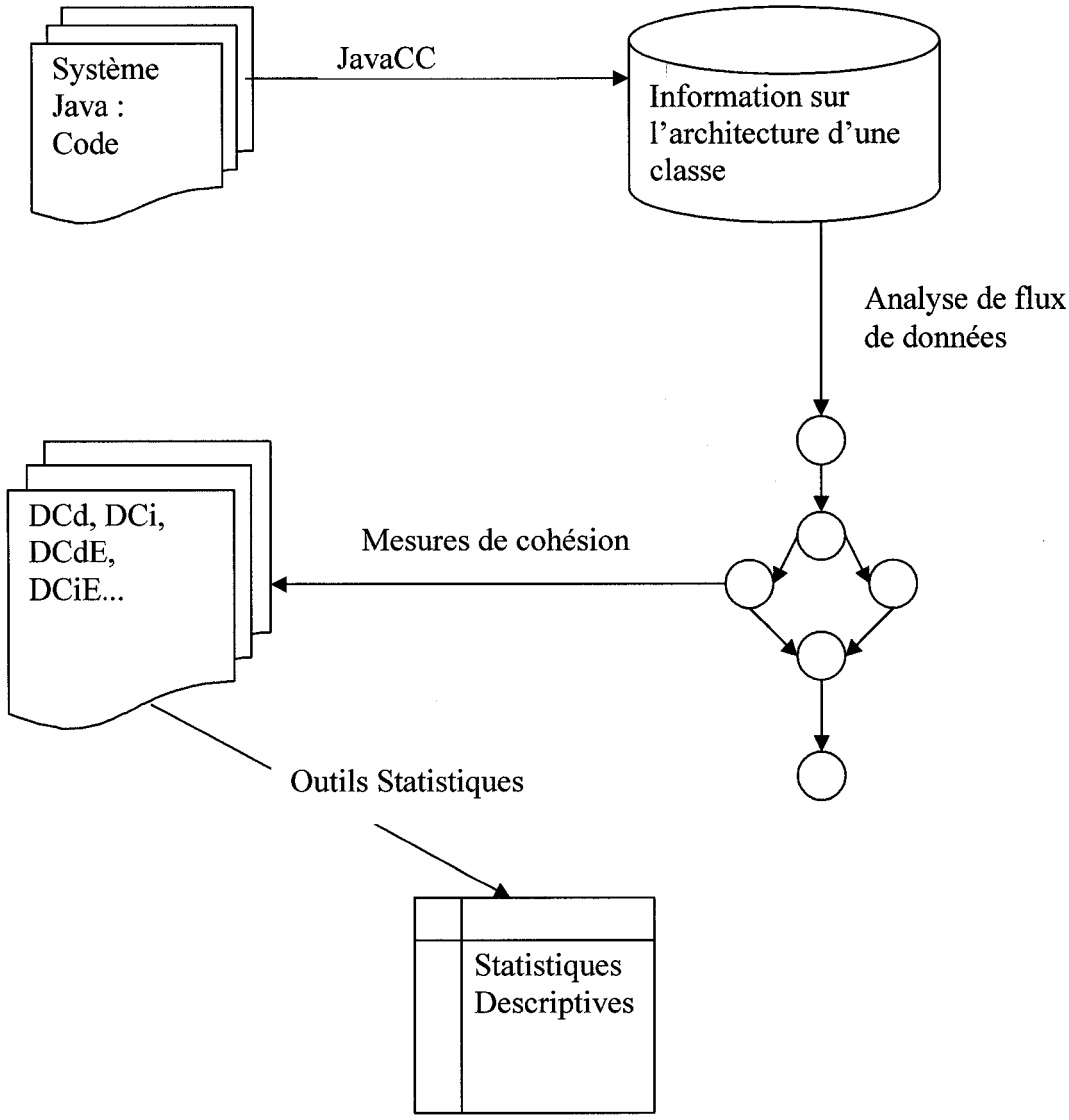


Figure 1 : Architecture de l'environnement de calcul des métriques.

4.4 Résultats

Les métriques sélectionnées sont décrites ci dessous :

DCd : le pourcentage de paires de méthodes qui sont directement reliées

DCi : le pourcentage de paires de méthodes qui sont directement ou indirectement reliées

DCd* : le pourcentage de paires de méthodes qui sont directement reliées en considérant, en plus, le partage d'objet au niveau des arguments.

DCi* : le pourcentage de paires de méthodes qui sont directement ou indirectement reliées en considérant, en plus, le partage d'objet au niveau des arguments.

En plus des valeurs de cohésion des classes pour l'ensemble des systèmes sélectionnés, le tableau III fournit également quelques descriptions statistiques de ces systèmes.

Tableau III

Valeurs moyennes de cohésion

Systèmes	Des. Stat	DCd	DCd*	DCi	DCi*
Jiu1	Moyenne	0,16027	0,17384	0,1922	0,2178
	Sdt.dev	0,13686	0,1378	0,1638	0,2178
Jiu2	Moyenne	0,2497	0,2635	0,3102	0,3350
	Sdt.dev	0,16466	0,1714	0,2292	0,2246
Fujaba	Moyenne	0,01597	0,05244	0,0207	0,0656
	Sdt.dev	0,01479	0,05861	0,0201	0,0739
Wbemservices	Moyenne	0,08138	0,2286	0,1013	0,2747
	Sdt.dev	0,14164	0,2051	0,1678	0,2332

Les résultats obtenus pour DCd* et DCi* montrent clairement que ces deux métriques capturent plus de paires de méthodes reliées que DCd et DCi.

Les figures 2, 3, 4 et 5 montrent les valeurs moyennes des métriques pour les systèmes 3 et 4. Les résultats montrent que DCd* et DCi* capturent un aspect additionnel des propriétés de classe. C'est dû, à notre avis, à la combinaison du critère rajouté. Cet aspect sera discuté et validé dans la prochaine section. L'objectif principal de ce travail était de démontrer l'efficacité du nouveau critère de cohésion. Pour cette raison, nous ne discuterons pas en détail les valeurs de cohésion des systèmes testés. Les résultats dans le tableau 3 prouvent clairement que les systèmes testés ne sont pas cohésifs.

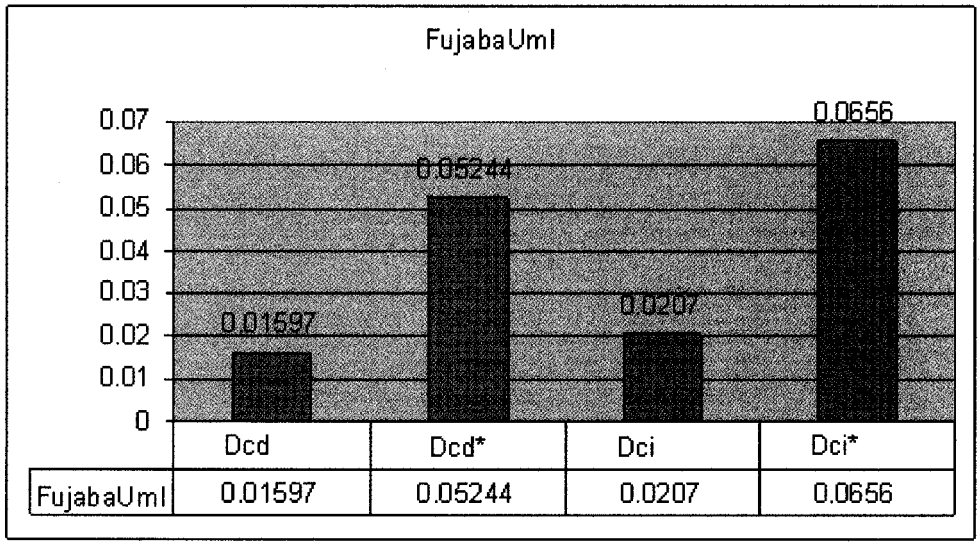


Figure 2 Représentation des valeurs moyennes dans FujabaUml.

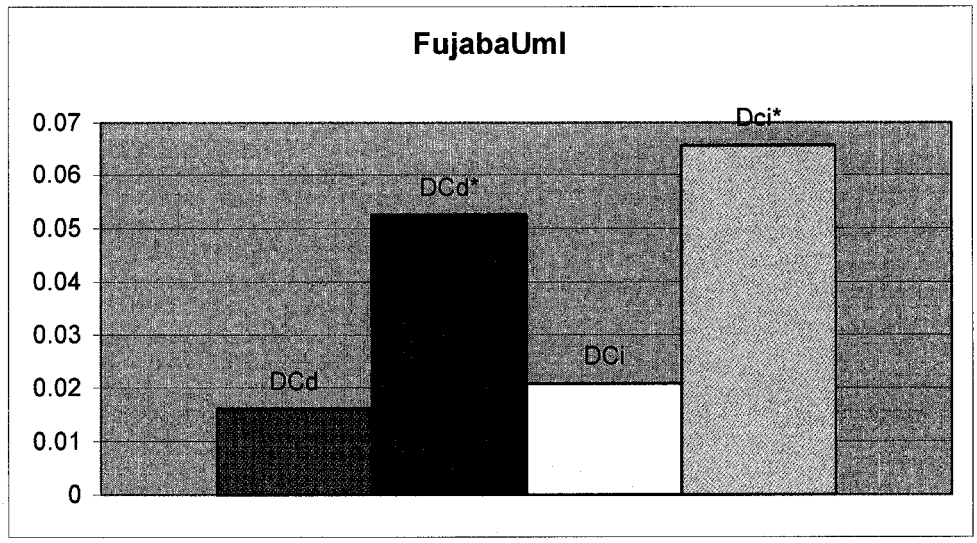


Figure 3 Comparaison des valeurs moyennes dans FujabaUml.

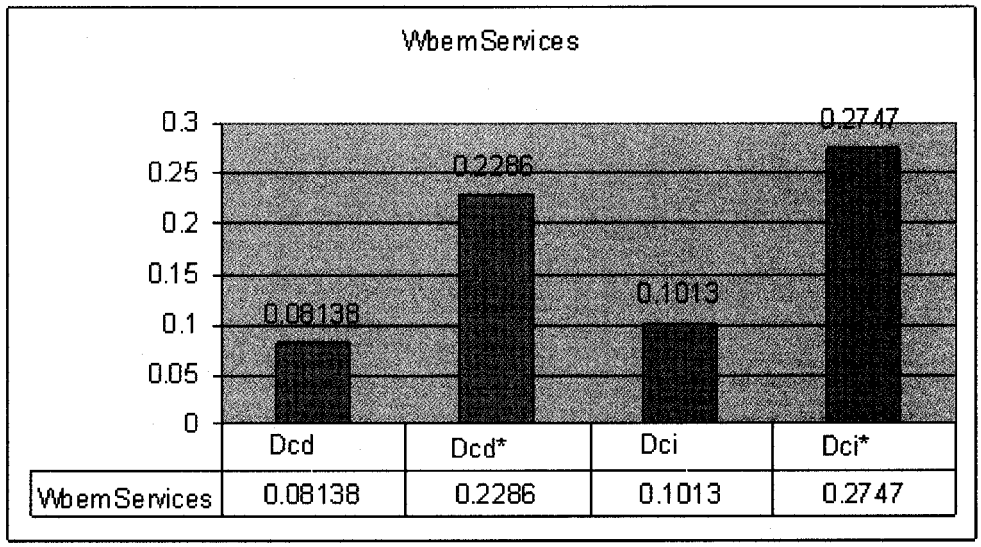


Figure 4 Représentation des valeurs moyennes dans Wbemservices.

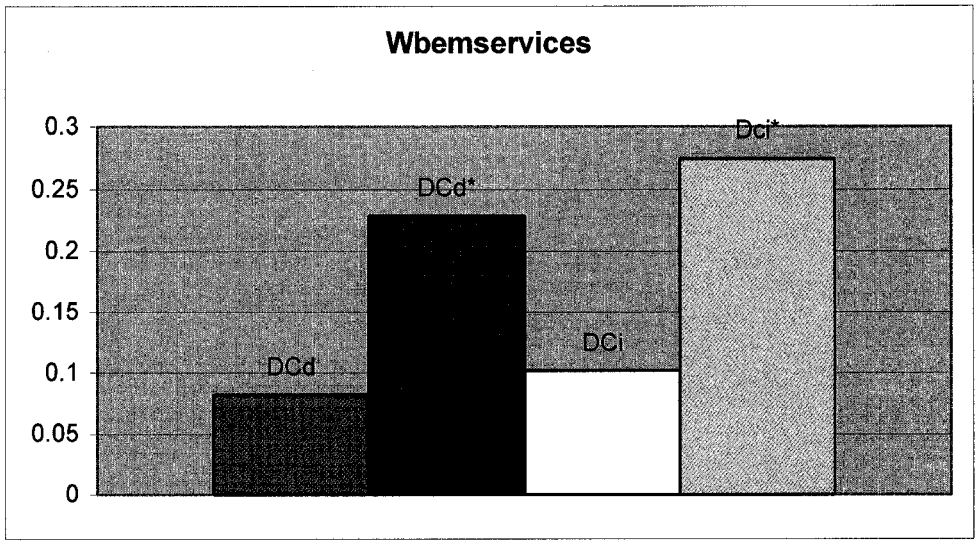


Figure 5 Comparaison des valeurs moyennes dans Wbemservices.

4.5 Validation des métriques étendues (test statistique)

Dans cette section, nous sommes intéressés à comparer les résultats de DCd et DCd* d'un coté et ceux de DCi et DCi* de l'autre. L'objectif étant de voir si la différence apportée par le nouveau critère introduit est statistiquement significative.

Notre but est donc de démontrer que DCd* et DCi* sont plus significatives que DCd et DCi et permettent de capturer plus de paires de méthodes reliées.

Pour valider cette hypothèse, nous utilisons un test statistique approprié : le Student's t-test [Hin 03] :

- Soit $\mu 1$ la valeur moyenne de DCd* (ou DCi*).
- Soit $\mu 2$ la valeur moyenne de DCd (ou DCi).

Nous avons à présent les 2 hypothèses suivantes :

H0 : $\mu 1 = \mu 2$ les métriques sont équivalentes.

H1 : $\mu 1 > \mu 2$ DCd* (DCi*) est plus significative que DCd (DCi).

Soit Diff la valeur ($\mu 1 - \mu 2$). Le test ci-dessus est équivalent à :

H0 : Diff = 0.

H1 : Diff >0.

Le test statistique est :

$Z = d / [Sd / sqrt(N)]$

Avec d : la valeur moyenne de l'échantillon Diff

Sd : la déviation standard de l'échantillon Diff et

N : le nombre de classes qui composent l'échantillon.

Les tableaux IV et V représentent respectivement les comparaisons entre DCd et DCd* d'un coté et DCi et DCi* de l'autre.

Tableau IV

Comparaison entre DCd et DCd*

Systèmes	Des. Stat	DCd	DCd*	Diff	Z	Zα
Jiu1	Moyenne	0,16027	0,17384	0,01356	1,799	2,326
	Sdt .dev	0,13686	0,1378	0,01685		
Jiu2	Moyenne	0,2497	0,2635	0,0228	2,4635	2,326
	Sdt.dev	0,16466	0,1714	0,0207		
Fujaba	Moyenne	0,01597	0,05244	0,03646	2,6547	2,326
	Sdt.dev	0,01479	0,05861	0,05663		
Wbemservices	Moyenne	0,08138	0,2286	0,1472	4,7917	2,326
	Sdt.dev	0,14164	0,2051	0,1869		

Tableau V

Comparaison entre DCi et DCi*

Systèmes	Des. Stat	DCi	DCi*	Diff	Z	Zα
Jiu1	Moyenne	0,1922	0,2178	0,0255	1,620	2,326
	Sdt.dev	0,1638	0,2178	0,0352		
Jiu2	Moyenne	0,3102	0,3350	0,02485	1,5498	2,326
	Sdt.dev	0,2292	0,2246	0,0358		
Fujaba	Moyenne	0,0207	0,0656	0,0448	2,6494	2,326
	Sdt.dev	0,0201	0,0739	0,0697		
Wbemservices	Moyenne	0,1013	0,2747	0,1734	5,7969	2,326
	Sdt.dev	0,1678	0,2332	0,1819		

La procédure consiste à comparer Z, pour chaque système, à la valeur quantile Zα (la valeur de α choisie est 0.05). Si la valeur de Z est supérieure à la valeur Zα, on rejette l'hypothèse H0 : Diff = 0 et on accepte l'hypothèse H1 : Diff > 0. Dans ce cas, le test statistique sera significatif et nous pourrons conclure que la métrique DCd* (DCi*) est plus significative que la métrique DCd (DCi). Cela signifie que le critère rajouté est significatif et permet de capturer un aspect additionnel des propriétés de classes. Nous avons collecté les données des métriques à partir des systèmes sélectionnés et calculés Diff et Z pour ces systèmes. Ces résultats sont présentés dans les tableaux IV et V.

Ils montrent clairement pour la majorité des systèmes testés que Z est supérieur à Zα. Les systèmes pour lesquels Z est inferieur à Zα sont des systèmes où la valeur de n est assez faible. Dans l'ensemble, les résultats montrent que DCd* (DCi*) est plus significatif que DCd (DCi).

4.6 Conclusion

Cette partie a porté essentiellement sur la validation statistique de la pertinence du nouveau critère de cohésion. Les résultats obtenus confirment nos hypothèses. Ils montrent que les métriques de cohésion étendues, basées sur l'ajout du critère proposé capturent plus de paires de méthodes reliées que les métriques DCd et DCi existantes.

Après cette première étape de validation statistique, il nous reste maintenant à les valider au complet comme étant reliées au couplage ou aux changements, pour les considérer empiriquement comme indicateurs de changeabilité.

CHAPITRE 5

RELATION ENTRE LE COUPLAGE ET LA COHÉSION

5.1 Introduction

Une croyance, largement tenue dans la communauté du génie logiciel, déclare qu'une cohésion élevée est liée à un bas couplage et vice versa [Pre 01]. Imran Baig dans [Bai 04] montre qu'une relation inverse existe entre sa métrique de cohésion et trois types de métriques de couplage (i.e. association, héritage et dépendance). Cependant, il mentionne que cette relation inverse entre la cohésion de classe et les métriques de couplage de classes n'est pas toujours garantie. Avec les systèmes étudiés, on arrive effectivement à observer visuellement cette dépendance en faisant des séries de courbes de distribution de la cohésion et du couplage. Les figures ci-dessous en sont des illustrations pour quelques systèmes étudiés:

Système 1 : *Gnujsp*

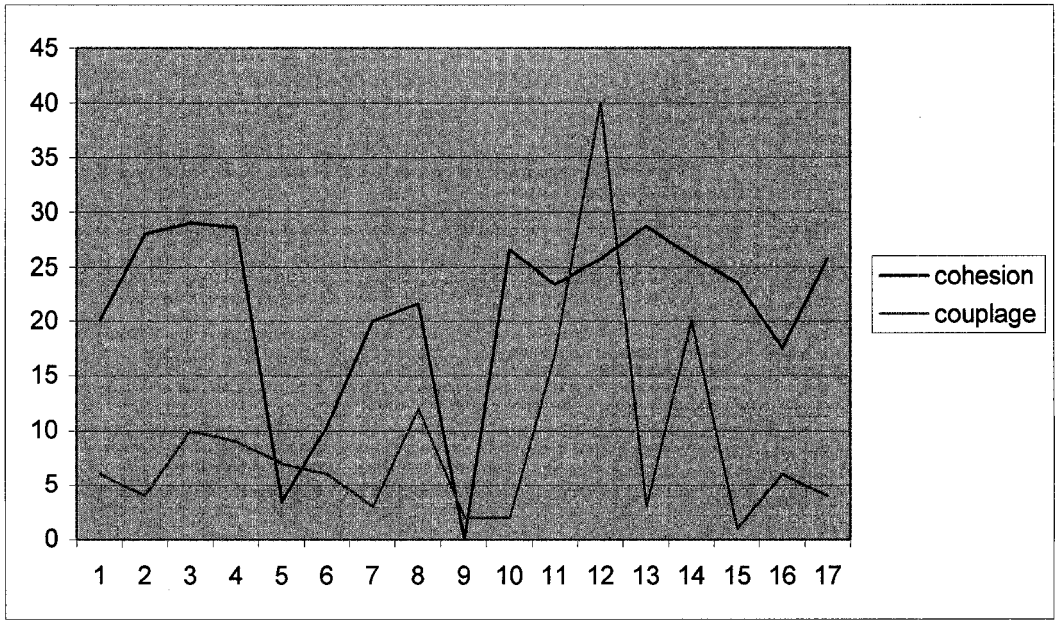


Figure 6 Courbes de distribution couplage – cohésion dans Gnujsp.

Système 2: *fujabaUml*

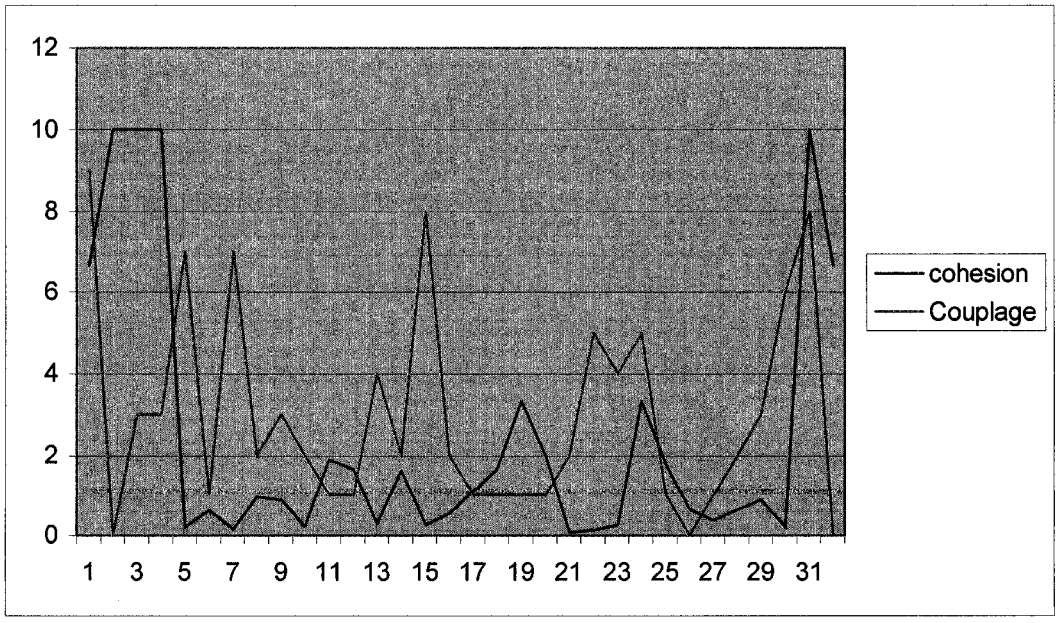


Figure 7 Courbes de distribution couplage - cohésion dans Fujaba.

Système 3 : *Jiu*

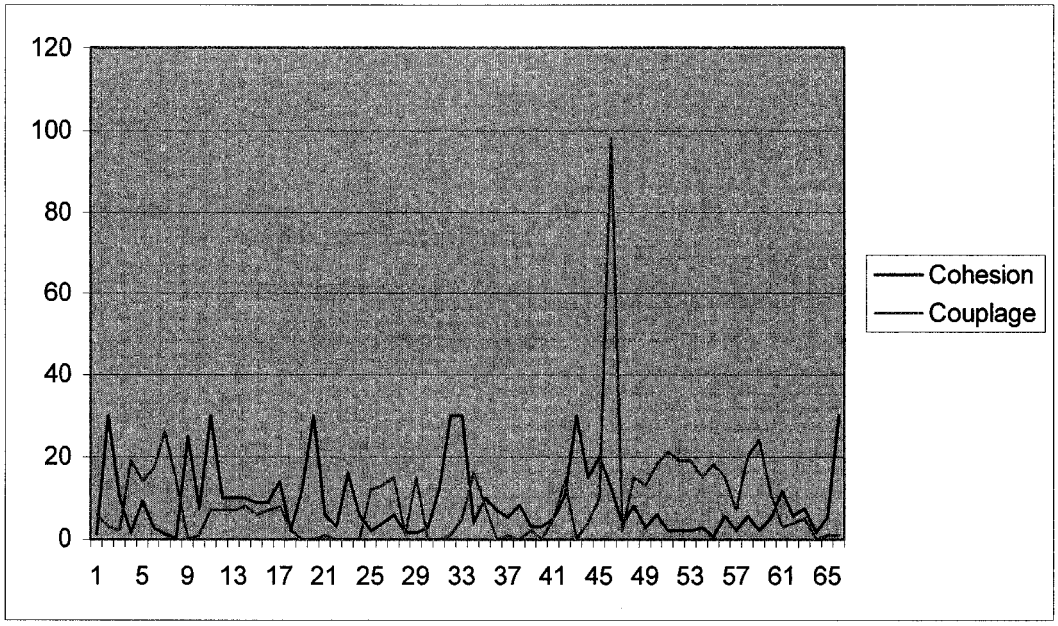


Figure 8 Courbes de distribution couplage – cohésion dans Jiu.

Système 4 : *Moneyjar*

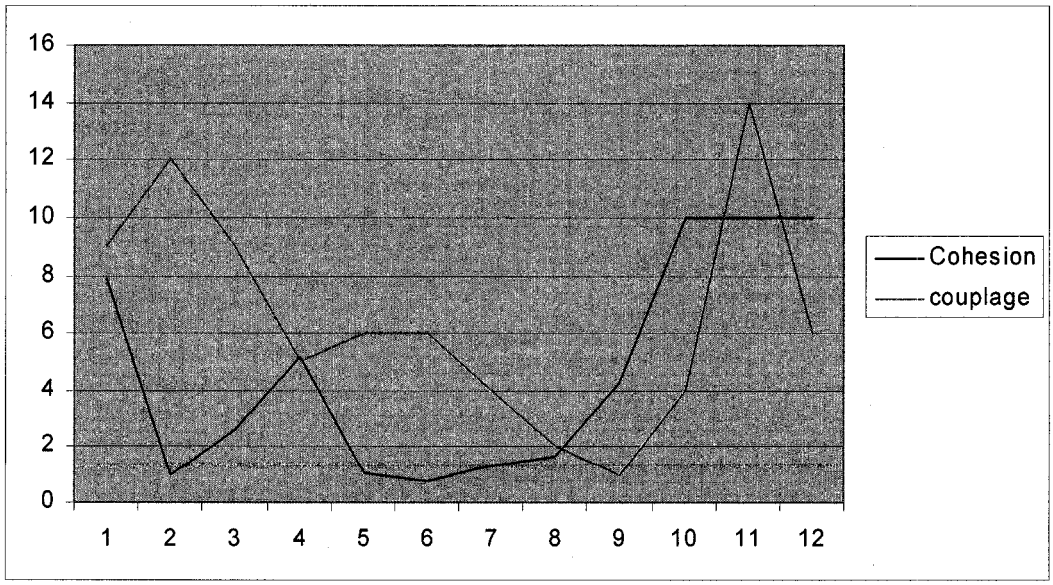


Figure 9 Courbes de distribution couplage – cohésion dans Moneyjar.

Système 5: *JexcelApi*

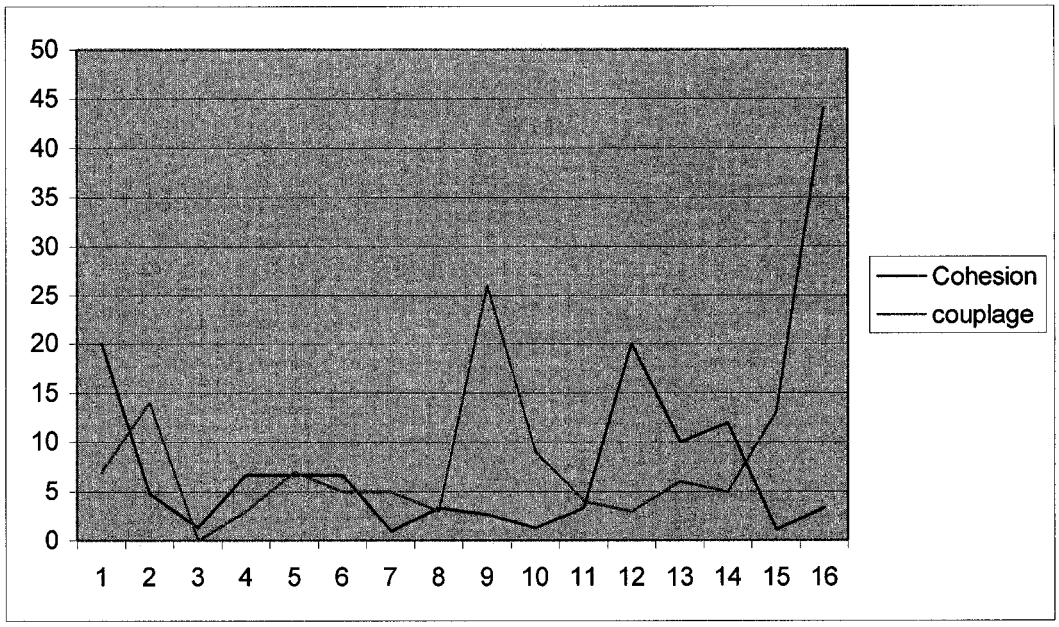


Figure 10 Courbes de distribution couplage – cohésion JexcelApi.

5.2 Procédure expérimentale

Dans ce qui suit, nous allons tenter d'apporter une explication et une manière de vérifier la relation pouvant exister entre les métriques de cohésion et celles de couplage.

Pour tester l'hypothèse à savoir si la cohésion est corrélée au couplage, nous adoptons les métriques de cohésion connues comme TCC, LCC, DCi, DCd, les DC étendues et la métrique de couplage CBO.

Dans l'expérimentation, nous extrayons d'abord les métriques sélectionnées des systèmes testés et par la suite nous utilisons le coefficient de Spearman (les statistiques de rang). Des analyses de jeux de données sont effectuées en calculant les coefficients de

dépendance de Spearman pour chaque paire (Métrique de cohésion, Métrique de couplage).

La statistique de Spearman est basée sur les rangs des observations. Ainsi des colonnes Ri et Si sont créées et contiennent les rangs de chaque observation. La valeur de la statistique de Spearman est un nombre compris entre -1 et 1, -1 correspondant à une dépendance négative parfaite et +1 à une dépendance positive parfaite.

5.3 Environnement

Dans cette section, nous présentons en premier les systèmes testés dans l'expérimentation. Puis, l'environnement dans lequel l'expérimentation a été conduite est décrit. Six systèmes Java ont été considérés pour évaluer les différentes métriques de cohésion et de couplage proposées et démontrer la validité de la dépendance entre la cohésion et le couplage. Ils varient en taille (nombre de classes) et en domaine d'application.

Le premier système testé est *Gnujsp 1.0.1*, *GNUJSP* est une implémentation gratuite de Java Server Pages de Sun. (http://klomp.org/gnujsp).

Le second système testé est *jiu 0.12*, *JIU* (Java Imaging Utilities) est une bibliothèque en Java pour le chargement, l'édition, l'analyse et la sauvegarde de pixel de fichiers images (http://sourceforge.net/projects/jiu).
Le troisième système testé est *fujabaUml 4, FujabaUML* est un outil de développement de logiciel qui permet d'étendre facilement UML et le développement Java par l'ajout de plug-ins (http://www.fujaba.de).

Le quatrième système testé est *jexcelapi 2.6, JExcelApi* est une bibliothèque de Java qui donne la capacité de lire, écrire, et modifier des feuilles de Microsoft Excel. (http://sourceforge.net/projects/jexcelapi).

Le cinquième système testé est *moneyjar 0.8, Moneyjar* est une librairie Java pour des applications financières. Elle simplifie la gestion de fortunes, la conversion de devises, le calcul d'impôts et la gestion de factures (http://sourceforge.net/projects/moneyjar).

Le dernier système testé est *wbemservices 1.0.0, Wbemservices* est une implémentation open source Java de Web Based Enterprise Management (WBEM) pour des applications commerciales et non commerciales. C'est un projet composé d'APIs, de serveur, d'applications clientes et d'outils (http://wbemservices.sourceforge.net/).

La table 3 fournit le nombre de classes de chaque système testé.

Tableau VI

Nombre de classes des systèmes testés.

Systèmes testés	Nombre de classes des systèmes
Gnujsp	56
jiu	77
fujabaUml	60
jexcelapi	110
moneyjar	20
wbemservices	180

Pour calculer les métriques utilisées dans l'expérimentation, nous utilisons un outil développé avec JavaCC, un parseur, qui analyse le code des systèmes testés. L'information issue du parseur contient les données de toutes les classes du système. L'information capturée est livrée sous forme de métriques.

Pour ce qui est de la métrique de couplage utilisée dans notre étude, nous utilisons l'outil de développement Togheter developer de Borland. L'outil Together developer analyse le système comme données d'entrée et nous livre la métrique de couplage en sortie.

Une fois l'ensemble des métriques collecté, elles sont exportées vers les outils SAS (Statistical Analysis System). SAS a pour vocation de collecter les informations provenant des différents systèmes opérationnels, d'effectuer sur elles des traitements statistiques classiques, de présenter les résultats entre autres de façon graphique et aussi clairement que possible afin de produire une bonne interprétation.

5.4 Résultats

5.4.1 Étude de régression

Une étude de régression entre le couplage et les différentes métriques de cohésion a été effectuée. Chaque métrique de cohésion est associée à la métrique de couplage afin de faire une régression simple entre ces deux variables. Le but de cette analyse statistique est d'identifier toute relation entre les métriques de cohésion et le couplage.

Quelques termes utilisés dans ce rapport nécessitent d'être définis. Ces termes sont définis comme suit :

Modèle de régression: c'est le modèle de régression utilisé. Les variables indépendantes sont les métriques de cohésion DCdE, DCiE, DCd, DCi, TCC, LCC et la variable dépendante de couplage CBO.

Variable dépendante: Une variable aléatoire à prédire.
Variable indépendante : une variable prédictive.

R2 (r-square) : le pourcentage de variance de la variable dépendante expliqué par les variables indépendantes dans le modèle de régression pour échantillon de la population.

Population : l'ensemble des classes prises en compte au niveau du test.

Adjusted R-square: le pourcentage de variance de la variable dépendante expliquée par les variables independantes dans un modèle de régression pour une population.

Sum of squares of regression: la variance de la variable dépendante expliquée par le modèle de régression.

Sum of squares of residual: la variance non expliquée dans la variable dépendante.

Mean squares of residual: la somme des carrés des résidus divisée par le degré de liberté des résidus.

Le tableau qui suit donne les valeurs de R2 obtenus.

Tableau VII

Valeur de R2 au niveau des différents systèmes.

Système	Métrique de cohésion	R2 vs Couplage
FujabaUml	DCdE	0.0118
	DCiE	0.0081
	DCd	0.0081
	DCi	0.0054
	TCC	0.0172
	LCC	0.0133
Système	Métrique de cohésion	R2 vs Couplage
Gnujsp	DCdE	0.2835
	DCiE	0.2676
	DCd	0.4657
	DCi	0.4506
	TCC	0.2835
	LCC	0.2676
Système	Métrique de cohésion	R2 vs Couplage
JIU	DCdE	0.0228
	DCiE	0.0267
	DCd	0.0186
	DCi	0.0221
	TCC	0.0236
	LCC	0.0274
Système	Métrique de cohésion	R2 vs Couplage
Moneyjar	DCdE	0.0226
	DCiE	0.0237
	DCd	0.0320
	DCi	0.0331
	TCC	0.0230
	LCC	0.0241

Pour étudier une autre variante de cette relation entre métriques de cohésion et métriques de couplage, une autre variable qui est le logarithme de la valeur de couplage a été définie. Une régression entre ce logarithme et la valeur de la cohésion est effectuée. Les résultats se présentent comme suit (tableau VIII):

Tableau VIII

R2 obtenu par log du couplage.

Système	Métrique de cohésion	R2 vs log Couplage
FujabaUml	DCdE	0.0027
	DCiE	0.0012
	DCd	0.0019
	DCi	0.0008
	TCC	0.0035
	LCC	0.0015
Système	Métrique de cohésion	R2 vs log Couplage
Gnujsp	DCdE	0.0032
	DCiE	0.0066
	DCd	0.0628
	DCi	0.0504
	TCC	0.0032
	LCC	0.0066
Système	Métrique de cohésion	R2 vs log Couplage
JIU	DCdE	0.0341
	DCiE	0.0430
	DCd	0.0459
	DCi	0.0545
	TCC	0.0372
	LCC	0.0464
Système	Métrique de cohésion	R2 vs log Couplage
Moneyjar	DCdE	0.0150
	DCiE	0.0148
	DCd	0.0168
	DCi	0.0162
	TCC	0.0151
	LCC	0.0148

5.4.2 Étude de la corrélation de Spearman (statistique de rang)

Nous mesurons les valeurs de corrélation entre les métriques de cohésion et de couplage pour l'ensemble des systèmes sélectionnés. La table IX nous fournit quelques descriptions statistiques de ces systèmes.

Tableau IX

Résultats de la méthode des statistiques de rang (Spearman).

Système	Statistique	Dcie-cbo	Dcde-cbo	Dci-cbo	Dcd-cbo	Lcc-cbo	Tcc-cbo
Gnujsp	Coef de Spearman	-0.354545	-0.35892	-0.35455	-0.35892	-0.44101	-0.43982
	Stat du test	-2.786373	-2.8258	-2.78637	-2.8258	-3.61087	-3.59872
	P-value	0.0036697	0.003299	0.00367	0.003299	0.000334	0.000347
jiu	Coef de Spearman	-0.50857	-0.47888	-0.50337	-0.47584	-0.48699	-0.44084
	Stat du test	-5.11527	-4.72409	-5.04502	-4.68533	-4.8287	-4.25334
	P-value	1.17E-06	5.27E-06	1.53E-06	6.11E-06	3.54E-06	3E-05
fujabaUml	Coef de Spearman	-0.425590442	-0.43809	-0.29225	-0.31195	-0.29025	-0.29894
	Stat du test	-3.5817696	-3.71155	-2.3273	-2.5005	-2.30989	-2.38579
	P-value	0.000349459	0.000232	0.011731	0.007625	0.012237	0.010164
jexcelapi	Coef de Spearman	-0.18723	-0.22039	-0.19318	-0.21987	-0.27665	-0.29179
	Stat du test	-1.98076	-2.34805	-2.04612	-2.3423	-2.99185	-3.17031
	P-value	0.02508	0.010346	0.021587	0.010499	0.001718	0.000991
moneyjar	Coef de Spearman	-0.00602	-0.01955	-0.02105	-0.03459	-0.31128	-0.31128
	Stat du test	-0.02552	-0.08295	-0.08934	-0.14683	-1.38968	-1.38968
	P-value	0.48996	0.467402	0.4649	0.442451	0.090788	0.090788
wbemservices	Coef de Spearman	-0.242732	-0.295322901	-0.26708	-0.3055	-0.36385	-0.40613
	Stat du test	-3.338282	-4.124041493	-3.69767	-4.28049	-5.21164	-5.92951
	P-value	0.0005133	2.8523E-05	0.000145	1.52E-05	2.57E-07	7.72E-09

5.5 Analyse

Le but de cette expérimentation était de trouver un lien, une corrélation entre les métriques de cohésion et de couplage sélectionnées. Nous rappelons que cette question a déjà fait l'objet de quelques rares travaux qui n'ont pu, en fonction des expérimentations effectuées, valider ce lien. À notre connaissance, aucun travail de recherche n'a validé ce lien. Ceci n'a pas empêché la communauté génie logiciel de considérer ce lien de façon intuitive et largement compréhensible.

Concernant la première expérimentation, c'est-à-dire celle de l'analyse de régression, nous nous basons sur les valeurs de la valeur statistique R2 afin d'interpréter la relation liant éventuellement le couplage et la cohésion. Par exemple, pour le système JIU au niveau du tableau VII, les valeurs 0.0228 et 0.0267 respectivement des métriques DCdE et DCiE représentent les pourcentages de variance du couplage expliqués par les métriques de cohésion. Ainsi 2.28% et 2.67% de la variance du couplage est expliqué respectivement par les métriques de cohésion DCdE et DCiE. Concernant le tableau VIII, pour le même système JIU, les valeurs 0.0341 et 0.0430 respectivement des métriques DCdE et DCiE représentent les pourcentages de variance du logarithme du couplage expliqués par les métriques de cohésion. Ainsi 3.41% et 4.3% de la variance du logarithme du couplage est expliqué respectivement par les métriques de cohésion DCdE et DCiE

Avec la deuxième expérimentation, pour vérifier si la corrélation était significativement (en termes statistiques) plus petite que 0 (dépendance négative), un test a été effectué. La statistique du test qui doit par la suite être comparée à une variable de Student à n-2 degrés de liberté, n étant la taille de l'échantillon, est calculée.

La p-value indique la probabilité d'obtenir une telle valeur sous l'hypothèse d'absence de dépendance. En général, si p-value < 0.05, on conclut à une dépendance négative significative.
Ainsi, pour l'ensemble des systèmes testés au niveau du tableau IX, seul le système *moneyjar* présente des valeurs de p-value > 0.05 pour toutes les combinaisons métriques de cohésion – métrique de couplage. Nous observons les valeurs de corrélation 0.48996, 0.4649, 0.442451, 0.090788 et 0.090788 pour respectivement les métriques DCiE, DCdE, DCi, DCd, LCC,TCC par rapport au métrique de couplage CBO.

Pour le reste des systèmes étudiés, les valeurs de p-value sont toutes < 0.05 pour l'ensemble des combinaisons métrique de cohésion - métrique de couplage. Donc, tous les systèmes au niveau du tableau IX présentent une dépendance négative significative entre la cohésion et le couplage à l'exception du système *moneyjar*. Une explication possible est qu'au niveau du tableau VI, c'est-à-dire celui du nombre de classes des systèmes testés, seul le système *moneyjar* dispose d'un nombre de classes inferieur à 50 : 20 classes. Donc pour observer une dépendance négative significative, il serait intéressant de considérer des systèmes ayant un nombre de classes élevé.

5.6 Conclusion

D'après les résultats obtenus, il y'a bien une forte dépendance entre les métriques de cohésion et la métrique de couplage. Plus le nombre de classes dans un système augmente plus la relation de dépendance entre cohésion et couplage est visible. Ceci montre que cette relation de dépendance est non linéaire. Bien que les résultats obtenus, selon les divers systèmes considérés, confirment nettement la relation qui existe entre le couplage et la cohésion (lorsque l'un augmente l'autre diminue), il est tout de même nécessaire de continuer à explorer cette relation pour de nombreux autres systèmes pour arriver à tirer des conclusions plus globales.

CHAPITRE 6

L'IMPACT DE CHANGEMENT

6.1 Introduction

Un impact est l'effet ou l'impression d'une chose sur une autre. Dans notre contexte, l'impact peut être vu comme la conséquence d'un changement. Les effets quantitatifs et qualitatifs d'un changement sont étudiés dans l'analyse de l'impact. L'analyse de l'impact est une activité importante de la maintenance dont l'objectif est de déterminer l'étendue d'une requête de changement. L'analyse de l'impact de changement estime les éléments au niveau code source et la documentation affectés lorsqu'un changement est effectué. Elle est définie comme le processus d'évaluation des effets d'un changement proposé sur d'autres composants du système [Boh 96, Law 03, Kab 01].

Bohner et Arnold [Boh 96] définissent l'analyse de l'impact de changement comme l'identification des conséquences potentielles d'un changement ou l'estimation de ce qui nécessite une modification afin d'accomplir un changement.

L'analyse de l'impact peut contribuer à mesurer le coût d'un changement. Lorsque ce dernier cause plusieurs autres changements, le coût total augmente. Une analyse préliminaire du changement permet d'estimer le coût et d'aider les gestionnaires à choisir le compromis entre les changements alternatifs. Un changement dont l'impact est grand mérite d'être examiné une seconde fois afin de déterminer s'il est valable.

L'analyse de l'impact de changement peut être utilisée pour indiquer la vulnérabilité des sections critiques du code source. Lorsqu'une section fournit des fonctionnalités

critiques en plus de dépendre de plusieurs parties du système, elle est amenée à dévier de sa fonctionnalité première quand des changements sont effectués [Kab 01].
L'analyse de l'impact peut être aussi employée pour réaliser les tests de régression; déterminer les parties du système qui nécessitent d'être re-testées après l'application d'un changement. Les tests de régression sont une activité de la maintenance. Ils se rapportent aux tests répétés, destinés à prouver que le comportement du système est inchangé.

Nous nous intéresserons surtout à ce dernier point c'est-à-dire, à travers des tests statistiques, à quel point l'impact d'un changement affecte les propriétés d'un système. A présent quelques modèles de changements en rapport avec notre étude vont être présentés.

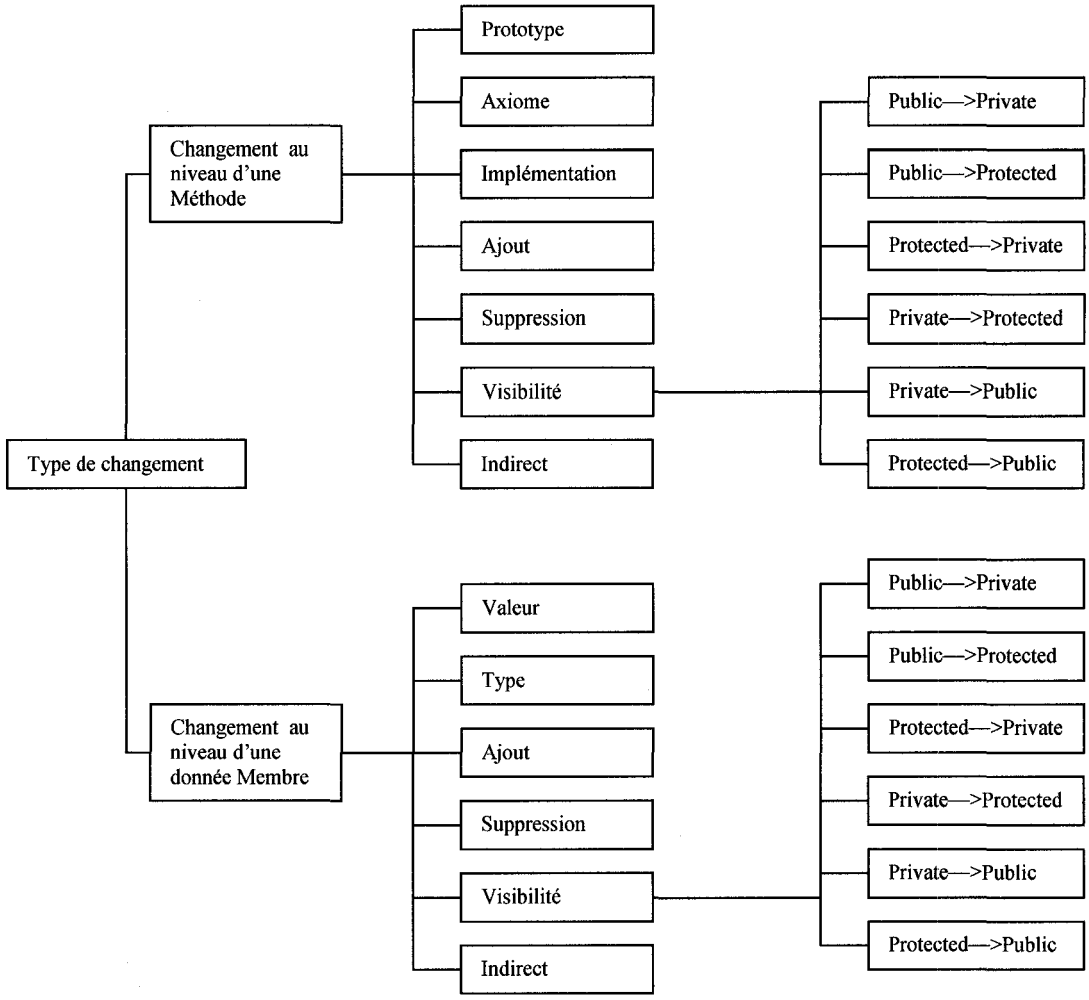
6.2 Les modèles de changements

Dans la littérature, plusieurs modèles de changements ont été proposés. Certains d'entre eux suscitent beaucoup d'intérêts pour notre travail. Les principaux changements considérés lors de ces travaux sont présentés ci-dessous :

Li et Henry [Li 93] analysent un nombre possible de changements dans les systèmes orientés objet et comment ces changements affectent les classes des systèmes. Le tableau ci dessus représente les différents types de changements considérés. Au niveau abstraction, ils sont au nombre de treize. Six pour les données membres et sept pour les méthodes.

Tableau X

Changements considérés par Li et Henry



Kung et al. [Kun 94] discutent des types de changement pouvant être appliqués à une librairie orientée objet. Ils décrivent aussi une méthode pour identifier les classes affectées suite à un changement dans une classe.

Tableau XI

Les différents changements considérés par Kung.

Components		Changes	
Data	component changes	1	change data definition/declaration/uses
		2	change data access scope/mode
		3	add/delete data
Method	interface changes	4	add/delete external data usage
		5	add/delete external data updates
		6	add/delete/change a method call/a message
		7	change its signature
	structure changes	8	add/delete a sequential segement
		9	add/delete/change a branch/loop
	component changes	10	change a predicate
		11	add/delete/change local data
		12	change a sequential segement
Class	component changes	13	change a defined/redefined method
		14	add/delete a defined/redefined method
		15	add/delete/change a defined data attribute
		16	add/delete a virtual abstract method
		17	change an attribute access mode/scope
	relationship changes	18	add/delete a superclass
		19	add/delete a subclass
		20	add/delete an object pointer
		21	add/delete an aggregated object
		22	add/delete an object message
Class Library	component changes	22	change a class (defined attributes)
	relationship changes	23	add/delete a relation between classes
		24	adde/delete a class and its relations
		25	add/delete an independent class

Kabaili et al. [Kab 01], dans leur travail, ont d'abord étudié la relation entre la cohésion et le couplage, puis la relation entre la cohésion et les impacts de changements. Ils sont

passés à la deuxième suite à un échec lors d'une première tentative de validation de la relation couplage-cohésion. Les changements considérés dans cette étude sont listés dans le tableau ci-dessous.

Tableau XII

Changements considérés par Kabaili

	Changement	Expression d'impact
1.	Type de variable	S + L
2.	Portée d'une variable de public à protected	SH'F'
3.	Signature de méthode	I + L
4.	Portée d'une méthode de public à protected	H'IF'
5.	Dérivation de classe de public à protected	H'F' (S+I)
6.	Ajout de classe abstraite dans la structure d'héritage	S+G+H+I+L

Abdi et al. [Abd 06] adaptent le modèle complet, en C++, développé dans le cadre du projet Spool [Cha 98A, Kab 01] au langage Java. Cette adaptation donne une liste de 52 changements susceptibles de se produire dans un système java (tableau XIII).

Tableau XIII

Changements pour un système java

Change Id	Change Description	Impact Expression
v.1.1	Variable value change	-
v.1.2	Variable type change	S+L
v.1.3	Variable addition	-
v.1.4	Variable deletion	S+L
v.1.5	Variable scope change	
v.1.5.1	Public → Private	S
v.1.5.2	Public → Protected	S~H
v.1.5.3	Protected → Private	SH
v.1.5.4	Protected → Public	-
v.1.5.5	Private → Public	-
v.1.5.6	Private → Protected	-
v.1.6	Variable change (Static/Non-static)	
v.1.6.1	Static → Non-static	S+L
v.1.6.2	Non-static → Static	-
m.2.1	Method change (Static/Non-static)	
m.2.1.1	Static → Non-static	I+L
m.2.1.2	Non-static → Static	L
m.2.2	Method change (Abstract/Non-abstract)	
m.2.2.1	Abstract → Non-abstract	H+ie(3.1.2)+L
m.2.2.2	Non-abstract → Abstract	H+ie(3.1.1)+L
m.2.3	Method Return type change	
m.2.3.1	Non-abstract method	H+ie(3.1.2)+L
m.2.3.2	Abstract method	H+L
m.2.4	Method implementation change	L
m.2.5	Method signature change	
m.2.5.1	Non-abstract method	I+ie(3.1.2)+L
m.2.5.2	Abstract method	H+L
m.2.6	Method Scope change	
m.2.6.1	Public → Private	
m.2.6.1.1	Non-abstract method	I
m.2.6.1.2	Abstract method	-
m.2.6.2	Public → Protected	
m.2.6.2.1	Non-abstract method	~H I
m.2.6.2.2	Abstract method	-
m.2.6.3	Protected → Private	
m.2.6.3.1	Non-abstract method	H I
m.2.6.3.2	Abstract method	-
m.2.6.4	Protected → Public	
m.2.6.4.1	Non-abstract method	-
m.2.6.4.2	Abstract method	-
m.2.6.5	Private → Public	
m.2.6.5.1	Non-abstract method	-
m.2.6.5.2	Abstract method	
m.2.6.6	Private → Protected	
m.2.6.6.1	Non-abstract method	-
m.2.6.6.2	Abstract method	-
m.2.7	Method addition	
m.2.7.1	Abstract method	ie(3.1.1)
m.2.7.2	Non-abstract method	I+ie(3.1.2)+L
m.2.8	Method deletion	
m.2.8.1	Abstract method	ie(3.1.2)
m.2.8.2	Non-abstract method	I + ie(3.1.1)+ L
c.3.1	Classe change (Abstract/Non-abstract)	
c.3.1.1	Non-abstract → Abstract	G+H+I+L
c.3.1.2	Abstract → Non-abstract	H+L
c.3.2	Classe deletion	
c.3.2.1	Non-abstract class	S+G+H+I
c.3.2.2	Abstract class	S+H+I
c.3.4	Class inheritance derivation	
c.3.4.1	Public → Private	S+I
c.3.4.2	Public → Protected	~H(S+I)
c.3.4.3	Protected → Private	H(S+~SG+~S I)
c.3.4.4	Protected → Public	-
c.3.4.5	Private → Public	-
c.3.4.6	Private → Protected	-
c.3.5	Class addition	-
c.3.6	Class inheritance structure	
c.3.6.1	Abstract class addition	S+G+H+I+ ie(3.1.1)+ L
c.3.6.2	Non abstract class addition	H+L
c.3.6.3	Abstract class deletion	H+ie(3.1.2)+ L
c.3.6.4	Non abstract class deletion	H+L

6.3 Le Modèle retenu

Lorsqu'un changement est considéré, il est nécessaire d'identifier les composantes du système qui seront affectés. Notre but est d'étudier comment le système réagit à un changement. Selon [Abd 06], en général, un système absorbe facilement un changement si le nombre de composantes affectées est faible.

Un système est vu comme un ensemble de classes connectées par des liens. Une classe est définie comme un groupe de méthodes définissant le comportement des objets de la classe et une section de variables définissent l'état des objets de la classe. Un composant correspond à une classe, une méthode ou une variable. Comme exemples de changement, on peut avoir une suppression d'une variable, un changement de portée d'une méthode de « public » à « protected ».

Le tableau ci-dessous, tiré de [Abd 06], montre les principaux changements dans les systèmes orientés objet, à un niveau de conception. Ils sont catégorisés selon le composant qu'ils affectent et un total de 13 changements sont identifiés.

Tableau XIV

Les principaux changements au niveau abstraction.

Composante	Description du changement
Variable	Changement de type
	Changement de portée
	ajout
	suppression
Méthode	Changement de type de retour
	Changement d'implémentation
	Changement de signature
	Changement de portée
	ajout
	suppression
Classe	Changement de structure d'héritage
	ajout
	suppression

La majorité des études sur l'impact de changement porte sur l'étude des liens [Kab 01, Abd 06] entre les classes d'un système. Dans la plupart de ces études, le couplage est souvent utilisé afin de quantifier les effets probables d'un changement.

Dans notre approche, nous allons plutôt considérer la cohésion comme indicateur de changeabilité. La plupart des études ont porté sur le couplage [Cha 98A, Kab 01, Abd 06]. Il serait intéressant de voir un autre indicateur. Aussi le couplage est difficile à mesurer et consomme du temps du fait qu'il constitue une propriété interclasses. En effet, pour la mesurer, la connaissance de l'ensemble du système et de tous les liens

entre les classes doit être maitrisée. La cohésion de classe, qui est une propriété intra-classe, nécessite juste de considérer la classe étudiée pour la mesurer.

Les métriques de cohésion des chapitres 3 et 4 utilisent dans leurs définitions les variables d'instance et les méthodes. Ceci motive notre modèle à considérer uniquement les changements en rapport avec les définitions des métriques de cohésion à savoir les changements au niveau des variables d'instance et des méthodes (tableau XV).

Tableau XV

Les principaux changements sur les variables et les méthodes

Composante	Description du changement
Variable	Changement de type
	Changement de portée
	ajout
	suppression
Méthode	Changement de type de retour
	Changement d'implémentation
	Changement de signature
	Changement de portée
	ajout
	suppression

Au niveau du tableau ci-dessus, concernant les variables, un changement de type ou de portée n'a aucun effet sur le calcul des métriques de cohésion retenues. Même constat, pour les méthodes, quand il s'agit d'un changement de type de retour ou d'implémentation. Le tableau suivant présente les types de changement considérés pour les métriques de cohésions retenus.

Tableau XVI

Les principaux changements pour la cohésion

Composante	Changement
Variable	ajout
	suppression
méthode	Changement de signature
	Changement de portée
	ajout
	suppression

CHAPITRE 7

ENVIRONNEMENT DES EXPÉRIMENTATIONS

7.1 Description des environnements

Un analyseur de métriques a été construit pour collecter des métriques de cohésion OO à partir de codes sources écrits en Java. L'analyseur a été implémenté avec JavaCC dans un environnement Windows. Il s'exécute en deux étapes. La première consiste à parcourir une classe dans un système et à stocker les informations nécessaires la concernant. La seconde étape calcule les métriques à partir des données collectées lors de la première étape. Une fois les données métriques obtenues, l'outil SAS (Statistical Analysis System), qui a pour vocation de collecter les informations provenant des différent systèmes opérationnels, effectue sur celles-ci des traitements statistiques classiques et présente les résultats, entre autres de façon graphique, aussi clairement que possible afin de permettre une bonne interprétation. L'outil Excel de Microsoft a été également utilisé à des fins statistiques. En particulier, pour effectuer les statistiques de rang avec la corrélation de Spearman. L'architecture globale de l'environnement est donnée par la figure 11.

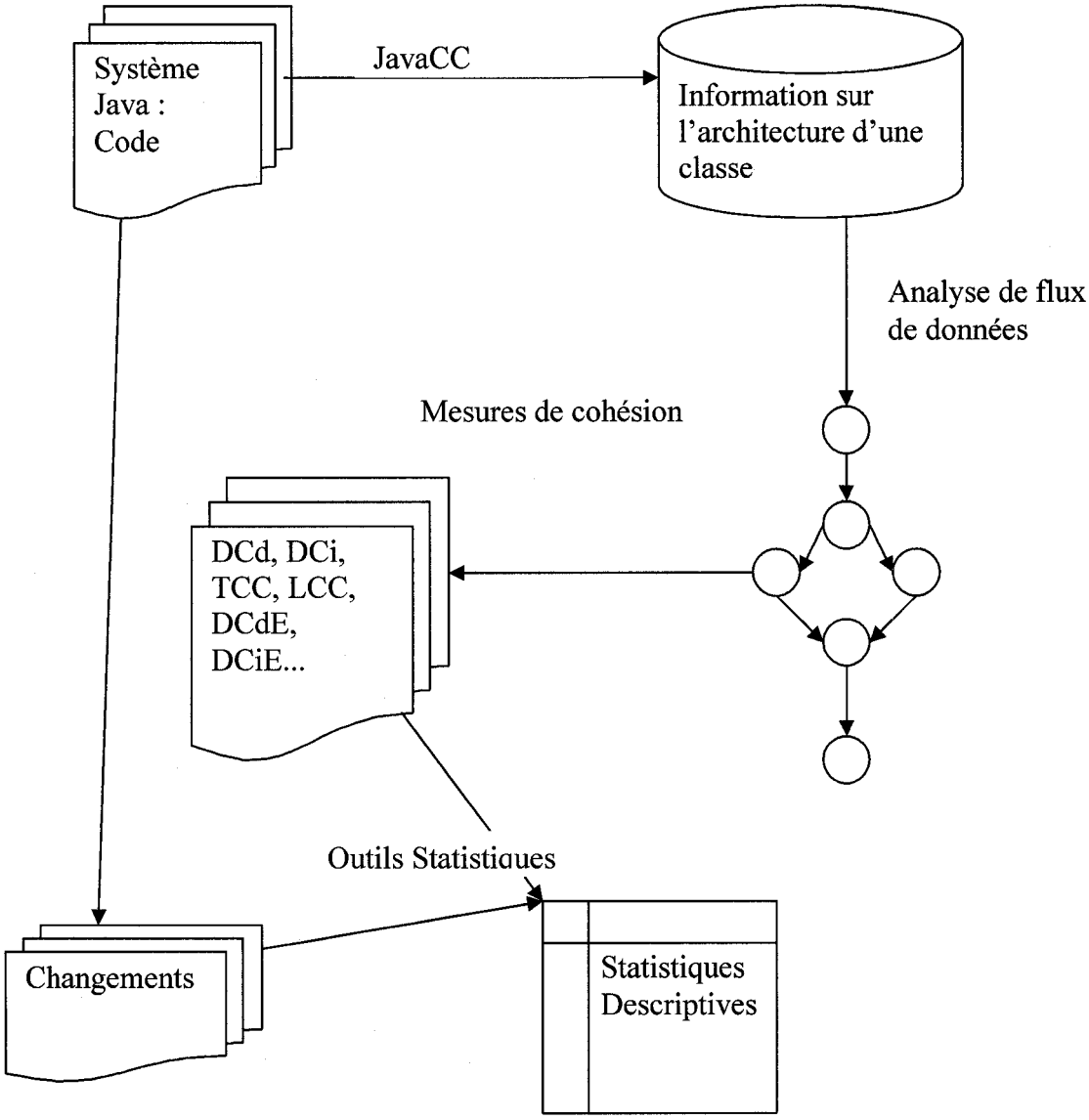


Figure 11 Architecture de l'environnement de calcul de l'impact.

7.2 Les systèmes Sélectionnés

Pour évaluer la changeabilité, nous avons procédé à l'analyse du code source Java de plusieurs systèmes.
Système 1 : *XXL* (eXtensile and fleXible Library : est une librairie Java qui contient une riche infrastructure pour implémenter une fonctionnalité de processus de requête avancée (advanced query processing functionality). (http://dbs.mathematik.uni-marburg.de/?search=Research_Projects_XXL). La première version (1.0) de ce système contient 623 classes. La deuxième version (1.1beta) de ce système contient 823 classes.

Système 2 : *Moneyjar* est une librairie Java pour des applications financières. Elle simplifie la gestion des fortunes, la conversion des devises, le calcul d'impôts et la gestion des factures. (http://sourceforge.net/projects/moneyjar). La première version (0.1) de ce système contient 33 classes. La deuxième version (0.8) de ce système contient 83 classes.

Système 3 : *FujabaUML* est un outil de développement logiciel qui permet d'étendre facilement UML et le développement Java par l'ajout de plug-ins. (http://www.fujaba.de). La première version (4) de ce système contient 186 classes. La deuxième version (5) de ce système contient 201 classes.

7.3 Sélections des métriques

Les métriques utilisées dans notre étude empirique peuvent être classées en trois groupes, à savoir métriques de taille, métriques de couplage et métriques cohésion. Comme métrique de taille, nous avons choisi LOC, la plus simple et la plus utilisée généralement dans la littérature des métriques de taille. Pour la métrique de couplage, nous avons considéré CBO (couplage beetwen Objects) définie par Chidamber et Kemerer [Dag 03, Chi 94]. Il s'agit d'une métrique qui ne tient pas compte de l'héritage. Le choix des métriques de cohésion a porté sur celles définies par Bieman et Kang c'est-

à-dire TCC (Tight Class Cohesion) et LCC (Lack Class Cohesion) [Bie 95] et celles définies par Badri et al. [Bad 03] : DCd et DCi. Les métriques DCdE et DCiE, définies dans le chapitre 4, qui constituent un raffinement des métriques DCd et DCi sont aussi considérées.

Tableau XVII

Liste des métriques retenues

Métriques	Type
DCd*	Cohésion Badri direct étendue
DCi*	Cohésion Badri indirect étendue
DCd	Cohésion Badri direct
DCi	Cohésion Badri indirect
TCC	Cohésion Bieman direct
LCC	Cohésion Bieman indirect
CBO	Couplage
LOC	Taille

7.4 Les données de changement

Les données de changement sont obtenues à partir de deux versions des systèmes étudiés. Les codes sources des deux versions sont comparés grâce à l'outil de comparaison qui est inclus dans Eclipse. Pour chaque classe modifiée, les changements sont répertoriés et classés selon notre modèle de changements défini dans le chapitre précédent.

7.5 Les données métriques

Notre analyseur, développé à partir de JavaCC, extrait les différentes métriques considérées des systèmes testés.

CHAPITRE 8

ÈTUDE(S) EMPIRIQUE(S)

8.1 Contexte – Objectifs

Une manière de valider la changeabilité des systèmes logiciels, ou de valider certaines métriques comme indicateurs de changeabilité, est de trouver quelques propriétés de conception, capturées par les métriques, et démontrer qu'il existe un lien entre ces métriques et la changeabilité. Dans ce cas, ces métriques peuvent être utilisées comme indicateurs de changeabilité. Dans le domaine des systèmes orientés objet, plusieurs expérimentations ont été conduites montrant par exemple que le couplage entre les classes est un bon indicateur de changeabilité. Chaumun et al. [Cha 99] ont observé une forte corrélation entre la changeabilité et quelques métriques de couplage, à travers différents systèmes industriels et à travers plusieurs types de changements.

La cohésion est une propriété intra classe. Pour la mesurer, nous avons juste besoin de considérer la classe étudiée. Partant d'une large croyance dans la communauté logicielle soutenant le fait qu'une forte cohésion est reliée à un faible couplage et que la plupart des études [Cha 98A, Kab 01, Dag 03, Abd 06] ont porté sur le couplage, nous avons décidé d'étudier la cohésion comme indicateur de changeabilité. Si cette investigation s'avère concluante, la validation de la cohésion comme indicateur de changeabilité (et son utilisation) serait moins couteuse que celle du couplage [Kab 01].

8.2 Corrélation entre les métriques

Pour comprendre les relations entre les métriques, nous avons crée et analysé des matrices de corrélation pour chaque système étudié. L'outil statistique SAS a été utilisé

pour créer les matrices de corrélation. Chaque matrice de corrélation a un coefficient de corrélation (r) pour chaque paire de métriques possible et peut varier entre -1 et 1. Lorsque la valeur de r est +1, elle représente une corrélation positive parfaite. Si la valeur est de -1, elle représente une corrélation négative parfaite. Toute valeur entre 0.7 et 1 est acceptée comme une corrélation positive forte, alors que toute valeur entre -1 et -0.7 est acceptée comme corrélation négative forte. Ces valeurs sont prises comme base de notre estimation.

Tableau XVIII

Matrice de corrélation pour XXL.

	y	ΔDCdE	ΔDCiE	ΔDCd	ΔDCi	ΔTCC	ΔLCC	ΔCBO	ΔLOC
y	1.00000	-0.01019	-0.00351	-0.07265	-0.08592	-0.06350	-0.07458	0.89128 <.0001	0.49817
ΔDCdE		1.00000	0.99655	0.99759	0.99366	0.98941	0.98880	-0.02968	-0.06623
ΔDCiE			1.00000	0.99314	0.99584	0.98018	0.98331	-0.02491	-0.06851
ΔDCd				1.00000	0.99653	0.99138	0.99114	-0.08052	-0.08748
ΔDCi					1.00000	0.98298	0.98674	-0.09418	-0.09602
ΔTCC						1.00000	0.99823	-0.06641	-0.08183
ΔLCC							1.00000	-0.08190	-0.09100
ΔCBO								1.00000	0.40361
ΔLOC									1.00000

Pour le système XXL, on note une corrélation significative entre les deltas de métrique de cohésion entre eux et une corrélation significative entre le changement et de delta du couplage avec une valeur de 0.89128.

Tableau XIX

Matrice de corrélation pour MoneyJar

	y	ΔDCdE	ΔDCiE	ΔDCd	ΔDCi	ΔTCC	ΔLCC	ΔCBO	ΔLOC
y	1.00000	0.08873	0.14468	0.00905	0.01747	-0.16455	-0.14244	-0.24304	0.72992
ΔDCdE		1.00000	0.99777	0.99555	0.99670	0.76288	0.76586	-0.22598	-0.37892
ΔDCiE			1.00000	0.98743	0.98965	0.74308	0.74912	-0.24930	-0.31997
ΔDCd				1.00000	0.99982	0.77835	0.77733	-0.21298	-0.45433
ΔDCi					1.00000	0.77405	0.77409	-0.22225	-0.44095
ΔTCC						1.00000	0.99828	0.00681	-0.70057
ΔLCC							1.00000	-0.01795	-0.66767
ΔCBO								1.00000	-0.20091
ΔLOC									1.00000

Pour le système MoneyJar, on note une corrélation significative entre les deltas de métrique de cohésion entre eux et une corrélation significative entre le changement et de delta ligne de code avec une valeur de 0.72992.

Tableau XX

Matrice de corrélation pour FujabaUml.

	y	ΔDCdE	ΔDCiE	ΔDCd	ΔDCi	ΔTCC	ΔLCC	ΔCBO	ΔLOC
y	1.00000	-0.09463	-0.09225	-0.09422	-0.14701	-0.1218	-0.20294	0.06543	0.33118
ΔDCdE		1.00000	0.62456	0.97018	0.59624	0.96320	0.60030	-0.40663	-0.20769
ΔDCiE			1.00000	0.57885	0.92742	0.55906	0.90857	-0.30866	-0.09104
ΔDCd				1.00000	0.63173	0.99092	0.63336	-0.37889	-0.19217
ΔDCi					1.00000	0.61227	0.98390	-0.25824	-0.04059
ΔTCC						1.00000	0.63775	-0.39209	-0.17767
ΔLCC							1.00000	-0.28118	-0.02573
ΔCBO								1.00000	0.03987
ΔLOC									1.00000

Pour le système FujabaUml, on note une corrélation significative seulement entre les deltas de métrique de cohésion entre eux.

Les observations que nous pouvons déduire à partir des valeurs issues des matrices de corrélations sont:

- Il y a une corrélation forte entre ΔCBO et le changement pour le système XXL.
- Il y a une corrélation forte entre ΔLOC et le changement pour le système Moneyjar.
- Il n'y a pas de corrélation entre les métriques de cohésion et les changements.
- Il n'y a pas de corrélation entre les métriques de cohésion et CBO.
- Il n'y a pas de corrélation entre les métriques de cohésion et LCO.
- Il y a une forte corrélation entre ΔDCdE, ΔDCd et ΔTCC.
- Il y a une forte corrélation entre ΔDCiE, ΔDCi et ΔLCC.

8.3 Modèle d'analyse de régression

Une régression se définit comme la modélisation et la prédiction de variables.
Une étude de régression entre le changement et les différentes métriques de cohésion, de couplage et de taille a été effectuée. Le but de cette analyse statistique est de déceler toute relation entre les métriques, spécialement celles de cohésion et le changement. Quelques termes utilisés dans ce qui suit nécessitent d'être définis.

Modèle de régression: c'est le modèle de régression utilisé. Les variables indépendantes sont les métriques de cohésion DCdE, DCiE, DCd, DCi, TCC, LCC , CBO et LOC et la variable dépendante, le changement.

Variable dépendante: Une variable aléatoire à prédire.

Variable indépendante : une variable prédictive.

R2 (r-square) : le pourcentage de variance de la variable dépendante expliqué par les variables indépendantes dans le modèle de régression pour échantillon de la population.

Population : l'ensemble des classes prises en compte au niveau du test.

Adjusted R-square: le pourcentage de variance de la variable dépendante expliquée par les variables indépendantes dans un modèle de régression pour une population.

Sum of squares of regression: la variance de la variable dépendante expliquée par le modèle de régression.

Sum of squares of residual: la variance non expliquée dans la variable dépendante.

Mean squares of residual: la somme des carrés des résidus divisée par le degré de liberté des résidus.

Il y a une hypothèse principale qui est testée dans cette analyse statistique. Cette hypothèse est: il y a une forte relation entre les métriques de cohésion et les changements.

8.4 Relation entre les métriques et les changements – méthode de régression

Un de nos objectifs est de trouver une corrélation entre les métriques de cohésion et les changements. Deux tests ont été effectués à savoir, calculer les coefficients de corrélation entre les métriques et le changement, d'une part, et une analyse de régression, d'autre part. Le but de cette étape consiste à déterminer les mesures importantes pour prédire la maintenabilité (les changements) d'une classe.

Au niveau des matrices de corrélation, plus la valeur du coefficient de corrélation est proche de 1, plus il y a une forte corrélation. Le modèle de régression est utilisé pour

déterminer les mesures importantes pour prédire la maintenabilité des classes. La variable « changement » est utilisée comme étant notre variable dépendante et toutes les autres métriques sélectionnées, dans ce travail, comme variables indépendantes.

Étape 1

L'analyse de régression multiple est la première étape de l'analyse. Elle est appliquée à l'ensemble des métriques retenues pour l'étude. L'objectif consiste à voir quelle sera la relation des différentes métriques prisent ensemble avec les changements.

Variable dépendante = changement.
Variables indépendantes = DCiE + DCdE + DCd + DCi + TCC + LCC + CBO + LOC.
Probabilité > F = <.0001 (xxl), ... (moneyjar), 0.1620 (FujabaUML).
R-Square = 0.9524 (xxl), (moneyjar), 0.6393 (FujabaUML).
Adjusted R-Square = 0.9323 (xxl), ... (moneyjar), 0.3187 (FujabaUML).

Il y a une variable dépendante et huit variables indépendantes dans ce modèle. La variable dépendante « changement » est une mesure de la changeabilité. Toutes les variables indépendantes sont des valeurs métriques.

La « Probabilité > F », communément appelée «p-value », donne une indication sur le degré de signification de la régression. Plus la probabilité est petite, plus la régression est significative.

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001,..., 0.1620. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

R-Square est un indicateur de la qualité de la régression qui mesure la qualité des prédictions ; c'est à dire combien la variance de la variable dépendante est expliquée par les variables indépendantes dans l'échantillon. Adjusted R-Square mesure le même aspect que R-Square mais au niveau de la population, avec un ajustement dépendant et de la taille de l'échantillon et du nombre de variables indépendantes.

Dans le système XXL, plus de 95% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 93% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, le nombre de variables métriques est supérieur à celui dans l'échantillon. Aucun résultat ne peut être déduit. Dans le système FujabaUML, plus de 63% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 31% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de l'ensemble des métriques est possible pour le système XXL.

Étape 2

Une deuxième étape de l'analyse de régression, toujours une analyse de régression multiple, a été appliquée à des sous ensembles de métriques (CBO + LOC + une métrique de cohésion (pour toutes les métriques de cohésion)).

1)
Variable dépendante = changement.
Variables indépendantes = DCiE + CBO + LOC.
Probabilité > F = <.0001 (xxl), 0.2597 (moneyjar), 0.6225 (FujabaUML).
R-Square = 0.8181 (xxl), 0.6937 (moneyjar), 0.1148 (FujabaUML).
Adjusted R-Square = 0.7954(xxl), 0.3873(moneyjar),-0.0749 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.2597, 0.6225. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 69% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 38% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 11% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 7% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion DCiE est possible pour le système XXL.

2)

Variable dépendante = changement.
Variables indépendantes = DCdE + CBO + LOC.
Probabilité > F = <.0001 (xxl), 0.2642 (moneyjar), 0.6309 (FujabaUML).
R-Square = 0.8179 (xxl), 0.6898 (moneyjar), 0.1124 (FujabaUML).
Adjusted R-Square = 0.7952 (xxl), 0.3797(moneyjar),-0.0777 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.2642, 0.6309. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 68% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 37% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 11% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 7% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion DCdE est possible pour le système XXL.

3)

Variable dépendante = changement.

Variables indépendantes = DCd + CBO + LOC.

Probabilité > F = <.0001 (xxl), 0.2754 (moneyjar), 0.6304 (FujabaUML).

R-Square = 0.8174 (xxl), 0.6804 (moneyjar), 0.1126 (FujabaUML).

Adjusted R-Square = 0.7945 (xxl), 0.3608 (moneyjar),-0.0776(FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.2754, 0.6304. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 68% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 36% (la valeur de adjusted R-Square)

dans la population. Dans le système FujabaUML, plus de 11% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 7% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion DCd est possible pour le système XXL.

4)

Variable dépendante = changement.

Variables indépendantes = DCi + CBO + LOC.

Probabilité > F = <.0001 (xxl), 0.2793 (Moneyjar), 0.5762 (FujabaUML).

R-Square = 0.8173(xxl), 0.6771 (moneyjar), 0.1279 (FujabaUML).

Adjusted R-Square = 0.7945 (xxl), 0.3543(moneyjar),-0.0590 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.2793, 0.5762. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 67% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 35% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 57% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 5% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion DCi est possible pour le système XXL.

5)

Variable dépendante = changement.

Variables indépendantes = TCC + CBO + LOC.

Probabilité > F = <.0001 (xxl), 0.1748 (moneyjar), 0.6229 (FujabaUML).

R-Square = 0.8173 (xxl), 0.7690 (moneyjar), 0.1147 (FujabaUML).

Adjusted R-Square = 0.7945 (xxl), 0.5380(moneyjar),-0.0751 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.1748, 0.6229. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 76% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 53% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 11% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 7% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion TTC est possible pour le système XXL.

6)

Variable dépendante = changement.

Variables indépendantes = LCC + CBO + LOC.

Probabilité > F = <.0001 (xxl), 0.1983 (moneyjar), 0.5104 (FujabaUML).

R-Square = 0.8173 (xxl), 0.7474 (moneyjar), 0.1475 (FujabaUML).

Adjusted R-Square = 0.7945 (xxl), 0.4949(moneyjar),-0.0352 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.1983, 0.5104. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 81% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 79% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 74% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 49% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 14% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir des métriques de couplage, de taille et de la métrique de cohésion LCC est possible pour le système XXL.

Étape 3

Puis, dans une troisième étape, nous nous intéressons à la seule combinaison des métriques de cohésion pour voir la relation de la combinaison avec les changements.

Variable dépendante = changement.
Variables indépendantes = DCiE + DCdE + DCd + DCi + TCC + LCC Probabilité > F = <.0001 (xxl), 0.2562 (moneyjar), 0.1353 (FujabaUML).
R-Square = 0.8265 (xxl), 0.9767 (moneyjar), 0.5341 (FujabaUML).
Adjusted R-Square = 0.7769 (xxl), 0.8601 (moneyjar), 0.2800 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de .0001, 0.2562, 0.1353. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 28% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 77% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 97% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 86% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 53% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 28% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de toutes les métriques de cohésion considérées est possible pour le système XXL.

Étape 4

Enfin, dans une quatrième étape, une analyse de régression simple est appliquée à toutes les métriques de cohésion pour voir la relation avec les changements.

1)
Variable dépendante = changement.
Variable indépendante = DCiE
Probabilité > F = 0.9891 (xxl), 0.7569 (moneyjar), 0.5384 (FujabaUML).
R-Square = 0.0000 (xxl), 0.0209 (moneyjar), 0.0241 (FujabaUML).
Adjusted R-Square=-0.0385(xxl),-0.1749(moneyjar),-0.0369 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de 0.9891, 0.7569, 0.5384. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, 0 % du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 2% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 17% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 2% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion DCiE n'est pas possible.

2)

Variable dépendante = changement.
Variables indépendantes = DCdE
Probabilité > F = 0.9660 (xxl), 0.8500 (moneyjar), 0.8504 (FujabaUML).
R-Square = 0.0001 (xxl), 0.0079 (moneyjar), 0.0023 (FujabaUML).
Adjusted R-Square=-0.0384(xxl),-0.1906(moneyjar),-0.0601 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML sont à un niveau de 0.9660, 0.8500, 0.8504. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, environ 0.01% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 4%

(la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, environ 0.79% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 2% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, environ 0.23 % du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 6% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion DCdE n'est pas possible.

3)

Variable dépendante = changement.

Variable indépendante = DCd

Probabilité > F = 0.7206 (xxl), 0.9846 (moneyjar), 0.8471 (FujabaUML).

R-Square = 0.0050 (xxl), 0.0001 (moneyjar), 0.0024 (FujabaUML).

Adjusted R-Square =-0.0333(xxl),-0.1999(moneyjar),-0.0600 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de 0.7206, 0.9846, 0.8471. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, environ 0.5 % du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3.3% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, environ 0.01% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 2% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, environ 0.24% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 6% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion DCd n'est pas possible.

4)

Variable dépendante = changement.

Variable indépendante = DCi

Probabilité > F = 0.6675 (xxl), 0.9703 (moneyjar), 0.4274 (FujabaUML).

R-Square = 0.0072 (xxl), 0.0003(moneyjar), 0.0398 (FujabaUML).

Adjusted R-Square=-0.0310(xxl),-0.1996(moneyjar),-0.0202 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de 0.6675, 0.9703, 0.4274. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, environ 0.7% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3.1% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 0.03% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 19.96% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 3.9% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 2% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion DCi n'est pas possible.

5)

Variable dépendante = changement.

Variable indépendante = TCC.
Probabilité > F = 0.7598 (xxl), 0.7244 (moneyjar), 0.9225 (FujabaUML).
R-Square = 0.0037 (xxl), 0.0271 (moneyjar), 0.0006 (FujabaUML).
Adjusted R-Square = -0.0347(xxl),-0.1675(moneyjar),-0.0619 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de 0.7598, 0.7244, 0.9225. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 0.37% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3.4% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 2.7% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 16.75% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 0.06% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 6.19% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion TTC n'est pas possible.

6)
Variable dépendante = changement.
Variable indépendante = LCC.
Probabilité > F = 0.7150 (xxl), 0.7606 (moneyjar), 0.3206 (FujabaUML).
R-Square = 0.0052 (xxl), 0.0203 (moneyjar), 0.0616 (FujabaUML).
Adjusted R-Square=-0.0330(xxl), -0.1757 (moneyjar), 0.0030 (FujabaUML).

La « Probabilité > F » dans les régressions des systèmes XXL, Moneyjar et FujabaUML est à un niveau de 0.7150, 0.7606, 0.3206. La petite valeur .0001 indique, avec un haut niveau de confiance, qu'une prédiction est possible.

Dans le système XXL, plus de 0.5 % du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 3.3% (la valeur de adjusted R-Square) dans la population. Dans le système Moneyjar, plus de 2% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et moins de 17% (la valeur de adjusted R-Square) dans la population. Dans le système FujabaUML, plus de 6.1% du total de la variance (la valeur de R-Square) dans les changements est expliqué par les métriques dans l'échantillon et plus de 0.3% (la valeur de adjusted R-Square) dans la population.

Conclusion : la prédiction des changements à partir de la métrique de cohésion LCC n'est pas possible.

8.5 Corrélation de rang de Spearman entre les métriques et les changements

Nous mesurons les valeurs de corrélation de Spearman (statistiques de rang) entre les deltas des métriques par rapport aux changements pour l'ensemble des systèmes sélectionnés. Le tableau XXI nous fournit quelques descriptions statistiques de ces systèmes.

Tableau XXI

Description statistique : Statistiques de rang de Spearman

Systèmes		Coef de Spearman	Stat du test	P-value
XXL	ΔDCdE-Changes	-0.19649699	-1.0218638	0.15812994
	ΔDCiE-Changes	-0.28297756	-1.50439806	0.0722654
	ΔDCd - Changes	-0.26902025	-1.42424499	0.08313397
	ΔDCi - Changes	-0.32703886	-1.76461198	0.04468772
	ΔTCC - Changes	-0.1803503	-0.93494044	0.17921093
	ΔLCC - Changes	-0.2484948	-1.30811105	0.1011416
	ΔCBO-Changes	0.13875205	0.7144098	0.2406696
	ΔLOC - Changes	0.63136289	4.15136491	0.00015736
Moneyjar	ΔDCdE-Changes	0.39285714	0.95525853	0.19165844
	ΔDCiE-Changes	0.39285714	0.95525853	0.19165844
	ΔDCd - Changes	-0.14285714	-0.32274861	0.37997265
	ΔDCi - Changes	0	0	0.5
	ΔTCC - Changes	-0.25	-0.57735027	0.29436222
	ΔLCC - Changes	-0.10714286	-0.2409658	0.40957543
	ΔCBO-Changes	-0.96428571	-8.1408063	0.00022707
	ΔLOC - Changes	0.17857143	0.40582063	0.35082897
FujabaUML	ΔDCdE-Changes	-0.03405573	-0.13630197	0.44664141
	ΔDCiE-Changes	-0.11042312	-0.44441018	0.33134922
	ΔDCd - Changes	-0.0753354	-0.30220037	0.38319614
	ΔDCi - Changes	-0.1496388	-0.60537125	0.27671145
	ΔTCC - Changes	-0.1372549	-0.55426531	0.29353077
	ΔLCC - Changes	-0.27966976	-1.16517385	0.13051324
	ΔCBO-Changes	0.00103199	0.00412797	0.4983787
	ΔLOC - Changes	0.22910217	0.94144897	0.18023654

8.6 Relation entre les métriques et le changement avec la corrélation de rang

Le but de cette expérimentation est de trouver un lien, une corrélation entre les deltas de métriques et les changements entre les versions des systèmes sélectionnées. Pour vérifier si la corrélation est significativement (en termes statistiques) plus petite que 0 (dépendance négative), un test a été effectué. La statistique du test, qui doit être par la suite comparée à une variable de Student à n-2 degrés de liberté (n étant la taille de l'échantillon), est calculée.

La p-value indique la probabilité d'obtenir une telle valeur sous l'hypothèse d'absence de dépendance. En général, si p-value < 0.05, on conclut à une dépendance négative significative.

Pour l'ensemble des systèmes étudiés dans le tableau XXI, les valeurs de p-value sont en générale toutes > 0.05 pour l'ensemble des combinaisons delta de métrique - changement.
Dans tous les systèmes, on ne note aucune dépendance négative significative entre les deltas de métrique et le changement. Donc aucune corrélation significative.

8.7 Discussion et interprétation des résultats

Dans la première expérimentation que nous avons effectuée, les résultats obtenus au niveau des matrices de corrélation [Tableau XIII, Tableau XIV, Tableau X] montrent une corrélation entre les métriques de taille LOC et de couplage CBO et les changements. Ces résultats confirment d'autres travaux tels que Kabaili et Dagpinar [Kab 01, Dag 03].

Dans notre deuxième expérimentation, les résultats de la régression de la combinaison de toutes les métriques est toujours supérieur à celui de la combinaison des deltas de métriques CBO, LCO à laquelle on rajoute un delta de métrique de cohésion comme le montre le tableau XXII ci dessous.

Tableau XXII

Valeurs de R2 dans les modèles de régression

	r-square		
Type de combinaison	XXL	MoneyJar	FujabaUML
Toutes les métriques	0.9526	1	0.7573
CBO + LOC + DCdE	0.8179	0.6898	0.1124
DCdE	0.0001	0.0079	0.0023
CBO + LOC + DCiE	0.8181	0.6937	0.1148
DCiE	0	0.0209	0.0241
CBO + LOC + DCd	0.8174	0.6804	0.1126
DCd	0.0050	0.0001	0.0024
CBO + LOC + DCi	0.8173	0.6771	0.1279
DCi	0.0072	0.0003	0.0398
CBO + LOC + TCC	0.8173	0.7690	0.1147
TCC	0.0037	0.0271	0.0006
CBO + LOC + LCC	0.8173	0.7474	0.1475
LCC	0.0052	0.0203	0.0616

Dans les cas ou le r-square est satisfaisant, on remarque toujours la présence de LOC et de CBO.
Au niveau du tableau XXII, en enlevant les deux métriques LOC et CBO de la régression, les métriques de cohésion par rapport aux changements donnent un r-square très faible. Par exemple, pour la combinaison CBO, LOC et DCdE on a une valeur de r-Square égale à 0.8179. Pour le même procédé, mais cette fois ci avec juste le DCdE, on obtient une valeur de 0.001. Il en va de même pour toutes les autres métriques de cohésion. D'après ces remarques, nous pouvons en déduire que les valeurs satisfaisantes des combinaisons sont totalement dues à l'apport des métriques LOC et CBO.

Le test sur les statistiques de rang de Spearman ne révèle pas une forte dépendance entre les deltas de métrique et les changements. Une explication possible est peut être que le nombre de classes considérées avec des changements pour chaque système est trop petit pour mener cette étude statistique. Plus le nombre de classes est important plus la relation de dépendance est visible comme le montre l'étude sur la relation entre le couplage et la cohésion au chapitre 5. Cette expérimentation n'a pu être étendue aux changements par manque d'informations sur l'historique des changements. Ces dernières, et d'une manière générale, sont très difficiles à obtenir.

Dans l'ensemble des tests effectués, les métriques de cohésion ne montrent aucun lien avec les changements. Plusieurs travaux antérieurs [Kab 01, Dag 03] en sont arrivés à la même conclusion et les résultats trouvés dans les différentes études appellent à une révision de la définition actuelle des métriques. Chae et Kwon [Cha 00] observent que quelques méthodes spéciales doivent être prises en compte et traitées pour ne pas compromettre la valeur des métriques de cohésion. Ils suggèrent aussi que les métriques de cohésion prennent en compte quelques caractéristiques additionnelles de la classe, par exemple, les patterns d'interaction parmi les membres d'une classe. Cependant, toutes ces propriétés additionnelles ne sont pas faciles à mesurer du moment qu'elles sont sémantiques.

Bien qu'ayant pris en compte un nouveau critère au niveau de la définition des nouvelles métriques au chapitre 4 les conclusions des travaux antérieurs, par rapport à la relation métrique de cohésion et le changement, restent inchangées. Nous pouvons conclure dès à présent que, aussi longtemps qu'une nouvelle métrique de cohésion ne sera pas définie, prenant en compte les importants facettes de propriété de cohésion, les métriques de cohésion actuelles ne peuvent pas être vues comme des indicateurs de changeabilité. Une autre possibilité peut également expliquer le fait que nous n'ayons pas pu prouver qu'une relation entre les métriques de cohésion et les changements existe. Elle est due au fait que les systèmes considérés ne présentent pas assez de changements pour pouvoir observer une relation statistiquement significative. D'autres études plus approfondies restent donc nécessaires pour pouvoir tirer des conclusions finales à ce sujet.

CONCLUSIONS GLOBALES

Dans ce mémoire, notre but majeur était de valider les métriques de cohésion comme indicateur de changeabilité. Cette idée découle, d'un coté, du fait que plusieurs études donnent le couplage comme un indicateur de changeabilité et de l'autre des principes de conception orientée objet, qui stipulent qu'une bonne conception montrant une forte cohésion de classes s'accompagne d'un faible impact de changements [Pre 01, Som 04]. Une relation devrait donc exister entre la cohésion et la changeabilité.

Le couplage est difficile à mesurer et consomme du temps du fait qu'il constitue une propriété interclasses. En effet, pour la mesurer, la connaissance de l'ensemble du système et de tous les liens entre les classes doit être maitrisée. De plus, la plupart des travaux sur la changeabilité ont porté sur le couplage [Cha 98A, Abd 06]. Il serait intéressant d'investiguer un autre indicateur : la cohésion des classes. La cohésion, qui est une propriété intra-classe, nécessite juste de considérer la classe étudiée pour la mesurer. Des études [Bri 99, Kab 01] montrent que le couplage représente un bon indicateur de changeabilité. Une croyance largement tenue dans la communauté de conception, déclare qu'une cohésion élevée est liée à un bas couplage et vice versa [Pre 01]. Plusieurs auteurs du domaine citent cette relation, facile à comprendre intuitivement.

Plusieurs métriques ont été proposées dans la littérature dans le but de mesurer la cohésion des classes dans les systèmes orientés objet. Ces métriques de cohésion sont essentiellement basées sur l'usage des variables d'instance et des invocations de méthodes. Cependant, certains travaux appellent à leur raffinement [Cha 00, Kab 01]. Ils suggèrent aussi que les métriques de cohésion prennent en compte quelques

caractéristiques additionnelles de la classe, par exemple, les patterns d'interaction parmi les membres d'une classe.

Dans le but de capturer des caractéristiques additionnelles des classes et de mieux mesurer les propriétés de cohésion, nous avons introduit un nouveau critère de cohésion de classe qui est basé sur le partage de type d'argument au niveau des méthodes. Notre but principal à ce niveau était de valider statistiquement ce nouveau critère introduit. Les résultats obtenus montrent que les nouvelles métriques que nous avons proposées et qui tiennent compte du nouveau critère, capturent plus de paires de méthodes connectées que les métriques existantes. Le test statistique a été concluant.

Après avoir défini et validé statistiquement ces nouvelles métriques, nous avons étudié dans un premier temps la relation entre la cohésion et le couplage. Une cohésion élevée est liée à un bas couplage et vice versa [Pre 01]. Les résultats obtenus confirment cette hypothèse. En effet, une forte dépendance a été observée entre ces métriques de couplage et de cohésion avec la méthode de corrélation de rang de Spearman. À notre connaissance, cette validation n'a fait l'objet d'aucun article dans la littérature jusqu'à date. Les quelques travaux qui ont abordé cette question sont arrivés à des échecs.

Après le chapitre 5, qui apporte une validation statistique de la relation couplage - cohésion, associée à l'idée du couplage comme indicateur de changeabilité et qui est démontrée par plusieurs études [Kab 01, Dag 03, Abd 06], nous avons investigué la cohésion en tant qu'indicateur de changeabilité. Les résultats trouvés ne sont pas concluants.

La seule conclusion que nous pouvons en tirer est que les métriques de cohésion définies à l'heure actuelle ne sont pas de bons indicateurs de changeabilité. On arrive, en fait, à la même conclusion que Kabaili [Kab 01].

BIBLIOGRAPHIE

[Abb 94] D. H. Abbott, T. D. Korson, et J.D. McGregor. *A proposed design complexity for object-oriented development.* Technical Report TR 94-105, Clemson, University, Clemson, Ontario, Canada K7L 3N6, 1994.

[Abd 06] M.K Abdi, H. lounis and H. Sahraoui. *Using Coupling Metrics for Change Impact Analysis in Object-Oriented Systems.* In QAOOSE 06, pages 61-70, 2006.

[Abo 97] Joe R. Abounader and David A. Lamb. *A data model for object-oriented design metrics.* Technical Report 409, Department of Computing and Information, Science, Queen University, Kingston, Ontario, Canada K7L 3N6, 1997.

[Abr 91] F. Brito e Abreu and W. Melo. *Evaluating the impact of object-oriented design on software quality.* In First workshop on Information Technologies and Systems (WITS'91), pages 285–307, December 1991.

[Abr 92] F. Brito e Abreu and Carapua. *Object oriented software design metrics.* In OOPSLA'92, workshop on OO Metric, 1992.

[Abr 96] F. Brito e Abreu and W. Melo. *Evaluating the impact of object-oriented design on software quality.* In METRICS'96. IEEE, March 1996.

[Ama 02] H. Aman, K. Yamasaki, H. Yamada and MT. Noda, A proposal of class cohesion metrics using sizes of cohesive parts, *Knowledge-Based Software Engineering*, T. Welzer et al. (Eds), pp. 102-107, IOS Press, September 2002.

[Ant 99] G. Antoniol, G. Canfora and A. De Lucia. *Estimating the size of changes for evolving object-oriented systems: a Case Study.* In Proceedings of the 6th International Software Metrics Symposium, pages 250-258, Boca Raton, Florida, Nov 1999.

[Bad 03] L. Badri and M. Badri. *A New Class Cohesion Criterion: An empirical study on several systems.* QAOOSE 03, 2003.

[Bad 04] Linda Badri and Mourad Badri: *A Proposal of a New Class Cohesion Criterion: An Empirical Study.* Journal of Object Technology 3(4): 145-159, 2004.

[Bad 95] L. Badri, M. Badri and S. Ferdenache. *Towards Quality Control Metrics for Object-Oriented Systems Analysis.* TOOLS (Technology of Object-

Oriented Languages and Systems) Europe'95, Versailles, France, Prentice-Hall, March 1995.

[Bai 04] Imran Baig. *Measuring Cohesion and Coupling of Oriented-object Systems- Derivation and Mutual Study of cohesion and coupling.* Master's thesis, school of Engineering, Blekinge Institute of Technology, 2004.

[Bas 96] V.R. Basili, L.C. Briand and W. Melo. *A validation of object-oriented design metrics as quality indicators.* IEEE Transactions on Software Engineering, 22 (10), pp. 751-761, October 1996.

[Ben 97] Saida Benlarbi. . *Object-Oriented Design Metrics for Early Quality Prediction.* CRIM, Montreal, Quebec, Canada OOPSLA 97, 1997.

[Bie 95] J.M. Bieman and B.K. Kang. *Cohesion and reuse in an object-oriented system.* Proceedings of the Symposium on Software Reusability (SSR'95), Seattle, WA, pp. 259-262, April 1995.

[Boh 96] S. A. Bohner and R. S. Arnold. *An introduction to software change impact analysis.* In S. A. Bohner and R. S. Arnold, editors, Software Change Impact Analysis, pages 1--26. IEEE Computer Society Press, 1996.

[Boo 94] G. Booch, *Object-Oriented Analysis and Design with Applications,* Second edition, Benjamin/Cummings, 1994.

[Bri 01] L.C. Briand, J. Wust, H. Lounis. *Replicated Case Studies for Investigating Quality Factors in Object-Oriented Designs.* In Empirical Software Engineering, an International Journal, 6(1):11-58, Kluwer Academic Publishers, March 2001.

[Bri 97] L. C. Briand, J. Daly, V. Porter, and J. Wuest. *The Dimensions of Coupling in Object-Oriented Design.* Fraunhofer IESE, Kaiserslautern, Germany, OOPSLA97, 1997.

[Bri 98] L.C. Briand, J. Daly and J. Wust. *A unified framework for cohesion measurement in object-oriented systems.* Empirical Software Engineering, 3 (1), pp. 67-117, 1998.

[Bri 98A] L. C. Briand, S. J. Carriére, R. Kazman, J. Wüst. *A Comprehensive Framework for Architecture Evaluation.* International Software Engineering Research Network Report ISERN-98-28, 1998.

[Bri 99] L. C. Briand, J. Wuest, and H. Lounis. *Using Coupling Measurement for*

Impact Analysis in Object-Oriented Systems. In proceedings of the International Conference on Software Maintenance ICSM'99, Oxford, England, August 30 – September 3, 1999.

[Can 01] R. Cantave. Abstractions *via un modèle générique d'application orientée objet*. Master's thesis, Université Laval, Canada, Avril 2001.

[Cha 00] H. S. Chae, Y. R. Kwon and D H. Bae. *A Cohesion measure for object-oriented classes*. Software Practice and Experience, No. 30, pp. 1405-1431, 2000.

[Cha 00A] M. A. Chaumun, H. Kabaili, R. K. Keller, F. Lustman. *Design Properties and Object-Oriented Software Changeability*. In Proceeding of the Fourth Euromicro Working Conference on Software Maintenance and Reengineering, pages 45-54, Zurich, Switzerland, February 2000.

[Cha 04] H. S. Chae, Y. R. Kwon. *Improving Cohesion Metrics for class by considering dependant instance Variable*. IEEE, 2004.

[Cha 98] H. S. Chae and Y.R. Kwon. *A cohesion measure for classes in object-oriented systems*. Proceedings of the fifth International Software Metrics Symposium, Bethesda, MD, pp. 158-166, November 1998.

[Cha 98A] M. A. Chaumun. *Change Impact Analysis in Object-Oriented Systems: Conceptual Model and Application to C++*. Master's thesis, Université de Montréal, Canada, November 1998.

[Cha 99] M. A. Chaumun, H. Kabaili, R. K. Keller and F. Lustman. *A Change Impact Model for Changeability Assessment in Object-Oriented Software Systems*. In Proceedings of the Third Euromicro Working Conference on Software Maintenance and Reengineering CSMR'99, pages 130-138, Amsterdam, The Netherlands, March 1999.

[Che 93] J.-Y Chen and J.F. Lu. *A new metric for object-oriented design*. Information and Software Technologies, 35(4): 232–240, 1993.

[Chi 94] S. R. Chidamber and C. F. Kemerer. *A Metrics Suite for Object Oriented Design*. In IEEE Transactions on Software Engineering, Vol. 20, No. 6, pages 476-493, June 1994.

[Dag 03] Dagpinar and Jens H. Jahnke. *Predicting maintainability with object-oriented metrics- An Empirical Comparison*. WCRE 2003, 2003.

[El 01] El Emam, K.; Benlarbi, S.; Goel, N. and Rai, S. N. *The confounding*

effect of class size on the validity of object oriented metrics. IEEE Transactions on Software Engineering, 27(7): 630-650, 2001.

[El 99] K. El Emam and W. Melo. *The prediction of faulty class using object-oriented design metrics.* National Research Council of Canada NRC/ERB 1064, 1999.

[Fen 99] Fenton, N. and Neil, M. *A critique of software defect prediction models.* IEEE Transactions on Software Engineering, 25(5):675–689. 1999.

[Han 97] J. Han. *Supporting Impact Analysis and Change Propagation in Software Engineering Environments.* In Proceedings of the STEP97, London, England, pages 172-182, July 1997.

[Hen 96] B. Henderson-sellers, *Object-Oriented Metrics Measures of Complexity*, Prentice-Hall, 1996.

[Hin 03] W. W. Hines, D. C. Montgomery, D. M. Goldsman and C. M. Borror, Probability and statistics in engineering, Fourth edition, John Wiley & Sons, Inc., 2003

[Hit 95] M. Hitz and B. Montazeri. *Measuring coupling in object-oriented systems.* Object Currents, 1(4), 1995.

[Kab 01] Hind Kabaili, Rudolf K. Keller and François Lustman. *Class Cohesion as Predictor of Changeability: An Empirical Study.* L'Objet, Vol. 6, No.4, Hermes Science Publications, Paris, France, 2001.

[Kun 94] D. Kung, J. Gao, P. Hsia, F. Wen, Y. Toyoshima, and C. Chen. *Change impact identification in object oriented software maintenance.* In Conference on Software Maintenance, pages 202-211, Piscataway, NJ, 1994.

[Kun 95] D. C. Kung, J. Gao, P. Hsia, J. Lin and Y. Toyoshima. *Class firewall, test order, and regression testing of object-oriented programs.* In JOOP, Vol. 8, No. 2, pages 51-65, May 1995.

[Lar 02] G. Larman. *Applying UML and Design Patterns, An introduction to object-oriented analysis and design and the unified process.* Second edition, Prentice Hall, 2002.

[Lee 98] M.L. Lee. *Change Impact Analysis for Object-Oriented Software.* PhD thesis, George Mason University, Virginia, USA, 1998.

[Li 93] W. Li and S. Henry, Object-Oriented Metrics that Predict Maintainability, In *Journal of Systems and Software,* 23:111-122, February, 1993.

[Li 95] W. Li, S. Henry, D. Kafura and R. Schulman. *Measuring Object-Oriented Design.* In Journal of Object-Oriented Programming, Vol. 8, No. 4, pages 48-55, July/August 1995.

[Li 96] L. Li and A. J. Offutt. *Algorithmic Analysis of the Impact of Changes to Object-Oriented Software.* In ICSM96, pages 171-184, 1996.

[Lou 97] H. Lounis, H.A. Sahraoui, and W.L. Melo. *Defining, Measuring and Using Coupling metrics in Object-Oriented Environment.* In SIGPLAN OOPSLA'97 Workshop on Object-Oriented Product Metrics, 1997, Atlanta, Georgia, USA, 1997.

[Pre 01] R. S. Pressman, *Software Engineering, A practitioner's approach*, Fifth edition, Mc Graw Hill, 2001.

[Roc 88] Rocacher, D. *Metrics definition for smalltalk.* Technical report, European Union ESPRIT Research Report 1257, 1988.

[Rum 98] J. Rumbaugh, I. Jacobson and G. Booch. *The Unified Modeling Language Reference Manual.* Addison-Wesley, Massachusetts, USA, 550 pages, Dec 1998.

[Som 04] Ian Sommerville. *Software Engineering.* Seventh edition, Pearson, Addison Wesley, 2004.

[Ste 74] W.P. Stevens, G.J. Myers and L.L. Constantine. *Structured Design.* In IBM Systems Journal, Vol. 13, No. 2, pages 115-139, May 1974.

[Suz 98] H. Suze. *A Framework of Software Measurement.* Walter de Gruyter, 1998.

[Wei 98] C. Davis Etzkorn L. and L. Wei. *A practical look at the lack of cohesion in methods metric. JOOP*, 11(5):27–34, 1998.

[Wil 98] F. G. Wilkie, B. A. Kitchenham. *Coupling Measures and Change Ripples in C++ Application Software.* Published in the proceedings of EASE'99, University of Keele, UK, 1998.

[You 79] E. Yourdon and L. Constantine. *Structured Design*, Prentice Hall, Englewood Cliffs, N.J., 1979.