Titre: Title:	Approche innovatrice de conception pour les réseaux de télécommunications basée sur l'architecture modulaire
Auteur: Author:	Charles Gagnon
Date:	2008
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Gagnon, C. (2008). Approche innovatrice de conception pour les réseaux de télécommunications basée sur l'architecture modulaire [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/8334/

UNIVERSITÉ DE MONTRÉAL

APPROCHE INNOVATRICE DE CONCEPTION POUR LES RÉSEAUX DE TÉLÉCOMMUNICATIONS BASÉE SUR L'ARCHITECTURE MODULAIRE

CHARLES GAGNON
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION DU DIPLÔME DE
MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
JUIN 2008

Your file Votre référence
ISBN: 978-0-494-46049-8
Our file Notre référence
ISBN: 978-0-494-46049-8

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

APPROCHE INNOVATRICE DE CONCEPTION POUR LES RÉSEAUX DE TÉLÉCOMMUNICATIONS BASÉE SUR L'ARCHITECTURE MODULAIRE

présenté par : GAGNON Charles
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury constitué de :

M. QUINTERO Alejandro, Doct., président.
M. CHAMBERLAND Steven, Ph.D., membre et directeur de recherche.
M. PIERRE Samuel, Ph.D., membre et codirecteur de recherche.
M. BEDARD Patrick, M.Sc.A.

À mon père qui m'a toujours encouragé dans mes études et qui aurait été fier...

Remerciements

Je tiens à remercier mon directeur de recherche Steven Chamberland et mon co-directeur Samuel Pierre pour leur soutien, leur encouragement et leurs conseils. Je tiens aussi à remercier Patrick Bédard et Yves Lepage de Bell Canada pour leur participation au projet.

Résumé

Les méthodes actuelles d'ingénierie des réseaux informatiques ne sont pas optimales au niveau des coûts et des performances. La diversité et la complexité des technologies rendent le développement de réseaux difficile. Pour faire face à la concurrence de plus en plus forte dans ce marché, les compagnies doivent rationaliser leurs méthodes de conception. Ce mémoire propose une approche innovatrice de conception pour les réseaux de télécommunications basée sur le concept d'architecture modulaire.

Le principe de l'architecture modulaire est d'augmenter l'abstraction du design en divisant l'architecture en couches hiérarchiques : pièce, composant, module. Le module est l'item central de l'architecture modulaire. Il y a trois types de modules : service, connectivité et inter-connectivité. L'assemblage de modules permet de créer un réseau fonctionnel. Les différents types de modules et leurs caractéristiques sont décrits en détails. Différentes métriques ont été définies et elles permettent de décrire la structure du module et ses performances.

Pour valider l'architecture modulaire proposée, un sous-ensemble restreint de modules (un de chaque type) est implémenté. L'impact des différents paramètres des métriques de design et de coût du modèle analytique est analysé. Les paramètres ayant le plus d'impact sur ces métriques et les valeurs permettant d'obtenir l'optimalité sont identifiés. Suite à cette étude analytique, un plan d'expérience est établi pour évaluer l'impact de ces paramètres sur le sous-ensemble de modules implémenté. Cette expérimentation permet d'établir l'impact de la décomposition de l'architecture d'un service en modules et en composants au niveau des métriques de design et de coût. Il n'est pas possible d'affirmer si l'architecture modulaire proposée est rentable. Ce qui est possible d'affirmer c'est que dans certaines conditions cette architecture modulaire est rentable. La décomposition de l'architecture modulaire en modules et en composants influence les différentes métriques. Il est possible d'évaluer les conditions de rentabilité de la décomposition de l'architecture d'un service en modules et en composants.

Abstract

The current design and engineering methods of Bell networks are non optimal for costs and performances. The diversity and the complexity of technologies make the development of networks difficult. To face increasingly strong competition in this market, the companies must rationalize their methods of design. This memory proposes an innovating approach of design for telecommunication networks based on the concept of modular architecture.

The principle of modular architecture is to increase the abstraction of the design by dividing it into hierarchical layers: part, component, module. The module is the central item of modular architecture. There are three types of modules: service, connectivity and inter-connectivity. The assembly of modules creates a functional network. The various types of modules and their characteristics are described in details. Different metrics were defined to describe the structure of the module and its performances.

To validate the suggested modular architecture, a restricted subset of modules (one of each type) is implemented. The impact of the various parameters of the metrics of design and cost of the analytical model is analyzed. The parameters having the most impact on these metrics and the values allowing optimality are identified. Following this analytical study, an experimental plan is established to evaluate the impact of these parameters on the implemented subset of modules. This experimentation establishes the impact of the decomposition of the architecture of a service in modules and components on the metrics of design and costs. It's not possible to affirm if the suggested modular architecture is profitable. What is possible to affirm it is that under certain conditions this modular architecture is profitable. The decomposition of the modular architecture in modules and components influences the different metrics. It's possible to evaluate the conditions of profitability of the decomposition of a service in modules and components.

Table des matières

Liste des tableaux

Liste des figures

Liste des sigles et abréviations

DSL	Digital Subscriber Line
EIE	Ethernet Internetworking
HSM	High Speed Metro
MPLS	MultiProtocol Label Switching
OSI	Open Systems Interconnection
PSTN	Public Switch Telephony Network
VoIP	Voice over IP
VPN	Virtual Private Network

CHAPITRE 1

INTRODUCTION

Le domaine des télécommunications est vaste et diversifié. Les différentes technologies évoluent rapidement et la gamme de services possibles ne cesse de s'élagir. Les ingénieurs font face à de nouveaux défis. Comment conserver et maintenir l'ensemble des services existants, parfois obsolets, tout en permettant l'ajout de nouveaux services ? Cette intégration doit se faire de façon homogène dans un contexte d'hétérogénéité de protocoles et de technologies. Chaque technologie et chaque service sont de plus en plus complexes. Dans un contexte où les ingénieurs doivent maîtriser un bon nombre de technologies et de services différents, il peut s'avérer utile de les manipuler à un plus haut niveau. En faisant abstraction des détails techniques et en se concentrant sur le fonctionnement des services, il serait possible de maîtriser un plus vaste éventail de services. Une façon d'y parvenir serait d'appliquer le concept d'architecture modulaire à la conception des réseaux de télécommunications. Dans ce chapitre d'introduction, il est question des concepts de base relatifs à l'architecture modulaire. Par la suite, les différents éléments de la problématique et les objectifs de recherche sont définis. Enfin, le plan du mémoire est exposé.

1.1 Définitions et concepts de base

Dans la littérature, il n'y a pas de concensus pour la notion de module. Souvent, il est question d'archtitecture modulaire sans que le concept de module soit défini. Dans le cadre de ce mémoire, un module est une entité fonctionnelle (matérielle et/ou logicielle) qui fait partie d'une architecture modulaire. Un module fournit un service réseau et il peut interragir avec les autres modules. Un module est défini par ses interfaces et ses fonctionnalités et non par son implémentation. L'architecture d'un réseau de télécommunications est similaire à celle des plans architecturaux d'un bâtiment. C'est un schéma des différents équipements de télécommunications, des emplacements géographiques et des interconnexions qui forment le réseau désiré.

Ce plan architectural du réseau de télécommunications doit être clair et bien défini. Dans le cadre de ce mémoire, une architecture modulaire est une architecture fonctionnelle d'un réseau de télécommunications formée par l'assemblage de modules. Cette architecture modulaire fournit un ou plusieurs services réseaux. Le concept de l'architecture modulaire est analogue aux blocs "légos" de construction. Des blocs de fonction (couleur), de taille (nombre de pattes) et de configuration (orientation horizontale ou verticale) variées peuvent être assemblés selon une multitude de patrons. Chaque bloc possède une interface uniformisée : nombre de pattes pair, de grosseur et de taille identique. Chaque assemblage de blocs est une architecture fonctionnelle. Le concept de l'architecture modulaire introduit la notion de réutilisabilité. Un bloc légo peut être réutilisé dans plusieurs assemblages différents. La réutilisabilité est une caractéristique recherchée dans une archtitecture modulaire. Un module doit pouvoir être utilisé et assemblé dans plusieurs architectures différentes. Le concept de la réutilisation réfère à la notion d'intéropérabilté. Il faut être en mesure d'assembler des modules parfois hétérogènes et ce de façon homogène.

Une façon de résoudre un problème complexe est de le diviser en sous-problèmes. Chaque sous-problème est typiquement plus simple à résoudre que le problème entier. Ce raffinement peut opérer à plusieurs niveaux. L'architecture modulaire se base sur ce principe. Pour un problème donné, plus le nombre de modules est grand, plus le coût unitaire de conception des modules diminue. Par contre, plus le nombre de modules est grand, plus il est complexe de les assembler. Il existe donc une zone de coût minimum où il y a un équilibre entre le coût unitaire d'un modules et le coût de réutilisation (assemblage). La figure [1.1] (Pressman, 2005) illustre cette relation.

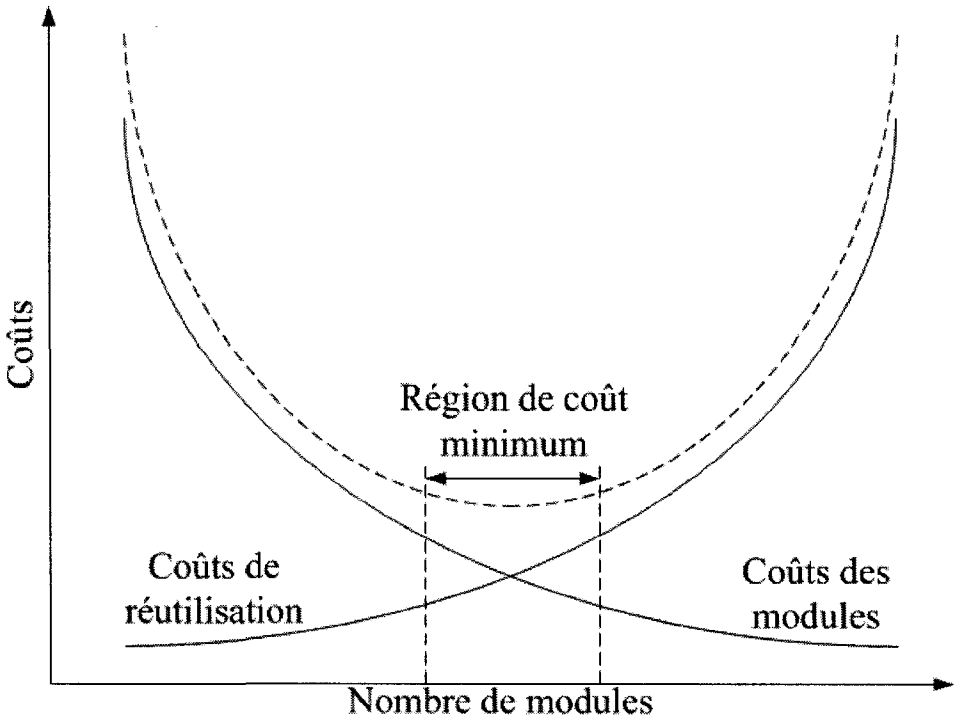

FIGURE 1.1 Coûts en fonction du nombre de modules

1.2 Éléments de la problématique

Les méthodes actuelles de conception et d'ingénierie des réseaux de télécommunications ne sont pas optimales au niveau des coûts et des performances. La diversité et la complexité des technologies rendent le développement de nouveaux réseaux difficile. Pour faire face à la concurrence de plus en plus forte dans ce marché, les compagnies doivent rationnaliser leurs méthodes de conception. Présentement, la plupart des architectures réseaux sont développées selon un mode de réutilisation ad hoc (non structuré). Un tel procédé implique que la conception d'une architecture pour un réseau de télécommunications est basée sur les connaissances personnelles et les préférences de travail (type d'équipement et manufacturier) des ingénieurs réseaux. Il n'y a pas d'uniformisation des architectures et par conséquent peu ou pas de réutilisation. Ce mode de conception entraîne trois conséquences principales. D'un point de vue organisationnel, il y a un manque d'uniformisation du design. Pour le même besoin, le même service, une architecture différente peut être fournie au client. D'un point de vue technique, l'architecture optimale n'est toujours fournie

au client et ce même si elle existe. D'un point de vue économique, ce mode de conception n'est pas le plus rentable. La propriété intellectuelle de la compagnie n'est pas ou peu réutilisée. Durant les années 1990, les grandes industries ont revu l'ensemble de leurs procédés de fabrication de façon à améliorer leur productivité et optimiser l'utilisation de leurs ressources. Le concept d'architecture modulaire et la notion de réutilisation ont été introduits et un effort de recherche y a été consacré. Dans le domaine des télécommunications, peu d'efforts de recherche ont été consacrés aux méthodes de conception et d'ingénierie des réseaux de télécommunications. La plupart des recherches relatives aux télécommunications touchent une technologie, un protocole ou un service donné au niveau des performances.

1.3 Objectifs de recherche

L'objectif de ce mémoire est de proposer une approche innovatrice de conception pour les réseaux de télécommunications basée sur le concept d'architecture modulaire. Cette approche de conception doit permettre de réduire les coûts de conception, d'améliorer la performance des réseaux, d'uniformiser le module optimal pour la réutilisation et de maximiser l'interopérabilité, la flexibilité et la réutilisabilité des modules.

L'approche de conception proposée doit être générique de façon à pouvoir intégrer le plus de services réseau possibles. Cette approche ne doit pas être orientée vers une technologie ou un service spécifique. Le but de cette méthode de conception est d'avoir des modules préalablement testés qui pourraient être assemblés pour obtenir une architecture réseau fonctionnelle. Pour un service donné, il y aurait déjà un ou des module(s) prêt(s) à être utilisé(s) avec une configuration de base. De manière plus spécifique, ce mémoire vise à :

- définir les requis de l'architecutre modulaire et les différentes métriques qui caractérisent cette architecture ;
- définir les modules et leurs interactions à l'intérieur de l'architecture ;
- implémenter un sous-ensemble de modules de cette architecture ;
- évaluer les différentes métriques du sous-ensemble de modules.

1.4 Plan du mémoire

Ce mémoire comporte cinq chapitres. Le présent chapitre est celui de l'introduction. Le second chapitre est une revue de littérature des différents types d'architectures modulaires et leurs principales caractéristiques. Le troisième chapitre décrit l'approche de conception proposée. Le chapitre quatre est celui des résultats. Des modules sont conçus et différentes métriques relatives au design sont analysées. Le coût de réutilisation d'un module est aussi analysé. Le dernier chapitre est une synthèse de l'approche proposée et des résultats obtenus. Il est aussi question des limitations des travaux et des pistes de recherche futures.

CHAPITRE 2

CONCEPT D'ARCHITECUTRE MODULAIRE

Le concept de l'architecture modulaire est souvent employé sans être bien défini. Qu'est-ce qu'un module ? Qu'est-ce que la modularité ? Quelles sont les caractéristiques d'une architecture modulaire ? Quel est le coût d'une telle architecture ? Toutes ces questions sont répondues en partie dans différents travaux. Ce chapitre est une revue de littérature des différents aspects qui concernent le concept de l'architecture modulaire. Dans un premier temps, il est question des définitions et des caractéristiques de l'architecture modulaire. Par la suite, la notion de réutilisation est abordée. Différentes métriques relatives au design de l'architecture modulaire sont ensuite définies avant de traiter des modèles de coûts de réutilisation dans le contexte d'une architecture modulaire. Finalement, la représentation et la documentation du design dans un contexte de réutilisation sont présentées.

2.1 Définitions et caractéristiques

Une façon de résoudre un problème complexe est de le diviser en sous-problèmes moins complexes qui peuvent être traités séparément. La fonction d'un système est décomposée en sous-fonctions. Cette approche de conception comporte deux avantages : simplification et rapidité. La résolution d'un sous-problème du système est généralement plus simple que le système en soi. La décomposition du problème en sous-problèmes permet de traiter parallèlement chacun de ces sous-problèmes. Selon Whitney (1992), la modularité est souvent vue purement comme un processus de décomposition de l'architecture d'un produit en sous-assemblage. Ulrich et Eppinger (1995) définissent l'architecture d'un produit comme le schéma selon lequel les éléments décomposés sont arrangés en sous-assemblages . Pimmler et Eppinger (1994) reprennent cette définition pour proposer une approche de conception modulaire en

trois étapes :

1. décomposition du système en éléments ;
2. documentation des interactions entre les éléments ;
3. groupement des éléments en sous-assemblages.

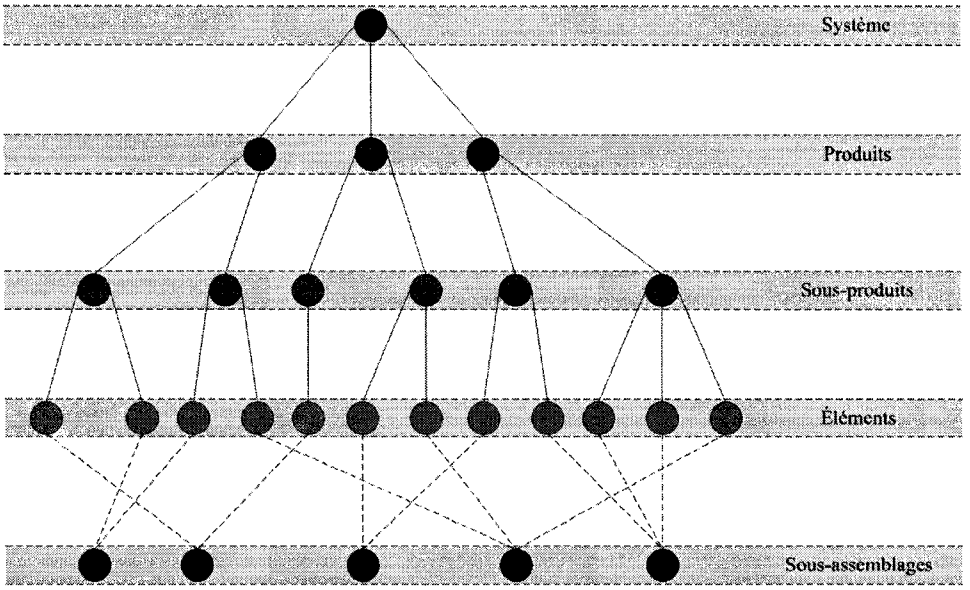

FIGURE 2.1 Décomposition et groupement

La Figure 2.1 illustre les étapes 1 et 3. Le système est décomposé en produits. Les produits sont décomposés en sous-produits. Les sous-produits sont décomposés en éléments. Les éléments sont ensuite regroupés en sous-assemblages selon les interactions entre les éléments et certains critères architecturaux. La documentation des interactions entre les éléments est un aspect important de l'architecture modulaire. Pimmler et Eppinger (1994) définissent quatre types génériques d'interaction :

- espace : nécessité d'une orientation particulière ou de la proximité de deux éléments ;
- énergie : nécessité d'un transfert d'énergie entre deux éléments ;
- information : nécessité d'un échange d'information entre deux éléments ;
- matériel : nécessité d'un échange de matériel entre deux éléments.

Ces interactions influencent le groupement des éléments en sous-assemblages. Il est possible d'illustrer cette approche avec un exemple. Soit un système de climatisation dans une automobile. Le système de climatisation concerne à la fois la ventilation

de l'habitacle interne et la ventilation du moteur. Le système a été décomposé en seize éléments. La Figure 2.2 (Pimmler et Eppinger, 1994) qui prend la forme d'une matrice énumère ces seize éléments et les interactions qui existent entre ceux-ci.

Plusieurs algorithmes permettent de réordonner les lignes et les colonnes de la ma-

		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Radiateur	A		1														
Ventilateur du moteur	B	1				1											
Moteur du chauffage	C																1
Conduits de chauffage	D																
Condensateur	E		1				1		1								
Compresseur	F					1			1	1							
Conduits d'évaporation	G																1
Évaporateur	H					1	1			1							1
Accumulateur	I						1		1								
Contrôleur de réfrigérations	J																
Contrôleur d'air	K																
Senseur	L																
Commande de distribution	M																
Déclencheurs	N																
Contrôleur de la ventilation	O																1
Moteur du ventilateur	P			1				1	1							1	

FIGURE 2.2 Matrice d'interractions des éléments

trice de la Figure 2.2 pour regrouper les interactions près de la diagonale. La Figure 2.3 (Pimmler et Eppinger, 1994) illustre cette réorganisation de la matrice d'interaction des éléments du système de climatisation. Il est ensuite possible de regrouper les éléments selon trois sous-assemblages. Cette approche nécessite la décomposition du produit en plus petits éléments que le niveau final désiré. Les sous-assemblages définissent l'architecture du produit.

Selon Yassine (2003), il y a trois configurations qui peuvent caractériser un système complexe :

- parallèle (concurrente) : les éléments du système n'interagissent pas ensemble ;
- séquentielle (dépendante) : un élément influence le comportement d'un autre

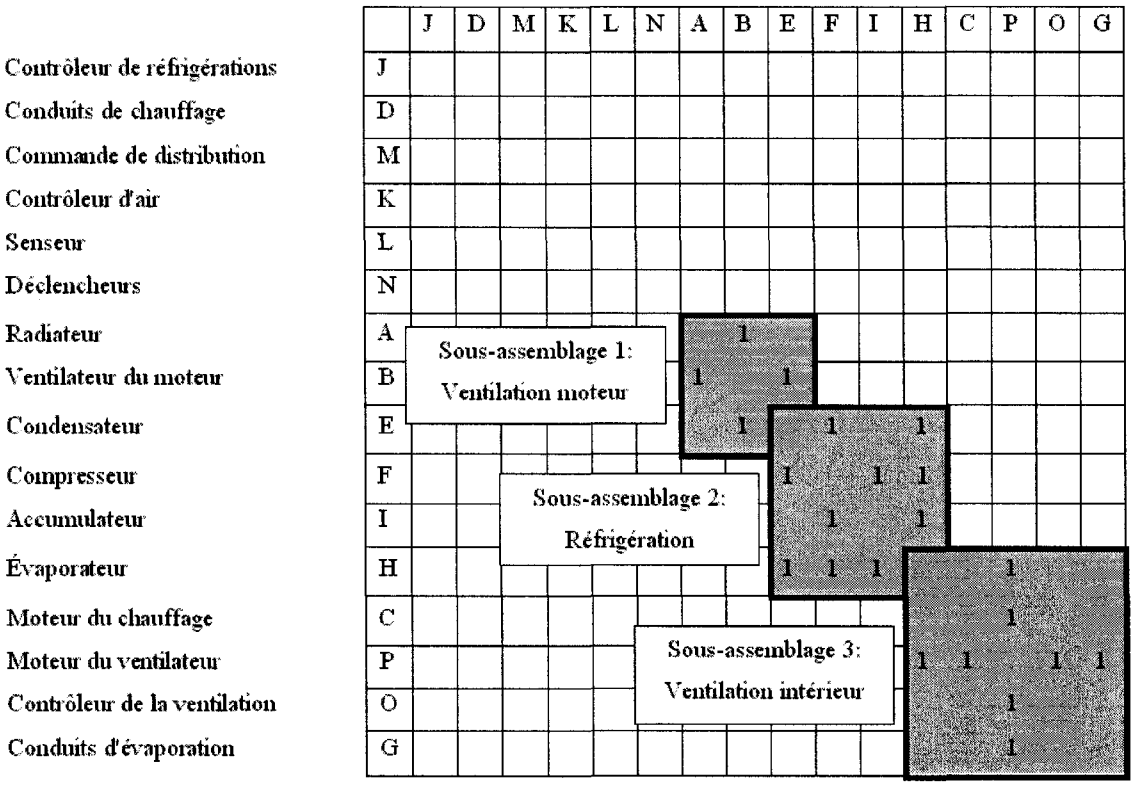

FIGURE 2.3 Matrice d'interractions des éléments réorganisée

élément ;

- couplée (interdépendante) : tous les éléments du système s'influencent mutuellement.

La Figure 2.4 illustre la représentation sous forme fonctionnelle et matricielle de ces trois configurations possibles.

Selon Marshall *et al.* (1998), les modules d'une architecture modulaire ont un certain nombre de caractéristiques qui fournissent des différences fondamentales permettant de les différencier. Il y a quatre caractéristiques principales :

- les modules sont des sous-systèmes coopératifs qui forment les produits ;
- les modules ont leurs interactions principales à l'intérieur plutôt qu'entre les modules ;
- les modules ont une ou plusieurs fonctions bien définies qui peuvent être testées de façon isolée du produit ;

- les modules sont indépendants et fonctionnels en soi et peuvent être combinés et configurés avec des unités similaires pour fournir un produit différent.

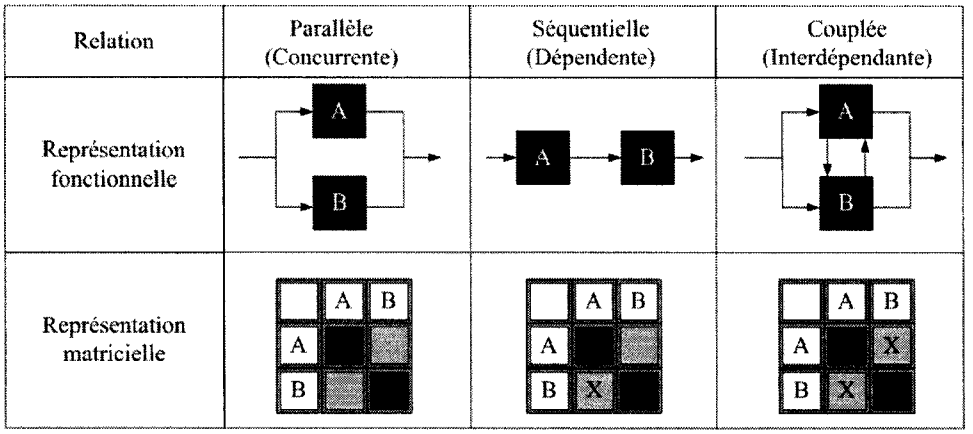

FIGURE 2.4 Configurations

Newcomb *et al.* (2003) définissent la modularité comme un concept qui consiste à décomposer un système en plusieurs parties ou modules indépendants qui peuvent être traités comme des unités logiques. La façon dont le produit est divisé en modules a un impact important sur l'assemblage. Selon Jablan (1997), la modularité consiste à utiliser plusieurs éléments de base (modules) pour construire une panoplie de structures modulaires différentes. Les différents modules sont des briques architecturales. La recombinaison de ces éléments de base selon un ensemble fini de patrons (configurations) permet d'obtenir une diversité de structures architecturales.

Ulrich et Tung (1991) utilisent le concept de modularité comme objectif de conception. Ils définissent la modularité selon deux caractéristiques :

- similarité entre l'architecture physique et fonctionnelle du design ;
- minimisation des interactions non désirées entre les composants physiques.

Selon eux, l'architecture d'un produit est l'arrangement des fonctionnalités d'un produit à l'intérieur de blocs physiques. Un module doit avoir une correspondance une à une entre les fonctionnalités dans la structure fonctionnelle et les composants physiques du produit. Ulrich et Tung ne considèrent pas la possibilité qu'un module puisse avoir plus d'une fonctionnalité. Ils considèrent que toutes les interfaces entre les modules sont découplées. Ils mettent toutefois en évidence le fait que les interactions entre les modules sont un aspect important de la modularité. Ulrich et Tung ont

aussi défini les bénéfices potentiels de la modularité. Ils ont identifiés neuf bénéfices :

- économie d'échelle des composants ;
- changement des produits ;
- variété des produits ;
- temps de conception ;
- découplage des tâches ;
- focus sur le design et la production ;
- vérification et test des composants ;
- consommation différentielle ;
- facilité de production, d'installation et d'utilisation.

Ils ont aussi défini les désavantages potentiels de la modularité. Ils ont identifié cinq désavantages :

- architecture statique des produits ;
- optimisation des performances ;
- facilité de la réingénierie ;
- augmentation des coûts variables des unités ;
- similarité excessive entre les produits.

Ulrich (1995) raffine sa notion de l'architecture d'un produit. Selon lui, l'arrangement d'un produit peut être définit comme l'arrangement des éléments fonctionnels en sous-assemblages qui deviennent les modules pour une famille de produit. L'architecture de ces produits est caractérisée par trois aspects :

- l'arrangement des éléments fonctionnels ;
- l'association entre les éléments fonctionnels et les éléments physiques ;
- la spécification des interfaces entre les composants physiques.

Selon Ulrich, il y a deux types d'architectures de produit :

- architecture intégrale ;
- architecture modulaire.

Dans une architecture modulaire, il y a une association un à un entre les éléments fonctionnels et les éléments physiques. Cette association implique que les interfaces entre les composants physiques soient découplées. Dans une architecture intégrale, il y a une association plusieurs à un ou un à plusieurs entre les éléments fonctionnels et les éléments physiques. Ces deux types d'association impliquent que les interfaces entre les composants ne sont pas découplées puisqu'elles sont dépendantes. La Figure

2.5 illustre ces deux types d'architecture.

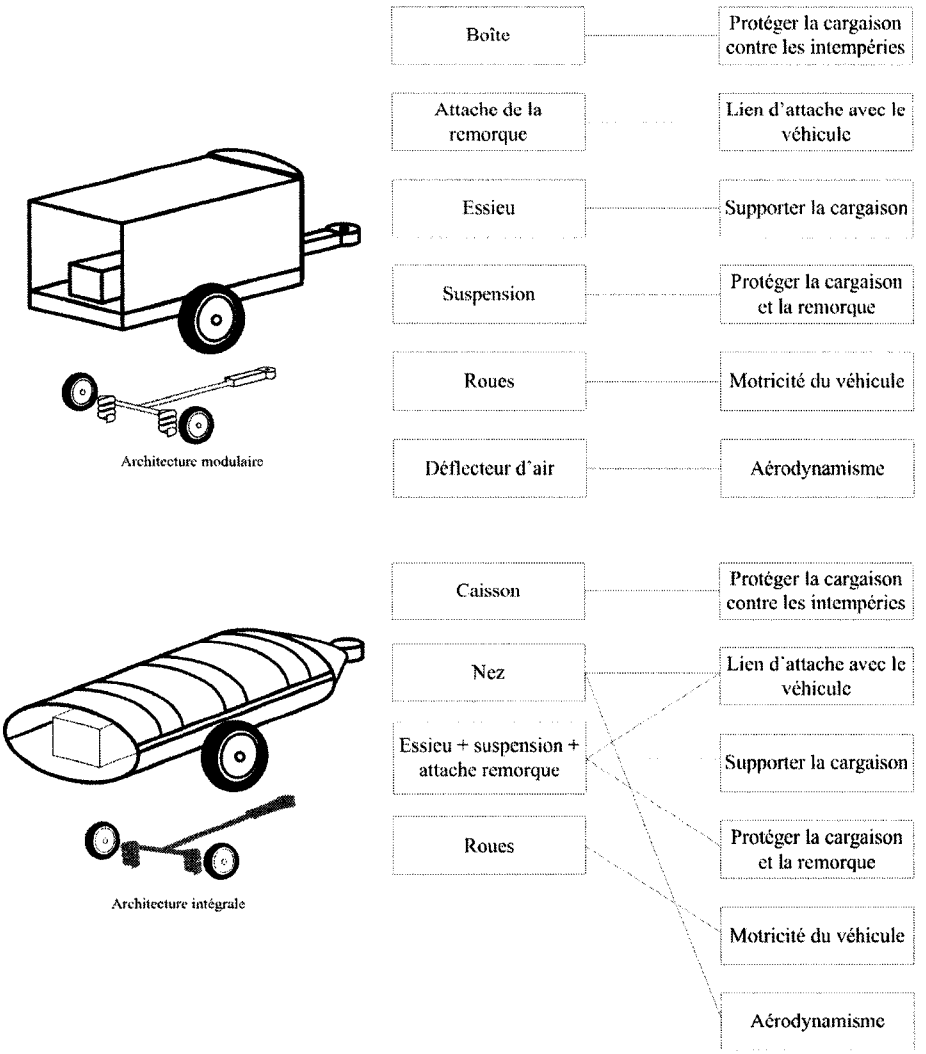

FIGURE 2.5 Types d'architecture

Selon Ulrich et Eppinger (2004), il existe trois types de modularité qui sont illustrées à la figure 2.6 :

- fente : interfaces différentes pour chaque module, arrangement des modules non interchangeables ;
- bus : mêmes interfaces pour tous les modules, arrangement interchangeables ;
- arête : mêmes interfaces pour tous les modules, arrangement interchangeables, assemblage direct entre les modules.

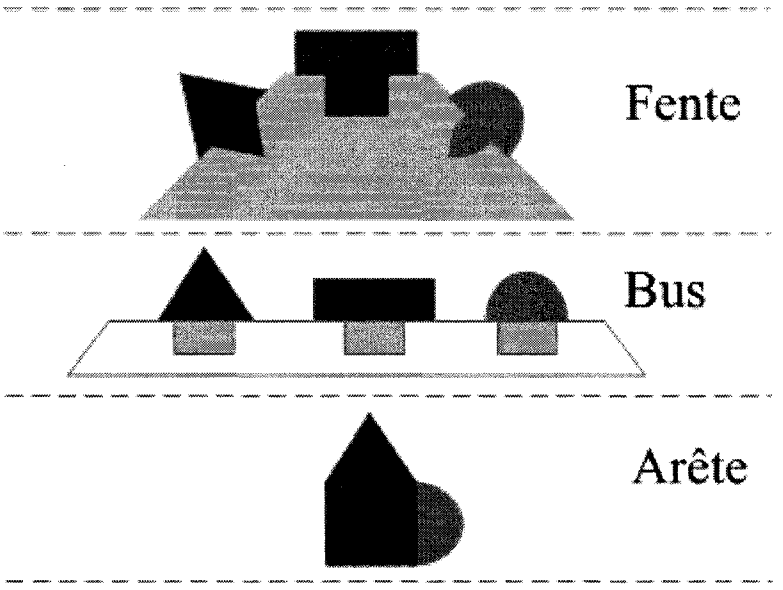

FIGURE 2.6 Types de modularité

Scheidt et Zong (1994) proposent une approche visant à réutiliser des modules de dispositifs électroniques. Pour maximiser la réutilisation, il est nécessaire de concevoir un produit avec des modules de fonctionnalités différentes plutôt que de concentrer toutes les fonctionnalités dans un seul bloc (module). Un produit modulaire est constitué de modules qui peuvent potentiellement être réutilisés dans d'autres produits. Il y a des désavantages possibles à la réutilisation. Un des arguments fréquent contre la réutilisation est la nécessité de l'innovation pour rester compétitif. Dans le passé, l'accent était mis sur l'optimisation des coûts de production. Maintenant, il faut aussi considérer les coûts de conception. Scheidt et Zong définissent trois requis nécessaires pour une approche de conception orientée vers la réutilisation :

- séparation claire des fonctionnalités des modules ;
- assemblage facile des modules ;
- possibilités techniques de vérification de la qualité des modules.

Selon Gershenson *et al.* (2003), il n'y a qu'un seul consensus dans la littérature en ce qui concerne la définition du concept de modularité : un produit modulaire est fait de modules, de blocs fonctionnels. Plus il y a de composants dans un module, plus le module est considéré comme modulaire. La définition de la modularité est

basée sur le concept de module, mais la plupart du temps la définition d'un module n'est pas donnée dans la littérature. Selon Baldwin et Clark (2000), le concept de modularité englobe deux idées. La première est l'interdépendance dans les modules et l'indépendance entre les modules. Un module est une unité dont les éléments structurels sont fortement liés entre eux et relativement peu liés aux éléments des autres unités. Il a plusieurs degrés de liaison entre les éléments et par conséquent il y a plusieurs degrés de modularité. La seconde idée englobe trois aspects : l'abstraction, le masquage de l'information et les interfaces. Un système complexe peut être traité en le divisant en morceaux plus petits et en les analysants séparément. Quand la complexité d'un des éléments dépasse un certain seuil, la complexité peut être isolée en définissant une nouvelle abstraction qui a une interface simple. L'abstraction masque la complexité de l'élément. L'interface indique comment les éléments interagissent avec le système. Cette définition de la modularité, basée sur les relations entre les structures, contraste avec celle d'Ulrich (1995) basée sur les fonctionnalités.

Selon Baldwin et Clark (2000), dans une perspective de modularité, un ensemble de règles de conception doivent adresser les aspects suivants :

- architecture : quels modules vont faire partie du système et quels seront leurs rôles ;
- interface : description détaillée de la façon dont les différents modules interagissent entre eux ;
- protocole d'intégration et normes de tests : ensemble de procédures qui permettent d'assembler le système et de déterminer qu'il fonctionne bien.

Nepal (2005) a déterminé que l'architecture d'un produit peut être convertit en architecture modulaire de deux façons :

- mise en grappe des éléments fonctionnels ;
- mise en grappe des composants physiques.

Selon Fixson (2007), la description d'un produit modulaire comporte deux dimensions :

- les éléments qui forment le produit, c'est-à-dire les modules ;
- les relations entre les éléments, c'est-à-dire les interfaces.

Les éléments sont représentés par des boîtes et les interfaces par des lignes connectant les boîtes. Fixson a regroupé les différentes définitions du concept de modularité en trois catégories qui sont illustrées à la figure 2.7 :

- paramétrique : forme la plus simple où les fonctionnalités des éléments sont fixes et certains éléments peuvent être remplacés ;
- configuration : regroupement de petits éléments en plus gros pour former des modules ;
- fondamentale : permet une réallocation de toutes les fonctionnalités des éléments.

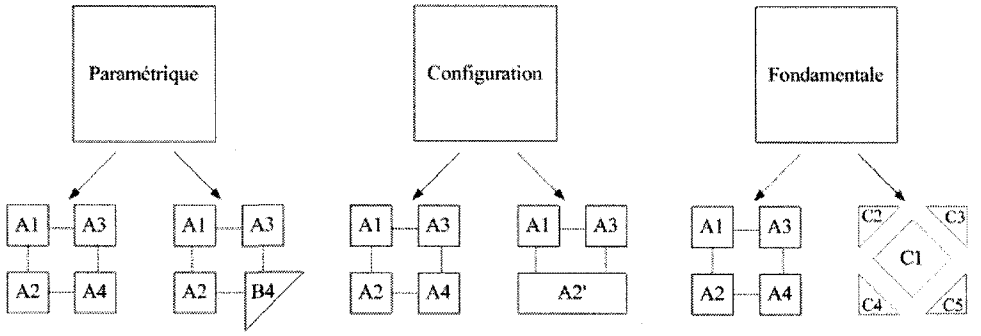

FIGURE 2.7 Catégories modularité

Les catégories diffèrent dans la façon dont les fonctionnalités sont attribuées aux éléments du produit. Fixson définit aussi trois catégories d'interfaces par leur niveau de détails dans leur description :

- niveau bas de détails : interfaces identiques ;
- niveau moyen de détails : nécessité de l'interopérabilité et introduction de la notion générale d'uniformisation ;
- niveau élevé de détails : introduction de mesures d'évaluation qualitatives de la qualité en fonction du niveau de dépendance et de la nature physique des éléments.

Suh (1990), définit deux axiomes de conception qui gouvernent un bon processus de design. Le premier axiome traite de la relation qui existe entre les requis fonctionnels et les composants physiques et le second traite de la complexité du design.

Axiome 1 : l'axiome de l'indépendance

Un design optimal maintient toujours l'indépendance des requis fonctionnels (RFs). Dans un design acceptable, les paramètres de design (PDs) et les RFs sont liés d'une façon telle qu'un PD spécifique peut être ajusté pour satisfaire sont RF correspondant sans affecté les autres RFs.

Axiome 2 : l'axiome de l'information

Le meilleur design est celui dont les fonctionnalités ne sont pas couplées et dont le contenu d'information du design est minimum. L'axiome 1 stipule que durant le processus de design, l'association entre les RFs dans le domaine fonctionnel et les PDs dans le domaine physique doit être telle qu'une perturbation dans un PD particulier n'affecte que son RF correspondant. L'axiome 2 stipule que parmi tous les designs qui satisfont l'axiome 1, celui avec le contenu d'information minimum est le meilleur. Le terme meilleur est utilisé de façon relative étant donné qu'il y a potentiellement une infinité de design qui satisfont l'axiome 1.

Vitharana *et al.* (2004) ont défini cinq objectifs et stratégies pour la conception de composants logiciels :

- rentabilité ;
- facilité d'assemblage ;
- facilité de personnalisation ;
- réutilisabilité ;
- facilité de maintenance.

Ils ont aussi défini cinq caractéristiques techniques pour la conception des composants logiciels :

- couplage ;
- cohésion ;
- nombre de composants ;
- taille des composants ;
- complexité.

Hopkins (2000) identifie le couplage et la cohésion comme des critères de conception important pour les systèmes complexes .

2.2 Réutilisation

Prieto-Diaz (1993) a identifié six facettes ou perspectives à la réutilisabilité : substance, portée, mode, technique, intention et produit. Le Tableau 2.1 résume les éléments qui composent chacune des six facettes.

TABLEAU 2.1 Perspectives de la réutilisabilité

Substance	Portée	Mode	Technique	Intention	Produit
Idées	Verticale	Planifiée	Composition-nelle	Boîte noire	Design
Concepts	Horizontale	Systématique	Générative	transparente	Architecture
Procédures		Ad-Hoc		grise	
Compétences		Opportuniste		blanche	
Composants					

La facette substance inclut la réutilisation d'idées (concepts), de procédures et de composants (modules). Elle indique quel type de réutilisation est utilisé. La réutilisation d'idées (concepts) implique une solution générique à une classe de problèmes donnés et par conséquent un degré d'abstraction élevé. La réutilisation de procédures implique la réutilisation d'expertise, de savoir-faire et d'expérience professionnelle. La réutilisation de composants implique la réutilisation de modules physiques existants. La facette portée comprend la réutilisation verticale et horizontale. La réutilisation verticale s'applique aux modules relatifs au même champ d'application. L'objectif de la réutilisation verticale est de dériver un module générique pour une famille de systèmes. La réutilisation horizontale s'applique aux modules relatifs à des champs d'application différents. L'objectif de la réutilisation horizontale est de dériver un module générique applicable à différents champs d'application. La facette mode concerne la manière dont la réutilisation est organisée. Il y a deux manières d'organiser la réutilisation : planifiée et ad hoc. La réutilisation planifiée fait usage de pratiques formelles, de lignes de conduite, de procédures et de métriques bien définies. La réutilisation est structurée. La réutilisation ad hoc n'est pas structurée par des pra-

tiques formelles ou des procédures. Elle opère à l'intérieur d'une équipe de travail ou d'un petit bureau. Une librairie générale où les modules sont entreposés est employée pour la réutilisation. Les modules réutilisés sont sélectionnés à l'intérieur de cette librairie. La facette technique comprend les techniques de réutilisation compositionnelle et générative. La réutilisation compositionnelle fait usage des modules existant pour développer de nouveaux systèmes. Ce type de réutilisation se base sur des interfaces uniformes. La réutilisation générative fait usage de modèles formels et spécifications. Les modules sont générés automatiquement à partir de modèles uniformes découlant des spécifications. La facette intention est une représentation conceptuelle du type de réutilisation. Il y a quatre types de réutilisation : boîte noire, boîte transparente, boîte grise et boîte blanche. La Figure 2.8 illustre cette facette. Un module de type boîte

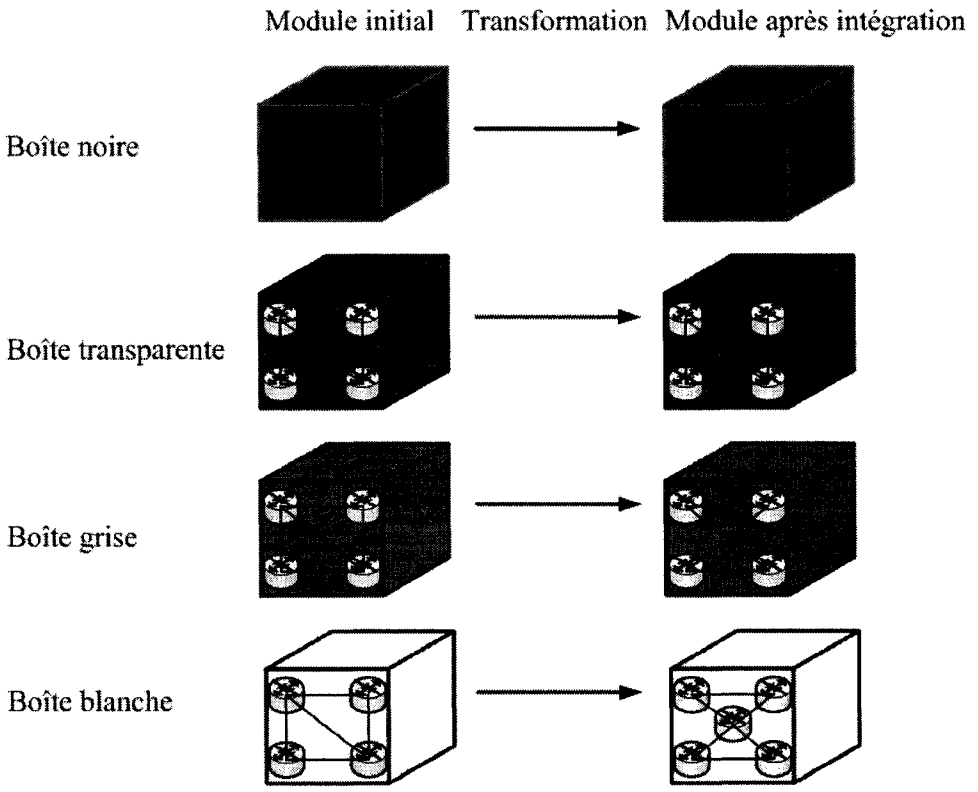

FIGURE 2.8 Types de réutilisation

noire est entièrement défini en termes de fonctionnalités et d'interfaces. Le schéma de la structure interne du design du module n'est pas accessible. Le design du module ne peut être modifié. Une description détaillée des fonctionnalités, des interfaces et

de l'environnement dans lequel le module doit être utilisé caractérise le module. Un module de type boîte transparente est un module dont le schéma de la structure interne du design du module est accessible, mais dont le design du module ne peut être modifié. L'impact et la performance de l'assemblage de deux modules de types boîte transparent peut être évalué étant donné que le design interne des modules est connu. Un module de type boîte grise est un module dont le schéma de la structure interne du design du module est accessible et dont certains éléments du design peuvent être modifiés. Un module de type boîte blanche est un module dont le schéma de la structure interne du design du module est accessible et dont tous les éléments du design peuvent être modifiés. Le module est fourni dans une forme initiale prédéfinie. La facette produit concerne le type de produit qui peut être réutilisé. Cette facette est une représentation détaillée des items de la facette substance. Le Tableau 2.2 illustre quelques exemples de produits réutilisés en lien avec la facette substance.

TABLEAU 2.2 Facette produit

Idées(concepts)	Procédures	Composants(modules)
Spécifications	Documentation	Design Architecture

2.3 Métriques du design

Vitharana *et al.* (2004) ont identifié cinq caractéristiques techniques pour les composants logiciels. Les équations (2.1) à (2.5) définissent chacune de ces caractéristiques : le couplage (2.1), la cohésion (2.2), le nombre de composants (2.3), la taille des composants (2.4) et la complexité (2.5). Le Tableau 2.3 définit les paramètres pour le couplage et pour la cohésion. Le Tableau 2.4 définit les paramètres pour la complexité. Le couplage entre deux classes peut être dérivé du diagramme de classe.

$$COUPL = \sum_{k=1}^{y} \sum_{i=1}^{n} \sum_{i=1, i \neq j}^{n} (x_{ik} * (1 - x_{ij}) * c_{ij}) \tag{2.1}$$

$$COHES = \sum_{k=1}^{y} \sum_{i=1}^{n} \sum_{i=1, i \neq j}^{n} (x_{ik} * x_{jk} * c_{ij}) \tag{2.2}$$

TABLEAU 2.3 Paramètres pour le couplage et la cohésion

y	=		nombre de composants du domaine
n	=		nombre total de classe dans le domaine
x_{ik}	=	1	si la classe i est placée dans le composant k
	=	0	sinon
c_{ij}	=		couplage entre la classe i et j $(i, j \geq 0$ entier ; $i \neq j)$

$$NCOMP = y \tag{2.3}$$

$$CSIZE = \sqrt{\frac{\sum_{j=1}^{y} \left(\sum_{i=1}^{n} x_{ij}\right)^2}{y}} \tag{2.4}$$

$$COMPL = \sum_{j=1}^{y} \left(\sum_{i=1}^{n} \left((w_{mcx} * m_i) + \left(w_{pcx} * \sum_{k=1}^{m_i} \left(\sum_{l=1}^{p_{ik}} p_{ikl} \right) \right) \right) * x_{ij} \right)^2 \tag{2.5}$$

TABLEAU 2.4 Paramètres pour la complexité

y	=		nombre de composants du domaine
n	=		nombre total de classe dans le domaine
x_{ik}	=	1	si la classe i est placée dans le composant k
	=	0	sinon
w_{mcx}	=		importance relative de la complexité de la méthode $(0 \leq w_{mcx \leq 1})$
m_i	=		nombre de méthodes publiques dans la classe i $(m_i \geq 0)$
w_{pcx}	=		importance relative de la complexité du paramètre $(0 \leq w_{pcx} \leq 1)$
p_{ik}	=		nombre de paramètres dans la méthode publique k de la classe i $(p_{ik} \geq 0)$
p_{ikl}	=		complexité relative du paramètre l dans la méthode k de la classe i $(0 \leq p_{ikl} \leq 1)$

Sigfried (1996) a estimé que la complexité d'un système contenant n item est n^2. Les valeurs de w_{mcs} , w_{pcx} et p_{ikl} sont assignées par le développeur lors du processus de conception. C'est lui qui assigne l'importance relative de la complexité des méthodes, paramètres globaux et paramètres individuels dans la conception des composants. Après avoir défini les différentes métriques de design et objectifs de conception, Vitharana et al. proposent un modèle d'optimisation basé sur ces métriques et objectifs.

Le vecteur R' représente l'importance relative des objectifs de conception. La matrice W représente la force de l'association entre les objectifs de conception et chacune des métriques de design. Le vecteur D représente la mesure des différentes métriques de design. L'objectif est de maximiser : $R'WD$. Les équations (2.6) à (2.8) illustrent les paramètres de l'optimalité. Le tableau 2.5 résume les différents paramètres qui permettent d'établir l'optimalité. Vitharana et al. cherchent à trouver l'équilibre entre l'atteinte des objectifs et stratégies de conception et les caractéristiques techniques des composants logiciels.

$$R' = \begin{bmatrix} R_{COST} & R_{ASBL} & R_{CUST} & R_{REUS} & R_{MNTN} \end{bmatrix} \tag{2.6}$$

$$W = \begin{bmatrix} w_{11} & w_{12} & w_{13} & w_{14} & w_{15} \\ w_{21} & w_{22} & w_{23} & w_{24} & w_{25} \\ w_{31} & w_{32} & w & w_{34} & w_{35} \\ w_{41} & w_{42} & w_{43} & w_{44} & w_{45} \\ w_{51} & w_{52} & w_{53} & w_{54} & w_{55} \end{bmatrix} \tag{2.7}$$

$$D = \begin{bmatrix} D_{COUPL} \\ D_{COHES} \\ D_{NCOMP} \\ D_{CSIZE} \\ D_{COMPL} \end{bmatrix} \tag{2.8}$$

Briand *et al.* (1998) ont proposé cinq propriétés relatives au couplage. $ClasseR(c)$ indique l'ensemble des relations de la classe c vers une autre classe où d'une autre classe vers la classe c. $InterR(C)$ indique l'ensemble des relations interclasse dans le système orienté objet C. $InterR(C)$ correspond à $\cup_{c \in C} ClasseR(C)$. Il y a trois propriétés importantes.

TABLEAU 2.5 Paramètres pour l'optimalité

R_{COST}	=	importance relative de la rentabilité
R_{ASBl}	=	importance relative de la facilité d'assemblage
R_{CUST}	=	imortance relative de la facilité de personnalisation
R_{REUS}	=	importance relative de la réutilisabilté
R_{MNTN}	=	importance relative de la facilité de maintenance
w_{ij}	=	force de l'association entre l'objectif i et l'aspect technique j
D_{COUPL}	=	mesure du couplage
D_{COHES}	=	mesure de la cohésion
D_{NCOMP}	=	mesure du nombre de composants
D_{CSIZE}	=	mesure de la taille des composants
D_{COMPL}	=	mesure de la complexité

Propriété 1 : La non-négativité

Le couplage de la classe c du système orienté objet C est non-négatif :

$$[Couplage(c) \geq 0 | Couplage(C) \geq 0]$$

Propriété 2 : Fusion de classes interconnectées

Soient c_1 et c_2 deux classes du système orienté objet C. Soit c' la classe résultant de l'union de c_1 et c_2. Soit C' le système orienté objet identique à C excepté que les classes c_1 et c_2 sont remplacées par c' . Alors :

$$[Couplage(c_1) + Coupalge(c_2) \geq Couplage(c') | Couiplage(C) \geq Couplage(C')]$$

Propriété 5 : Fusion de classes non-interconnectées

Soient c_1 et c_2 deux classes du système orienté objet C. Soit c' la classe résultant de l'union de c_1 et c_2. Soit C' le système orienté objet identique à C excepté que les classes c_1 et c_2 sont remplacées par c'. Alors :

$$[Couplage(c_1) + Coupalge(c_2) = Couplage(c') | Couiplage(C) = Couplage(C')]$$

Constantine *et al.* (1974) et Constantine et Yourdon (1979) ont défini sept niveaux de cohésions entre deux éléments :

- coïncidence (pire) : fonctions non-reliées ;
- logique : fonctions logiquement reliées ;
- temporelle : fonctions exécutées dans un même intervalle de temps ;
- procédurale : multiples fonctions reliées à une procédure générale ;
- communicationnelle : multiple fonctions ayant des données communes ;
- séquentielle : multiple fonctions bien définies dont les données sont le résultat de la fonction précédente ;
- fonctionnelle (meilleure) : une seule fonction bien définie.

Emerson (1984) a par la suite regroupé ces sept niveaux de cohésion en trois classes :

- I : communicationnel, séquentiel et fonctionnel ;
- II : temporel et procédural ;
- III : logique et coïncidence.

2.4 Coûts de réutilisation

Au niveau des coûts de réutilisation, la plupart des travaux se situent au niveau du génie logiciel. Les coûts de réutilisation sont évalués en fonction du nombre de lignes de codes, de classes, de fonctions et de paramètres. Barnes et Bollinger (1991) ont proposé un modèle économique pour la réutilisation de composants logiciels. Les équations (2.9) à (2.11) définissent le modèle. Le Tableau 2.6 décrit les différents paramètres du modèle.

$$RC = \left(b + \left(\frac{E}{N}\right) - 1\right) * R + 1 \tag{2.9}$$

$$RP = \frac{1}{RC} \tag{2.10}$$

$$N_0 = \frac{E}{1-b} \tag{2.11}$$

TABLEAU 2.6 Paramètres pour le modèle de Barnes et Bollinger

R	=	% du code fourni par des composants réutilisables
b	=	coût d'intégration des composants réutilisables
RC	=	coût relatif
RP	=	productivité relative
N	=	nombre de réutilisations
E	=	coût relatif de modification d'un composant pour le rendre réutilisable
N_0	=	seuil de rentabilité (nombre de réutilisations nécessaires pour amortir les coûts de développement des composants réutilisables)

À l'aide de ce modèle, Favaro (1991) a mené une étude industrielle qui a révélé que le coût de développement d'un composant réutilisable est souvent le double du coût de développement d'un composant non-réutilisable.

Dans une optique différente, Mili *et al.* (1995) ont quantifié les coûts de réutilisation d'un module de type boîte noire et boîte blanche. Le modèle qu'ils proposent se base sur la probabilité que le module recherché soit trouvé et sur la probabilité qu'il y ait des modifications à apporter.

$$C_{RBN} = R + (1-p) * DevMod \tag{2.12}$$

$$C_{RBN} = R + (1-p) * (RP + q * A + (1-q) * DevMod) \tag{2.13}$$

2.5 Représentation et documentation

Un des aspects de la réutilisation est la représentation et la documentation du design. Pour qu'une solution existante puisse être réutilisée, il faut qu'elle soit bien documentée. La forme sous laquelle la solution est représentée doit être structurée et uniformisée. À cet égard, il existe en génie logiciel un outil qui permet à la fois

TABLEAU 2.7 Paramètres pour les coûts de réutilisation selon Mili

R	=	coût moyen de recherche dans la liste des modules existans
p	=	probabilité que le module recherché soit trouvé
RP	=	coût moyen d'une recherche partielle (moins de restrictions) dans la liste des modules existants
q	=	probabilité qu'il y ait des modifications à apporter au module
$DevMod$	=	coût moyen de développement d'un module non réutilisable

de documenter et de représenter une solution sous une forme structurée : patron de conception.

Alexander *et al.* (1996) ont introduit la notion de patron de conception. Un patron de conception décrit un problème de design (informatique) qui est survenu à plusieurs reprises dans un champ d'application donné. Le patron de conception décrit le coeur de la solution pour le problème donné, d'une telle façon que la solution peut être utilisée un grand nombre de fois pour résoudre ce problème récurrent. L'idée du patron de conception s'appuie sur le principe du déjà-vu. Le patron de conception est une encapsulation des meilleurs pratiques concernant un problème donné déjà résolu. Selon Gamma *et al.* (1995), d'un point de vue général, un patron de conception contient quatre éléments :

- nom du patron ;
- problème ;
- solution ;
- conséquences.

Le nom du patron de conception est généralement composé de deux ou trois mots qui décrivent clairement le problème, la solution et les conséquences. Les différents noms des patrons de conception servent à bâtir un vocabulaire qui augmente le niveau d'abstraction. L'élément problème est une description des situations où le patron peut s'appliquer. Cette description explique la nature du problème à résoudre, son contexte et ses conditions d'applicabilité. L'élément solution est une description des éléments de design de la solution, leurs relations, leurs responsabilités et leurs collaborations. Ce n'est pas une description d'une implémentation concrète, mais plutôt une description abstraite d'une solution à un problème donné qui fournit un cadre de résolution. L'élément conséquences est une description des résultats résultant de l'application du

patron de conception, de l'aspect économique (coûts et bénéfices) et des impacts sur la flexibilité, l'évolutivité et la portabilité du design. La description formelle d'un patron de conception comporte douze aspects. Le Tableau 2.8 résume ces douze aspects.

TABLEAU 2.8 Aspects d'un patron de conception

Nom du patron	Nom équivoque qui résume l'essence du patron
Intention	Quel problème adresse le patron de conception
Motivation	Description d'un scénario qui illustre le problème de design
Applicabilité	Description des conditions d'applicabilité du patron de conception
Structure	Représentation graphique des classes présentes dans le patron de conception et leurs interractions
Participants	Description des classes et objets présents dans le patron de conception
Collaborations	Description des collaborations entre les participants
Conséquences	Description des résultats suite à l'utilisation du patron de conception
Implémentation	Description de pièges à éviter, conseils, techniques, langages... à connaître lors de l'implémentation
Échantillons	Fragments de code qui illustrent les bonnes pratiques
Usages connus	Description d'utilisations réelles du patron dans des domaines d'application différents
Patrons associés	Description des patrons qui sont conceptuellement proches et des collaborations possibles avec ceux-ci

2.6 Questions ouvertes

Suite à cette revue de littérature, plusieurs questions demeurent ouvertes. Dans le contexte des réseaux de télécommunications, il faut être en mesure d'appliquer les différents concepts de l'architecture modulaire. Il n'y a pas de travaux dans le domaine des télécommunications qui traitent de la conception basée sur l'architecture modulaire au niveau matériel. L'ensemble des travaux décrits dans cette revue de littérature portent sur le génie industriel et logiciel. Le Tableau 2.9 résume l'ensemble

des questions soulevées par cette revue de littérature.

TABLEAU 2.9 Questions ouvertes sur l'architecture modulaire

Qu'est-ce qu'un module dans le contexte des réseaux de télécommunication ?
Quelles sont les caractéristiques des modules ?
Que contient un module ?
Comment est-il représenté ?
Quelles métriques caractérisent un module ?
Comment déterminer l'optimalité d'un module ?
Comment décomposer l'architecture en modules ?
Comment assembler les modules ?
Combien coûte un module réutilisable ?
Combien coûte la réutilisation ?
Comment gérer les changements des modules dus aux changements rapides des nouvelles technologies ?

CHAPITRE 3

ARCHITECTURE MODULAIRE PROPOSÉE

Une façon de résoudre un problème complexe est de le diviser en plusieurs sous-problèmes. Chaque sous-problème est plus simple à résoudre que le problème entier. Ce raffinement peut opérer à plusieurs niveaux. L'architecture modulaire se base sur ce principe. L'objectif de ce mémoire est d'appliquer ce concept à la conception des réseaux de télécommunications. Dans un premier temps, les requis de l'architecutre modulaire sont établis. Par la suite, l'architecture modulaire proposée est exposée. Les différentes métriques qui permettent de caractériser cette architecture et les différents coûts liés à la conception et à la réutilisation d'une telle architecutre sont définis. Finalement, un mécanisme de réutilisation structuré basé sur cette architecture modulaire est proposé.

3.1 Requis de l'architecture

Avant de définir les requis pour l'architecture modulaire, il est nécessaire de définir la notion d'architecture modulaire dans l'optique des réseaux de télécommunications ainsi que la notion même de module.

3.1.1 Définitions

Architecture modulaire

Une architecture modulaire est une architecture fonctionnelle d'un réseau de télécommunications formée par l'assemblage de modules. Cette architecture modulaire fournit un ou plusieurs services réseaux.

Module

Un module est une entité fonctionnelle (matérielle et/ou logicielle) qui fait partie d'une architecture modulaire. Un module est complètement défini par ses interfaces et ses fonctionnalités. Il existe une séparation claire et distincte entre l'implémentation du module et sa caractérisation (interfaces et fonctionnalités). L'ensemble des fonctionnalités d'un module fournit un ou plusieurs services. Les différentes interfaces définissent les entrées et sorties d'un module. Celles-ci caractérisent les transactions possibles avec les modules adjacents. Une transaction est une interaction (échange d'informations) avec un module adjacent. Il existe deux types opposés de transaction : dépendance et complémentarité. Une transaction peut se situer entre les deux.

3.1.2 Requis architecture

L'architecture proposée doit permettre de :
- réduire les coûts de conception ;
- améliorer les performances réseaux ;
- uniformiser le module optimal pour la réutilisation.

Par hypothèse, le module optimal au niveau des performances doit être réutilisé. Le volet optimisation des performances n'est pas considéré dans ce mémoire. Ce mémoire se concentre sur une méthode de conception pour les réseaux de télécommunications basée sur l'architecture modulaire. La réduction des coûts de conception et l'uniformisation du module optimal pour la réutilisation sont abordées.

3.1.3 Requis modules

Un module doit posséder les cinq caractéristiques suivantes : indépendance, interopérabilité, réutilisabilité, évolutivité et dissociation des fonctionnalités et de l'implémentation.

Indépendance

C'est le caractère auto-suffisant du module. Le fonctionnement d'un module ne doit pas dépendre des modules avec lesquels il est interconnecté. Le couplage entre les modules doit être minimal. Deux modules indépendants peuvent être complémentaires.

La notion de complémentarité réfère à des transactions entre modules adjacents dont le contenu n'influence pas le fonctionnement de ces modules. Un module de service et un module de connectivité sont complémentaires. Un module de service et un module d'inter-connectivité sont indépendants. Un module d'inter-connectivité et un module de connectivité sont complémentaires. Une architecture décentralisée implique des modules de service complémentaires. Une architecture de centralisée implique une dépendance et une complémentarité entre les modules de service de la centrale et ceux des succursales.

Interopérabilité

La notion d'interopérabilité fait référence à la nécessité d'assembler de façon transparente des modules hétérogènes. Les interfaces entre les modules doivent être uniformisées de façon à permettre l'assemblage homogène des modules.

Réutilisabilité

Un module doit être utilisable dans plusieurs problèmes donnés appartenant au même champ d'application. Un module doit pouvoir être réutilisé pour plusieurs clients différents.

Évolutivité

Un module doit être flexible quant à l'ajout de nouvelles fonctionnalités et la modification et/ou le retrait de fonctionnalités existantes. La structure du module doit permettre la maintenance et la modernisation du design.

Dissociation fonctionnalités vs implémentation

Un module est défini par ses fonctionnalités et non par son implémentation. L'implémentation est transparente aux fonctionnalités.

3.1.4 Métriques

Il existe deux types de métriques. Le premier type concerne la structure du design. Le deuxième type concerne la performance des réseaux.

Structure du design

La structure du design d'un module peut être caractérisée par quatre métriques :
- couplage ;
- cohésion ;
- taille ;
- complexité.

Performance

Un module peut être caractérisé par différentes métriques de performances :
- délai ;
- gigue ;
- taux de perte de paquets ;
- largeur de bande des interfaces ;
- temp de recouvrement d'une panne ;
- etc.

3.2 Architecture modulaire

Le principe de l'architecture modulaire est d'augmenter le niveau d'abstraction du design de façon à faciliter la conception de réseaux. À ce moment, le travail de l'ingénieur réseau devient plus un travail d'intégration que de conception. La division de l'architecture en couches hiérarchiques permet d'obtenir cette abstraction du design. L'architecture modulaire peut être décomposée en quatre catégories d'items qui sont reliés ensemble par des liens et/ou tubes de connectivité et qui sont distribués géographiquement selon trois catégories d'emplacement.

3.2.1 Item

L'architecture d'un réseau peut être décomposée en quatre catégories d'items :
- pièce ;
- composant ;
- module ;
- patron architectural.

La Figure 3.1 illustre les différentes catégories d'items de l'architecture modulaire.

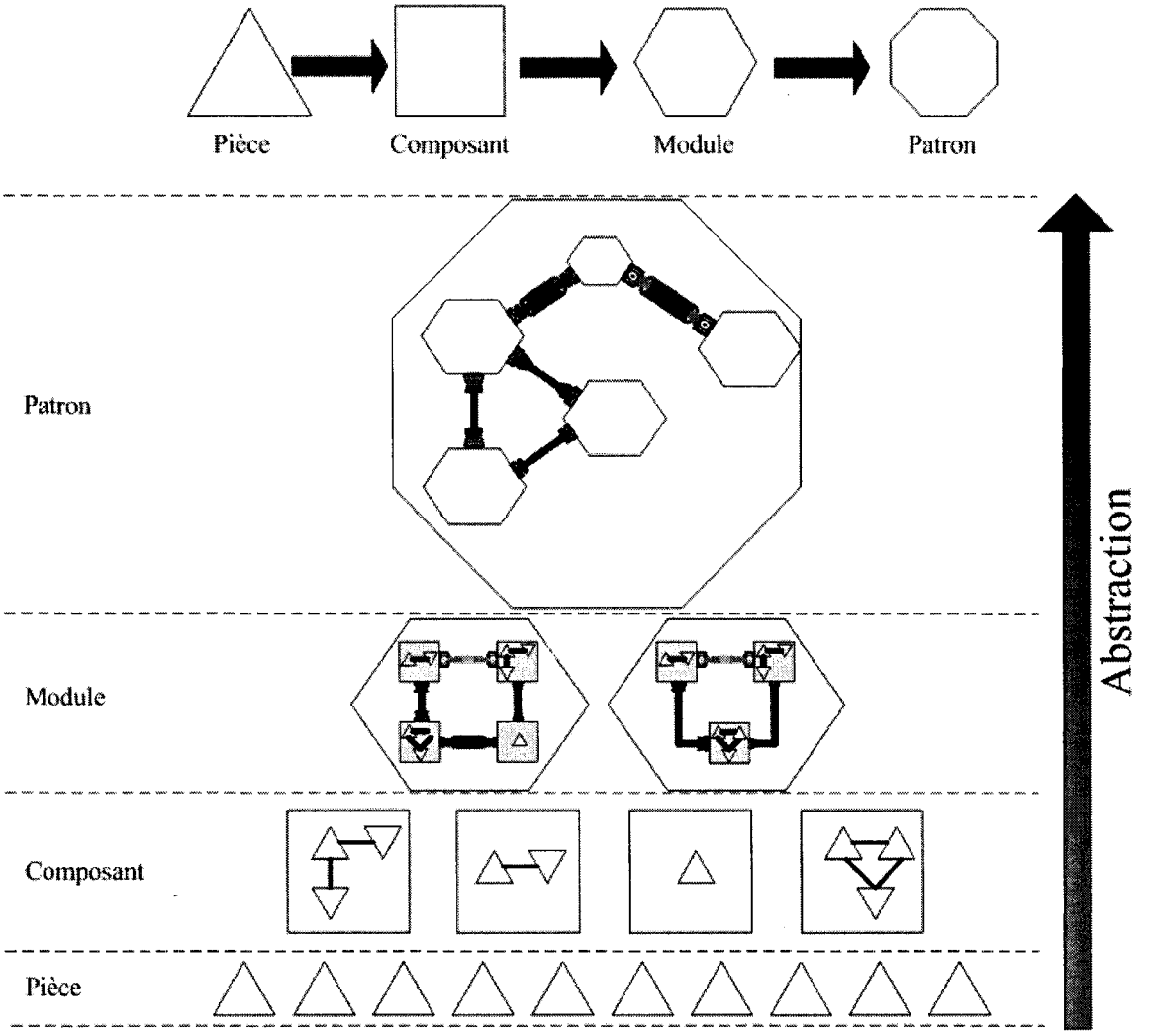

FIGURE 3.1 Niveaux d'abstraction

3.2.2 Pièce

Une pièce est un item simple qui est assemblé en composants. Elle possède des fonctionnalités limitées et n'est pas auto-suffisante en termes de fonctionnalités. Une pièce peut être un élément physique (équipement) ou applicatif (logiciel). Une pièce applicative doit être liée à une pièce physique dans un composant.

3.2.3 Composant

Un composant est un item plus complexe qu'une pièce. Il est formé par l'assemblage de pièces. Un composant fournit une ou plusieurs fonctionnalités bien définies. Un composant est auto-suffisant en termes de fonctionnalités. Les composants sont assemblés en modules selon des interfaces uniformes. Le caractère auto-suffisant du composant permet de remplacer un composant à l'intérieur d'un module par un composant différent qui fournit la ou les mêmes fonctionnalités et ce, sans affecter le comportement du module.

3.2.4 Module

Un module est un item plus complexe qu'un composant. Il est formé par l'assemblage de composants. Un module est une structure récurrente dans l'assemblage des composants. Un module fournit un ensemble de fonctionnalités qui définissent un ou plusieurs services. Un module est auto-suffisant en termes de services. Les modules sont assemblés selon des interfaces uniformes pour créer un patron architectural. Le caractère auto-suffisant du module permet de remplacer un module à l'intérieur d'un patron architectural par un autre module qui fournit le ou les mêmes services et ce sans affecter le comportement du patron.

3.2.5 Patron

Un patron architectural est un item plus complexe qu'un module. Il est formé par l'assemblage de modules. Un patron architectural est une structure récurrente dans l'assemblage des modules. Un patron architectural fournit une architecture complète. Un patron architectural est auto-suffisant à tous les niveaux.

3.2.6 Lien et tube de connectivité

Les différentes unités fonctionnelles qui composent l'architecture modulaire sont reliées ensemble par un lien et/ou un tube de connectivité. Une unité fonctionnelle est un composant ou un module. Un tube est une agrégation de liens de même type. Les liens et/ou tube de connectivité sont définis selon les interfaces des différentes unités fonctionnelles. Une interface est une frontière physique d'une unité fonctionnelle qui sert de point d'interconnexion avec une autre unité fonctionnelle de même

type d'interface. Il existe plusieurs types d'interface : RJ45, RJ21, port série, port USB, sans-fil, port optique, etc. La Figure 3.2 illustre différents tubes de connectivité. La Figure 3.3 illustre la relation entre un lien et un tube de connectivité.

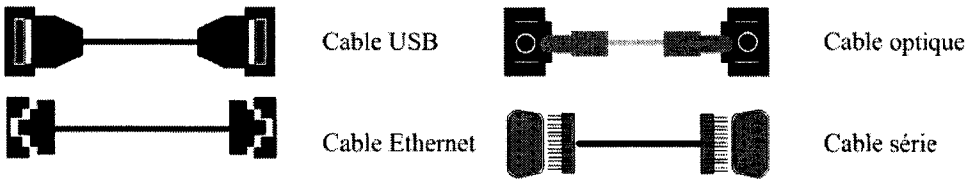

FIGURE 3.2 Liens de connectivité

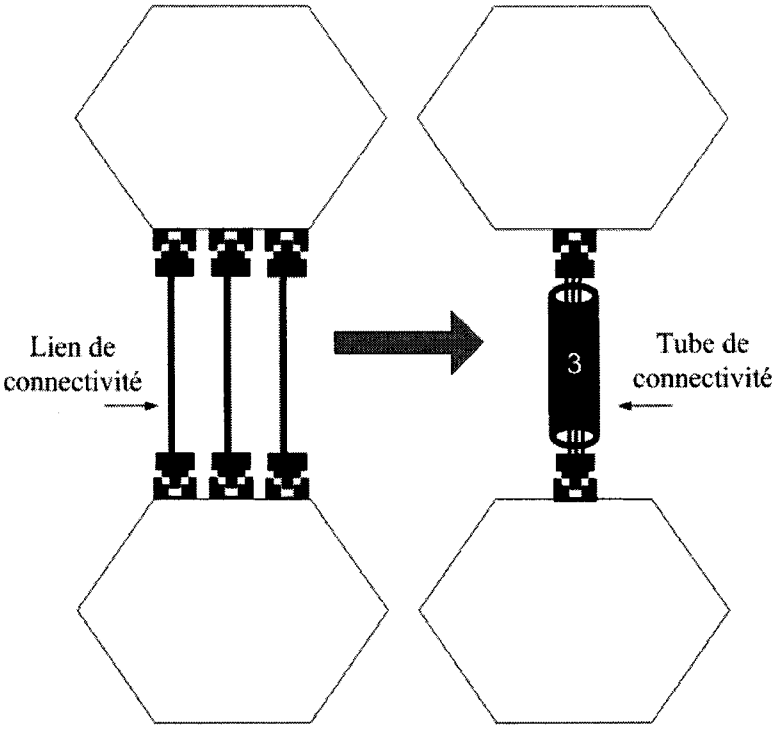

FIGURE 3.3 Lien et tube de connectivité

Lien de connectivité

Un lien de connectivité permet de relier directement deux unités fonctionnelles. C'est un lien physique (cable coaxial, paire torsadée, fibre optique, etc.) caractérisé par une interface.

Tube de connectivité

Un tube de connectivité est un regroupement de plusieurs liens de connectivité de même type entre deux unités fonctionnelles. Un tube possède les mêmes propriétés qu'un lien de connectivité. Un tel tube permet une représentation simplifiée sur un schéma d'architecture réseau.

3.2.7 Emplacements

L'architecture d'un réseau peut être décomposée en trois catégories d'emplacements géographiques.

- salle : plus petit emplacement situé à l'intérieur d'un bureau ;
- bureau : formé par une ou plusieurs salles ;
- campus : formé par plusieurs bureaux géographiquement regroupés.

Il existe deux types de bureaux : succursale et centrale. La Figure 3.4 illustre les trois types d'emplacements géographiques.

3.3 Types

Il y a trois types de modules :

- service ;
- connectivité ;
- inter-connectivité.

3.3.1 Service

Un module de service fournit un ou plusieurs services client. Les modules de services sont assemblés ensemble pour fournir une architecture cliente complète. La Figure 3.5 illustre le concept de module de service. Le module possède quatre composants, trois liens de connectivité et un tube de connectivité.

3.3.2 Connectivité

Un module de connectivité est un connecteur inter-modulaire. Il permet de relier deux modules de service qui sont géographiquement distants. IP-VPN (MPLS), HSM (*High Speed Metro*) et EI (*Ethernet Inter-networking*) sont des exemples de modules

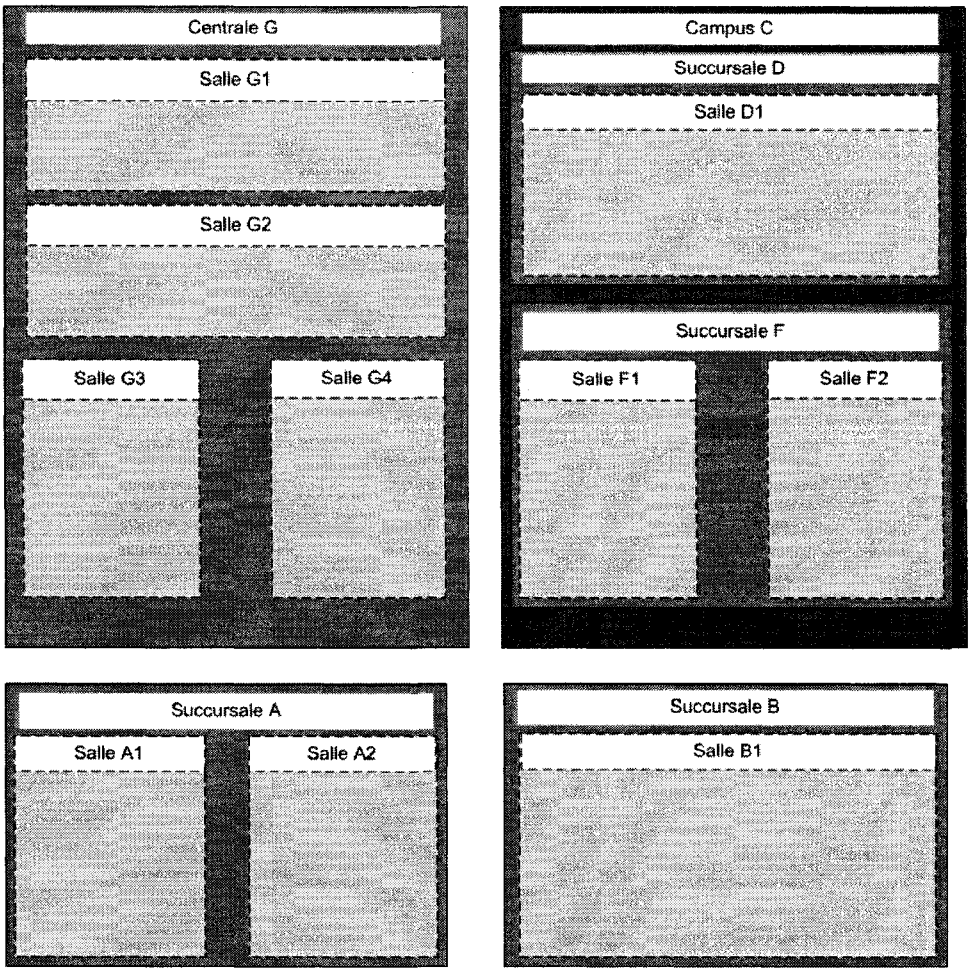

FIGURE 3.4 Emplacements géographiques

de connectivité. Un module intermédiaire d'inter-connectivité peut être nécessaire entre le module de service et le module de connectivité. La Figure 3.6 illustre le concept de module de connectivité.

3.3.3 Inter-connectivité

Un module d'inter-connectivité est un adapteur inter-modulaire. Il permet à deux modules à priori incompatibles d'être reliés. Cet adapteur sert d'intermédiaire entre un module de service et un module de connectivité. Par exemple, pour passer d'un lien Ethernet à un lien DSL, il faut préalablement passer par un modem. La Figure 3.7 illustre le concept de module d'inter-connectivité. La Figure 3.8 illustre les relations

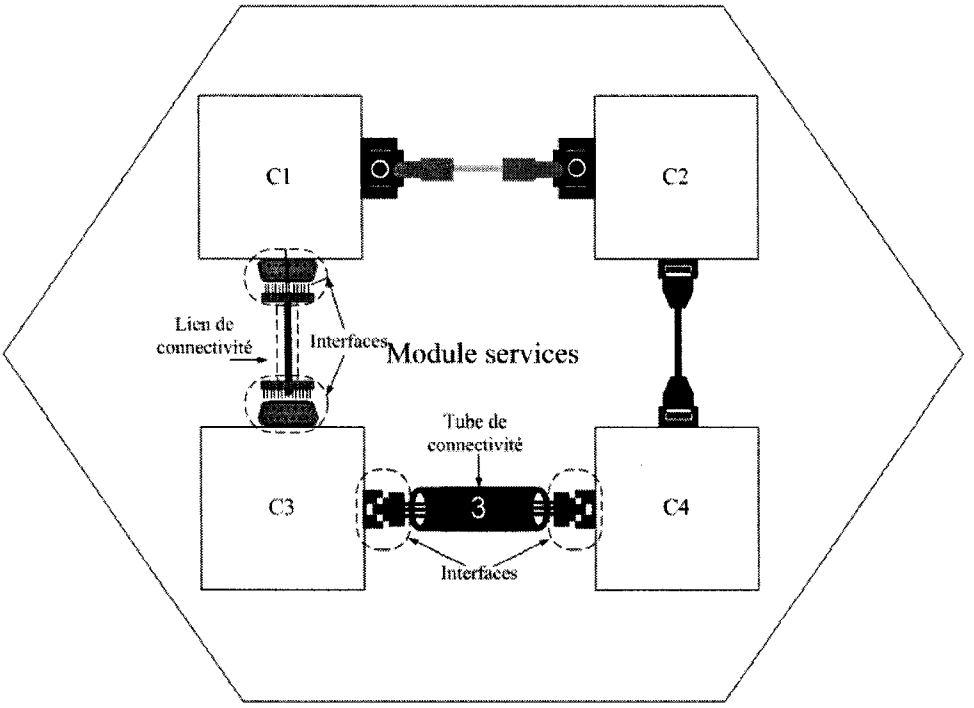

FIGURE 3.5 Module de service

entre les trois types de modules.

3.4 Assemblage

Le processus d'assemblage des modules comporte deux volets. Le premier concerne les incompatibilités entre les modules. Le deuxième concerne le processus de réutilisation des modules.

3.4.1 Incompatibilité

Deux modules sont considérés incompatibles s'ils nécessitent la présence d'un tiers-module pour être reliés. Il y a quatre types d'incompatibilité entre les modules :

- technologie ;
- protocole ;
- comportement ;
- performance.

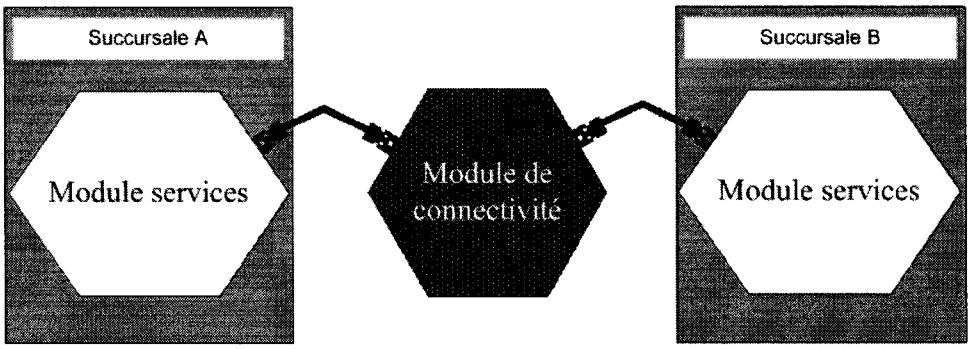

FIGURE 3.6 Module de connectivité

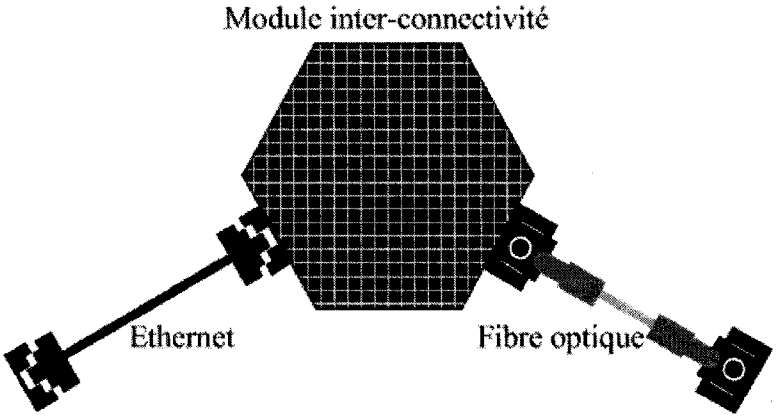

FIGURE 3.7 Module d'inter-connectivité

Une incompatibilité de type technologie peut survenir au niveau du type d'interface du lien et/ou du tube de connectivité entre deux modules. Un module d'inter-connectivité peut solutionner l'incompatibilité. Une incompatibilité de protocole peut survenir à chaque couche du modèle OSI. Au niveau applicatif, un logiciel peut faire la conversion pour solutionner l'incompatibilité. Au niveau physique, un module d'inter-connectivité peut solutionner l'incompatibilité. Une incompatibilité de type comportement peut survenir si les services des modules adjacents sont incompatibles entre eux. Un module d'inter-connectivité ne peut solutionner l'incompatibilité. Une incompatibilité de type performance peut survenir si les propriétés non-fonctionnelles entre les modules ne sont pas satisfaites. Les interfaces entre les modules peuvent être compatibles, mais les paramètres de performance tels que la largeur de bande, le délai, la gigue, etc. peuvent ne pas être satisfaits. Un module d'inter-connectivité peut solutionner l'incompatibilité.

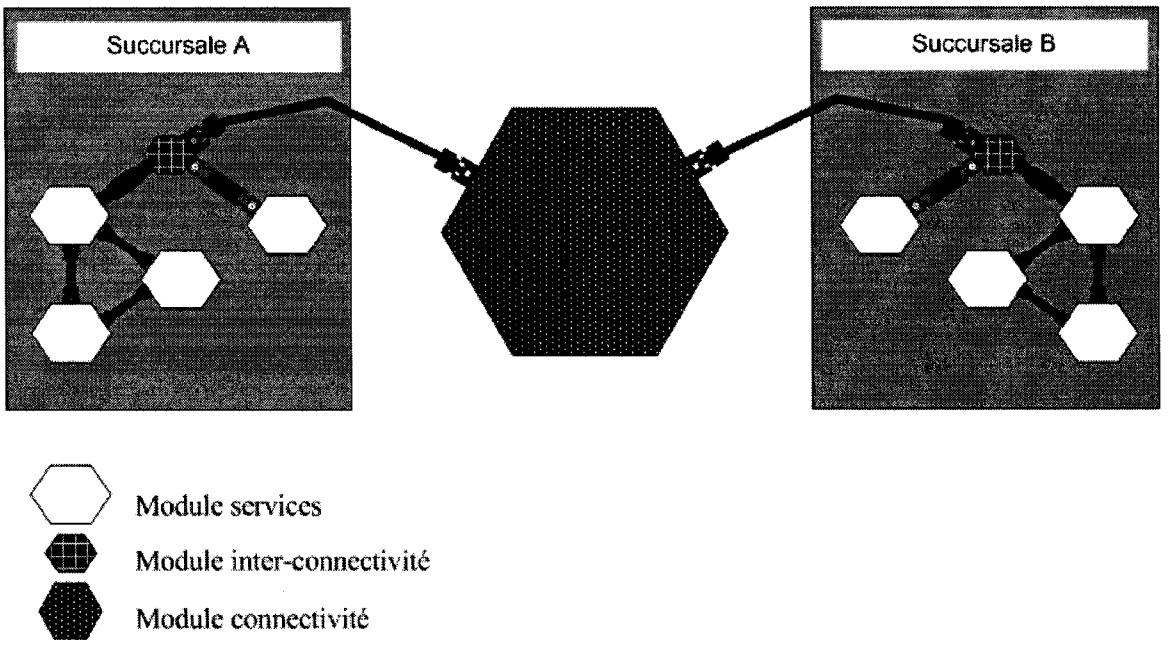

FIGURE 3.8 Relations modules

3.4.2 Réutilisation

Un des objectifs d'une architecture modulaire est de faciliter le processus de réutilisation des modules. Le processus de réutilisation comporte cinq volets :

- sélection ;
- modification-adaptation ;
- assemblage ;
- vérification-validation ;
- intégration.

La sélection consiste à choisir les modules nécessaires dans l'annuaire des modules existants. La modification-adaptation consiste à apporter les changements-corrections nécessaires aux modules pour l'assemblage. L'assemblage consiste à unir les différents modules sélectionnés selon la bonne configuration de façon à pouvoir fournir les services désirés. Le processus de détection d'incompatibilités comporte deux volets :

- vérification ;
- validation.

La vérification consiste à s'assurer que le comportement des modules entre eux est compatible. La validation consiste à établir si l'ensemble des contraintes techniques (comptabilité des équipements et protocoles) et des métriques de performances sont satisfaites. L'intégration consiste à insérer les différents modules à l'architecture réseau. À partir de la définition du problème (requis, spécifications, contraintes), les modules doivent être assemblés ensemble et vérifiés-validés pour générer une architecture fonctionnelle. La Figure 3.9 illustre deux types d'assemblages. Le premier type d'assem-

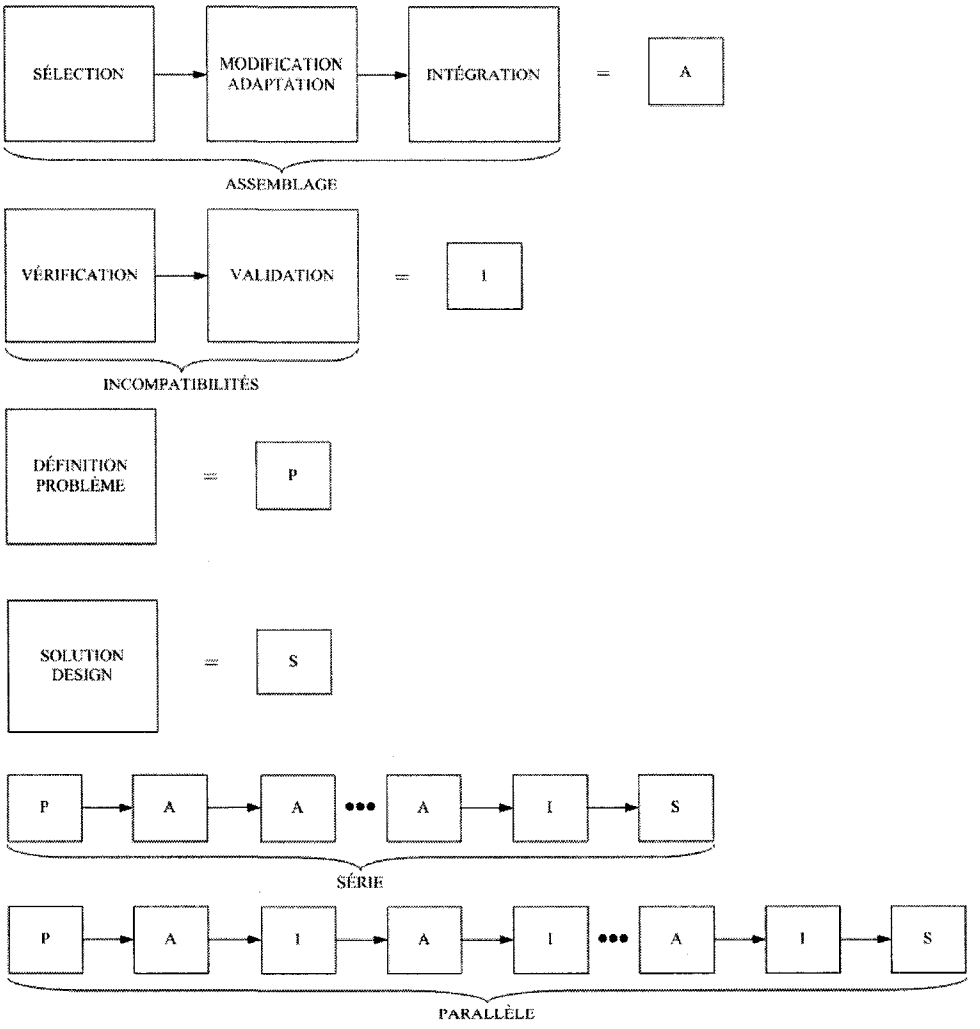

FIGURE 3.9 Assemblage des modules

blage (série) consiste à assembler tous les modules et à effectuer la vérification et la validation à la fin. Le design est vérifié et validé une seule fois lorsque l'assemblage est

terminé. Le deuxième type d'assemblage (parallèle) consiste à assembler un module à la fois et à effectuer la vérification et la validation avant d'intégrer un nouveau module. Le design est vérifié et validé au fur et à mesure. Une fois que le design est vérifié et validé, la dernière étape consiste à l'intégrer.

3.5 Métriques de design

Un module possède quatre métriques qui permettent de caractériser sa structure :
- couplage ;
- cohésion ;
- taille ;
- complexité.

3.5.1 Couplage

Le couplage d'un module est une mesure du niveau d'interconnexion (dépendance) avec les modules adjacents. Le couplage est caractérisé par le nombre et le type d'interfaces et le nombre de transactions potentielles avec un autre module. Une interface est une entrée-sortie du module qui permet d'effectuer des transactions avec un autre module. Il faut caractériser le trafic sortant de façon à le différencier. Il faut aussi caractériser les ports de sorties du module. On peut distinguer deux types de trafic et de ports : voix et données. Il faut aussi différencier le type de transaction entre deux modules. Une transaction se situe entre la complémentarité et la dépendance. La complémentarité est le type de transaction idéal entre deux modules. Une transaction de ce type indique que les modules sont indépendants ($w_{kid=0}$ ou $w_{kiv} = 0$). La dépendance est le pire type de transaction. Une transaction de ce type indique que les modules sont complètement dépendants ($w_{kid=1}$ ou $w_{kiv} = 1$). Une transaction peut se situer entre le deux, c'est-à-dire que certaines transactions sont de type complémentaire et d'autres sont de type dépendantes. Le tableau 3.1 illustre les paramètres pour le couplage. Les équations (3.1) à (3.5) définissent ces paramètres et le couplage. Le couplage est défini comme la somme de toutes les transactions potentielles de l'ensemble des pièces du module avec un module adjacent. Le nombre de transactions de voix (T_v) et de données (T_d) potentielles est calculé. Un module

possède un nombre défini d'interfaces de type voix (x_v) et données (x_d). Pour communiquer avec un module adjacent, un trafic de type données doit passer par une interface de type données du module. Il en va de même pour le trafic de type voix. Le nombre de transactions potentielles avec un module adjacent correspond donc à la somme des produits $T_d x_d$ et $Tv_x v$.

TABLEAU 3.1 Paramètres pour le couplage

N	=		nombre de composants
p_i	=		nombre de pièces du composant i
y_{kid}	=	1	si la pièce k du composant i doit communiquer (données) avec un module adjacent
	=	0	sinon
y_{kiv}	=	1	si la pièce k du composant i doit communiquer (voix) avec un module adjacent
	=	0	sinon
w_{kid}	=		poids complémentarité-dépendance entre la pièce k du composant i et un module adjacent (données)
w_{kiv}	=		poids complémentarité-dépendance entre la pièce k du composant i et un module adjacent (voix)
x_d	=		nombre de ports inter-modulaires (données)
x_v	=		nombre de ports inter-modulaires (voix)
T_d	=		nombre de transactions potentielles avec les modules adjacents (données)
T_v	=		nombre de transactions potentielles avec les modules adjacents (voix)

$$T_d = \sum_{i=1}^{N} \sum_{k=1}^{p_i} w_{kid} y_{kid} \tag{3.1}$$

$$T_v = \sum_{i=1}^{N} \sum_{k=1}^{p_i} w_{kiv} y_{kiv} \tag{3.2}$$

$$0 \leq w_{kid} \leq 1 \tag{3.3}$$

$$0 \leq w_{kiv} \leq 1 \tag{3.4}$$

$$Couplage = T_d x_d + T_v x_v \tag{3.5}$$

3.5.2 Cohésion

La cohésion d'un module est une mesure du niveau d'interconnexion entre les composants internes du module. La cohésion est caractérisée par le nombre de transactions potentielles entre les différents composants et par le type de cohésion. Il existe trois types principaux de cohésion qui sont en ordre décroissant de force la cohésion fonctionnelle, la cohésion séquentielle et la cohésion communicationnelle. La cohésion fonctionnelle indique qu'un module est constitué de composants qui fournissent un seul service. Toutes les fonctionnalités des différents composants du module communiquent ensemble pour fournir un seul service. La cohésion séquentielle indique qu'un module est constitué de composants qui fournissent plus d'un service telle que la sortie d'un service est l'entrée du service suivant. L'ordre d'exécution des fonctionnalités est séquentiel. La cohésion communicationnelle indique qu'un module est constitué de composants qui fournissent plus d'un service telle que l'ordre d'exécution n'est pas important. Dans ce cas, tous les composants utilisent les mêmes entrées. Comme pour le couplage, il faut différencier le type de trafic : données et voix. Il y aussi une caractérisation du type de transaction en fonction du type de cohésion. La cohésion idéale est de type fonctionnel. La pire cohésion est de type communicationnel. La cohésion peut se situer entre les deux. Le Tableau 3.2 illustre les paramètres de la cohésion. Les équations (3.6) à (3.10) définissent ces paramètres et la cohésion. La cohésion est définie comme la somme de toutes les transactions internes potentielles entre l'ensemble des composants du module.

$$T_{ijd} = \sum_{k=1}^{p_i} y_{kijd} \tag{3.6}$$

$$T_{ijv} = \sum_{k=1}^{p_i} y_{kijv} \tag{3.7}$$

$$0 \leq w_d \leq 1 \tag{3.8}$$

TABLEAU 3.2 Paramètres pour la cohésion

N	=		nombre de composants
p_i	=		nombre de pièces du composant i
y_{kijd}	=	t	si la pièce k du composant i doit communiquer (données) avec t pièces du composant j
y_{kijv}	=	t	si la pièce k du composant i doit communiquer (voix) avec t pièces du composant j
w_d	=		poids associé au type de cohésion (données)
w_v	=		poids associé au type de cohésion (voix)
c_{ijd}	=		nombre de chemins différents allant du composant i à j (données)
c_{ijv}	=		nombre de chemins différents allant du composant i à j (voix)
T_{ijd}	=		nombre de transactions potentielles entre les pièces du composant i et du composant j (données)
T_{ijv}	=		nombre de transactions potentielles entre les pièces du composant i et du composant j (voix)

$$0 \leq w_v \leq 1 \tag{3.9}$$

$$Cohesion = \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} (w_d c_{ijd} T_{ijd} + w_v c_{ijv} T_{ijv}) \tag{3.10}$$

3.5.3 Taille

Le nombre de composants et de fonctionnalités d'un module caractérisent la taille d'un module. Le Tableau 3.3 illustre les paramètres de la taille. L'équation (3.11) définit la taille.

TABLEAU 3.3 Paramètres pour la taille

N	=	nombre de composants
f_i	=	nombre de fonctionnalités du composant i
F	=	nombre de fonctionnalité du module

$$F = \sum_{i=1}^{N} f_i \tag{3.11}$$

3.5.4 Complexité

La complexité d'un module dépend des trois premières métriques : couplage, cohésion et taille. C'est une mesure relative. Elle permet de comparer des modules qui fournissent le ou les même(s) service(s). C'est une mesure pondérée qui se base sur les ratios entre le couplage et la cohésion et entre le nombre de composants et de fonctionnalités. Le Tableau 3.4 illustre les paramètres pour la complexité. L'équation (3.12) définit la complexité et les équations (3.13) à (3.16) définissent les contraintes de conception. Ces contraintes indiquent que la cohésion doit être supérieure au couplage et qu'il y a au minimum une fonctionnalité par composant. Plus la cohésion est forte par rapport au couplage et plus le ratio nombre de composants sur nombre de fonctionnalités est proche de 1, plus la complexité est faible.

TABLEAU 3.4 Paramètres pour la complexité

w_{cc}	=	poids associé au ratio $Couplage/Cohesion$
w_{nf}	=	poids associé au ratio N/F
$Couplage$	=	couplage du module
$Cohesion$	=	cohésion du module
N	=	nombre de composants du module
F	=	nombre de fonctionnalité du module

$$Complexite = \left(w_{cc} \frac{Couplage}{Cohesion} + w_{nf} \left(1 - \frac{N}{F} \right) \right)^2 \tag{3.12}$$

où

$$w_{cc} + w_{nf} = 1 \tag{3.13}$$

$$0 \leq \frac{Couplage}{Cohesion} \leq 1 \tag{3.14}$$

$$0 \leq \left(1 - \frac{N}{F}\right) \leq 1 \tag{3.15}$$

$$0 \leq Complexite \leq 1 \tag{3.16}$$

3.6 Coûts

Un des objectifs recherchés par cette architecture modulaire est la réduction des coûts de conception. Il faut donc évaluer les coûts de conception des différents modules. Il faut aussi évaluer les coûts inhérants à la réutilisation. À partir des coûts de conception et de réutilisation, il faut chercher à évaluer la rentabilité d'une telle méthode de conception et les gains qu'il est possible d'en retirer.

3.6.1 Coûts de réutilisation

Le nombre de modules dans une architecture modulaire influence les coûts. Le coût unitaire d'un module diminue lorsque le nombre de modules augmente. Le coût de réutilisation augmente lorsque le nombre de modules augmente. Le coût total de l'architecture modulaire dépend des coûts unitaires des modules et des coûts de réutilisation. Il existe une région de coût minimum où le couplage est minimisé et la cohésion maximisée. Une des caractéristiques désirées d'un module est l'indépendance. Un couplage faible et une cohésion forte correspondent à un module indépendant. Il doit y avoir un équilibre entre le couplage et la cohésion. Cet équilibre résulte en un nombre adéquat de modules qui permet de minimiser à la fois le coût unitaire des modules et le coût de réutilisation. Si un module est trop gros-couplage très faible et cohésion très forte-il va coûter très cher à concevoir. À l'opposé, si les modules sont très petits-couplage élevé et cohésion faible-l'assemblage de ces modules va coûter très cher étant donné que les modules sont fortement dépendants. Il est nécessaire de trouver un équilibre entre le couplage et la cohésion de façon à obtenir la meilleure relation entre les coûts de conception et les coûts de réutilisation. Si le concept d'architecture modulaire n'est pas bien appliqué, les coûts de conception ne seront pas réduits. Le coût moyen de réutilisation d'un module de type boîte grise correspond à l'équation (3.17). Le coût moyen de réutilisation d'un module de type boîte transparente correspond à l'équation (3.20).

$$C_{RBG} = R + p * (1 - q) * A + (1 - p) * DevMod \tag{3.17}$$

$$A = (Complexite)^2 * DevMod \tag{3.18}$$

$$A \leq DevMod \tag{3.19}$$

$$C_{RBT} = R + (1 - p) * DevMod \tag{3.20}$$

Le Tableau 3.5 illustre les paramètres pour les différents coûts de réutilisation. L'équation (3.19) est une hypothèse qui indique que le coût de modification/adaptation ne dépasse jamais le coût de développement d'un module non réutilisable.

TABLEAU 3.5 Paramètres pour les coûts de réutilisation

R	=	coût moyen de recherche dans la liste des modules existants
p	=	probabilité que le module recherché soit trouvé
q	=	probabilité que des modifications ne soient pas nécessaires
A	=	coût moyen de modification du module
$DevMod$	=	coût moyen de développement d'un module non réutilisable pour ce type de service

3.6.2 Coûts en fonction du nombre de modules

Pour un service donné, le nombre de modules influence le coût de conception des modules réutilisables et le coût de réutilisation. Le coût total dépend des valeurs du coût de conception des modules réutilisables et de leur coût de réutilisation en fonction du nombre de modules. Le Tableau 3.6 illustre les paramètres des différents coûts en fonction du nombre de modules. Les équations (3.21) à (3.31) définissent ces paramètres et le coût total en fonction du nombre de modules.

Les équations (3.21) et (3.22) sont basées sur le principe illustré à la Figure 1.1 de la section (1.1). Plus le nombre de modules augmente, plus le coût unitaire de

TABLEAU 3.6 Paramètres pour le coût total

N_C	=	nombre de composants
N_M	=	nombre de modules
$N_{M_{max}}$	=	nombre maximal de modules
$DevModR^*$	=	coût de développement unitaire d'un module réutilisable pour N_M modules
$DevMod^*$	=	coût de développement unitaire d'un module non réutilisable pour N_M modules
R^*	=	coût de recherche unitaire pour N_M modules
q^*	=	probabilité qu'un des N_M modules recherché soit trouvé
p^*	=	probabilité que des modifications ne soient pas nécessaires sur un des N_M modules
$C^*_{RN_M}$	=	coût de réutilisation unitaire pour N_M modules
A^*	=	coût unitaire de modification-adaptation pour N_M modules
$Complexite^*$	=	complexité unitaire d'un module pour N_M modules

conception d'un module diminue. Dans sa forme la plus simple, chaque composant renferme une seule fonctionnalité et chaque module un seul composant. Il suffit alors de décomposer l'architecture en services et en fonctionnalités.

$$DevModR^* = \frac{DevModR}{N_M^2} \tag{3.21}$$

$$DevMod^* = \frac{DevMod}{N_M^2} \tag{3.22}$$

Les équations (3.23) à (3.25) sont basées sur le principe que le service offert ne change pas. La décomposition en composants et en modules change, mais le service ne change pas.

$$R^* = R \tag{3.23}$$

$$q^* = q \tag{3.24}$$

$$p^* = p \tag{3.25}$$

L'équation (3.26) modifie l'équation (3.12) de la complexité. Comme le ratio $\frac{Couplage}{Cohesion} \leq 1$, alors $Couplage \leq Cohesion$. Lorsque le nombre de modules N_M augmente, le couplage augmente et la cohésion diminue. Il en résulte la condition suivante : $\lim_{N_M \to F} \frac{Couplage}{Cohesion} = 1$.

$$Complexite^* = \left(w_{cc} * \left(\frac{Couplage_0}{Cohesion_0} + \frac{N_M}{\alpha} \right) + w_{nf} * \left(1 - \frac{N_C}{F} \right) \right)^2 \tag{3.26}$$

$$\alpha = \frac{N_{M_{max}}}{1 - \frac{Couplage_0}{Cohesion_0}} \tag{3.27}$$

Les équations (3.28) et (3.29) sont équivalentes aux équations (3.17) et (3.18). Les valeurs de R, p, q, A et $DevMod$ sont remplacées par $R*$, $p*$, $q*$, $A*$ et $DevMod*$ respectivement pour tenir compte du nombre de modules.

$$C^*_{RN_M} = R^* + p^* * (1 - q^*) * A^* + (1 - p^*) * DevMod^* \tag{3.28}$$

$$A^* = (Complexite^*)^2 * DevMod^* \tag{3.29}$$

$$CoutTotal = C^*_{RN_M} * N_M + DevModR^* * N_M \tag{3.30}$$

L'équation (3.31) est la somme des coûts de conception et de réutilisation en fonction du nombre de modules.

$$CoutTotal = N * \left(R + p * (1 - q) * \left(\left(w_{cc} * \left(\frac{Couplage_0}{Cohesion_0} + \alpha \right) + w_{nf} * \left(1 - \frac{N_C}{F} \right) \right)^2 * \frac{DevMod}{N_M^2} \right) + (1 - p) * \frac{DevMod}{N_M^2} + \frac{DevModR}{N_M^2} \right) \tag{3.31}$$

3.6.3 Seuil de rentabilité

Le coût de réutilisation n'est pas le seul facteur à considérer. L'amortissement du coût de développement du module réutilisable en fonction de son degré de réutilisation doit aussi être considéré. Le coût de développement d'un module réutilisable (Dev-

ModR) est plus élevé qu'un module non réutilisable (DevMod) : équation (3.32). Pour que l'utilisation d'un module réutilisable soit rentable, il faut que la contrainte (3.33) soit respectée.

$$DevModR > DevMod \tag{3.32}$$

$$\beta * C_R + DevModR \leq \beta * DevMod \tag{3.33}$$

L'équation (3.33) peut être réorganisée différemment : équations (3.34) et (3.35). β et ϵ sont des valeurs entières. β représente le nombre de réutilisations du module. ϵ est la borne inférieure de β pour rentabiliser le coût de développement du module réutilisable.

$$\beta \geq \epsilon \tag{3.34}$$

$$\beta \geq \frac{DevModR}{DevMod - C_R} \tag{3.35}$$

3.6.4 Gains

Il est aussi possible d'évaluer les gains ($) générés par la réutilisation des modules en fonction du nombre de réutilisation β. L'équation (3.36) représente l'expression de ces gains.

$$Gains = \beta * DevMod - \beta * C_R - DevModR \tag{3.36}$$

3.7 Description d'un module

La description d'un module comporte huit aspects :

- Nom ;
- Service(s) ;
- Contexte ;
- Fonctions-Options ;
- Propriétés ;

- Structure-Design ;
- Coûts-Bénéfices ;
- Résumé.

La différence fondamentale entre la description d'un module et la description d'un patron de conception classique réside dans le fait que le module contient une implémentation physique et fonctionnelle tandis qu'un patron de conception contient une description abstraite de solution.

3.7.1 Nom

Le nom du module doit être explicite. Il doit résumer la nature du module (ex. : Modèle VoIP pour un bureau de 1 à 30 postes).

3.7.2 Service(s)

Il s'agit d'une description général du ou des service(s) offert(s) par le module (ex. : VoIP).

3.7.3 Contexte

Il s'agit d'une description des conditions sous-lesquelles le module est utilisable et applicable (ex. : bureau de 1 à 30 postes reliés à une centrale).

3.7.4 Fonctions-Options

Il s'agit d'une description détaillée des différentes fonctionnalités de base et des options du module (ex. : appel en attente, messagerie vocale, interurbains...).

3.7.5 Propriétés

Il s'agit d'une description :

- des différentes interfaces du module ;
- des métriques de design : propriétés fonctionnelles : couplage, cohésion, taille et complexité ;
- des métriques de performance en terme de largeur de bande, de délai, de gigue, de taux de perte de paquets...

3.7.6 Structure-Design

Il s'agit d'une représentation graphique du design du module de type boîte noire, transparente, grise ou blanche.

3.7.7 Coûts-Bénéfices

Il s'agit d'une description des coûts de réutilisation du module et des bénéfices relatifs à l'utilisation du module.

3.7.8 Résumé

Le résumé répond à la question suivante : que fait le module ? C'est une description claire et concise des services offerts, de son contexte d'utilisation, des fonctionnalités et options disponibles, des caractéristiques des différentes interfaces, de la structure interne du module et des coûts et bénéfices résultant.

3.8 Mécanisme de réutilisation

Après avoir défini l'architecture modulaire et les différentes métriques et coûts qui permettent de caractériser cette architecture, il faut concevoir un mécanisme de réutilisation qui permet de l'appliquer adéquatement. Il y a déjà des services standards pour certains types de services. Lorsque les besoins d'un client ne peuvent être comblés par ces services standards, il doit y avoir un mécanisme structuré qui permet la réutilisation de solutions existantes. Lorsqu'aucun service standard et aucune solution n'existent pour une problématique donnée, ce même mécanisme doit permettre de structurer la nouvelle solution de façon à permettre une réutilisation future. La Figure 3.10 illustre les grandes lignes de la méthode de conception proposée.

La première étape consiste à définir la problématique du client en fonction de ses besoins. Si une solution existante permet de satisfaire ces besoins, le mécanisme de réutilisation est enclenché. Si aucune solution existante ne permet de satisfaire ces besoins, il faut définir, caractériser et documenter une nouvelle solution dans l'optique d'une réutilisation future. Il est à noter que l'agencement des services standards et

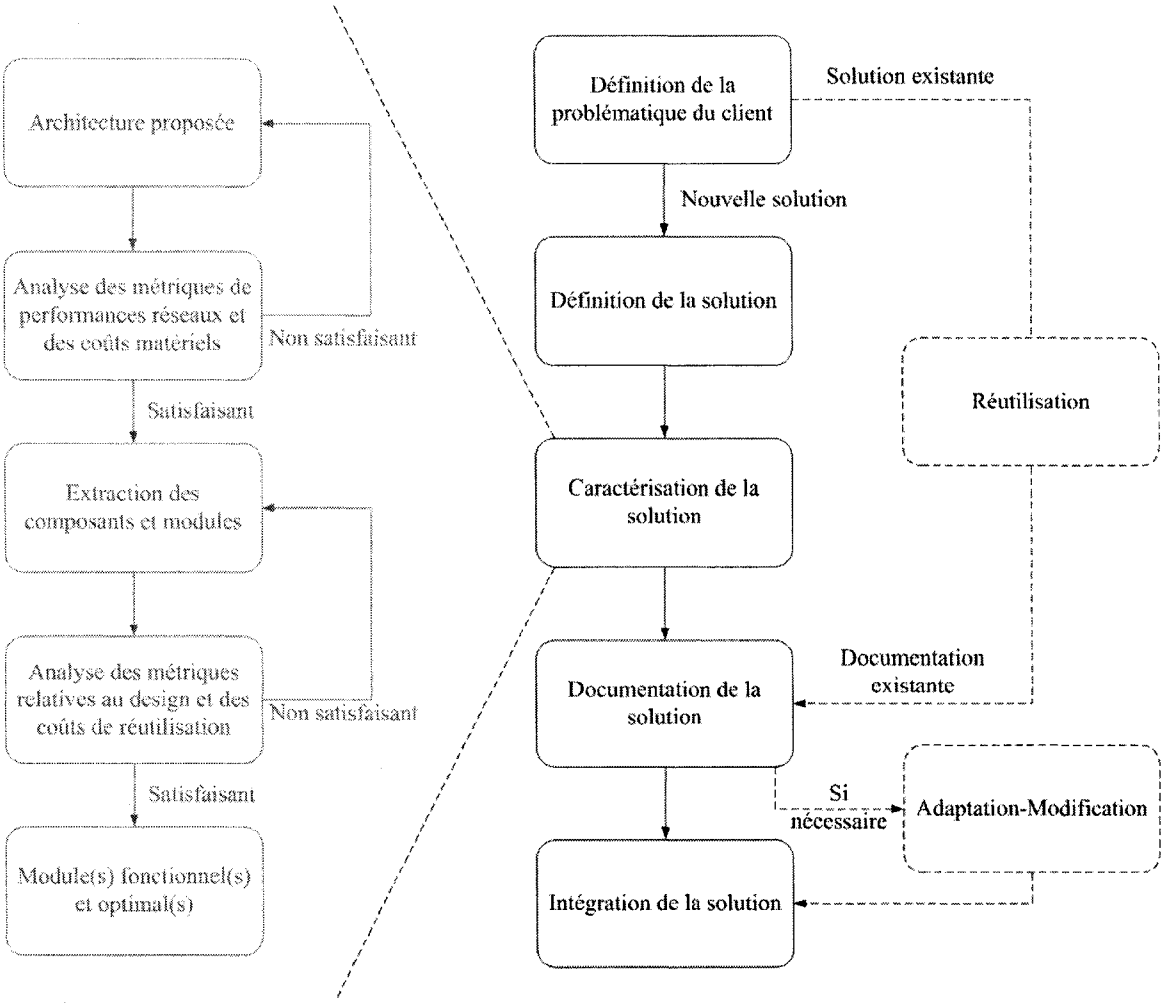

FIGURE 3.10 Mécanisme de réutilisation

des solutions existantes qui sont réutilisées doit se faire d'une façon homogène et transparente.

3.8.1 Définition de la solution

Après avoir défini la problématique du client, il faut déterminer les bases de la solution en fonction des besoins. Au niveau de la problématique, les besoins sont définis en termes fonctionnels. Il faut déterminer les besoins d'affaire du client. Au niveau de la définition de la solution, les besoins sont définis en termes techniques et en mesures de performances escomptées. Au niveau de la problématique, il est question de besoins qualitatifs, tandis qu'au niveau de la définition de la solution il

est question des besoins quantitatifs.

3.8.2 Caractérisation de la solution

Par la suite, il faut caractériser la solution. Ce volet de conception concerne l'implémentation et la représentation de la solution. La caractérisation de la solution procède en deux étapes :

- définition de l'architecture réseau ;
- définition des modules.

Définition de l'architecture réseau

Une fois que la problèmatique a été établie et que les grandes lignes de la solution ont été définies, il faut implémenter la solution. Une architecture est donc proposée. Les métriques de performances réseau et les coûts matériels sont analysés. Une fois que l'architecture proposée est satisfaisante en regard de critères fixés, il est possible de passer à l'extraction des modules.

Définition des modules

Une fois que l'architecture réseau est définie, il faut décomposer l'architecture en services et fonctionnalités de façon à pouvoir en extraire les pièces, composants et modules. Il faut procéder en deux temps : domaine fonctionnel et domaine physique. Au niveau du domaine fonctionnel, il y a trois étapes :

1. services ;
2. fonctionnalités ;
3. composants.

Tout d'abord, il faut isoler les services réseaux présents dans l'architecture. Ensuite, il faut identifier les fonctionnalités pour chaque service. Finalement, il faut grouper les fonctionnalités en composants selon les affinités entre les fonctionnalités. Au niveau du domaine physique, il faut associer les pièces d'équipements réseau aux différents composants fonctionnels. Il faut différencier le domaine physique du domaine fonctionnel de façon à pouvoir dissocier les fonctionnalités de l'implémentation. La Figure

3.11 illustre la décomposition de l'architecture en services, fonctionnalités, composants et pièces. Une fois que les services, les fonctionnalités et les composants ont

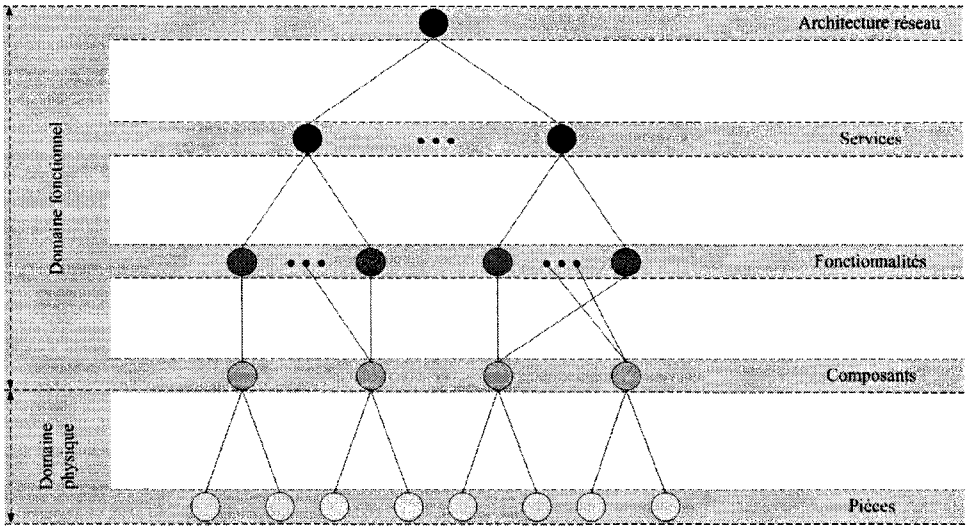

FIGURE 3.11 Extraction composants

été identifiés, il est possible d'extraire les modules. Pour ce faire, il faut grouper les composants en modules selon l'affinité entre les composants. L'idée maîtresse derrière cette décomposition est d'isoler chaque fonctionnalité dans un composant et chaque service dans un ou plusieurs modules. La Figure 3.12 illustre le groupement des composants en modules. Après avoir extrait les composants et les modules, il faut analyser

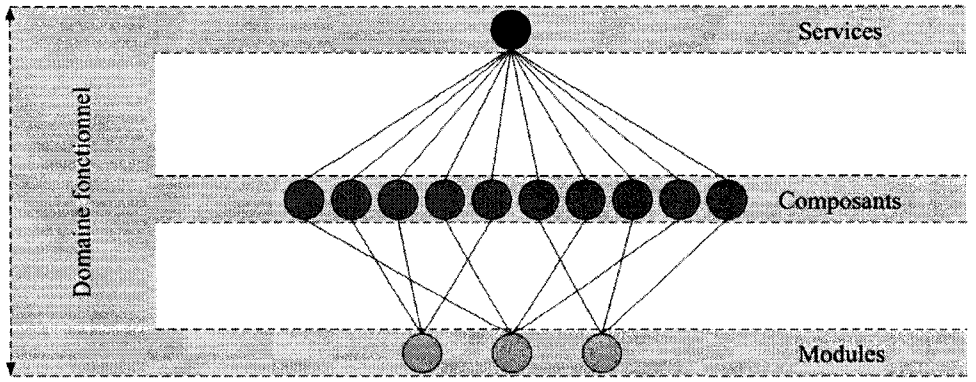

FIGURE 3.12 Extraction modules

les métriques relatives au design et les différents coûts de réutilisations. Il faut évaluer si cette décomposition est satisfaisante. Si la décomposition n'est pas satisfaisante, il

faut retourner à l'étape de l'extraction des composants et des modules et reformuler une nouvelle solution. Lorsque la décomposition est satisfaisante, les modules sont fonctionnels et optimaux en regard des critères fixés.

3.8.3 Documentation

Après avoir caractérisé la solution, il faut la documenter de façon à ce qu'elle puisse être réutilisée. Il faut décrire chacun des modules avec les huits aspects définis à la section (3.7). Lorsqu'une solution existante est réutilisée, la documentation existante sur la solution est utilisée de façon à déterminer si des adaptations et modifications sont nécessaires avant l'intégration.

3.8.4 Réutilisation

Lorsqu'une solution existante permet de combler les besoins du client, un mécanisme doit permettre la réutilisation de cette solution. Après avoir défini la problématique du client, il faut déterminer s'il existe une solution qui puisse satisfaire les besoins du client. Il faut donc chercher dans une liste exhaustive des solutions existantes en accédant à la documentation des différents modules. Une fois que les modules qui vont être réutilisés ont été identifiés, il faut déterminer les modifications et adaptations qui sont nécessaires avant l'intégration. La situation idéale serait qu'aucune modification et adaptation ne soit nécessaire. Toutefois, considérant la spécificité des besoins et la panoplie d'options disponibles, quelques ajustements risquent d'être nécessaires.

3.8.5 Intégration

La dernière étape consiste à intégrer la solution chez le client.

CHAPITRE 4

IMPLÉMENTATION ET RÉSULTATS

Dans ce chapitre, un sous-ensemble de modules est implémenté de façon restreinte. Par la suite, le modèle analytique des métriques relatives au design et aux coûts est analysé. L'impact des différents paramètres sur ces métriques est évalué. En dernier lieu, une expérimentation est menée sur un module de service. Les différentes métriques en fonction de certains paramètres sont analysées sur ce module. Un plan d'expérience est défini et des résultats expérimentaux sont obtenus.

4.1 Implémentation restreinte de l'architecture

Dans cette section, un sous-ensemble de modules est implémenté. Dans l'architecture modulaire proposée, il y a trois types de modules : service, connectivité et inter connectivité. Un exemple pour chaque type de module est donné. Chacune des implémentations est détaillée. Les modules sont ensuite assemblés dans une architecture décentralisée et centralisée.

4.1.1 Module de service

Pour illustrer un module de service de l'architecture modulaire et la méthode de conception proposée, un exemple d'architecture de VoIP pour une petite compagnie est utilisé. La figure 4.1 illustre une architecture de VoIP pour une petite compagnie construite à partir d'équipement Nortel. Le tableau 4.1 définit la fonctionnalité de chacun des équipements. Il est pris pour acquis que cette architecture est la meilleure pour ce type de service et que c'est cette architecture que l'entreprise voudrait fournir à tous ses clients requérant un tel service. L'hypothèse de base de l'architecture modulaire proposée est que l'optimisation des performances a préalablement été faite. Il faut donc extraire les composants et les modules de cette architecture. Il faut

considérer les caractéristiques énumérées précédemment :

- isoler la portion de l'architecture associée au service de VoIP ;
- isoler les fonctionnalités dans le but de dissocier les fonctionnalités de l'implémentation ;
- identifier les types d'interfaces possibles pour supporter l'interopérabilité ;
- grouper les fonctionnalités dans les composants ;
- grouper les composants en modules ;
- uniformiser la représentation du design.

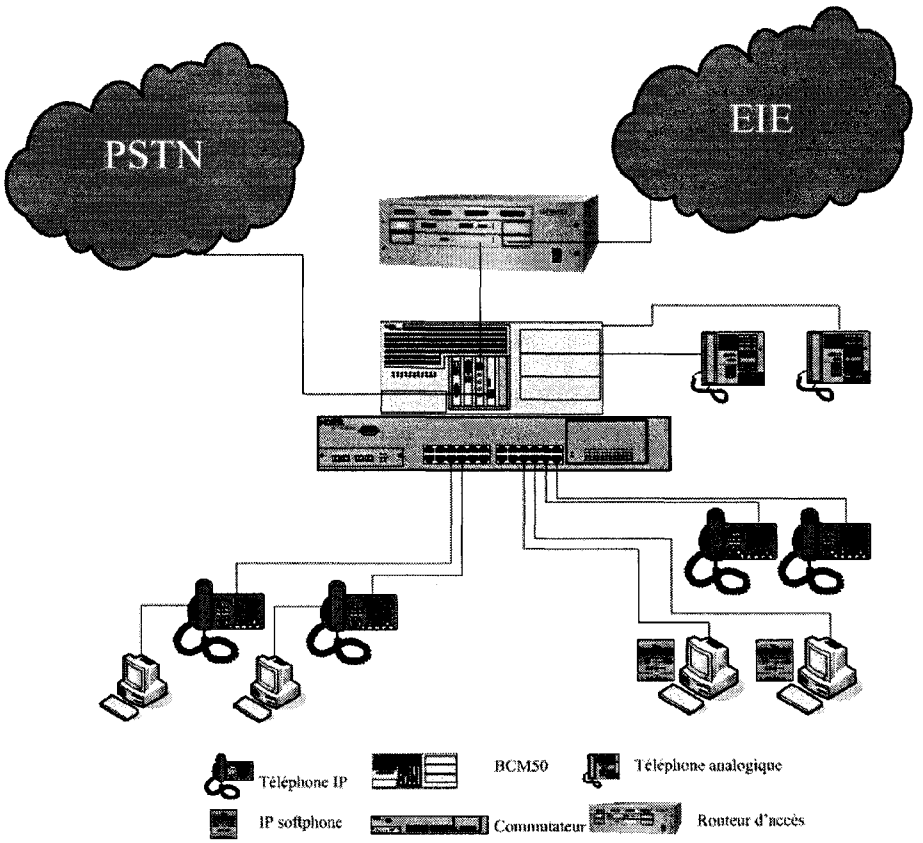

FIGURE 4.1 VoIP : Nortel

De l'architecture de VoIP présentée à la figure 4.1, un module composé de huit composants a été extrait. La taille de certains composants peut varier en fonction du nombre d'utilisations désirées. Le module et les composants sont présentés à la figure

TABLEAU 4.1 Fonctionnalités des équipements Nortel pour le service de VoIP

Équipement	Fonctionnalités
BCM50	Gestionnaire d'appels et boîte vocale
Commutateur	Permet l'établissement de plusieurs appels simultanément
Routeur d'accès	Permet la connexion à un réseau d'entreprise ou à Internet
Téléphone IP	Téléphone permettant la transmission de paquets de voix IP
IP softphone	Logiciel de téléphonie IP
Téléphone analogique	Téléphone traditionnel

4.2. Le module est représenté par un hexagone et les composants par des boîtes. Les fonctionnalités ont été isolées dans les composants et le service dans un seul module. Les différentes connexions entre les composants (boîtes) sont équivalentes à celles présentes sur le schéma original. Elles sont simplement représentées d'une manière uniforme par le biais de liens et tubes de connectivité. Il y a deux types différents de liens et/ou tubes de connectivité entre les composants du module. Le module a trois différents types d'interfaces (ports) avec les modules adjacents. Les différents types d'équipements (pièces) qui remplissent la même fonctionnalité ont été groupés ensemble. Les fonctionnalités sont clairement identifiées. En isolant les fonctionnalités, un composant pourrait être remplacé par un composant différent implémentation qui est compatible avec l'architecture existante sans altérer le comportement du module. En autant que les interfaces et les fonctionnalités soient les mêmes, le changement se ferait de façon transparente. Ceci rejoint la dissociation des fonctionnalités et de l'implémentation. Le module résultant est un module de type boîte grise. Certains items peuvent nécessiter une modification ou une adaptation.

4.1.2 Module de connectivité

Le module de service de VoIP présenté à la section précédente est auto-suffisant en termes de services. Il ne dépend pas d'un autre module pour fonctionner correctement. Il opère à l'intérieur d'une succursale. Ce module peut être reproduit dans plusieurs succursales. Chaque module est alors relié à un module de connectivité.

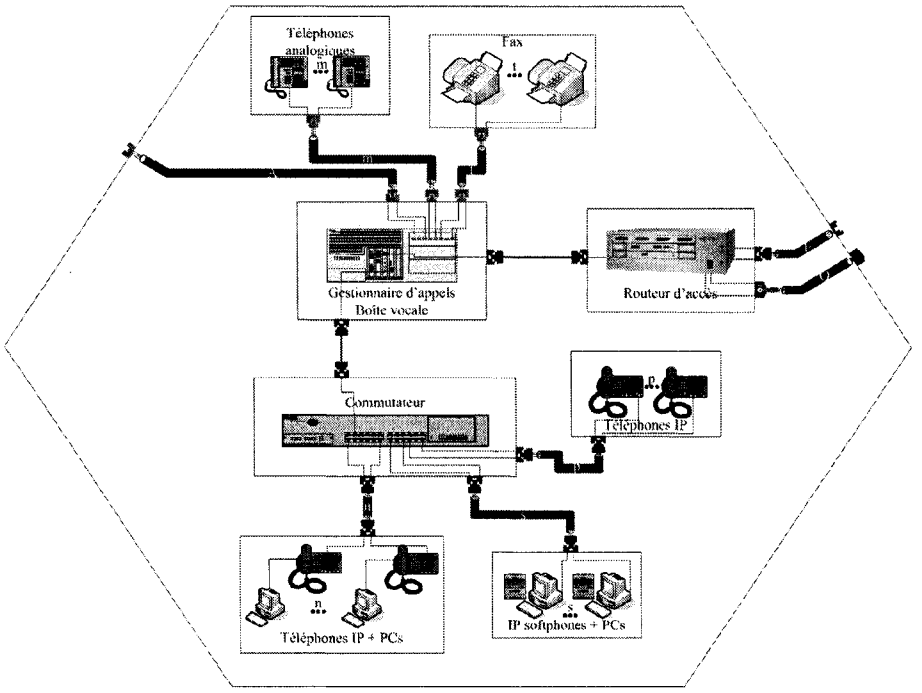

FIGURE 4.2 Module de VoIP : Nortel

Le module de connectivité permet aux succursales de communiquer entre elles sans avoir à passer par le PSTN. Pour illustrer un tel module de connectivité, un exemple d'architecture d'un réseau EIE (*Ethernet Internetworking*) est utilisé. La figure 4.3 illustre l'architecture simplifiée du service EIE construite avec de l'équipement Nortel. Les modules de connectivité sont en général des modules de type boîte transparente. C'est un service existant et implémenté. Les modules de services se rattachent au module de connectivité. C'est pour cette raison que l'architecture est simplifiée dans le module. Ce module pourrait aussi être représenté comme un module de type boîte noire. Il faut donc extraire les composants et les modules de cette architecture.

De l'architecture présentée à la figure 4.3, un module composé de huit composants a été extrait. Le module et les composants sont présentés à la figure 4.4. Les fonctionnalités ont été isolées dans chaque composant. Il y a deux types de liens de connectivité entre les composants du module. Le module a un seul type d'interface (port) avec les modules adjacents. Il y a quatre ports externes au module. Tel que mentionné, ce module est un module de type boîte transparente. Aucun item ne peut

être modifié dans ce module.

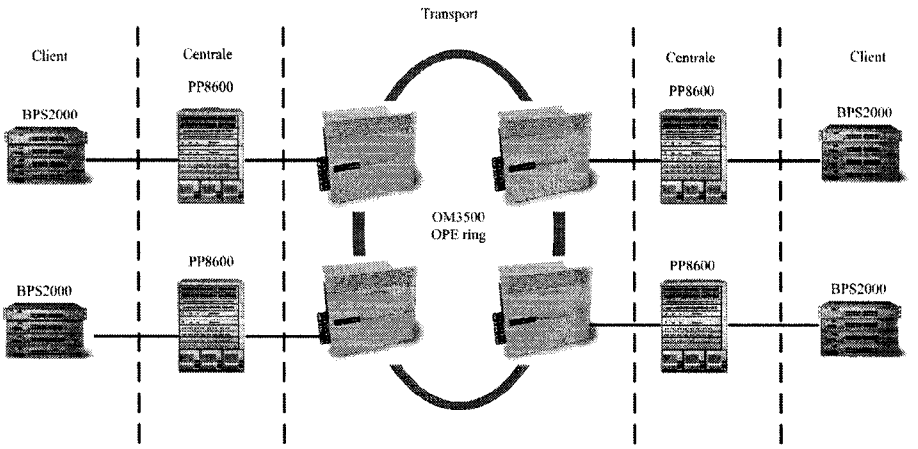

FIGURE 4.3 Connectivité : Nortel

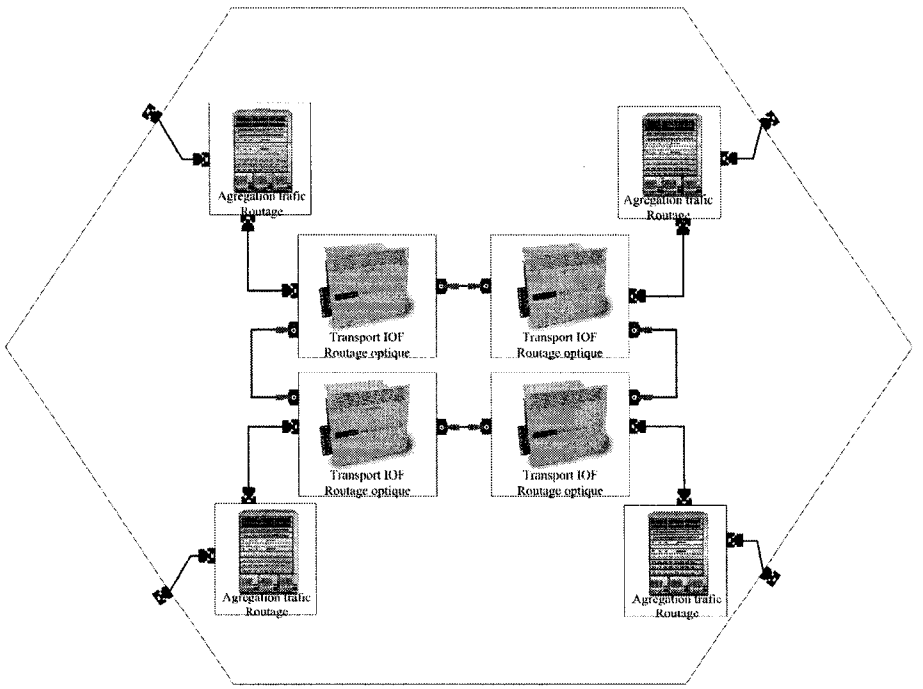

FIGURE 4.4 Module de connectivité : Nortel

4.1.3 Module d'inter connectivité

Les modules de services de VoIP et de connectivité sont indépendants. Ils ne sont pas conçus à priori pour être assemblés. Les interfaces sont compatibles, mais il manque une pièce d'équipement dans le module de service de VoIP pour faire le traitement adéquat de l'information. Il y a donc une incompatibilité de type comportement. Un module d'inter connectivité est donc nécessaire pour faire le pont entre le module de service et le module de connectivité. Un module d'inter connectivité est le type de module le plus simple. Généralement, c'est un module de type boîte transparente. La figure 4.5 illustre un exemple simple de module d'inter connectivité. C'est un module composé uniquement d'un composant. Les fonctionnalités sont toutes concentrées dans ce seul composant. Il possède un seul type d'interface. Si le module d'inter connectivité était nécessaire pour résoudre une incompatibilité de type technologie au niveau du lien, il y aurait eu deux types d'interfaces différents. Dans ce cas-ci, il s'agit d'une incompatibilité de type comportement.

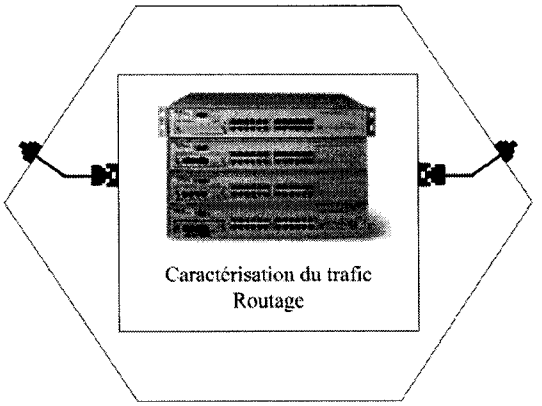

FIGURE 4.5 Module d'inter connectivité

4.1.4 Assemblage

Il y a trois types de modules : service, connectivité et inter connectivité. Un exemple de module de service de VoIP pour une petite entreprise a été donné. Des exemples de module de connectivité EIE et un module d'inter connectivité entre le module de VoIP et d'EIE ont aussi été donnés. Ces modules peuvent être assemblés

ensemble pour donner une architecture décentralisée de VoIP à plusieurs succursales. Chaque module de VoIP (un par succursale) est relié au PSTN et au module de connectivité EIE. La connexion entre le module de VoIP et le mdoule EIE se fait par le biais du module d'inter connectivité. La figure 4.6 illustre cet assemblage. L'architecture est composée de quatre succursales géographiquement distantes reliées entre elles par un module EIE. Comme mentionné, les modules de service de VoIP sont auto-suffisants en termes de services. Ils ne nécessitent par la présence d'un autre module pour fonctionner. Chaque succursale opère indépendamment des autres. Ceci veut dire que si une succursale tombe en panne, les autres peuvent continuer à communiquer entre elles sans altérations du comportement et des performances. L'ajout ou le retrait d'une succursale n'influencerait pas le comportement des autres succursales.

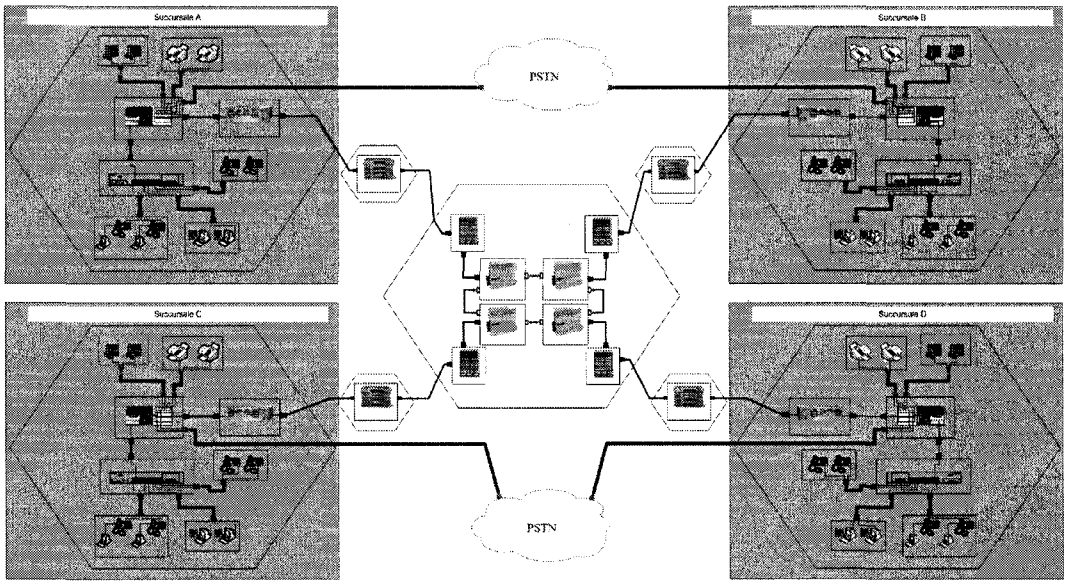

FIGURE 4.6 4 succursales de VoIP : Nortel

Les modules de service de VoIP ont été reproduits dans une architecture décentralisée. Ils pourraient aussi l'être dans une architecture centralisée. Chaque succursale serait connectée à une centrale qui aurait la responsabilité de faire le traitement des tâches relatives à la VoIP. Les modules de VoIP des succursales seraient reliés à la centrale par le même module de connectivité EIE que dans l'architecture décentralisée. Les

modules de VoIP de l'architecture centralisée et décentralisée seraient les mêmes. Les modules de VoIP de l'architecture centralisée posséderaient de l'équipement redondant qui leur permettrait d'opérer même si la centrale tombait en panne. La figure 4.7 illustre le module de VoIP de la centrale. Il est composé de deux composants et de deux types d'interface. La figure 4.8 illustre l'architecture de VoIP centralisée. Elle est composée de trois succursales et d'une centrale qui sont reliées par le module de connectivité EIE. Chaque module de service est connecté au module EIE par le biais d'un module d'inter connectivité.

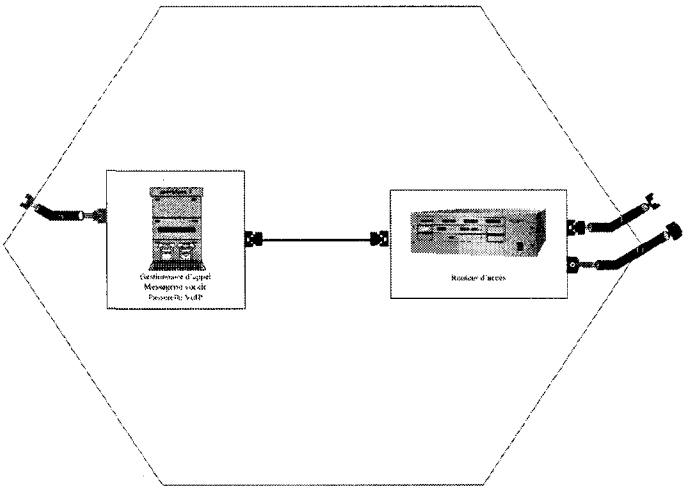

FIGURE 4.7 Module VoIP centrale : Nortel

4.2 Modèle analytique des métriques de design et de coûts

Dans cette section, l'impact des paramètres sur les différentes métriques de design et sur les différents coûts est analysé. L'objectif est d'identifier les paramètres qui peuvent maximiser l'optimalité, c'est-à-dire la rentabilité de l'architecture modulaire.

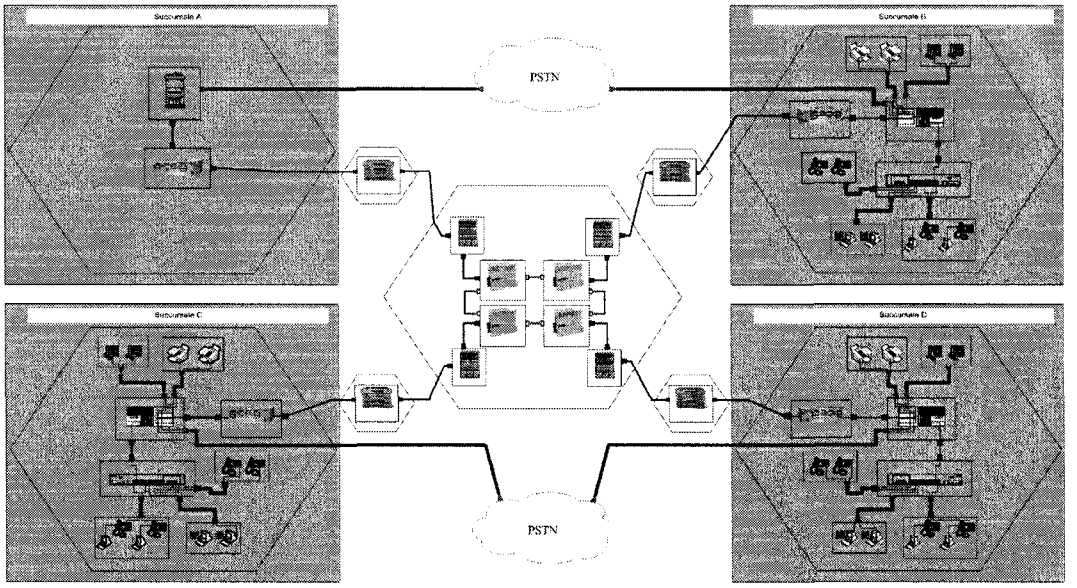

FIGURE 4.8 3 succursales et 1 centrale de VoIP : Nortel

4.2.1 Évaluation des paramètres des métriques de design

Il y a quatre métriques de design : couplage, cohésion, taille et complexité. Les trois premières définissent la complexité. Il faut donc évaluer l'impact des trois premiers paramètres sur la complexité.

Complexité

La mesure de complexité est une mesure relative. Elle permet de comparer l'arrangement des composants et des modules pour un même service. C'est une mesure pondérée qui se base sur les ratios entre le couplage et la cohésion et entre le nombre de composants et de fonctionnalités. Le couplage est une mesure du niveau d'inter connexion avec les modules adjacents. La cohésion est une mesure du niveau d'inter connexion entre les composants internes d'un module. Les équations (3.12) à (3.16) définissent la complexité et les différentes contraintes.

Le tableau 4.2 résume les valeurs des différents paramètres de la complexité utilisés pour évaluer l'influence de chacun d'entre eux sur la complexité. En fixant tous les paramètres et en faisant varier un paramètre à la fois, il est possible d'évaluer

l'impact de celui-ci. Les figures 4.9 à 4.14 illustrent la complexité en fonction des différents paramètres. Les figures 4.9 et 4.10 illustrent la complexité en fonction des deux ratios. La complexité diminue de façon quadratique lorsque le ratio $\frac{N}{F}$ augmente. Un ratio $\frac{N}{F}$ près de 0 indique que le nombre de fonctionnalités du module est beaucoup plus grand que le nombre de composants. Un ratio $\frac{N}{F}$ près de 1 indique que le nombre de fonctionnalités du module se rapproche du nombre de composants. C'est donc une association un à un entre les fonctionnalités et les composants qui entraîne une complexité minimale. À l'inverse, plus le ratio $\frac{Couplage}{Cohesion}$ augmente, plus la complexité augmente. Un ratio $\frac{Couplage}{Cohesion}$ près de 0 indique que la cohésion est beaucoup plus grande que le couplage. Un ratio $\frac{Couplage}{Cohesion}$ près de 1 indique que le couplage se rapproche de la cohésion. C'est donc un module indépendant, c'est-à-dire un module dont le couplage est nul par rapport à la cohésion qui entraîne une complexité minimale. Les figures 4.11 à 4.14 viennent illustrer l'impact individuel des paramètres des ratios $\frac{N}{F}$ et $\frac{Couplage}{Cohesion}$. Un couplage fort entraîne une augmentation de la complexité. Une cohésion forte entraîne une diminution de la complexité. Un nombre élevé de composants entraîne une diminution de la complexité. Finalement, un nombre élevé de fonctionnalités entraîne une augmentation de la complexité. Il faut donc chercher à minimiser le couplage et maximiser la cohésion. Il faut aussi chercher une association une à une entre les composants et les fonctionnalités.

TABLEAU 4.2 Paramètres pour l'évaluation des paramètres de la complexité

w_{cc}	=	0.5
w_{nf}	=	0.5
$Couplage$	=	10
$Cohesion$	=	30
N	=	8
F	=	14

4.2.2 Évaluation des paramètres des métriques de coût

Il y a différentes métriques de coûts qui caractérisent un module : coût de conception, coût de réutilisation, seuil de rentabilité et gains. Ces différentes métriques peuvent aussi être évaluées en fonction du nombre de modules qui composent le service.

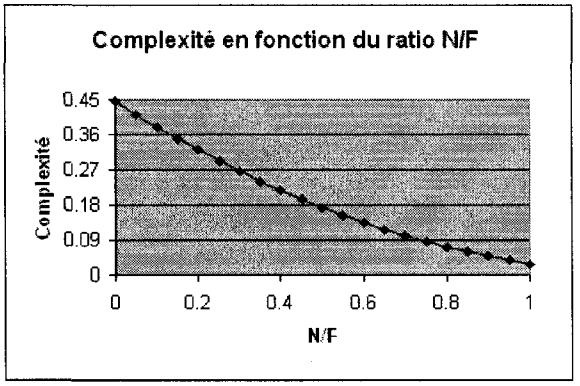

FIGURE 4.9 Complexité en fonction du ratio N/F

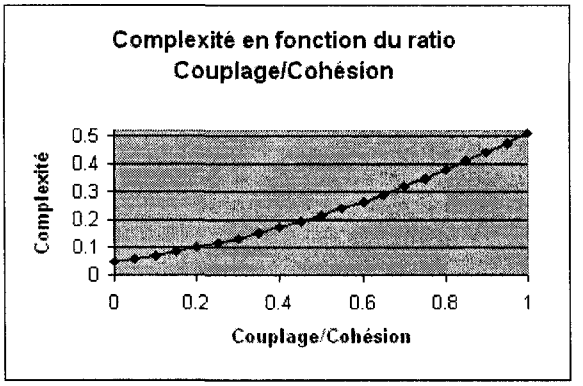

FIGURE 4.10 Complexité en fonction du ration Couplage/Cohésion

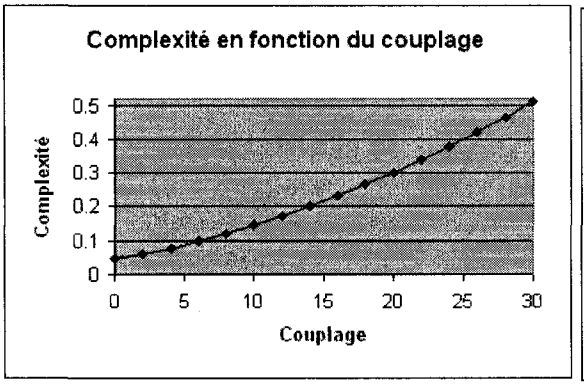

FIGURE 4.11 Complexité en fonction du couplage

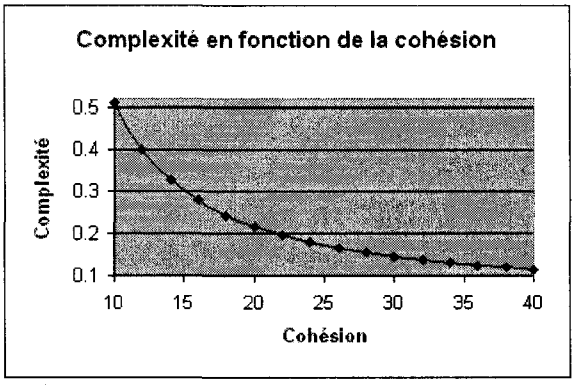

FIGURE 4.12 Complexité en fonction de la cohésion

Coûts de conception

L'évaluation du coût de conception n'a pas été abordé dans cette thèse. Le coût de conception est une variable employée pour établir certaines métriques de coûts.

Coûts de réutilisation

Un des aspects importants de l'architecture modulaire proposée est la réutilisabilité. Un module doit être réutilisable. Il est nécessaire d'évaluer les coûts liés à la réutilisation pour être en mesure d'en évaluer la rentabilité. Dans la section (3.6.1), des équations permettant d'évaluer les coûts de réutilisation d'un module de type boîte grise (3.17) et boîte transparente (3.20) ont été proposées.

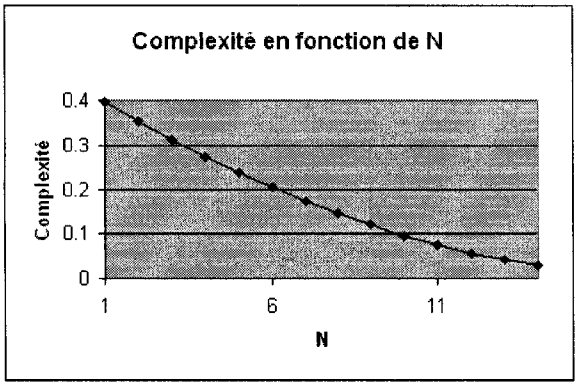

FIGURE 4.13 Complexité en fonction du nombre de composants

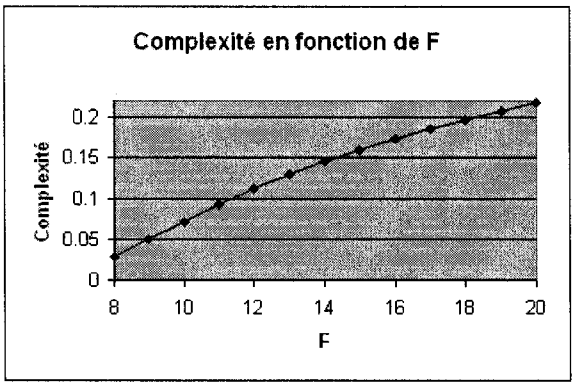

FIGURE 4.14 Complexité en fonction du nombre de fonctionnalités

Un module de type boîte grise est un module dont les entrées et sorties, les fonctionnalités et l'implémentation sont connues et dont il est possible de modifier certains items. Un module de type boîte transparente est un module dont les entrées et les sorties, les fonctionnalités et l'implémentation sont connues, mais dont aucun item ne peut être modifié. Le tableau 4.3 résume les valeurs des différents paramètres du coût de réutilisation des deux types de modules utilisés pour évaluer le coût de réutilisation. Pour un module de type boîte grise, il y a cinq paramètres : p, q, *complexite*, $DevMod$ et R. Pour un module de type boîte transparente, il y a trois paramètres : p, $DevMod$ et R. En fixant tous les paramètres et en faisant varier un paramètre à la fois, il est possible d'évaluer l'impact de celui-ci. Les figures 4.15 à 4.19 illustrent le coût de réutilisation d'un module de type boîte grise en fonction des différents paramètres. Plus la complexité, le coût de développement d'un module non réutilisable $DevMod$ et le coût de recherche R augmentent, plus le coût de réutilisation augmente. La complexité influence le coût de modification et d'adaptation A. La valeur de A augmente de façon quadratique lorsque la complexité augmente. Le coût de développement d'un module non réutilisable influence aussi le coût de modification et d'adaptation A. La valeur de A augmente de façon linéaire lorsque $DevMod$ augmente. $DevMod$ influence aussi de façon linéaire le coût associé à la probabilité que le module recherché ne soit pas trouvé et ce, même s'il existe. Finalement, le coût de recherche influence linéairement le coût de réutilisation. Une augmentation de R influence directement le coût de réutilisation. À l'inverse, plus les probabilités p et q sont grandes, plus

le coût de réutilisation diminue. La probabilité p correspond à la probabilité que le module recherché soit trouvé. La probabilité q correspond à la probabilité qu'il n'y ait pas de modifications ou d'adaptations à faire sur le module. Les probabilités p et q influencent linéairement le coût de réutilisation. Elles influencent le coût de modification et d'adaptation A. Il faut donc chercher à maximiser les probabilités p et q. Il faut aussi chercher à minimiser la complexité et le coût de recherche des différents modules. Les figures 4.20 à 4.22 illustrent le coût de réutilisation d'un module de type boîte transparente en fonction des différents paramètres. L'influence de R, p et $DevMod$ est la même que pour le module de type boîte transparente. Le principe de base de la réutilisation est que le coût de réutilisation doit être inférieur au coût de développement d'un module non réutilisable. Il faut chercher à minimiser le coût de réutilisation.

TABLEAU 4.3 Paramètres pour l'évaluation du coût de réutilisation

$DevMod$	=	30
R	=	10
p	=	0.75
q	=	0.75
$complexite$	=	0.5

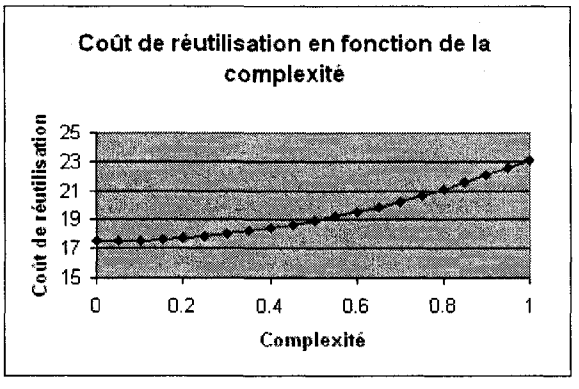

FIGURE 4.15 Coût de réutilisation en fonction de la complexité

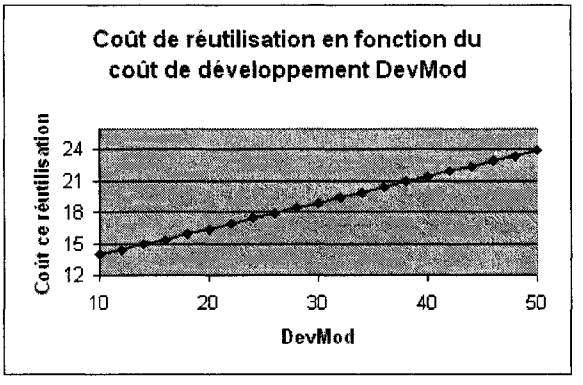

FIGURE 4.16 Coût de réutilisation en fonction du coût de DevMod

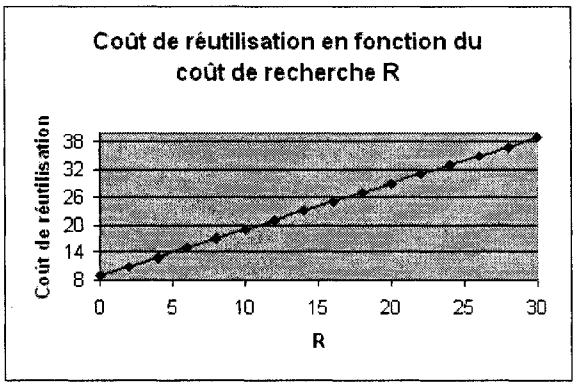

FIGURE 4.17 Coût de réutilisation en fonction du coût de recherche

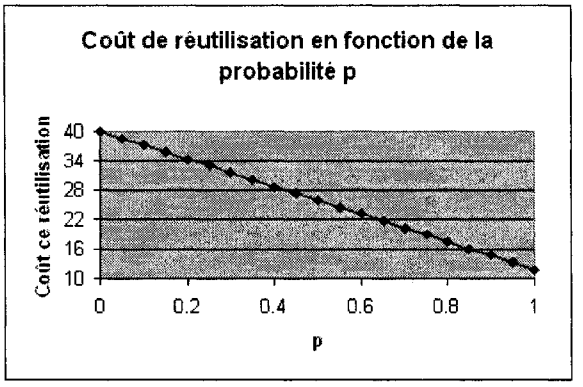

FIGURE 4.18 Coût de réutilisation en fonction de la probabilité p

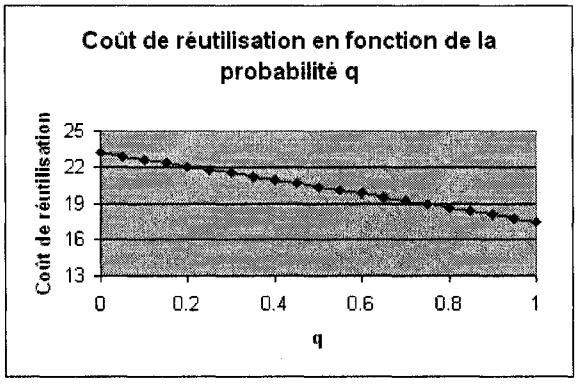

FIGURE 4.19 Coût de réutilisation en fonction de la probabilité q

Seuil de rentabilité

Le coût de réutilisation d'un module réutilisable est un facteur important à considérer. Le coût de développement du module réutilisable l'est tout autant. Il existe une relation entre le coût de développement et le coût de réutilisation d'un module réutilisable qui permet d'en évaluer la rentabilité. L'équation (3.33) illustre la contrainte de rentabilité. Les équations (3.34) et (3.35) illustrent le seuil de rentabilité. β représente le nombre de réutilisations du module réutilisable. ϵ représente la borne inférieure de β pour rentabiliser le coût de développement du module réutilisable. Le coût de développement du module réutilisable est amorti à chaque réutilisation. Il faut donc que le nombre de réutilisations du module soit suffisant.

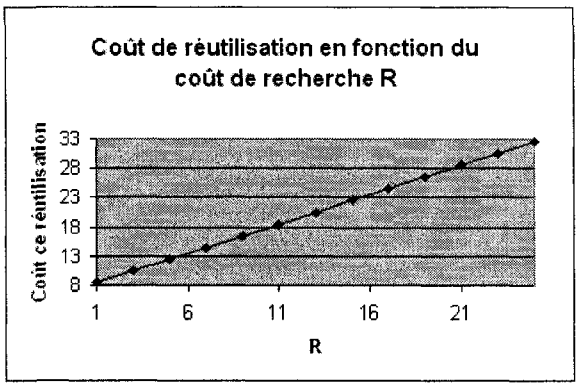

FIGURE 4.20 Coût de réutilisation en fonction du coût de recherche

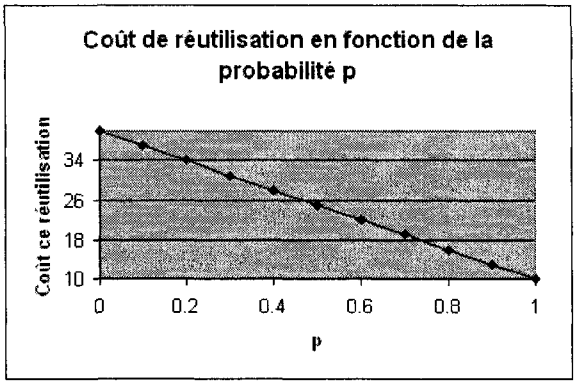

FIGURE 4.21 Coût de réutilisation en fonction de la probabilité p

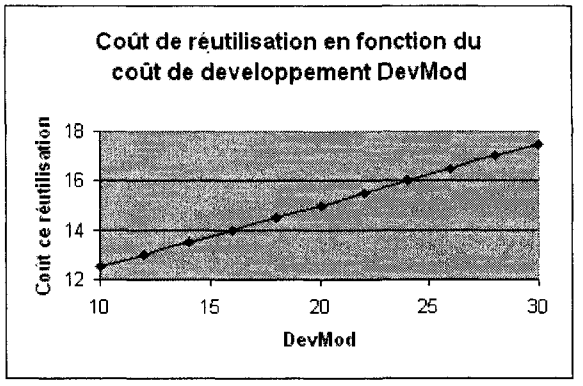

FIGURE 4.22 Coût de réutilisation en fonction du coût de développement DevMod

Pour évaluer l'impact des différents paramètres sur le seuil de rentabilité β, il faut fixer ces différents paramètres et en faire varier un à la fois. Le tableau 4.4 résume les valeurs des différents paramètres. Les figures 4.23 à 4.25 illustrent le seuil de rentabilité ϵ en fonction des différents paramètres. Plus le coût de réutilisation se rapproche du coût de développement du module non réutilisable, plus ϵ augmente de façon exponentielle. Il est à noter qu'un coût de réutilisation supérieur au coût de développement d'un module non réutilisable rend la réutilisation non rentable. À l'inverse, lorsque le coût de réutilisation est fixe et que le coût de développement d'un module non réutilisable augmente, ϵ diminue de façon exponentielle. Le coût de développement du module réutilisable est amorti tranquillement. Finalement, l'augmentation du coût de développement du module réutilisable fait augmenter ϵ de façon linéaire. Plus ce

coût est élevé, plus de nombre de réutilisations nécessaires pour l'amortir est élevé.

TABLEAU 4.4 Paramètres pour l'évaluation du seuil de rentabilité

$DevMod$	=	10
$DevModR$	=	20

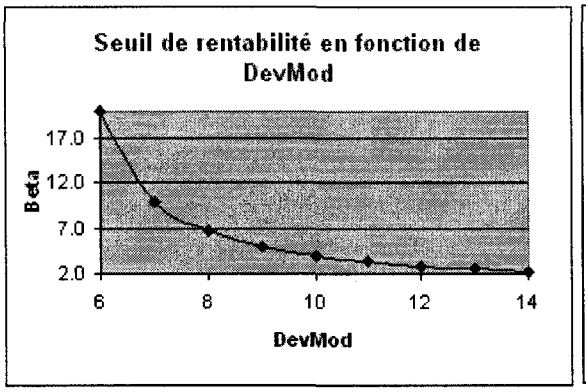

FIGURE 4.23 Seuil de rentabilité en fonction du coût de développement des modules DevMod

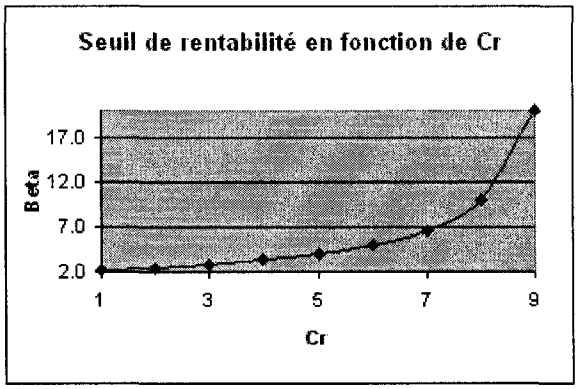

FIGURE 4.24 Seuil de rentabilité en fonction du coût de réutilisation des modules C_R

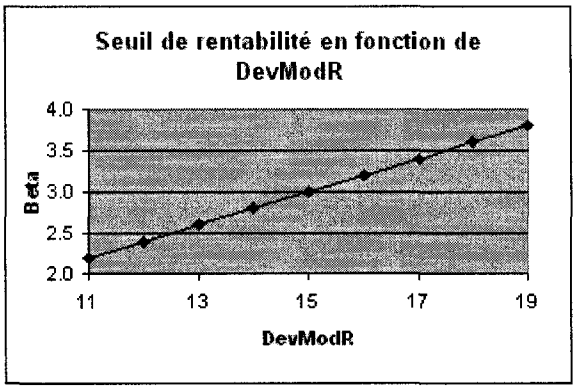

FIGURE 4.25 Seuil de rentabilité en fonction du coût de développement DevModR

Gains

Les coûts de réutilisations et de développement des modules réutilisables influencent le seuil de rentabilité. Au-delà du nombre de réutilisations nécessaires pour

rentabiliser le coût de développement du module réutilisable, il y a les gains générés par la réutilisation. Il peut être intéressant de chiffrer ces gains en fonction du nombre de réutilisations. L'équation (3.36) illustre l'équation des gains. β représente le nombre de réutilisations. Un gain positif correspond à un profit et un gain négatif correspond à une perte. Le gain s'exprime en dollars.

Pour évaluer l'impact des différents paramètres sur les gains, il faut fixer les différents paramètres et en faire varier un à la fois. Le tableau 4.5 résume les valeurs des différents paramètres. Les figures 4.26 à 4.29 illustrent les gains en fonction des différents paramètres. Plus le nombre de réutilisations β et le coût de développement d'un module non réutilisable $DevMod$ augmentent, plus les gains augmentent. L'augmentation des gains est linéaire. À l'inverse, plus le coût de réutilisation C_R et le coût de développement d'un module réutilisable $DevModR$ augmentent, plus les gains diminuent jusqu'à devenir des pertes.

TABLEAU 4.5 Paramètres pour l'évaluation du seuil de rentabilité

$DevMod$	=	25
$DevModR$	=	100
β	=	10
C_R	=	10

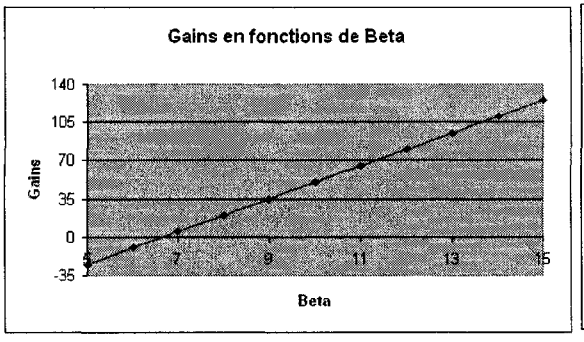

FIGURE 4.26 Gains en fonction du nombre de réutilisations

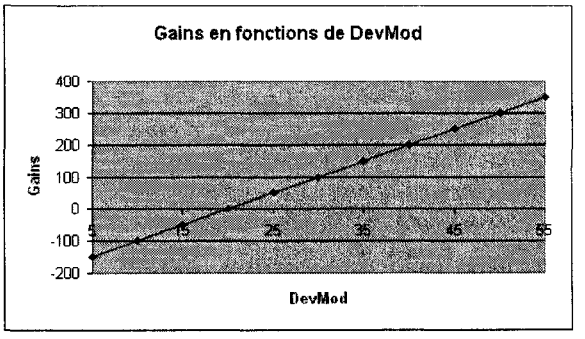

FIGURE 4.27 Gains en fonction du coût de développement DevMod

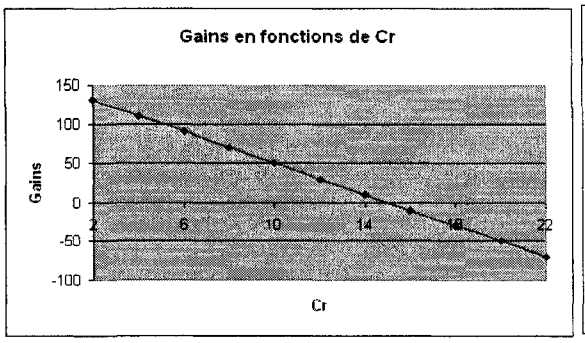

FIGURE 4.28 Gains en fonction du coût de réutilisation

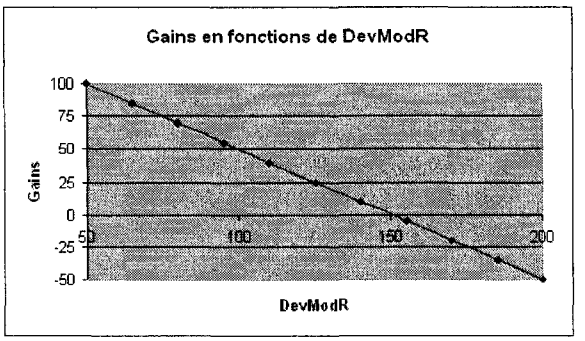

FIGURE 4.29 Gains en fonction du coût de développement DevModR

Coûts en fonction du nombre de modules

Le nombre de modules pour un service donné influence le coût de conception et de réutilisation. Le coût total dépend à la fois des coûts de conception et de réutilisation en fonction du nombre de modules. Les équations (3.23) à (3.31) illustrent les différents coûts et paramètres du coût total en fonction du nombre de modules

La figure 1.1 de la section (1.1) illustre la relation recherchée dans une architecture modulaire entre les coûts de conception et de réutilisation en fonction du nombre de modules. Plus le nombre de modules augmente, plus les coûts de conception des modules diminuent. À l'inverse, plus le nombre de modules augmente et plus le coût de réutilisation des modules augmente. Le coût total correspond à la somme des coûts de conception et de réutilisation. Il existe une zone de coût minimum où il y a un équilibre entre les coûts de conception et de réutilisation. C'est le fondement du concept de l'architecture modulaire. Il faut comparer le modèle proposé à la section (3.6) à celui de la figure 1.1. Pour ce faire, il faut fixer les paramètres du coût total et faire varier le nombre de modules. Le tableau 4.6 résume les valeurs des différents paramètres. La figure 4.30 illustre le coût unitaire de conception des modules réutilisables en fonction du nombre de modules. Lorsque le nombre de modules augmente, le coût unitaire de conception des modules réutilisables diminue. Le coût total de conception des modules réutilisables diminue lui aussi lorsque le nombre de modules augmente. La figure 4.31 illustre le coût unitaire de réutilisation d'un module et le coût total de réutilisation en fonction du nombre de modules. Lorsque le nombre de modules augmente, le coût unitaire de réutilisation diminue. Toutefois, lorsque le nombre de modules augmente,

le coût total de réutilisation augmente. La diminution du coût unitaire n'est pas assez prononcée pour engendrer une baisse du coût total de réutilisation. Lorsque les coûts totaux de conception et de réutilisation sont combinés, le coût total est obtenu. La figure 4.32 illustre le coût total en fonction du nombre de modules. Il existe une zone de coût minimum entre $N = 3$ et $N = 5$. Il faut toutefois considérer l'amortissement du coût des modules réutilisables. La figure 4.33 illustre le seuil de rentabilité en fonction du nombre de modules. Pour $N = 1$ à $N = 4$, c'est rentable. Pour $N \geq 5$, ce n'est pas rentable. Un seuil négatif indique que le coût de réutilisation est supérieur au coût de développement d'un module non réutilisable. Considérant le seuil de rentabilité, $N = 4$ devrait correspondre à l'optimalité. Toutefois, au-delà du seuil de rentabilité, il faut considérer les gains en fonction du nombre de réutilisations β envisagées. La figure 4.34 illustre les gains en fonction du nombre de réutilisations pour les différentes valeurs du nombre de modules qui sont rentables ($N = 1$ à $N = 4$). À court et moyen termes, c'est pour $N = 2$ que c'est le plus rentable. À long terme ($\beta = 39$), $N = 1$ finirait par être plus rentable. Donc, la figure 4.32 n'est pas suffisante pour déterminer l'optimalité. Il faut considérer le seuil de rentabilité et les gains pour y parvenir. Il faut considérer le nombre de réutilisations envisagées pour déterminer l'optimalité qui s'y rattache.

TABLEAU 4.6 Paramètres pour l'évaluation du coût total en fonction de N

$DevModR$	=	100
$DevMod$	=	25
R	=	5
p	=	0.75
q	=	0.75
w_{cc}	=	0.5
w_{nf}	=	0.5
$\frac{Couplage}{Cohesion}$	=	0.5
N	=	5
F	=	10

Influence des paramètres pour les métriques de coûts en fonction du nombre de modules

Quelle est l'influence des différents paramètres sur le coût total, le seuil de rentabilité et les gains en fonction du nombre de modules ? L'évaluation de l'impact

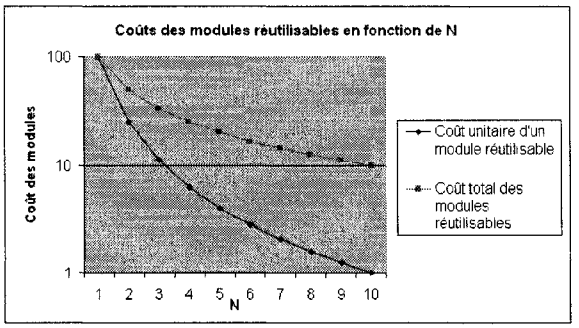

FIGURE 4.30 Coût des modules réutilisables en fonction du nombre de modules

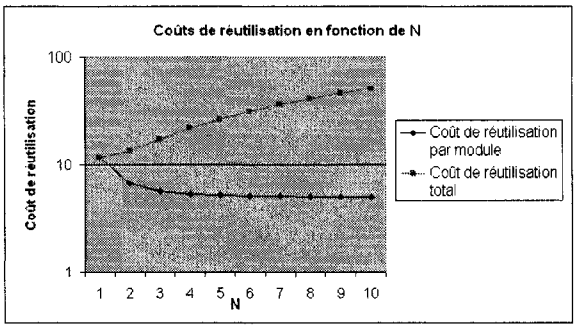

FIGURE 4.31 Coût de réutilisation des modules en fonction du nombre de modules

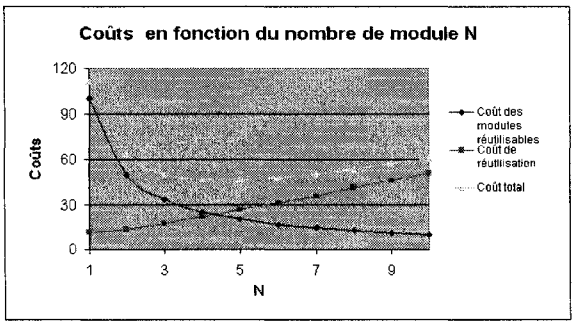

FIGURE 4.32 Coûts en fonction du nombre de modules

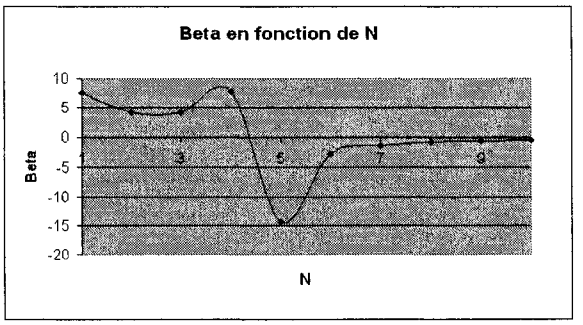

FIGURE 4.33 Seuil de rentabilité en fonction du nombre de modules

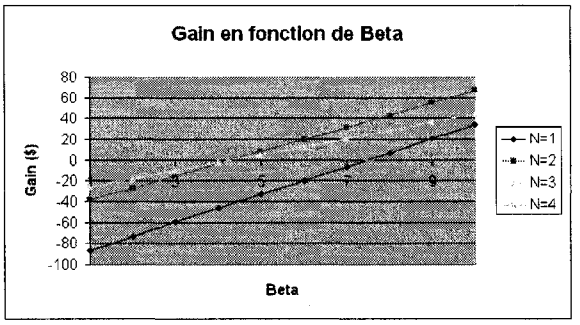

FIGURE 4.34 Gains en fonction du nombre de modules

des différents paramètres en fonction du nombre de modules permet de distinguer les caractéristiques recherchées dans une bonne architecture modulaire. Il faut donc fixer les différents paramètres et en faire varier un à la fois. Le tableau 4.7 résume

les valeurs de ces paramètres. Les figures 4.35 à 4.54 illustrent l'impact des différents paramètres sur les coûts, le seuil de rentabilité et les gains. Une analyse détaillée expliquant l'impact des différent paramètres sur le coût total en fonction de N est donnée à la fin de cette section.

TABLEAU 4.7 Paramètres pour l'évaluation de l'impact des paramètres sur le coût total

$DevModR$	=	100
$DevMod$	=	25
R	=	2.5
p	=	0.75
q	=	0.75
w_{cc}	=	0.5
w_{nf}	=	0.5
$\frac{Couplage}{Cohesion}$	=	0.5
N	=	5
F	=	10

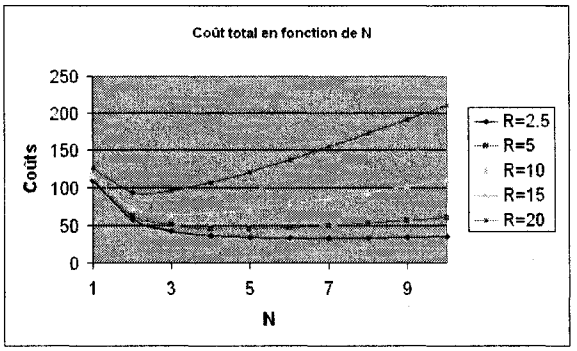

FIGURE 4.35 Coûts en fonction du coût de recherche

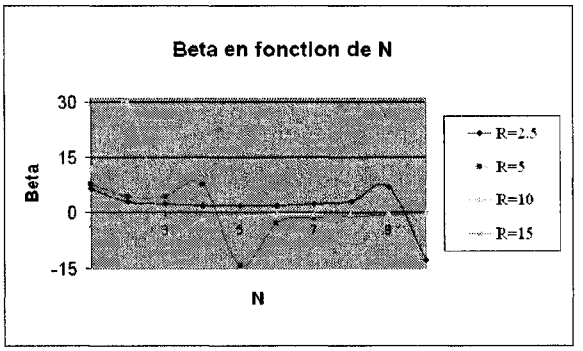

FIGURE 4.36 Seuil de rentabilité en fonction du coût de recherche

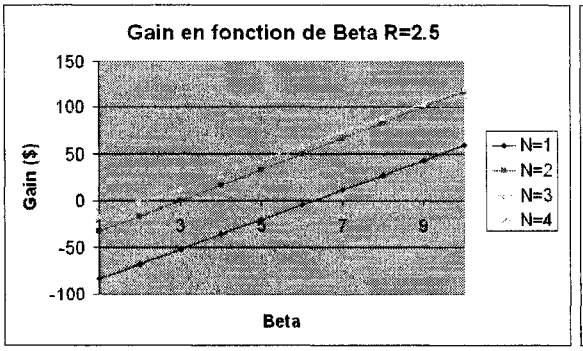

FIGURE 4.37 Gains en fonction de Beta (R=2.5)

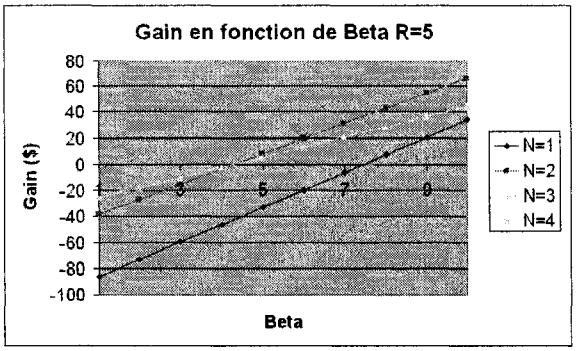

FIGURE 4.38 Gains en fonction de Beta (R=5)

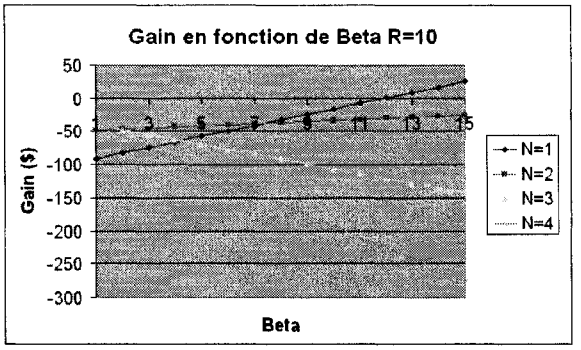

FIGURE 4.39 Gains en fonction de Beta (R=10)

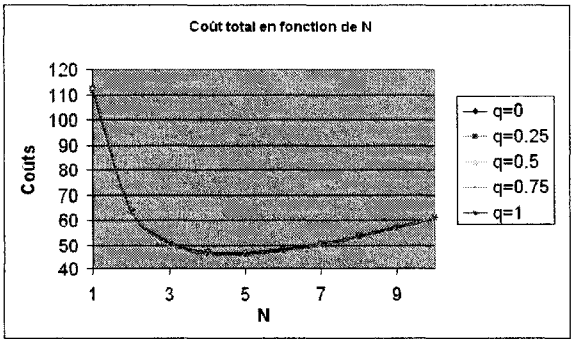

FIGURE 4.40 Coûts en fonction de la probabilité q

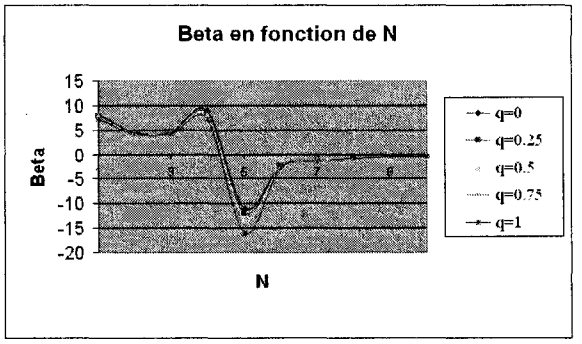

FIGURE 4.41 Seuil de rentabilité en fonction de la probabilité q

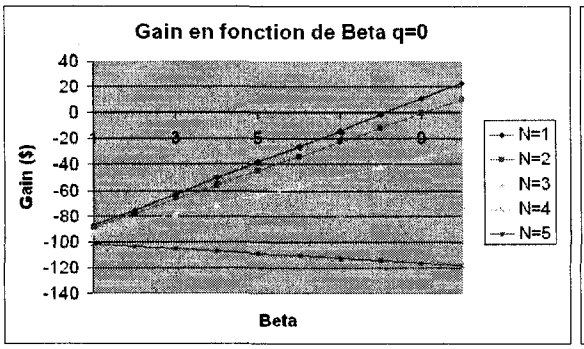

FIGURE 4.42 Gains en fonction de Beta (q=0)

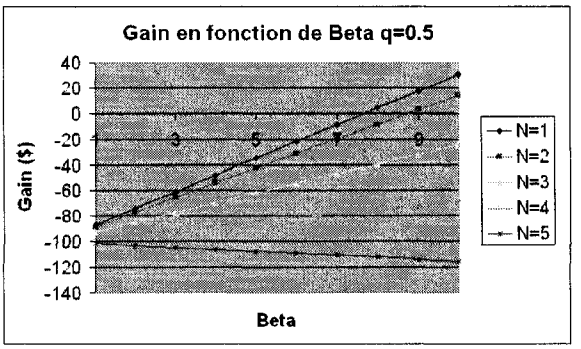

FIGURE 4.43 Gains en fonction de Beta (q=0.5)

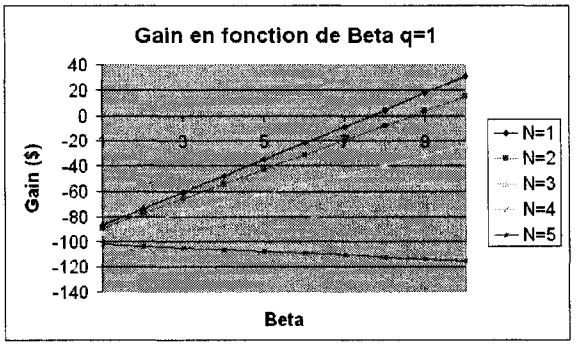

FIGURE 4.44 Gains en fonction de Beta (q=1)

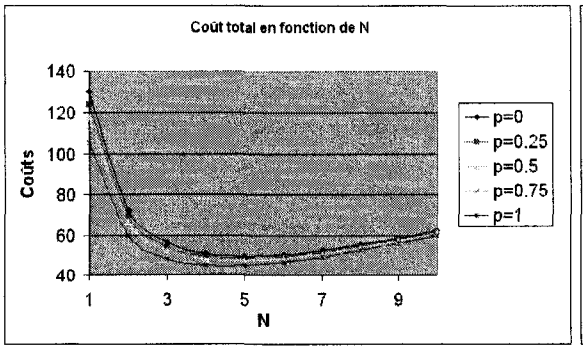

FIGURE 4.45 Coûts en fonction de la probabilité p

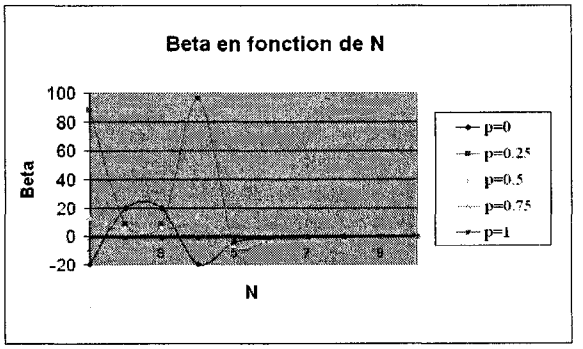

FIGURE 4.46 Seuil de rentabilité en fonction de la probabilité p

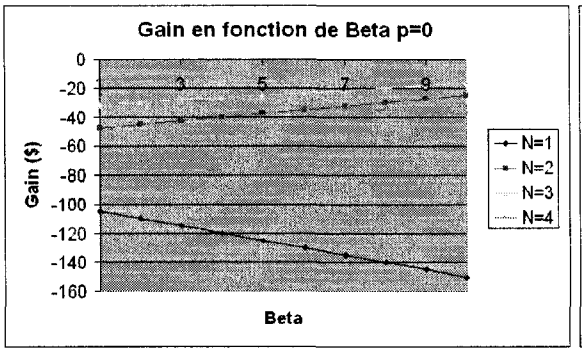

FIGURE 4.47 Gains en fonction de Beta (p=0)

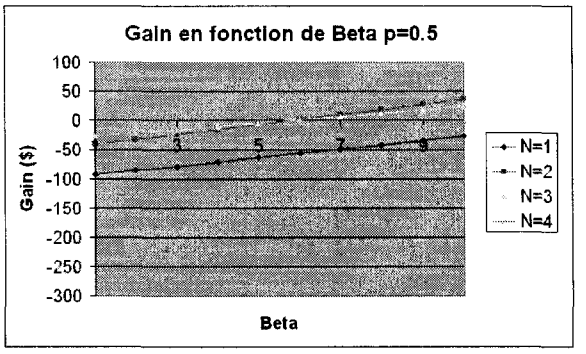

FIGURE 4.48 Gains en fonction de Beta (p=0.5)

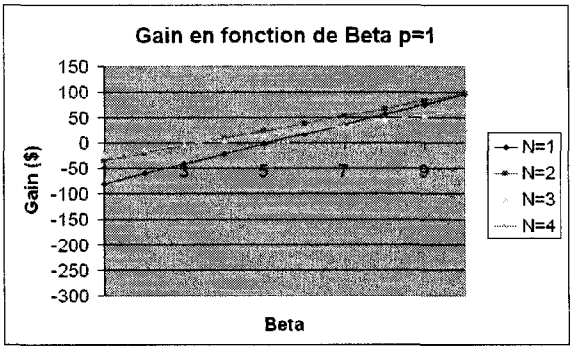

FIGURE 4.49 Gains en fonction de Beta (p=1)

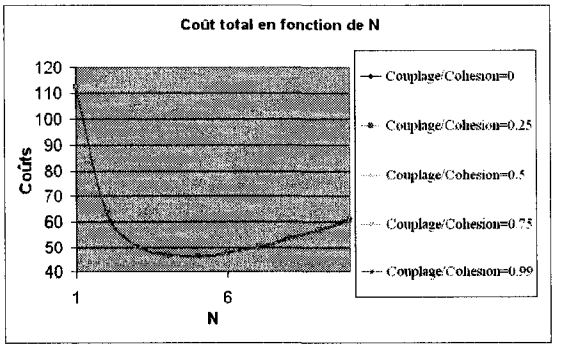

FIGURE 4.50 Coûts total en fonction du ratio initial Couplage/Cohésion

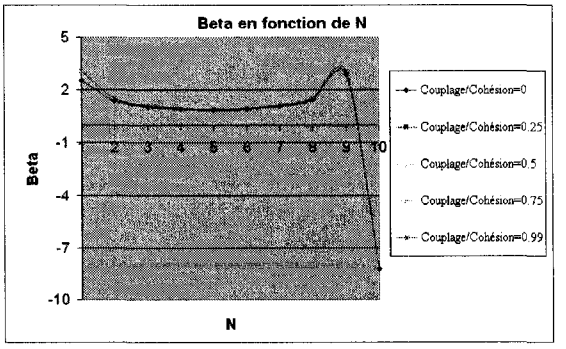

FIGURE 4.51 Seuil de rentabilité en fonction du ratio initial Couplage/Cohésion

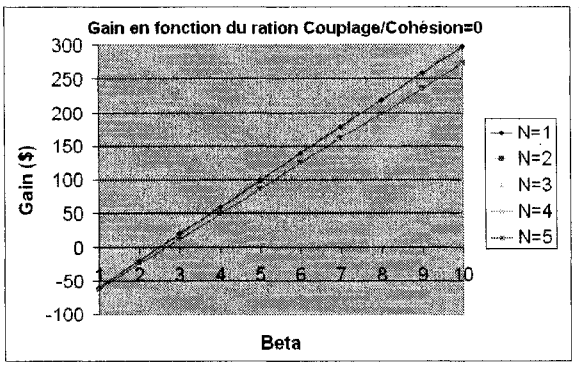

FIGURE 4.52 Gains en fonction de Beta (Couplage/Cohésion=0)

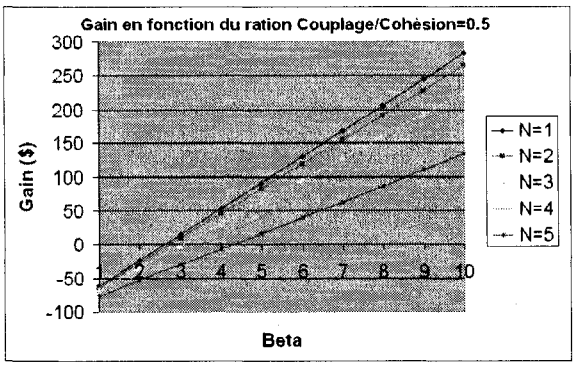

FIGURE 4.53 Gains en fonction de Beta (Couplage/Cohésion=0.5)

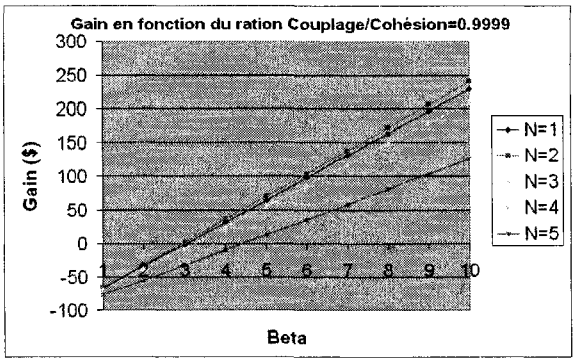

FIGURE 4.54 Gains en fonction de Beta (Couplage/Cohésion=0.9999)

Analyse

Le tableau 4.8 résume l'impact des différents paramètres sur le coût total, le seuil de rentabilité et les gains.

Les différents coûts sont influencés par N. Dans un premier temps, l'augmentation de N provoque une diminution des coûts et une augmentation des gains. À partir d'un certain niveau de réutilisation, l'augmentation de N entraîne une diminution des gains. Le seuil subit ce même changement causé par l'augmentation du nombre de modules et du nombre de réutilisations. Le seuil subit un effet de cloche. Au niveau des gains, le même phénomène se produit. Par contre l'effet est linéaire. Les figures

TABLEAU 4.8 Impact des paramètres sur les coûts

		Coût total	Seuil	Gains
N	↑	↑	↓ ↑	↑ ↓
R	↑	↑	↑	↓
p	↑	↓	↓	↑
q	↑	↓	↓	↑
$\frac{Couplage}{Cohesion}$	↑	↑	↑	↓

4.55 et 4.56 illustrent l'effet de cloche du seuil et l'effet linéaire des gains respectivement.

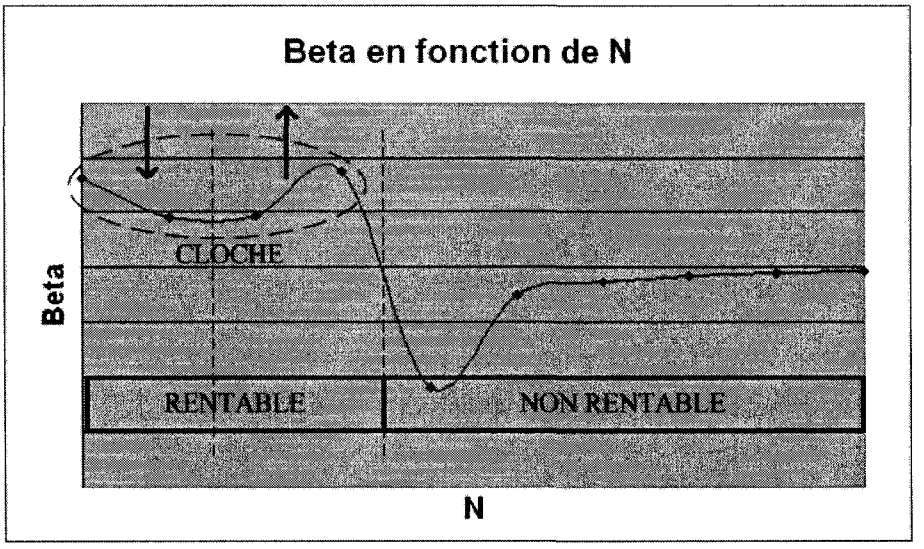

FIGURE 4.55 Seuil de rentabilité : effet de cloche

Le coût de recherche influence directement le coût de réutilisation. Tel que mentionné, plus le coût de recherche est grand, plus le coût total est grand. Par conséquent, lorsque R augmente, le seuil de rentabilité augmente et les gains diminuent. La probabilité q, c'est-à-dire la probabilité qu'il n'y ait pas de modifications ou d'adaptations à apporter, influence le coût de réutilisation. Lorsque q augmente, le coût de réutilisation diminue. Une augmentation de q entraîne une diminution du coût total et du seuil de rentabilité. À l'inverse, une augmentation de q entraîne une augmentation des gains. La probabilité p, c'est-à-dire la probabilité que le module recherché

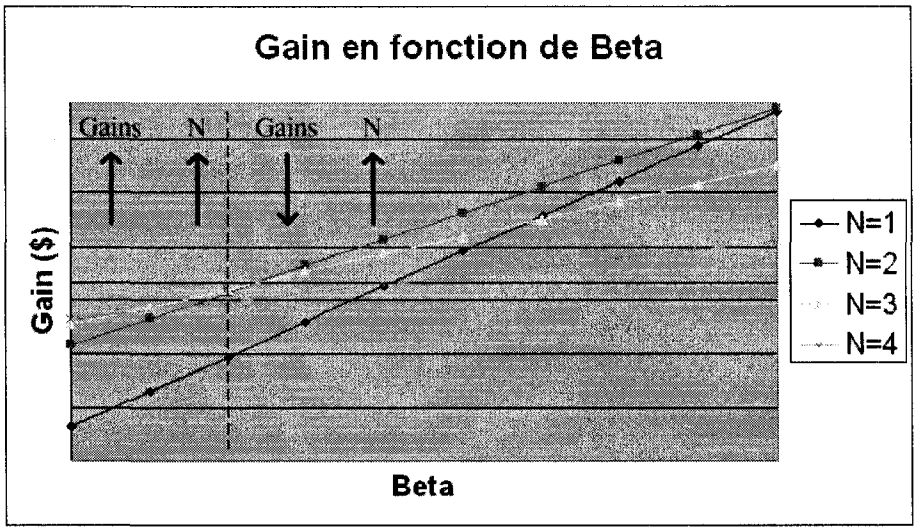

FIGURE 4.56 Gains : effet linéaire

soit trouvé, influence aussi le coût de réutilisation. Le comportement d'une augmentation de la probabilité p est le même qu'une augmentation de la probabilité q. Une augmentation de p entraîne une diminution du coût total et du seuil de rentabilité et une augmentation des gains. Le ratio initial $\frac{Couplage}{Cohesion}$ influence directement la complexité du module. La complexité influence le coût de modification et d'adaptation. Une augmentation du ratio initial $\frac{Couplage}{Cohesion}$ entraîne une augmentation du coût total et du seuil de rentabilité. À l'inverse, une augmentation du ratio initial $\frac{Couplage}{Cohesion}$ entraîne une diminution des gains.

4.3 Plan d'expérience

Dans les sections précédentes, l'impact des paramètres sur les différentes métriques relatives au design et les différents coûts a été évalué. Des exemples de modules de service, de connectivité et d'inter connectivité ont aussi été donnés. Il s'agit maintenant d'évaluer ces différentes métriques sur un module de service de façon à pouvoir évaluer l'impact global du changement d'un paramètre. Il faut d'abord définir les objectifs de l'expérience. Ensuite, il faut identifier les métriques qui seront évaluées. La dernière étape avant l'exécution de l'expérience consiste à définir les méthodes d'analyse.

4.3.1 Objectifs

Les objectifs de cette expérimentation peuvent se résumer en trois volets. Tout d'abord, il faut évaluer les métriques relatives au design. Il faut faire varier la décomposition en composants et modules de l'architecture du service de VoIP présenté à la figure 4.2 de la section (4.1.1) et évaluer l'impact de ces changements. Il faut chercher à déterminer quelle est la meilleure décomposition de l'architecture du service de VoIP en regard de ces métriques.

Par la suite, la décomposition du service de VoIP en modules et en composants influence aussi les différents coûts. Il faut chercher à déterminer quel est la décomposition optimale en composants et en modules du service de VoIP en regard des différents coûts. Il faut aussi évaluer la rentabilité de l'architecture modulaire proposée.

Finalement, il faut déterminer si l'optimalité du module de service de VoIP en regard des métriques de design concorde avec celle des coûts.

4.3.2 Métriques

Il y a quatre métriques relatives au design d'un module qu'il faut évaluer :

- couplage ;
- cohésion ;
- taille ;
- complexité.

Il y a différents coûts qui peuvent être évalués pour un service donné en fonction du nombre de composants et de modules qui composent ce service :

- réutilisation ;
- conception ;
- total.

À partir de ces coûts, il est possible d'établir le seuil de rentabilité. À partir de ces coûts et du nombre de réutilisations envisagées, il est possible d'établir les gains.

4.3.3 Méthodes d'analyse

Pour évaluer les métriques relatives au design, les différents coûts et être en mesure de déterminer l'optimalité, il faut faire varier la décomposition en composants et modules du service. Il faut donc faire varier le nombre de composants et de modules qui composent le service. Ensuite, il faut évaluer la complexité pour chaque nouvelle décomposition du service en composants et en modules. Comment évaluer la complexité d'un module ? Une fois que les composants sont identifiés, il est possible de définir des matrices d'interactions entre chaque composant en différenciant le type de trafic (voix et données). La figure 4.57 illustre un exemple de matrices d'interactions. Tout d'abord, il y a une matrice d'interaction entre les composants du module qui ne différencie pas le type de trafic. Un 1 indique que les deux composants doivent communiquer (échange d'information) ensemble (au moins une transaction). Les deux matrices sous cette matrice sont les matrices respectives des données et de la voix. La matrice du niveau supérieur correspond à la somme de ces deux matrices. La matrice d'interactions entre les pièces est celle qui définit les interactions entre les composants. Pour chaque paire de composants, il faut définir la matrice d'interactions des pièces. Le composant i possède P_i composants et le composant j possède P_j composants. Ce sont ces matrices qui permettent de déterminer la cohésion du module.

Le tableau 4.9 résume la syntaxe du fichier de cohésion qui permet de représenter l'ensemble des matrices d'interactions. La ligne 1 indique le nombre de composants du module. Les lignes 2 et 3 définissent le type de cohésion pour la voix et les données. Ensuite, il faut définir les matrices d'interactions composant par composant. La ligne 4 indique le numéro du composant. La ligne 5 indique le nombre de pièces de ce composant. Les lignes 6 à 10 définissent les interactions de chacune des pièces avec les autres composants, et ce, pour la voix. Il y a autant de colonnes que de composants. Chaque colonne correspond à un composant. Chaque ligne correspond à une pièce. Un 1 indique qu'il y a une interaction entre cette pièce et le composant concerné. Les lignes 11 à 15 font de même pour la voix. Il faut répéter pour chaque composant. La ligne 76 définit la matrice de chemins. Entre chaque paire de composants et pour

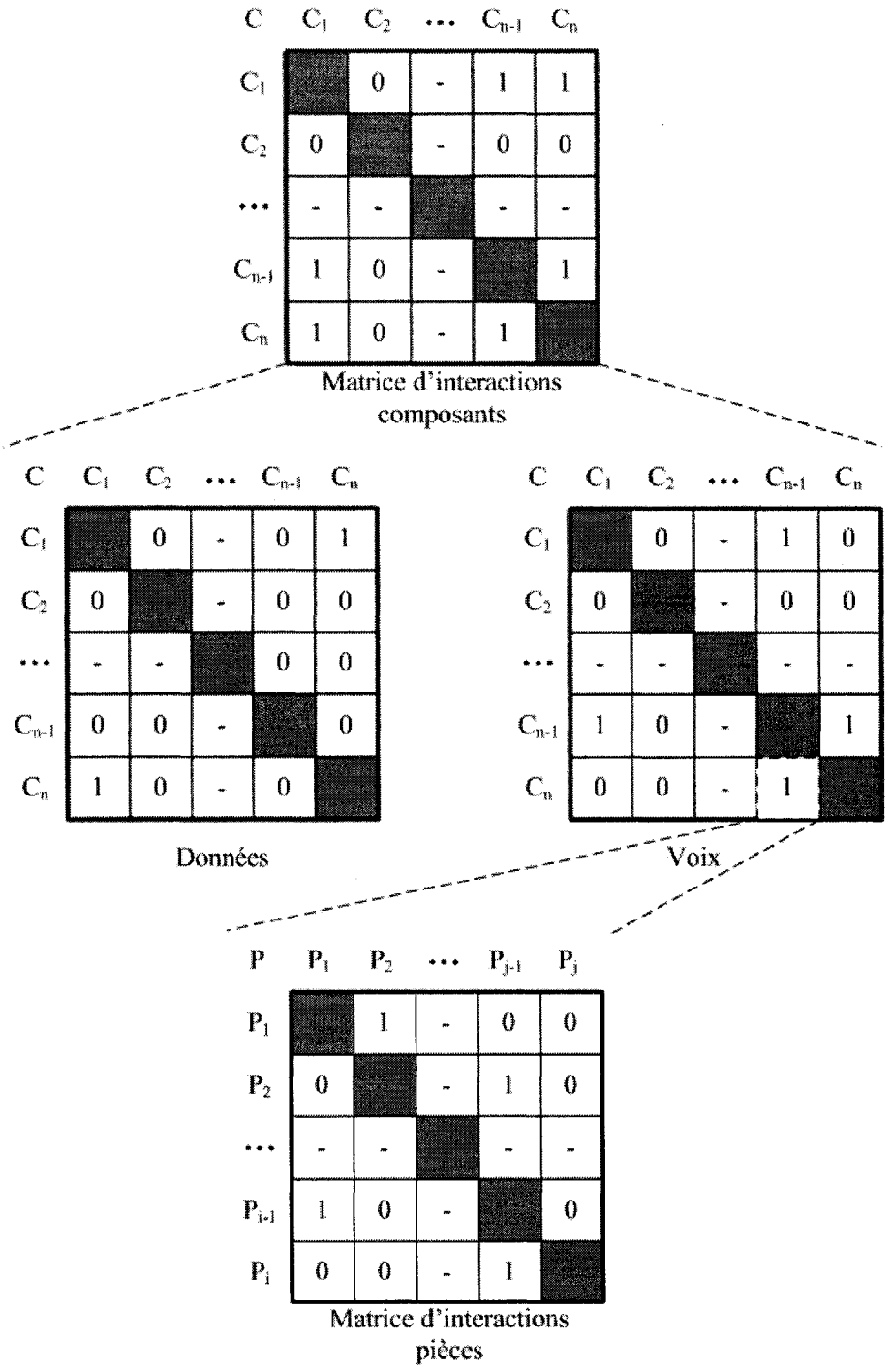

FIGURE 4.57 Matrices d'interactions

chaque type de trafic, il faut définir le nombre de chemins différents que peut emprunter un paquet pour se rendre à destination. Les lignes 77 à 84 correspondent aux données et les lignes 85 à 92 à la voix. Chaque colonne correspond à un composant. Chaque ligne correspond à un composant. À partir de ce fichier, il est possible de calculer la cohésion en appliquant les équations (3.6) à (3.10).

TABLEAU 4.9 Syntaxe du fichier cohesion

Ligne	
1	C= 8
2	wd= 0.5
3	wv= 0.5
4	C1
5	p1= 5
6	0 0 0 0 0 0 0 0
	...
10	0 0 0 0 0 0 0 0
11	0 0 2 1 0 7 4 6
	...
15	0 0 2 1 0 7 4 6
76	cij
77	0 0 1 1 0 0 0 0
	...
84	0 0 0 0 0 0 0 0
85	0 0 1 1 1 1 1 1
	...
92	1 0 1 1 1 1 1 0

Le tableau 4.10 résume la syntaxe du fichier couplage. Le couplage implique les relations que pourraient avoir des composants avec d'autres modules. La ligne 1 indique le nombre de composants du module. Les lignes 2 et 3 indiquent le nombre de ports pour les données et la voix qui permettent de communiquer avec d'autres modules. Ensuite, il faut définir pour chaque composant les transactions potentielles avec d'autres modules et leur type. Pour chaque composant, il y a autant de colonnes que de pièces. Pour chaque pièce, il faut définir s'il y a une transaction potentielle avec un autre module. La ligne 4 indique le numéro du composant. La ligne 5 indique le nombre de pièces du composant. La ligne 6 définit les transactions potentielles des

pièces pour les données. La ligne 7 fait de même pour la voix. Un 1 indique que la pièce en question doit communiquer avec un module adjacent. Les lignes 8 et 9 définissent le type de transaction pour les données et la voix respectivement. Il faut répéter la même chose pour chaque composant. Il y a autant de colonnes que de pièces dans le composant.

TABLEAU 4.10 Syntaxe du fichier couplage

Ligne						
1	C=	8				
2	xd=	1				
3	xv=	5				
4	C1					
5	p1=	5				
6	0	0	0	0	0	
7	1	1	1	1	1	
8	0	0	0	0	0	
9	0.5	0.5	0.5	0.5	0.5	
	...					
46	C8					
47	p8=	6				
48	0	0	0	0	0	0
49	1	1	1	1	1	1
50	0	0	0	0	0	0
51	0.5	0.5	0.5	0.5	0.5	0.5

Le tableau 4.11 résume la syntaxe du dernier fichier nécessaire, celui des paramètres. Il s'agit de spécifier le nombre de composants (ligne 1), le nombre de fonctionnalités (ligne 2) et les poids de la complexité (ligne 3 et 4). Une fois ces fichiers définis, il est possible de calculer la complexité. À partir de celle-ci, il est possible de calculer les différents coûts.

TABLEAU 4.11 Syntaxe du fichier paramètres

Ligne	
1	N= 8
2	F= 9
3	wcc= 0.5
4	wnf= 0.5

4.4 Résultats expérimentaux

Dans cette section, l'expérience définie à la section précédente est réalisée. L'influence de la décomposition en composants et en module de l'architecture du service de VoIP de la figure 4.1 est analysée.

4.4.1 Influence des composants

Le tableau 4.12 résume les valeurs des paramètres utilisés pour évaluer l'influence des composants. Les tableaux 4.13 à 4.19 illustrent les sept décompositions possibles du module de service de VoIP.

TABLEAU 4.12 Paramètres pour l'évaluation de l'influence des composants

$DevMod$	=	25
$DevModR$	=	50
$Couplage$	=	62
R	=	10
p	=	0.9
q	=	0.7
N	=	9

TABLEAU 4.13 Complexité : 2 composants

C1	=	téléphones analogues, faxs, gestionnaire d'appels et boîte vocale, routeur d'accès
C2	=	téléphones IP, téléphones IP et PCs, IP softphones et PCs, commutateur

TABLEAU 4.14 Complexité : 3 composants

C1	=	téléphones analogues, faxs
C1	=	gestionnaire d'appels et boîte vocale, routeur d'accès, commutateur
C3	=	téléphones IP, téléphones IP et PCs, IP softphones et PCs

TABLEAU 4.15 Complexité : 4 composants

C1	=	téléphones analogues, faxs,
C2	=	gestionnaire d'appels et boîte vocale, routeur d'accès, commutateur
C3	=	téléphones IP
C4	=	téléphones IP et PCs, IP softphones et PCs

TABLEAU 4.16 Complexité : 5 composants

C1	=	téléphones analogues, faxs,
C2	=	gestionnaire d'appels et boîte vocale, routeur d'accès
C3	=	commutateur
C4	=	téléphones IP
C5	=	téléphones IP et PCs, IP softphones et PCs

TABLEAU 4.17 Complexité : 6 composants

C1	=	téléphones analogues, faxs,
C2	=	gestionnaire d'appels et boîte vocale
C3	=	routeur d'accès
C4	=	commutateur
C5	=	téléphones IP
C6	=	téléphones IP et PCs, IP softphones et PCs

Métriques relatives au design

Dans la section (4.2.1), l'impact individuel de chacun des paramètres de la complexité a été évalué. Le changement d'un paramètre ne tenait pas compte des changements possibles des autres paramètres. Par exemple, une augmentation du nombre de composants d'un module devrait entraîner un changement au niveau de la cohésion. Pour évaluer l'impact du changement d'un paramètre sur les autres, il faut évaluer

TABLEAU 4.18 Complexité : 7 composants

C1	=	téléphones analogues, faxs,
C2	=	gestionnaire d'appels et boîte vocale
C3	=	routeur d'accès
C4	=	commutateur
C5	=	téléphones IP
C6	=	téléphones IP et PCs
C7	=	IP softphones et PCs

TABLEAU 4.19 Complexité : 8 composants

C1	=	téléphones analogues
C2	=	faxs,
C3	=	gestionnaire d'appels et boîte vocale
C4	=	routeur d'accès
C5	=	commutateur
C6	=	téléphones IP
C7	=	téléphones IP et PCs
C8	=	IP softphones et PCs

ceux-ci sur un module implémenté. Dans ce cas-ci, le module de service de VoIP illustré à la figure 4.2 est utilisé.

Pour évaluer l'impact que peut avoir le nombre de composants sur les autres métriques de design, il faut faire varier la décomposition de l'architecture en composants et réévaluer les différentes métriques. Pour chaque nouvelle décomposition du module en composants il faut redéfinir les fichiers cohésion et paramètres. Le couplage ne change pas, étant donné que le nombre de modules ne change pas. Le fichier cohésion correspond aux matrices d'interactions entre les composants. Il y a neuf fonctionnalités dans le module. Deux d'entre-elles ne peuvent êtres séparées : gestionnaire d'appels et boîte vocale. Il peut donc y avoir un maximum de huit composants. Le nombre de composants varie de $N = 2$ à $N = 7$. Pour chacune de ces décompositions de l'architecture en composants, la cohésion, le couplage, la taille et la complexité sont évalués en fonction des matrices d'interactions. La figure 4.58 illustre la cohésion du module de service de VoIP en fonction du nombre de composants. Lorsque le nombre de composants augmente, la cohésion augmente. Les interactions

internes aux composants ne sont pas considérées dans le calcul de la cohésion. Lorsque le nombre de composants augmente, une partie de ces interactions est transformée en cohésion. Le couplage reste constant étant donné que la matrice d'interactions avec les modules adjacents est établie au niveau des pièces. La décomposition du module en composants n'influence pas le calcul du couplage. Étant donné que la cohésion augmente lorsque le nombre de composants augmente et que le couplage reste constant, la complexité diminue avec une augmentation du nombre de composants. La figure 4.59 illustre la complexité en fonction du nombre de composants. De plus, le terme $\left(1 - \frac{N}{F}\right)$ de l'expression de la complexité diminue lorsque le nombre de composants augmente. La complexité ne peut que diminuer avec une augmentation du nombre de composants. L'optimalité correspond à la décomposition à huit composants.

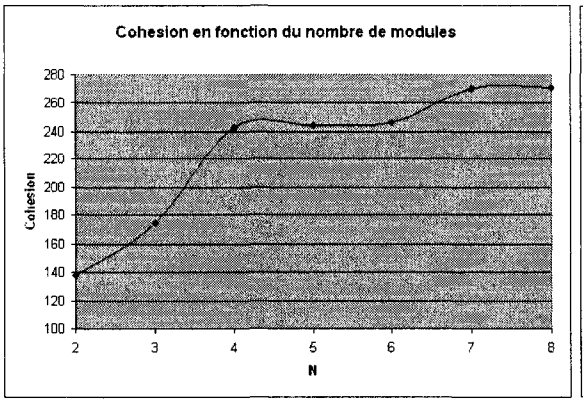

FIGURE 4.58 Cohésion en fonction du nombre de composants

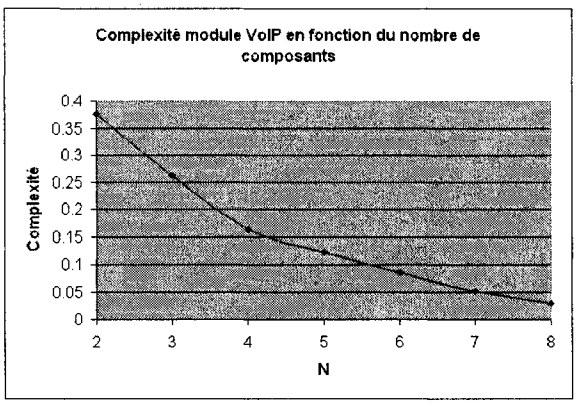

FIGURE 4.59 Complexité en fonction du nombre de composants

Coûts

La complexité influence directement les coûts de modification et d'adaptation qui font partie intégrante des coûts de réutilisation. Lorsque la complexité augmente, les coûts de réutilisation augmentent. Une diminution de la complexité produit l'effet inverse. Dans ce cas-ci, une augmentation du nombre de composants entraîne une diminution de la complexité et par conséquent une diminution du coût de réutilisation. La figure 4.60 illustre le coût de réutilisation en fonction du nombre de composants.

Lorsque le nombre de composants augmente, le seuil de rentabilité diminue. Le

nombre de réutilisations nécessaires pour rentabiliser le coût de développement du module réutilisable diminue. Comme la complexité diminue, les coûts de réutilisation diminuent. À chaque réutilisation, une plus grande part du coût de développement du module réutilisable est amortie. La figure 4.61 illustre le seuil de rentabilité en fonction du nombre de composants. Il est à noter que l'effet de cloche du seuil observé précédemment n'est pas présent

Le seuil de rentabilité et les gains sont intimement liés. Si le seuil de rentabilité diminue, le nombre de réutilisations nécessaires pour rentabiliser le coût de développement diminue. Mais est-ce que ça veut dire que les gains seront plus grands ? Est-ce que le fait de rentabiliser un module plus rapidement le rend plus rentable pour autant ? Si le nombre de modules est le même, la réponse est oui. Sinon, ce n'est pas le cas. La figure 4.62 illustre les gains pour chaque valeur du nombre de composants en fonction du nombre de réutilisations. Lorsque le nombre de composants augmente, les gains augmentent eux-aussi. Plus le nombre de réutilisations augmente, plus les gains sont élevés. L'optimalité correspond à $N = 8$.

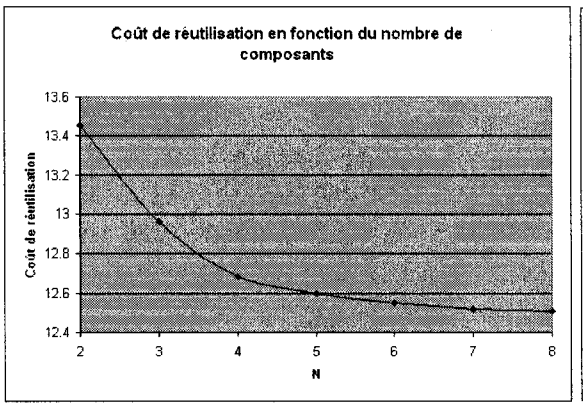

FIGURE 4.60 Coût de réutilisation en fonction du nombre de composants

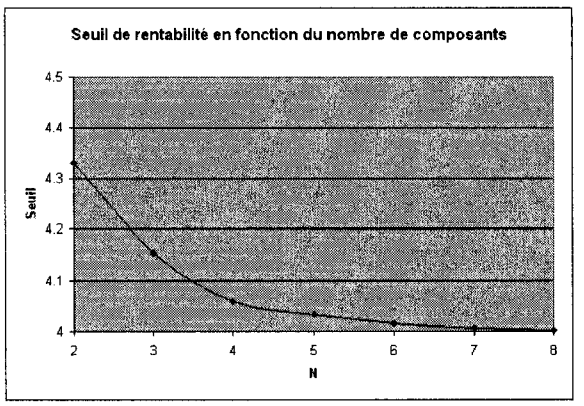

FIGURE 4.61 Seuil en fonction du nombre de composants

4.4.2 Influence des modules

Le tableau 4.20 résume les valeurs initiales des paramètres utilisés pour évaluer l'influence du nombre de modules d'un service. Les tableaux 4.21 à 4.23 illustrent les

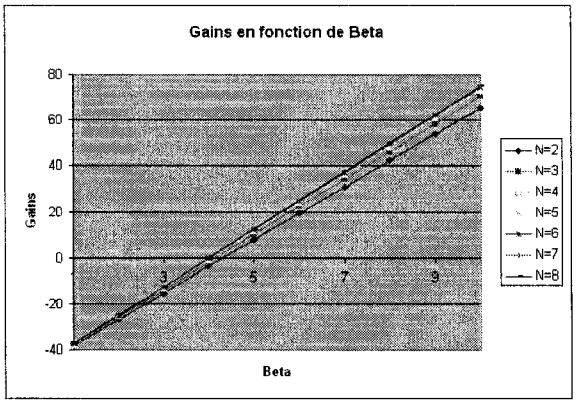

FIGURE 4.62 Gains en fonction du nombre de composants

trois décompositions possibles du service de VoIP en modules et composants.

TABLEAU 4.20 Paramètres pour l'évaluation de l'influence des modules

$DevMod$	=	25
$DevModR$	=	50
R	=	10
p	=	0.9
q	=	0.7
N	=	9

TABLEAU 4.21 Complexité : 1 module

N1		
C1	=	téléphones analogues
C2	=	faxs,
C3	=	gestionnaire d'appels et boîte vocale
C4	=	routeur d'accès
C5	=	commutateur
C6	=	téléphones IP
C7	=	téléphones IP et PCs
C8	=	IP softphones et PCs

TABLEAU 4.22 Complexité : 2 modules

N1		
C1	=	téléphones analogues
C2	=	fax
C3	=	gestionnaire d'appels et boîte vocale
C4	=	routeur d'accès
N2		
C1	=	commutateur
C2	=	téléphones IP
C3	=	téléphones IP et PCs
C4	=	IP softphones et PCs

TABLEAU 4.23 Complexité : 3 modules

N1		
C1	=	téléphones analogues
C2	=	fax
C3	=	téléphones IP
N2		
C1	=	gestionnaire d'appels et boîte vocale
C2	=	routeur d'accès
C3	=	commutateur
N3		
C1	=	téléphones IP et PCs
C2	=	IP softphones et PCs

Métriques relatives au design

Pour évaluer l'impact que peut avoir le nombre de modules sur les autres métriques de design, il faut faire varier la décomposition en modules et en composants de l'architecture du service de VoIP présenté à la figure 4.2. Pour chaque nouvelle décomposition, il faut évaluer les différentes métriques relatives au design ,et ce,

pour chaque module. Il faut donc pour chaque nouvelle décomposition redéfinir les différentes matrices d'interactions pour la cohésion et le couplage. Dans la section précédente, le couplage ne changeait pas lorsque le nombre de composants augmentait puisque le nombre de modules ne variait pas. Dans ce cas-ci, le couplage va varier d'un module à l'autre. Pour chaque décomposition et pour chaque module, il faut évaluer la cohésion, le couplage, la taille et la complexité. Trois décompositions ont été réalisées : $N = 1$ à $N = 3$. Il n'est pas possible de décomposer cette architecture en plus de trois modules. Au-delà de trois modules, la contrainte de conception 3.14 n'est pas satisfaite. Le couplage devient plus grand que la cohésion. La figure 4.63 illustre la cohésion et le couplage moyen en fonction du nombre de modules. La cohésion et le couplage moyen diminuent lorsque le nombre de modules augmente. Toutefois, la cohésion diminue beaucoup plus rapidement que le couplage. Ça veut dire que le couplage augmente par rapport à la cohésion. La figure 4.64 illustre la complexité en fonction du nombre de modules. Il y a deux courbes de complexité sur cette figure. La première est celle calculée à partir de l'équation (3.31). La seconde est calculée à partir de l'équation (3.12). La seconde est calculée à partir des valeurs de la cohésion et du couplage de la figure 4.63. Lorsque le nombre de modules augmente, la complexité augmente. La différence entre la complexité calculée à partir de la figure 4.63 s'atténue avec une augmentation du nombre de modules. La formule estime adéquatement la complexité en fonction du nombre de modules. L'optimalité correspond à la décomposition à un module.

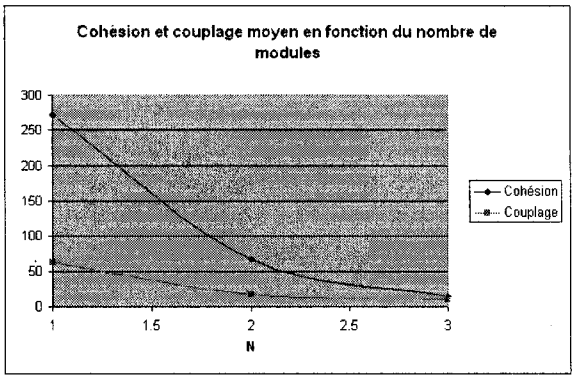

FIGURE 4.63 Cohésion et couplage en fonction du nombre de modules

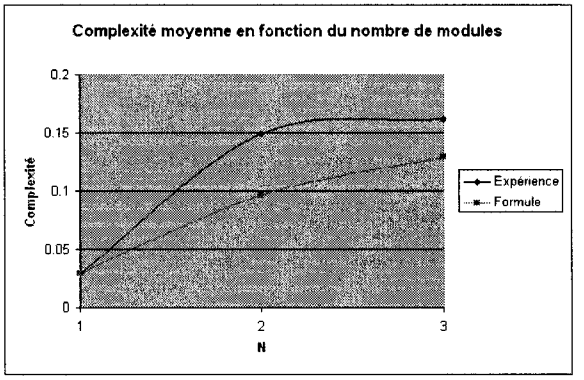

FIGURE 4.64 Complexité en fonction du nombre de modules

Coûts

La décomposition de l'architecture du service de VoIP en modules et en composants influence les différents coûts. Pour chaque décomposition présentée aux tableaux 4.21 à 4.23, il est possible d'évaluer le coût total, le seuil de rentabilité et les gains. La variation de la décomposition de l'architecture fait varier la complexité. C'est ce paramètre qui fait varier les différents coûts. Les valeurs des différents coûts sont calculés à partir des équations (3.21) à (3.31) en remplaçant l'équation (3.26) par la valeur de la complexité. La figure 4.65 illustre les différents coûts en fonction du nombre de modules. Les coûts de conception diminuent lorsque le nombre de modules augmente. Le coût total diminue puis augmente. Pour un grand nombre de modules l'augmentation des coûts de réutilisation serait plus prononcée que la diminution des coûts de conception. Il faut aussi considérer le seuil de rentabilité en fonction du nombre de modules. La figure 4.66 illustre le seuil de rentabilité en fonction du nombre de modules. Seule la décomposition à un module serait rentable. Sur la figure 4.65, le coût total associé à $N = 1$ était le plus élevé. Toutefois, la faible valeur des coûts de réutilisation compense pour le coût plus élevé de développement du module réutilisable. Il est à noter que l'effet de cloche observé précédemment pour le seuil est présent. La figure 4.67 illustre les gains en fonction du nombre de modules. Comme seule la décomposition à un module est rentable, seuls les gains de $N = 1$ sont des profits. Pour $N = 2$ et $N = 3$, plus le nombre de modules est grand, plus les pertes augmentent. Donc, l'optimalité correspond à $N = 1$.

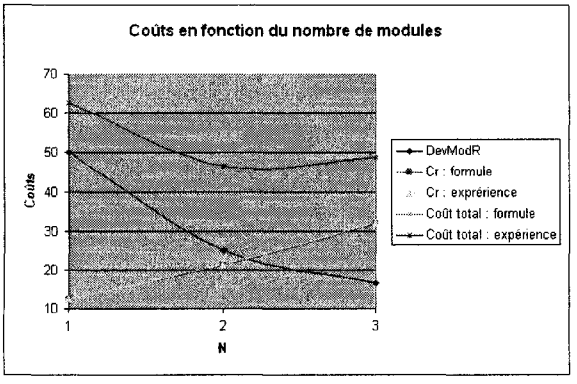

FIGURE 4.65 Coûts en fonction du nombre de modules

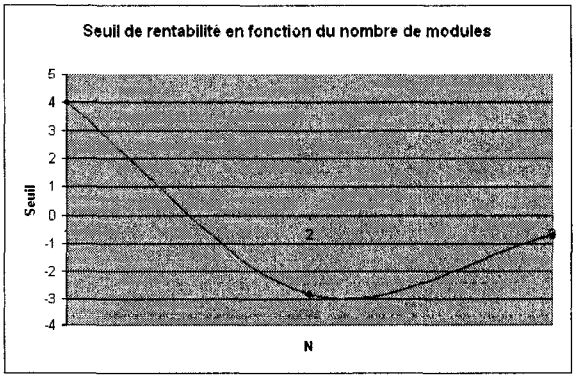

FIGURE 4.66 Seuil en fonction du nombre de modules

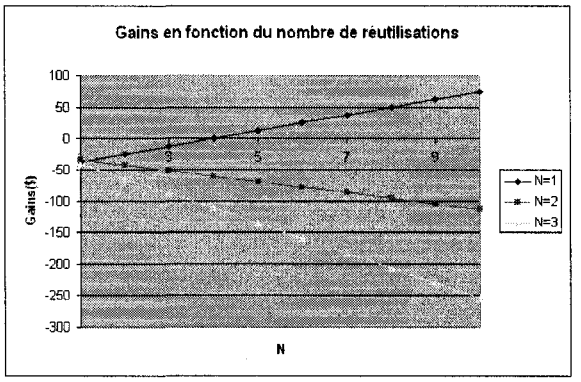

FIGURE 4.67 Gains en fonction du nombre de modules

4.4.3 Analyse des résultats

Pour un même service, plus le nombre de modules augmente, plus la complexité moyenne des modules augmente. Le couplage moyen des modules augmente par rapport à la cohésion moyenne. Étant donné que la complexité doit être minimisée, un service constitué d'un module est optimal. Pour un même module, plus le nombre de composants est grand, plus la complexité diminue. Une association une à une entre les composants et les fonctionnalités est optimale.

Pour un même service, plus le nombre de modules est grand, plus le seuil de rentabilité, c'est-à-dire le nombre de réutilisations nécessaires pour rentabiliser le coût de développement du module réutilisable, est grand. Il faut toutefois considérer l'effet de cloche du seuil de rentabilité. Plus le nombre de réutilisation β et le nombre de modules N sont grands, plus les gains diminuent. Un service composé d'un petit nombre de modules maximise les gains. Il faut toutefois considérer le nombre de réutilisations envisagées en regard de l'effet de cloche du seuil de rentabilité. En général, un service constitué d'un module est optimal au niveau des coûts.

L'optimalité en regard des métriques relatives au design concorde avec celle en regard des coûts. Considérant que l'optimalité correspond à un service composé d'un seul module, il faut se concentrer sur la décomposition du module en composants.

CHAPITRE 5

CONCLUSION

La méthodologie employée dans le cadre de ce mémoire est en trois étapes. Les travaux réalisés peuvent donc être classifiés selon trois volets. Le premier volet est une revue de littérature des travaux antérieurs dans le domaine de l'architecture modulaire. Le second volet propose une architecture modulaire pour la conception des réseaux de télécommunications en lien avec les problèmes et questions soulevés dans la revue de littérature. Le dernier volet évalue la solution proposée avec des résultats analytiques et expérimentaux à l'appui. La synthèse de ces travaux fait l'object de ce chapitre. Les limitations de la solution proposée sont identifiées. Finalement, des pistes pour des recherches futures sont données.

5.1 Synthèse des travaux

Les méthodes actuelles de conception et d'ingénierie des réseaux de télécommunications ne sont pas optimales au niveau des coûts et des performances. La diversité et la complexité des technologies rendent difficile le développement de nouveaux réseaux. Pour faire face à la concurrence de plus en plus forte dans ce marché, les compagnies doivent rationaliser leurs méthodes de conception. L'objectif de ce mémoire est de proposer une approche innovatrice de conception pour les réseaux de télécommunications basée sur le concept d'architecture modulaire. Cette approche de conception devait permettre de réduire les coûts de conception, d'améliorer les performances réseaux, d'uniformiser le module optimal pour la réutilisation et de maximiser l'interopérabilité, la flexibilité et la réutilisabilité des modules.

La revue de littérature a permis d'identifier plusieurs questions par rapport à l'architecture modulaire dans le contexte des réseaux de communications. Qu'est-ce qu'un module dans ce contexte ? Quelles sont ses caractéristiques ? Que contient-il ? Comment le représenter ? Quels sont ses coûts ? Comment déterminer l'optimalité ?

Comment décomposer l'architecture d'un réseau en modules ? La plupart des travaux réalisés sur l'architecture modulaire concernent l'amélioration de la productivité et l'optimisation des ressources dans un cadre industriel. Dans le domaine des télécommunications, peu d'efforts de recherche ont été consacrés aux méthodes de conception et d'ingénierie des réseaux de télécommunications.

L'architecture modulaire proposée répond à différents requis. Il y a les requis architecturaux et les requis propres aux modules. Différentes métriques ont été définies pour caractériser la structure du module et ses performances. Le principe de l'architecture modulaire proposée repose sur l'augmentation du niveau d'abstraction du design de façon à faciliter la conception des réseaux. La division de l'architecture du réseau en couches hiérarchiques permet d'obtenir cette abstraction du design. L'architecture modulaire peut être décomposée en quatre catégories d'items (patron architectural, module, composant et pièce) qui sont reliés ensemble par des liens et/ou tubes de connectivité et qui sont distribués géographiquement selon trois catégories d'emplacement (salle, bureau et campus). Il y a trois types de modules : services, connectivité et inter-connectivité. Lors de l'assemblage des modules, différents types d'incompatibilités peuvent survenir : technologie, protocole, comportement et performance. Le processus de réutilisation des modules comporte cinq volets : sélection, modification-adaptation, assemblage, vérification-validation et intégration. Afin de pouvoir déterminer l'optimalité d'un module, différentes métriques ont été définies. Un module possède quatre métriques qui permettent de caractériser sa structure : couplage, cohésion, taille et complexité. Par hypothèse, le module encapsulé est optimal au niveau des performances. Les métriques de performances sont identifiées mais elles ne sont pas définies. Il y a différents coûts inhérents à l'architecture modulaire. Il y a les coûts de conception et les coûts de réutilisation. À partir de ces coûts, le seuil de rentabilité et les gains liés à l'utilisation d'un module peuvent être dérivés. Après avoir défini les différents concepts liés à l'architecture modulaire dans le contexte des réseaux de communications, un mécanisme de réutilisation est défini de façon à pouvoir appliquer adéquatement ces différents concepts.

La dernière partie du mémoire concerne l'implémentation et les résultats. Un sous-ensemble restreint de modules est implémenté. Ce sous-ensemble comporte un module de chaque type. Ces différents modules sont assemblés pour former une archi-

tecture fonctionnelle. Par la suite, l'impact des différents paramètres des métriques de design et de coût du modèle analytique est analysé. L'objectif est d'identifier les paramètres qui ont le plus d'impact sur ces métriques et quelles valeurs permettent d'obtenir l'optimalité. Suite à cette étude analytique, un plan d'expérience est établi pour évaluer l'impact de ces paramètres sur le sous-ensemble de modules implémenté. L'expérimentation permet d'établir quel est l'impact du nombre de composants d'un module sur les métriques relatives au design et sur les coûts. Elle permet aussi d'établir l'impact du nombre de modules d'un même service sur ces mêmes métriques de design et de coûts. L'impact de la décomposition d'une architecture en modules et en composants est analysé. Il n'est pas possible d'affirmer si une telle architecture est rentable. Ce qui est possible d'affirmer, c'est que dans certaines conditions cette architecture modulaire est rentable. La décomposition de l'architecture en modules et en composants influence les métriques relatives au design et les différents coûts. Il est possible d'évaluer la rentabilité de la décomposition de l'architecture en modules et en composants et d'identifier les conditions de rentabilité.

5.2 Limitations de la solution proposée

Un sous-ensemble de modules (un module de chaque type) a été implémenté. L'assemblage de modules de services différents n'a pas été considéré. Il faudrait voir pour un grand nombre de modules comment l'assemblage se comporte. L'hypothèse de base de l'architecture modulaire proposée est que le module encapsulé est optimal au niveau des performances. L'optimalité des performances au niveau individuel des modules ne garantit par l'optimalité au niveau de l'architecture globale (assemblage). Au niveau des coûts, il n'est pas possible d'évaluer le coût de conception d'un module réutilisable. Le coût de conception d'un module réutilisable est utilisé comme paramètre dans l'évaluation du coût total, du seuil de rentabilité et des gains. Les valeurs employées sont fixées arbitrairement. Ce coût n'est pas évalué en fonction d'une formule ou de données réelles. Il en va de même pour le coût de recherche et les probabiltés p et q, c'est-à-dire la probabilité que le module recherché soit trouvé et qu'il n'y ait pas de modifications ou d'adaptations à apporter. Les valeurs employées dans l'évaluation du modèle analytique et l'expérimentation sont arbitraires.

5.3 Travaux futurs

Cinq champs d'application pourraient être approfondis. Le premier concerne la décomposition de l'architecture en modules et en composants à l'aide d'heuristiques. Le second concerne l'analyse des performances au niveau module et au niveau architectural. Le troisième concerne les mécanismes d'assemblage avec un traitement automatisé des incompatibilités avec des processus de vérification et de validation intelligents. Le quatrième concerne l'analyse des coûts de conception des modules réutilisables. Le cinquième et dernier concerne une analyse plus approfondie des coûts de réutilisation. Chacun de ces champs d'application pourrait faire l'object d'une thèse.

Références

ALEXANDER, C., ISHIKAWA, S., SILVERSTEIN, M., JACOBSON, M. et FIKSDAHL-KING, I. (1996). Software Reuse : Metrics and Models. *Oxford University Press*, pp. 1216.

BALDWIN, C. et CLARK, K. (2000). Design Rules : The Power of Modularity. *The MIT Press*, pp. 471.

BARNES, B. et BOLLINGER, T. (1991). Making software reuse cost effective. *IEEE Software*, pp. 13–24.

BRIAND, L., DALY, J. et WUST, J. (1998). A unified frameworks for coupling measurement in object-oriented systems. *IEEE Transactions on software engineering*, vol. 25, pp. 91–121.

CONSTANTINE, L., MYERS, G. et STEVENS, W. (1974). Structured Design. *IBM Systems Journal*, vol. 13, pp. 115–139.

CONSTANTINE, L. et YOURDON, E. (1979). Structured Design : Fundamentals of a Discipline of Computer Program and Systems Design. *Prentice-Hall*, pp. 473.

EMERSON, T. (1984). A discriminant metric for module cohesion. *Proceedings of the 7th International Conference on Software Engineering*, pp. 294–303.

FAVARO, J. (1991). What price reusability ? A case study. *ACM Ada Letters*, vol. 11, pp. 115–124.

FIXSON, S. (2007). What exactly is product modularity ? The answer depends on who you ask. *MIT Sloan Working Paper*, vol. 4631, pp. 1–36.

GAMMA, E., HELM, R., JOHNSON, R. et VLISSIDES, J. (1995). Design Patterns : elements of reusable object oriented-software. *Addison-Wesley*, pp. 395.

GERSHENSON, J., PRASAD, G. et ZHANG, Y. (2003). Product modularity : definitions and benefits. *Journal of engineering design*, vol. 14, pp. 295–313.

HOPKINS, J. (2000). Component primer. *Communications of the ACM*, vol. 43, pp. 27–30.

JABLAN, S. (1997). Modularity in art. *http ://www.mi.sanu.ac.yu/%7Ejablans/d3.htm*, pp. 1–7.

MARSHALL, R., LEANEY, P. et BOTTERELL, P. (1998). Enhanced Product Realisation Through Modular Design : An Example of Products/Process. *Journal of INtegrated Design and Process Technology*, vol. 3, pp. 143–150.

MILI, A., MILI, H. et MILI, M. (1995). Reusing Software : issues and research directions. *IEEE Transactions on software engineering*, vol. 21, pp. 528–562.

NEPAL, B.-P. (2005). *An integrated framework for modular product architecture.* Thèse de doctorat, Graduate School of Wayne State University of Detroit.

NEWCOMB, P., ROSEN, D. et BRAS, B. (2003). Life Cycle MOdularity Metrics for Product Design. *Proceedings of EcoDesign2003 : Third International Symposium on Environmentally Conscious Design and Inverse Manufacturing Tokyo*, pp. 1–8.

PIMMLER, T. et EPPINGER, S. (1994). Integration analysis of product decompositions. *ASME Design Theory and Methodology Conference Minneapolis*, pp. 1–9.

PRESSMAN, R. (2005). Software Engineering A practitioners's Approach. *McGraw-Hill*, pp. 880.

PRIETO-DIAZ, R. (1993). Status report : software reusability. *IEEE Software*, pp. 61–66.

SCHEIDT, L. et ZONG, S. (1994). An approach to achieve reusability of electronic modules. *Transactions of the IEEE*, pp. 331–336.

SIGFRIED, S. (1996). Understanding Object-Oriented Software Engineering. *IEEE Press.*

SUH, N. (1990). The principles of design. *Oxford Unisersity Press*, pp. 401.

ULRICH, K. (1995). The role of the product archtiecture in the manufacturing firm. *Research Policy*, vol. 24, pp. 419–440.

ULRICH, K. et EPPINGER, K. (1995). Product design and development. *McGraw-Hill*, pp. 289.

ULRICH, K. et EPPINGER, K. (2004). Product design and development. *McGraw-Hill*, pp. 366.

ULRICH, K. et TUNG, K. (1991). Fundamental of product modularity. *Issues in design manufacture/integration ASME 1991*, vol. 39, pp. 73–79.

VITHARANA, P., JAIN, H. et ZAHEDI, M. (2004). Design, Retrieval and Assembly. *IEEE Transactions on Systems, Man and Cybernetics*, vol. 34, pp. 460–474.

WHITNEY, D. (1992). Systematic Design of Modular Products at Telemechanique. *Telemechanique R&D Laboratory*, pp. 1–7.

YASSINE, A. (2003). An Introduction to Modeling and Analyzing Complex Product Development Process Using the DSM. *Product Development Research Laboratory of University of Illinois*, pp. 1–17.