



Titre: Gérer les exigences dans un contexte distribué : défis et pistes de solution pour une entreprise de l'industrie aérospatiale
Title:

Auteur: Georges Réti
Author:

Date: 2007

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Réti, G. (2007). Gérer les exigences dans un contexte distribué : défis et pistes de solution pour une entreprise de l'industrie aérospatiale [Master's thesis, École Polytechnique de Montréal]. PolyPublie. <https://publications.polymtl.ca/7947/>
Citation:

 **Document en libre accès dans PolyPublie**
Open Access document in PolyPublie

URL de PolyPublie: <https://publications.polymtl.ca/7947/>
PolyPublie URL:

Directeurs de recherche: Mario Bourgault
Advisors:

Programme: Unspecified
Program:

UNIVERSITÉ DE MONTRÉAL

GÉRER LES EXIGENCES DANS UN CONTEXTE DISTRIBUÉ :
DÉFIS ET PISTES DE SOLUTION POUR UNE ENTREPRISE DE
L'INDUSTRIE AÉROSPATIALE

GEORGES RÉTI
DÉPARTEMENT DE MATHÉMATIQUES ET DE GÉNIE INDUSTRIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAITRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INDUSTRIEL)

AVRIL 2007



Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 978-0-494-29254-9

Our file Notre référence

ISBN: 978-0-494-29254-9

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.


Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

GÉRER LES EXIGENCES DANS UN CONTEXTE DISTRIBUÉ :
DÉFIS ET PISTES DE SOLUTION POUR UNE ENTREPRISE DE
L'INDUSTRIE AÉROSPATIALE

présenté par : RÉTI Georges

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. PELLERIN Robert, ing., Ph.D., président

M. BOURGAULT Mario, ing., Ph.D., membre et directeur de recherche

M. CHALIFOUR Denis, ing., M.B.A., membre

à ma famille

REMERCIEMENTS

Je tiens d'abord à remercier mon directeur de recherche, Mario Bourgault pour m'avoir permis de conduire cette recherche, mais aussi pour m'avoir donné l'opportunité d'écrire un article puis le présenter à la conférence des usagers de Telelogic. Je tiens également à le remercier de m'avoir permis de participer à d'autres projets intéressants comme le projet de plate forme collaborative ou encore les enquêtes sur la gestion de projet en équipes dispersées qui m'ont permis de mettre en pratique les connaissances que j'ai accumulées.

Je tiens également à remercier CAE pour avoir fourni un terrain d'étude captivant. Je tiens en particulier à remercier Denis Chalifour et Ronald Houde pour leur temps et l'intérêt qu'ils ont porté à mon projet de recherche. Ils m'ont permis de mettre une image concrète aux concepts que j'ai pu apprendre en classe ou lire dans la littérature.

Je tiens enfin à remercier les membres de l'équipe de ePoly d'une part pour toutes les choses que j'ai pu apprendre de leur domaine d'expertise, et qui ont permis d'enrichir la présente recherche grâce à des outils (notamment l'utilisation du logiciel Aris pour la cartographie de processus) ou grâce à des discussions intéressantes qui m'ont aidées dans la partie des liens avec d'autres disciplines. D'autre part je tiens à les remercier pour leur amitié et leur soutien qui ont fait que cette étude a été enrichissante tant sur plan des connaissances que sur le plan humain.

RÉSUMÉ

L'industrie fait de plus en plus appel à des équipes dont les membres sont répartis géographiquement pour mener à bien des projets qui deviennent de plus en plus complexes et risqués, notamment dans des domaines comme l'aéronautique, le développement logiciel ou les télécommunications. Au fil des années, des outils technologiques ont ouvert de nouvelles opportunités afin de tirer profit des ressources spécialisées dans le monde entier. Cependant, la dispersion complique encore souvent la coopération entre les membres de l'équipe.

L'ingénierie des exigences est l'un des processus pouvant être affecté par la dispersion géographique des membres d'une équipe. À ce jour, peu de recherches se sont intéressées spécifiquement à ce processus qui pourtant, joue un rôle fondamental dans la conception et le développement des systèmes. Afin d'étudier plus à fond cette question dans le contexte des équipes dispersées, une étude de terrain fut réalisée auprès de CAE, le chef de fil mondial dans le marché des simulateurs de vol et de formation. Cette recherche a consisté à étudier le processus de l'ingénierie des exigences de CAE et de l'adapter au contexte des équipes dispersées. Pour ce faire, certains outils comme la modélisation des exigences fonctionnelles à l'aide de diagrammes UML ont été utilisés.

Les résultats montrent que d'une part, le processus proposé répond aux contraintes de la dispersion de l'équipe, et que d'autre part, la modélisation des exigences fonctionnelles contribue à améliorer significativement la spécification des exigences du système. Ultimement, cet apport permet d'assurer une plus grande adéquation entre le système conçu et les besoins du client et des parties prenantes.

Mots-clés : Ingénierie des exigences, modélisation, équipe dispersée, gestion de projets

ABSTRACT

Distributed projects are increasingly dominant in high tech industries such as aerospace, software and telecommunications. Over the years, economical and technological changes have opened up new possibilities for organizations to tap into specialized resources around the world, and to gain access to emerging markets. However, distribution remains a major challenge for the collaboration in the project team.

Requirements engineering is one of the processes that can be impacted by the dispersion of the project team. Although it has a fundamental role to play in the system development, as of today, only a few studies have been conducted on this process. In order to have a better knowledge of this subject in a distributed context, a field study has been conducted in conjunction with CAE Inc., a leading provider of simulation and integrated training services for civil aviation and defence customers worldwide. The present research studied the actual CAE requirements engineering process and proposed some modifications to adapt it to distributed teams. Tools like modelling functional requirements using UML have been used to achieve this goal.

Results show that on the one hand, the proposed process complies with the constraints of the distributed project team. On the other hand, the functional requirements modelling leads to a better system requirements specification. Eventually, this helps lead to a better match between the system developed by the project team and customer needs.

Keywords : Requirements engineering, modelling, distributed team, project management

TABLE DES MATIÈRES

DÉDICACE	iv
REMERCIEMENTS	v
RESUMÉ	vi
ABSTRACT	vii
TABLE DES MATIÈRES.....	viii
LISTE DES TABLEAUX.....	xi
LISTE DES FIGURES	xii
LISTE DES ABRÉVIATIONS	xiii
LISTE DES ANNEXES.....	xiv
INTRODUCTION.....	1
CHAPITRE 1 : CONTEXTE GENERAL DE LA RECHERCHE : L'INGENIERIE	
DES EXIGENCES.....	4
1.1 Les exigences dans un contexte de projets d'ingénierie	5
1.1.1 Formulation des exigences.....	6
1.1.2 Propriétés des exigences	8
1.1.3 Spécification des exigences du système.....	9
1.1.4 Exigences tacites	10
1.1.5 Niveau d'exigence et traçabilité.....	10
1.1.6 Types d'exigences.....	13
1.2 L'ingénierie des exigences.....	14
1.2.1 Le processus d'ingénierie des exigences.....	15
1.2.2 Acquisition des exigences	16
1.2.3 Analyse des exigences.....	16
1.2.4 Négociation	17
1.2.5 Spécification.....	17
1.2.6 Allocation.....	17

1.2.7	Vérification	18
1.2.8	Suivi	18
1.3	Outils	18
1.3.1	Modélisation.....	19
1.3.2	Technologies de l'information supportant le processus d'IE.....	24
1.4	Intégration	27
1.5	Liens entre l'ingénierie des exigences et d'autres domaines spécifiques aux projets de développement de systèmes	28
1.5.1	Disciplines.....	29
1.5.2	Domaines spécifiques.....	33
1.6	Synthèse	34
CHAPITRE 2 : PROBLÉMATIQUE SPÉCIFIQUE ET OBJECTIFS DE RECHERCHE		36
2.1	Conditions menant à une IE distribuée	37
2.2	Avantages.....	37
2.3	Conditions favorisant la mise en place.....	38
2.4	Défis de l'ingénierie des exigences dans un contexte distribué.....	39
2.4.1	Défis inhérents à l'IE	39
2.4.2	Défis de l'IE liés au contexte distribué	40
2.5	Outils de communication	43
2.6	Synthèse de la recherche dans le domaine de l'IE distribuée	44
2.7	Objectifs de la recherche.....	45
CHAPITRE 3 : ASPECTS MÉTHODOLOGIQUES DE LA RECHERCHE.....		47
3.1	Description du milieu à l'étude	47
3.1.1	Choix du terrain: un projet de l'entreprise CAE	47
3.1.2	État de l'existant : l'ingénierie des exigences chez CAE.....	51
3.1.3	Pistes explorées dans la phase initiale de la recherche	54
3.2	Le volet empirique du projet de maîtrise chez CAE : les grandes étapes	55

3.2.1	1 ^{ère} grande étape : étude du processus actuel	56
3.2.2	2 ^{ème} grande étape : proposition d'un processus.....	57
3.2.3	3 ^{ème} grande étape : expérimentation du processus sur un projet réel	57
3.2.4	4 ^{ème} grande étape : évaluation des résultats.....	58
3.3	Synthèse	58
CHAPITRE 4 : PRÉSENTATION DES RÉSULTATS		60
4.1	L'approche générale proposée	60
4.2	Documentation	61
4.2.1	Cartographie du processus	61
4.2.2	Dictionnaire.....	64
4.2.3	Matrice d'assignation des responsabilités.....	65
4.3	Outils.....	66
4.3.1	VPN.....	67
4.3.2	Moyens de communication	68
4.3.3	Choix des outils de support pour l'IE	68
4.4	Évaluation du processus proposé	69
4.4.1	Résultats	69
4.4.2	Analyse des résultats.....	75
CHAPITRE 5 : DISCUSSION.....		80
5.1	Contributions théoriques	80
5.2	Contributions pratiques	80
5.3	Limites de la recherche	81
5.4	Avenues de recherche	82
CONCLUSION		84
BIBLIOGRAPHIE.....		86
ANNEXES		92

LISTE DES TABLEAUX

Tableau 1.1 : Fonctionnalités des logiciels supportant l'IE.....	27
Tableau 4.1 : Correspondance entre les grandes étapes du volet empirique de la recherche et des résultats.....	60
Tableau 4.2 : Conventions utilisées dans la cartographie de processus	64

LISTE DES FIGURES

Figure 1.1: Exemple d'utilisation de la traçabilité entre plusieurs documents dans un processus de développement de produit en V	13
Figure 1.2: Processus d'ingénierie des exigences	15
Figure 1.3: Illustration d'un cas d'utilisations pour un système à concevoir : un distributeur de billets de banque (guichet automatique)	22
Figure 1.4: Diagramme d'activités pour le cas d'utilisation "s'identifier"	23
Figure 1.5: Diagramme d'états transitions d'un distributeur de billets	24
Figure 1.6 : Intégration de l'outil de modélisation et de l'outil de support pour l'IE	28
Figure 1.7 : Cycle de développement en V	33
Figure 2.1: Nature des problèmes liés à la distance et implication dans le processus d'IE..	41
Figure 2.2 : Problématique, champs de recherche et chercheurs notables.....	45
Figure 3.1 : Composants d'un simulateur d'avion	49
Figure 3.2 : Échéancier du volet empirique du projet de maîtrise	56
Figure 4.1 : Intégration de l'outil de modélisation et de l'outil de support pour l'IE	62
Figure 4.2 : Extrait de la cartographie du processus d'IE de CAE.....	63
Figure 4.3 : Un exemple de description d'une activité dans le dictionnaire d'activités...	65
Figure 4.4 : Extrait de la matrice d'assignation des responsabilités.....	66
Figure 4.5: Topologie du VPN.....	67
Figure 4.6 : Évolution du nombre d'exigences au cours du temps pour trois projets comparables.....	71
Figure 4.7 : Évolution de la volatilité des exigences au cours du temps pour trois projets comparables.....	72
Figure 4.8: Exemple du cas d'utilisation "maintenance corrective"	74
Figure 5.1: Cartographie du processus "Acquisition des exigences"	110
Figure 5.2: Cartographie du processus "Analyse des exigences"	119
Figure 5.3: Cartographie du processus "revue des exigences système"	126

LISTE DES SIGLES ET ABRÉVIATIONS

ASQ	<i>American Society for Quality</i>
COTS	Prêt à l'usage (<i>Commercial Off The Shelf</i>)
IE	Ingénierie des Exigences
LAN	Réseau local (<i>Local Area Network</i>)
MRD	Document des exigences du marché (<i>Market Requirements Document</i>)
MTBF	Temps moyen entre deux pannes (<i>Mean Time Between Failure</i>)
MTTR	Temps moyen de réparation (<i>Mean Time To Repair</i>)
OCD	Document des concepts opérationnels (<i>Operational Concepts Document</i>)
PMI	<i>Project Management Institute</i>
RAM	Matrice d'assignation des responsabilités (<i>Responsibility Assignment Matrix</i>)
SBS	Décomposition du système (<i>System Breakdown Structure</i>)
SRR	Revue des exigences système (<i>System Requirements Review</i>)
SRS	Spécification des exigences système (<i>System Requirements Specification</i>)
VPN	Réseau privé virtuel (<i>Virtual Private Network</i>)
WAN	Réseau étendu (<i>Wide Area Network</i>)

LISTE DES ANNEXES

ANNEXE A : EXTRAIT DU DOCUMENT MST-PRS-004 «REQUIREMENTS MANAGEMENT AND DEVELOPMENT PROCESS ».....	92
ANNEXE B : CAS D'UTILISATIONS DE LA MAINTENANCE D'UN SIMULATEUR D'AVION.....	94
ANNEXE C : PROCESSUS D'INGÉNIERIE DES EXIGENCES PROPOSÉ.....	109

INTRODUCTION

L'industrie fait de plus en plus appel à des équipes dont les membres sont distribués géographiquement pour mener à bien des projets qui deviennent de plus en plus complexes et risqués, notamment dans des domaines comme l'aéronautique, le développement logiciel ou les télécommunications. Au fil des années, des outils technologiques ont ouvert de nouvelles opportunités afin de tirer profit des ressources spécialisées dans le monde entier. Cependant, la dispersion complique encore souvent la coopération entre les membres de l'équipe.

Ce projet traite principalement de cette problématique de dispersion appliquée à l'ingénierie des exigences. Ce processus a pour but de découvrir, analyser et spécifier les attentes des parties prenantes afin que la solution proposée au client réponde à ses besoins. Bien qu'on puisse reconnaître que l'ingénierie des exigences soit affectée par la dispersion des acteurs, il semble que très peu de recherches aient été menées jusqu'à maintenant sur le sujet. Pourtant, il s'agit d'un processus jugé essentiel pour le design de systèmes complexes.

Dans le cadre de cette recherche, une étude de terrain fut réalisée auprès de CAE, le chef de fil mondial dans le marché des simulateurs de vol et de formation. Les objectifs de l'étude étaient d'étudier les problèmes actuels de l'industrie dans le domaine de l'IE distribuée, développer et évaluer des solutions dans le but de répondre aux problèmes, expérimenter ces solutions sur un projet réel, apporter une expérience pratique sur un sujet encore peu documenté dans la littérature actuelle.

Afin d'atteindre ces objectifs, Cette étude s'est déroulée en quatre grandes étapes :

- **l'étude du processus actuel** : la 1^{ère} grande étape, qui consistait à se familiariser avec le processus de CAE et à identifier les activités sensibles à la dispersion des membres de l'équipe de projet d'une part, et de rechercher les zones d'opportunités pour la modélisation des exigences fonctionnelles avec UML.
- **la proposition d'un processus** : la 2^{ème} grande étape qui intègre des solutions aux problèmes liés à la distribution de l'équipe de projet et identifiés dans l'étape précédente. La proposition de processus intègre également des changements pour intégrer la modélisation des exigences fonctionnelles.
- **l'expérimentation du processus proposé** : la 3^{ème} grande étape consiste à expérimenter le processus proposé sur un projet de CAE, qui permet de mettre le ce processus à l'épreuve de la réalité de l'industrie. Certains indicateurs clés comme le nombre d'exigences et la volatilité des exigences ont été mesurés pendant cette étape.
- **l'évaluation des résultats** : la 4^{ème} grande étape qui analyse les résultats des mesures effectuées pendant l'expérimentation. Les livrables comme la spécification des exigences du système ou encore la documentation du processus proposé permettront également d'évaluer ce processus.

Chacune de ces étapes sera décrite et détaillée à travers cinq chapitres bien définis.

Le premier chapitre est destiné à familiariser le lecteur avec le contexte général de la recherche, à savoir l'ingénierie des exigences. La première partie définira ce que sont les exigences dans un contexte de projet, pour pouvoir définir ce qu'est l'ingénierie des exigences dans la seconde partie. La troisième partie décrit les outils comme la modélisation ou encore les technologies de l'information, auxquels l'ingénierie des exigences fait appel. La quatrième partie montre une vue d'ensemble qui intègre les concepts vus dans les parties précédentes. Enfin, la cinquième partie montre les liens qu'il existe entre l'ingénierie des exigences et d'autres disciplines et domaines spécifiques.

Le second chapitre traite de la problématique spécifique, la distribution de l'équipe de développement, et des objectifs de recherche. Dans un premier temps, les

conditions qui mènent à une ingénierie des exigences distribuée seront explorées. La seconde partie décrit les avantages d'une telle situation. La partie suivante montre les défis rencontrés par l'ingénierie des exigences dans un contexte distribué. Les outils de communication qui sont utilisés sont le sujet de la cinquième partie. La sixième partie donne un état de la recherche dans ce domaine, ce qui permet de définir les objectifs de la recherche dans la septième partie.

Les aspects méthodologiques de la recherche sont le sujet de la troisième partie. La première partie décrit le milieu de l'étude, alors que la seconde partie s'intéresse aux grandes étapes du volet empirique du projet de recherche chez CAE.

Le quatrième chapitre présente les résultats de l'étude empirique menée chez CAE. La première partie explique l'approche générale proposée. La seconde partie fait le détail de la documentation qui a été produite pour le processus proposé. La partie suivante montre les outils qui ont été sélectionnés pour supporter ce processus. La dernière partie présente les résultats des mesures qui ont été prises au cours de l'expérimentation du processus proposé sur un projet de CAE.

Le cinquième et dernier chapitre discute des contributions théoriques et pratiques de cette recherche respectivement dans les première et deuxième parties. La partie suivante discute des limites de cette recherche et enfin la quatrième partie discute des avenues de recherche.

CHAPITRE 1 :

CONTEXTE GENERAL DE LA RECHERCHE :

L'INGENIERIE DES EXIGENCES

La conception de systèmes complexes (avions, systèmes de télécommunications, etc.) représente un défi de taille, tant sur le plan technique que sur le plan de la gestion. Pour y répondre de manière efficace, les entreprises mettent en place des processus de travail qui assurent, d'une part, l'intégration parfaite des composants conçus et fabriqués par une multitude d'acteurs, et d'autre part, l'atteinte des objectifs en termes d'échéancier, de qualité, et de coûts. Il en résulte un ensemble très complexe de processus s'imbriquant les uns aux autres. L'un de ces processus, l'ingénierie des exigences (IE), est au cœur même de la gestion des besoins du client et constitue le thème principal de ce mémoire.

De façon très simplifiée, on peut dire que l'IE a pour but de comprendre les besoins du client et de les spécifier de manière explicite afin que le système développé réponde à toutes ses attentes.

Ce chapitre présente une introduction au domaine de l'ingénierie des exigences. Il décrit d'abord ce que sont les exigences, les différents types d'exigences qui existent, la façon de les formuler et de les gérer. Plus spécifiquement, le chapitre présente la terminologie associée au domaine, explique son origine, et décrit les principales activités qui constituent le processus d'IE. Deux outils de support à l'IE y sont également présentés: la modélisation des exigences et les technologies de l'information (logiciels). À la fin du chapitre, l'IE est mise en relation avec d'autres domaines importants liés aux

projets de conception de systèmes tels que la gestion de projets et l'ingénierie de systèmes.

1.1 Les exigences dans un contexte de projets d'ingénierie

Pour définir ce qu'est l'IE, il faut d'abord définir ce que sont les exigences. Une première définition simple est proposée par l'auteur Harwell [1] lorsqu'il suggère que:

« Si c'est quelque chose qui doit être fait, transformé, produit ou fourni, c'est une exigence »¹

Bien sûr, il existe des définitions plus précises dans la littérature spécialisée :

- dans la norme IEEE Std 1220-2005 [2], une exigence est une spécification identifiant une aptitude, une caractéristique physique ou un facteur de qualité qui devra être respecté ou atteint par le produit ou le processus et pour lequel il faudra trouver une solution¹;
- pour l'EIA (*l'Electronic Industry Alliance* est une alliance commerciale représentant les différents domaines de l'industrie électronique aux États Unis) [3], une exigence est une caractéristique qui identifie le niveau requis d'accomplissement afin d'atteindre un objectif spécifique pour un lot de conditions donné.¹;
- pour l'AFIS (*l'Association Française d'Ingénierie Système*) [4], une exigence prescrit une aptitude, une caractéristique, ou une limitation du système, d'un de ses constituants ou encore d'un produit ou d'un processus contribuant à leur cycle de vie.

Ainsi, une exigence est une expression prescriptive des attentes du client mais également des autres parties prenantes, en termes de besoins et non de solutions, et qui sert de commun entendement entre eux sur les caractéristiques et fonctionnalités du système qui sera développé. L'énoncé suivant est celui donné par la norme IEEE Std

¹ Traduction libre

1233-1998 [5] et sera exploré dans la section suivante. Il constitue, par exemple, une exigence type pour un système de transport ferroviaire :

« Le système doit déplacer des personnes entre Los Angeles et New York à une vitesse de croisière de 200 km/h, et une vitesse maximale de 300 km/h. »

1.1.1 Formulation des exigences

La formulation des exigences représente toujours une étape cruciale dans un projet d'ingénierie puisqu'elles déterminent en grande partie la portée du dit projet. En effet, c'est à partir des exigences que sont déterminés les tâches et le niveau d'effort requis pour satisfaire les besoins d'un client. Par ailleurs, la formulation des exigences représente également un défi de taille puisque les clients d'un projet ne sont pas toujours en mesure de les exprimer de façon précise et « utilisable » par les concepteurs. Lorsque le client spécifie des exigences dans un appel d'offre par exemple, il va le faire en ses propres mots, dans l'ordre qui lui semble bon. Ainsi, les exigences venant des clients ne peuvent pas être traitées directement car elles diffèrent sur bien des points (plan, complétude, niveau de détail, rigueur). Celles-ci sont appelées des exigences brutes (*raw requirements*). Éventuellement, ces exigences brutes sont précisées et détaillées suivant des règles précises.

De façon générale, les pratiques reconnues proposent de formuler les exigences suivant le même modèle afin de s'assurer que toutes les informations requises sont incluses afin de pouvoir d'une part connaître en détail l'exigence et d'autre part savoir quand cette exigence est acquise par le système. La norme IEEE Std 1233-1998 [5], par exemple, fournit des indications pour rédiger des exigences « bien formulées » (*well-formed*).

« Une exigence bien formulée est l'expression d'une fonctionnalité (ou d'une capacité) du système qui peut être validée, qui doit être implantée

dans le système afin de résoudre un problème ou atteindre un objectif du client, et qui est qualifiée par des conditions mesurables et limitée par des contraintes. »¹

Dans cette définition, on peut noter qu'une exigence bien formulée exprime trois aspects importants de la fonctionnalité désirée, soient :

- **Une dimension de capacité**, qui décrit une fonction, ou ce que le système doit être capable de faire. La capacité doit être spécifiée de façon suffisamment abstraite pour ne suggérer aucune solution. C'est l'élément essentiel d'une exigence.
- **Une ou des conditions**, critères qui permettent de mieux qualifier la capacité. Elles sont mesurables et permettent de vérifier, de valider l'exigence.
- **Une ou des contraintes**, qui restreignent volontairement les solutions disponibles. Les contraintes peuvent être imposées par le client (ex : utilisation d'un langage de programmation précis), l'environnement (ex : température), les lois (ex : limitation de vitesse, du bruit), etc.

L'exemple suivant, issu de la norme IEEE Std 1233-1998 [5] tiré du domaine des transports, illustre ces trois aspects :

L'attente « déplacer des personnes entre Los Angeles et New York » est l'expression d'une exigence brute (*raw requirement*), qui pourrait servir de base à l'établissement d'une exigence bien formulée. Suivant les trois dimensions mentionnées, il faut donc introduire contraintes et conditions à cette formulation. De façon plus exacte, on pourrait donc établir qu'une exigence contient les trois éléments suivants:

- **Capacité** : « déplacer des personnes entre Los Angeles et New York »
- **Condition** : « vitesse de croisière doit être de 200 km/h »
- **Contrainte** : « la vitesse ne doit pas dépasser 300 km/h »

¹ Traduction libre

Ainsi, l'exigence bien formulée se lirait ainsi :

« Le système doit déplacer des personnes entre Los Angeles et New York à une vitesse de croisière de 200 km/h, et une vitesse maximale de 300 km/h. »

On peut cependant noter que cet exemple contient certaines failles. Par exemple, le terme « vitesse de croisière » n'est pas défini, ou encore les conditions de voyage des passagers ne sont pas incluses (sécurité, confort). Dans un cas réel, il faudrait donc définir ces autres aspects (définitions, etc...) ou formuler des exigences additionnelles.

1.1.2 Propriétés des exigences

Tel qu'indiqué précédemment, les exigences sont l'expression d'un besoin et non d'une solution. Au début d'un projet, il importe donc de documenter le problème ou l'objectif du client sous forme d'exigences qui n'écartent pas de solution a priori, laissant ainsi aux développeurs, experts et ingénieurs, le soin d'envisager toutes les options possibles.

L'auteur Wiegers [6] et la norme IEEE Std. 1233-1998 [5] proposent une liste de propriétés (attributs) qui devraient caractériser l'ensemble des exigences dans un projet d'ingénierie. Selon eux, les exigences devraient être :

- **Abstraites** : les exigences doivent être l'expression d'un besoin et non d'une solution.
- **Univoques** : les exigences doivent être formulées de façon à ce qu'elles n'aient qu'une seule signification possible. En effet, les langues parlées induisent beaucoup d'ambiguïté dans les phrases. (par exemple, il n'est pas rare qu'un mot ait plusieurs significations). Pour éviter l'ambiguïté, il est possible d'utiliser un langage formel plutôt que d'utiliser une langue parlée comme l'anglais ou le français où les ambiguïtés sont fréquentes (par exemple un même mot peut avoir plusieurs sens).
- **Nécessaires** : chaque exigence doit exprimer un besoin du client. Il faut donc s'assurer que les besoins du client, des parties prenantes ou que des contraintes

sont à l'origine des exigences. (La manière de capturer l'information sur l'origine des exigences sera développée dans le chapitre 1.1.5).

- **Uniques** : un même besoin ne doit être exprimé qu'une seule fois. D'une part, cela permet de réduire le nombre d'exigences à gérer. D'autre part, des exigences redondantes peuvent mener à des conflits d'exigences (par exemple, lors des mises à jour)
- **Testables** : il doit être possible de démontrer qu'une exigence est respectée, soit par un test, une analyse ou une démonstration.
- **Complètes** : les exigences doivent décrire de façon complète les fonctionnalités du système. Si certains éléments sont encore indéterminés, il est possible de marquer les exigences en question afin de souligner ces lacunes, et de les compléter ultérieurement, à l'issue d'une réunion avec les parties prenantes.
- **Réalisables** : les exigences spécifiées doivent être réalisables. Les prototypes, preuves de concept ou encore le développement incrémental sont des approches qui permettent d'estimer la faisabilité des exigences. L'évaluation de la faisabilité peut aussi être confiée à un expert.
- **Pondérées** : les exigences doivent être priorisées car elles n'ont pas toutes la même importance. Le gestionnaire de projet saura grâce à cette pondération quelles exigences sont critiques. Dans un cas de retard ou de dépassement de budget, il pourra par exemple se concentrer sur les exigences critiques et laisser de côté les autres qui n'ont pas de répercussions majeures sur le système ou sur l'acceptation.

1.1.3 Spécification des exigences du système

L'ensemble des exigences d'un système sont rassemblées dans un document appelé « spécification des exigences du système » (*System Requirements Specification*, SRS). Celui-ci est approuvé par le client et aura préséance sur les documents précédents (documentation technique du produit *Technical Product Documentation*, par exemple). Il décrit toutes les exigences fonctionnelles et non fonctionnelles relatives au système, ainsi que leur méthode de vérification. Il existe un modèle de SRS dans la norme IEEE Std. 1233-1998. Les exigences qui ne sont pas relatives au système sont rassemblées dans d'autres documents comme l'énoncé des travaux (*Statement of Work*).

1.1.4 Exigences tacites

Les exigences recueillies auprès des parties prenantes ne sont généralement pas exploitables telles quelles. En plus de la formulation et des caractéristiques citées précédemment, les exigences brutes peuvent présenter des lacunes qu'il faut combler. Il arrive que le client omette de spécifier des exigences cruciales, soit parce qu'il ne connaît pas le domaine et qu'il n'en connaissait pas l'existence, ou au contraire parce qu'il le connaît très bien, et qu'il suppose que ces exigences sont évidentes pour les concepteurs. Dans l'exemple précédant sur les systèmes ferroviaires (déplacement Los Angeles – New York), l'exigence porte sur la vitesse. Cela implique « évidemment » que le système doit être capable de freiner, et que la force de freinage doit être adaptée à la sa vitesse. Ces exigences non exprimées sont dites « tacites ».

1.1.5 Niveau d'exigence et traçabilité

La découverte d'exigences tacites, la décomposition des exigences brutes et la compréhension des vrais besoins des parties prenantes mettent au jour de nouvelles exigences plus détaillées. On dit que les exigences brutes sont donc « dérivées », c'est à dire que l'ingénieur des exigences doit découvrir les exigences impliquées (plus détaillées, tacites ou omises) par les exigences brutes et les spécifier. En même temps, il peut détecter les exigences qui suggèrent une solution et les reformuler sous forme d'exigences, en accord avec les parties prenantes. À chaque nouvelle dérivation, on parle d'un nouveau « niveau » d'exigence. Dépendamment de la complexité du projet, de sa nature, du nombre de parties prenantes et du nombre d'itérations lors de l'analyse des exigences, le nombre de niveaux variera. Voici les quatre principaux niveaux proposés par Wiegers [6] :

- **Niveau d'exigences d'affaires** : ce sont des exigences de très haut niveau. Elles sont généralement liées aux préoccupations des dirigeants (haute direction, chef de département...) qui décident de (faire) développer le système et ont une vocation stratégique.

- **Niveau d'exigences utilisateur** : ces exigences décrivent les tâches que les utilisateurs du système devront être capables d'effectuer lorsqu'ils seront en interaction avec le système.
- **Niveau d'exigences fonctionnelles** : ces exigences décrivent les fonctions du système nécessaires pour que les usagers puissent effectuer les tâches décrites dans les exigences utilisateur.
- **Niveau d'exigences système** : ces exigences décrivent les exigences du système dans son ensemble, incluant les exigences non fonctionnelles (maintenabilité, sécurité, etc...)

On constate aisément que ces différents niveaux d'exigences sont interreliés et qu'il importe de garder un lien entre les exigences des différents niveaux. Ce lien entre exigences est appelé « traçabilité ». Il s'agit d'un concept central à la problématique des exigences dans un projet d'ingénierie. Voici quelques exemples de l'utilité de la traçabilité [6] :

- La traçabilité permet d'analyser l'impact d'un changement d'une exigence sur les autres exigences. En effet, si par exemple une exigence fonctionnelle change, seules les exigences du système qui y sont liées sont susceptibles d'être touchées. Dans l'exemple précédent (déplacement Los Angeles – New York), si l'exigence de vitesse est modifiée (exigence fonctionnelle), il se peut qu'il faille installer un moteur plus puissant (exigence système).
- La traçabilité permet également de savoir si toutes les exigences sont nécessaires. En effet, s'il n'est pas possible de trouver un lien de traçabilité d'une exigence système jusqu'aux exigences brutes, c'est que l'exigence est inutile pour le client et qu'elle n'a pas lieu d'être.
- La traçabilité permet de prouver au client que le système conçu remplit toutes ses attentes. Pour ce faire, il faut montrer que toutes les exigences brutes disposent d'un lien de traçabilité vers un élément du design.

Ces exemples utilisent des types de traçabilité différents. L'OMG (*Object Management Group*, la principale association américaine dont l'objectif est de standardiser et promouvoir le modèle objet sous toutes ses formes) [7] propose certaines recommandations sur la nature des liens à créer, mais l'utilisation et la création des liens

de traçabilité restent encore souvent à la discrétion de l'ingénieur des exigences et des analyses qu'il veut conduire. Voici quelques exemples courants de liens de traçabilité.

- **Dérive (*derive*)** : c'est le lien le plus courant, il permet de lier les exigences affiliées d'un niveau d'exigence à l'autre. C'est ce type de lien qui permet, entre autres, de faire l'analyse d'impact qui est décrit un peu plus haut dans ce chapitre.
- **Satisfait (*satisfy*)** : ce lien est créé plus tard dans le processus de développement du produit. Celui-ci représente le lien entre les exigences et les éléments du SBS (*System Breakdown Structure*) qui vont les satisfaire. Cela permet de s'assurer que toutes les exigences sont allouées (cf. cf. 1.2.6) à un élément du système, et ainsi s'assurer que le système satisfera toutes les exigences.
- **Vérifie (*verify*)** : il s'agit du lien entre une exigence et le test qui vérifie cette exigence. Ce lien est utilisé pour s'assurer que toutes les exigences sont testées.

La Figure 1.1 montre schématiquement l'utilisation de la traçabilité dans un projet de développement d'un système (le processus de développement en V sera plus amplement décrit dans le chapitre 1.5.1.2). Dans cet exemple, la traçabilité (représentée par les flèches) permet de lier les éléments de différents documents entre eux. Deux types de traçabilité y sont illustrés. Les exigences fonctionnelles sont déduites et sont « dérivées » des exigences de l'utilisateur. Les tests d'acceptation sont basés sur les exigences de l'utilisateur et permettent de les « vérifier ».

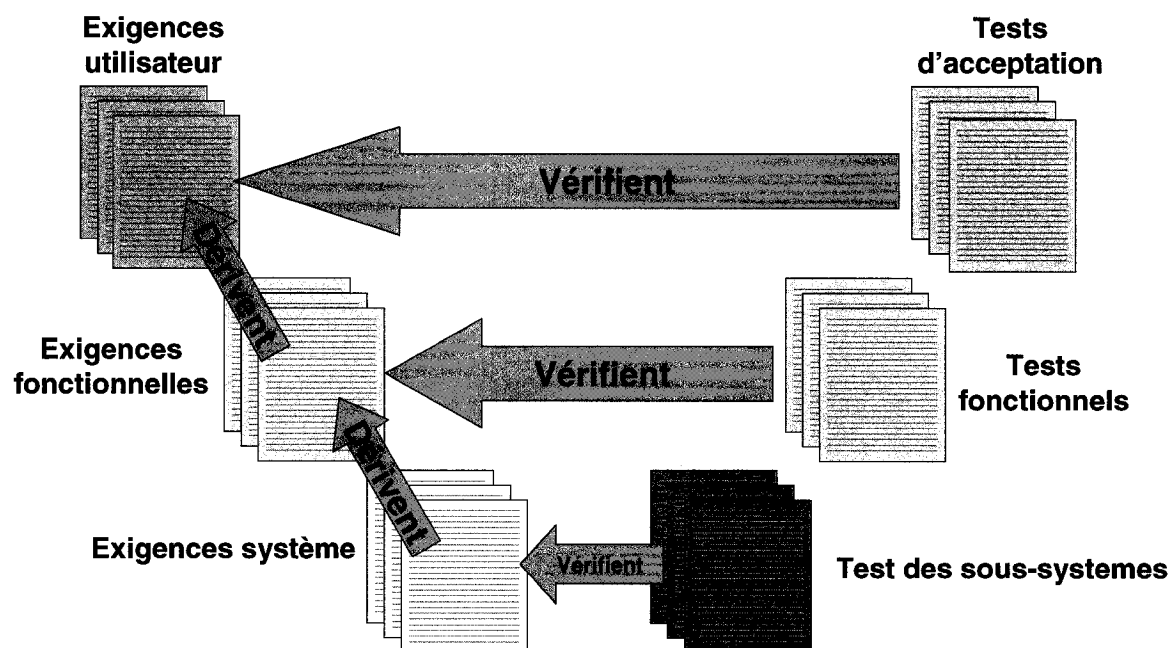


Figure 1.1: Exemple d'utilisation de la traçabilité entre plusieurs documents dans un processus de développement de produit en V¹

1.1.6 Types d'exigences

Young [8] distingue plus de 15 types d'exigences différents qu'il est possible de séparer en deux grande familles :

- **Les exigences non fonctionnelles** : ce sont les exigences qui ne réfèrent pas à une fonction du système, mais plutôt à certaines propriétés du système comme par exemple les conditions environnementales, la sécurité d'utilisation, la fiabilité, la maintenabilité, l'adaptabilité, etc... Des exigences non fonctionnelles typiques sont les valeurs de MTBF (*Mean Time Between Failure*) ou de MTTR (*Mean Time To Repair*).

¹ Traduction libre. Ce diagramme est issu des notes de l'atelier de formation de Telelogic « *Using DOORS for Requirements Management* »

- **Les exigences fonctionnelles :** ce sont celles qui spécifient les aptitudes du système. Dans l'exemple précédant, le fait de pouvoir transporter des personnes de Los Angeles à New York constitue une fonction du système à concevoir.

Ce dernier type d'exigence est directement en rapport avec le « niveau d'exigences fonctionnelles », car cette itération permet de découvrir les exigences de type fonctionnel. Elle se traduit par la génération d'un document de concepts opérationnels (*Operational Concepts Document*, OCD) qui rassemble les scénarii d'utilisation du système, c'est-à-dire quelles fonctions du système les utilisateurs auront besoin, ce qui se traduit par des exigences fonctionnelles. En revanche, les exigences non fonctionnelles peuvent apparaître à tout moment et proviennent par exemple de contraintes (par ex. lois, normes), des exigences brutes (maintenabilité, adaptabilité) ou encore de l'environnement (sécurité d'utilisation)

1.2 L'ingénierie des exigences

L'IE est la discipline qui traite de la gestion et du développement des exigences. Elle fait partie intégrante d'un plus grand ensemble, l'ingénierie des systèmes, et est présente tout au long du cycle de développement. Bien qu'en principe, elle soit applicable à bon nombre de projets de conception de produits et de systèmes, elle demeure surtout utilisée pour l'élaboration de systèmes complexes, comme par exemple dans l'aéronautique, les systèmes spatiaux, l'armement, la construction navale ou encore l'informatique.

Wiegiers [6] fait la distinction entre les termes de « gestion des exigences », « développement des exigences » et « ingénierie des exigences ». La discipline en elle-même est appelée « ingénierie des exigences », la « gestion des exigences » et le « développement des exigences » en étant deux parties distinctes.

- **La gestion des exigences :** elle consiste à « établir et maintenir un accord sur les exigences [...] avec le client ». Il s'agit donc de s'assurer de bien comprendre

les exigences du client et de s'assurer que celui-ci est d'accord avec l'interprétation que l'équipe de développement a faite des exigences. Si le client ou l'équipe de développement doit changer les exigences, cela a rapport à la gestion des exigences.

- **Le développement des exigences** : s'intéresse au contenu et à la signification des exigences. Le but du développement des exigences est de dériver les exigences brutes du client qui sont de très haut niveau en exigences de plus bas niveau. Cela permet de s'assurer qu'aucun élément du système n'est omis. Il se compose de l'acquisition, l'analyse, la spécification et la validation.

Bien que la littérature ne présente pas de consensus clair par rapport à cette terminologie, celle proposée par Wiegers [6] est utilisée dans le cadre du présent mémoire.

1.2.1 Le processus d'ingénierie des exigences

La norme IEEE Std. 1220-2005 [9] propose une description très détaillée des processus de l'ingénierie des systèmes, dont le processus d'IE. La présente section en présente une synthèse, adaptée de Hull et al. [10] selon les besoins de la recherche. La Figure 2 fournit un aperçu du processus.

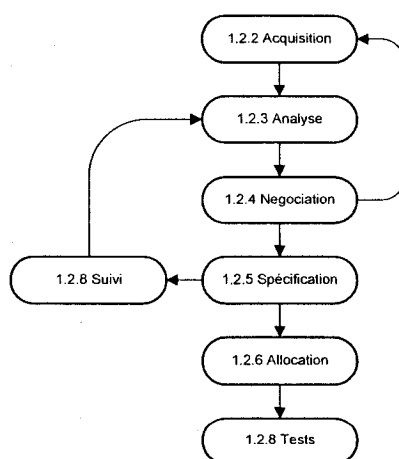


Figure 1.2: Processus d'ingénierie des exigences¹

¹ Traduction libre et adaptation de [10]

1.2.2 Acquisition des exigences

La première étape de l'IE est l'acquisition des exigences. Il s'agit de recueillir les besoins et attentes que les parties prenantes ont envers le système. Pour ce faire, Lloyd [11] et Boehm [12] ont identifié différentes techniques : les sondages, les entrevues avec le client, les remue méninges, le développement de scénarios (cas d'utilisations), le prototypage, la diffusion de questionnaires ou encore l'IE elle-même, dans le sens où de nouvelles exigences seront trouvées au cours des diverses dérivations. Le choix des techniques à utiliser est dépendant des parties prenantes (importance, distance, compétence, implication) et du projet (budget, complexité, temps, nature exploratoire), et reste à la discrétion de l'ingénieur des exigences. Pour s'assurer de capturer un maximum d'exigences, l'ingénieur des exigences peut utiliser plusieurs techniques simultanément : par exemple commencer par un remue méninges pour dégager les idées générales, puis creuser chacune des idées à l'aide de cas d'utilisations. Une des tâches les plus difficiles lors de l'acquisition des exigences est de détecter les fonctionnalités implicites (*assumptions*). Ce sont les fonctionnalités auxquelles le client s'attend, mais dont il ne parle pas, nécessairement puisqu'elle lui apparaissent « évidentes » [6]. Ce sont néanmoins parmi les plus importantes puisque si le système ne les respecte pas, le client ne sera pas satisfait.

1.2.3 Analyse des exigences

À cette étape, un bon nombre des exigences du client sont connues, mais elles ne sont pas exploitables telles quelles. En effet, celles-ci ne sont généralement pas bien formulées (tel qu'expliqué à la section 1.1.1). Elles n'ont certainement pas non plus les propriétés nécessaires pour être exploitées (ces propriétés ont été citées dans la section 1.1.2). Enfin en formulant correctement les exigences et en les corrigeant pour qu'elles soient conformes à la norme IEEE Std. 1233-1998, il se peut qu'une exigence brute se

traduise en plusieurs exigences dérivées. Le maintien de la traçabilité est très important à cette étape.

Les exigences seront également rangées soit par fonctionnalité afin de créer un arbre fonctionnel, soit par sous-système si l'architecture du système est déjà connue.

1.2.4 Négociation

Au moment de l'analyse, les exigences ont été reformulées. Il faut donc s'assurer que les parties prenantes soient d'accord avec les exigences « dérivées ». À cet instant, de nouvelles exigences peuvent apparaître et le processus reprendra alors à l'étape d'analyse pour celles-ci.

De plus, les résultats des études conduites au cours de l'analyse seront discutés. En fonction de l'analyse de risques et de coûts, le fournisseur peut proposer une solution moins risquée et/ou moins coûteuse si le client abandonne ou modifie certaines de ses exigences.

1.2.5 Spécification

Une fois que toutes les parties prenantes se sont entendues sur les exigences et leur signification, il reste à générer un SRS, document qui rassemble toutes les exigences relatives au système à développer dans un format prédéfini. Les exigences sont alors sauvegardées dans un référentiel (*baseline*) et tout changement devra être traité par un processus de suivi qui gère ces changements (le processus de suivi est expliqué dans la section 1.2.8). La norme IEEE Std 1233-1998 [5] propose un plan de SRS.

1.2.6 Allocation

Au cours de l'allocation, l'ingénieur des exigences va attribuer chacune des exigences à un élément du SBS grâce à un lien de traçabilité, de telle sorte à ce qu'il soit

possible de savoir, (1) quelles exigences sont assumées par quel élément du système, et (2) de vérifier que toutes les exigences sont bien assignées à un élément du système. Si ce n'est pas le cas, c'est que le système ne « satisfera » certainement pas certaines exigences (cela fait référence à la notion de traçabilité qui a été vue dans la section 1.1.5).

1.2.7 Vérification

La vérification vise à affirmer que le système construit est bien conforme aux exigences. Un test permet de le vérifier. L'analyse et l'inspection sont d'autres méthodes possibles de vérification. Cette activité du processus est conduite en fin de cycle de développement, cependant, les exigences système qui forment les critères de recette sont développées bien plus tôt. C'est pour cela que les exigences doivent être vérifiables.

À l'instar de l'allocation (vue dans la section 1.2.6) l'ingénieur des exigences va créer des liens de traçabilité entre les exigences et les tests à effectuer. Les exigences qui ne sont pas assignées à un test ne seront pas « vérifiées ». Le lien est appelé *vérifie*.

1.2.8 Suivi

Le suivi est utile tout au long du cycle de vie du projet. Il fait appel à la traçabilité des exigences. Ainsi, lorsqu'il y a un changement dans une exigence, on peut savoir quelles autres exigences sont touchées par ce changement et faire les autres changements nécessaires.

1.3 Outils

En pratique, le processus d'ingénierie des exigences implique plusieurs intervenants et génère une grande quantité de données qu'il faut pouvoir répertorier, traiter, échanger, ... Pour ce faire, il existe plusieurs outils commerciaux qui se

présentent sous la forme de bases de données spécialisées qui intègrent nombre de principes de l'IE, comme par exemple la traçabilité.

L'utilisation d'outils permet de faciliter ou même d'améliorer l'IE. D'une part, la modélisation permet de créer une représentation graphique des exigences et d'en faciliter, par le fait même, la compréhension et la gestion. Par ailleurs, les technologies de l'information (principalement des logiciels) automatisent certaines tâches du processus et disposent de fonctions avancées. Ces outils seront décrits dans cette partie.

1.3.1 Modélisation

La modélisation des exigences est une technique qui permet de donner une représentation graphique des exigences et de l'architecture du système à développer. Selon le moment dans le processus d'IE où l'on utilise la modélisation (acquisition des exigences, analyse, design,...), celle-ci présente les informations pertinentes grâce à différents modèles. De ce fait, l'utilisation de la modélisation au cours du processus d'IE constitue un premier pas vers l'adoption de la norme IEEE Std 1471 [13], qui suggère de montrer aux parties prenantes seulement les données pertinentes en fonction du / des rôle(s) qu'elles jouent dans le projet [14]. Modéliser les exigences consiste à faire abstraction de la complexité du système à développer et à le représenter par des dessins simples destinés à favoriser l'échange d'information.

1.3.1.1 Normes

La littérature fait référence à plusieurs normes pour la modélisation :

- i* (1993): Yu [15] étudie ce cadre de travail qui met en relation l'organisation et le système d'information. Il est constitué de deux principaux modèles. L'un se concentre sur la logique interne de l'organisation et les dépendances entre les acteurs (*Strategic Dependency*), l'autre s'intéresse aux tâches que les acteurs veulent réaliser avec le système et comment ils vont les réaliser (*Strategic Rationale*).

- **ADL** : (*Architecture Description Language*) caractérisent des langages de description d'architecture spécialisés fondés sur les mathématiques. Ils ont été principalement développés dans le monde académique. Kandé explique [14] qu'ils permettent de représenter mathématiquement des systèmes logiciels et ainsi faire des analyses avancées sur le système. Ils sont toutefois complexes à comprendre et ne sont donc pas adaptés à toutes les parties prenantes.
- **EFFBD** : (*Enhanced Functional Flow Block Diagrams*) il donne une description fonctionnelle et dynamique du système. Il permet de représenter les tâches à accomplir, les liens logiques et la simultanéité et des décisions (par ex. les itérations). Conrad [16] compare ce type de modélisation au diagramme d'activités en UML / SysML qui sera présenté dans la partie 1.3.1.3.
- **UML et SysML** : sont les deux normes les plus récentes et les dont les mises à jour sont les plus fréquentes. Elles ont le potentiel de s'imposer, car elles reprennent tous les concepts et avantages que les autres normes de modélisation ont. De plus, celles-ci sont à l'initiative de l'OMG (*Object Management Group*), le principal organisme de promotion du modèle objet, soutenu par l'INCOSE (*International Council On Systems Engineering*, la plus importante association de promotion de l'ingénierie système en Amérique du Nord). UML (*Unified Modelling Language*) est un langage graphique qui rassemble pas moins de 13 diagrammes standardisés qui présentent le système sous différentes vues (utilisation, implémentation, logique, déploiement et processus). UML [17] est principalement utilisé en génie domaine logiciel. SysML (*System Modeling Language*) est une variante d'UML spécialisée pour l'ingénierie des systèmes qui permet de supporter la spécification, l'analyse, le design et la vérification et validation du système. Pour ce faire, SysML réutilise 7 diagrammes d'UML et en apporte 2 nouveaux.

Les points suivants présentent les principaux diagrammes que l'on trouve dans UML / SysML. Seuls les diagrammes dont l'utilisation a été considérée dans la présente étude seront décrits (à moins qu'il soit spécifié explicitement autrement, les points suivants s'appliquent aussi bien à UML qu'à SysML)

1.3.1.2 Cas d'utilisations

Une première façon de définir simplement les cas d'utilisations est proposée par Bittner K. et Spence I. [18] :

« Les cas d'utilisations permettent de décrire des séquences d'événements qui ensemble conduisent le système à faire quelque chose d'utile. »¹

La spécification d'UML [17] définit les cas d'utilisations comme :

« le moyen de spécifier les usages requis d'un système. Ils sont habituellement utilisés pour capturer les exigences fonctionnelles d'un système, c'est-à-dire ce que le système est supposé faire. »

Les cas d'utilisations représentent donc les concepts opérationnels ou encore les fonctions du système que les utilisateurs voudront utiliser. Ils sont au cœur de plusieurs méthodologies (par exemple *Rational Unified Process*, RUP [19]) et sont utilisés à la fois pour spécifier les concepts opérationnels et pour l'acquisition des exigences. Ils permettent enfin d'identifier graphiquement les acteurs du système. Piechocki [20] définit les acteurs comme des entités externes au système qui agissent sur le système (opérateurs ou autres systèmes).

Utilisés dans l'acquisition des exigences, ils permettent de mettre les parties prenantes dans un contexte concret d'utilisation du système à développer et s'imaginer à la place des acteurs. Cette méthode a l'avantage de permettre aux parties prenantes de mieux imaginer et verbaliser ce qu'elles attendent du système, car elles peuvent se concentrer sur un contexte précis et concret proche de leur réalité. La norme IEEE Std 1362 [21] recommande de rassembler les cas d'utilisations dans le document de concepts opérationnels (*Operational Concepts Document*, OCD).

Par exemple, si l'on considère un distributeur de billets comme système, le diagramme de cas d'utilisations pourrait ressembler à la Figure 1.3. On y voit les « acteurs » () (client, technicien, préposé) ainsi que les utilisations qu'ils peuvent faire du système. Certains cas d'utilisations qui suggèrent plusieurs alternatives peuvent à leur

¹ Traduction libre

tour être décomposés en d'autres cas d'utilisations, sinon, ils sont détaillés dans des diagrammes d'activités (qui seront détaillés dans la section suivante 1.3.1.3) ou encore des diagrammes de séquence (non traités dans cette étude).

Alexander [22] propose d'identifier également les cas de mauvaises utilisations, c'est-à-dire les cas d'utilisations pour lesquels les acteurs ont une intention hostile (*misuse case*). Par exemple, un voleur qui cherche à voler un distributeur de billets représente un cas de mauvaise utilisation.

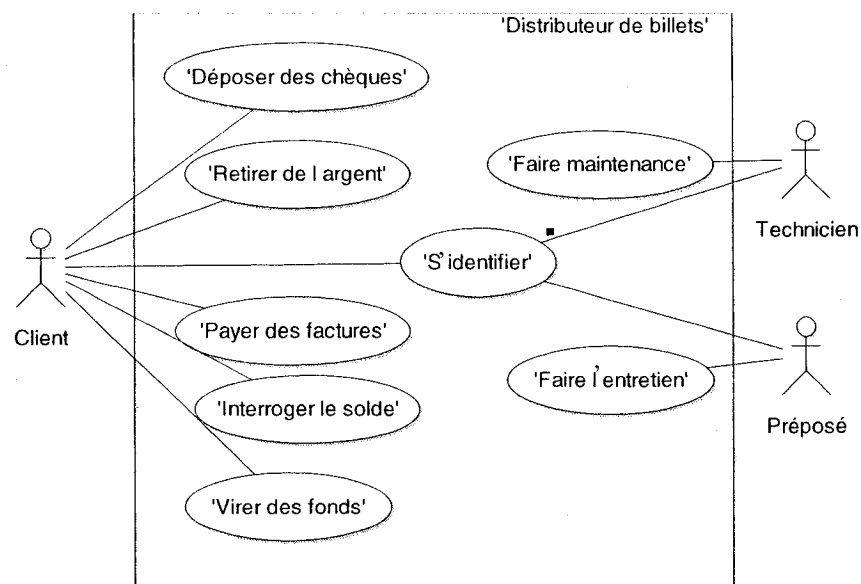


Figure 1.3: Illustration d'un cas d'utilisations pour un système à concevoir : un distributeur de billets de banque (guichet automatique)

1.3.1.3 Diagramme d'activités

Le diagramme d'activité est une vue dynamique qui permet de représenter graphiquement le déroulement d'un cas d'utilisation qui est décomposé en « activités ». Ce diagramme permet de montrer la simultanéité des activités, ainsi que de représenter plusieurs cheminements grâce à des décisions qui définissent la structure logique du

système. Enfin, les acteurs sont représentés dans des partitions (*swimlanes*) ce qui permet de savoir qui fait quoi. Les exigences qui sont dans la partition du système seront les seules que celui-ci devra être capable de faire, et ainsi ce sont les seules qui devront rester dans la spécification des exigences du système.

Le diagramme d'activités introduit la notion de début (●), de fin (●) et de décision (◇). Il s'apparente à une cartographie de processus, au détail près que l'on porte que peu d'attention aux données échangées. (le diagramme de séquence, qui ne sera pas traité dans ce document, s'intéresse à cet aspect). La Figure 1.4 montre l'exemple du diagramme d'activités pour les cas d'utilisation « s'identifier » issu de l'exemple précédent.

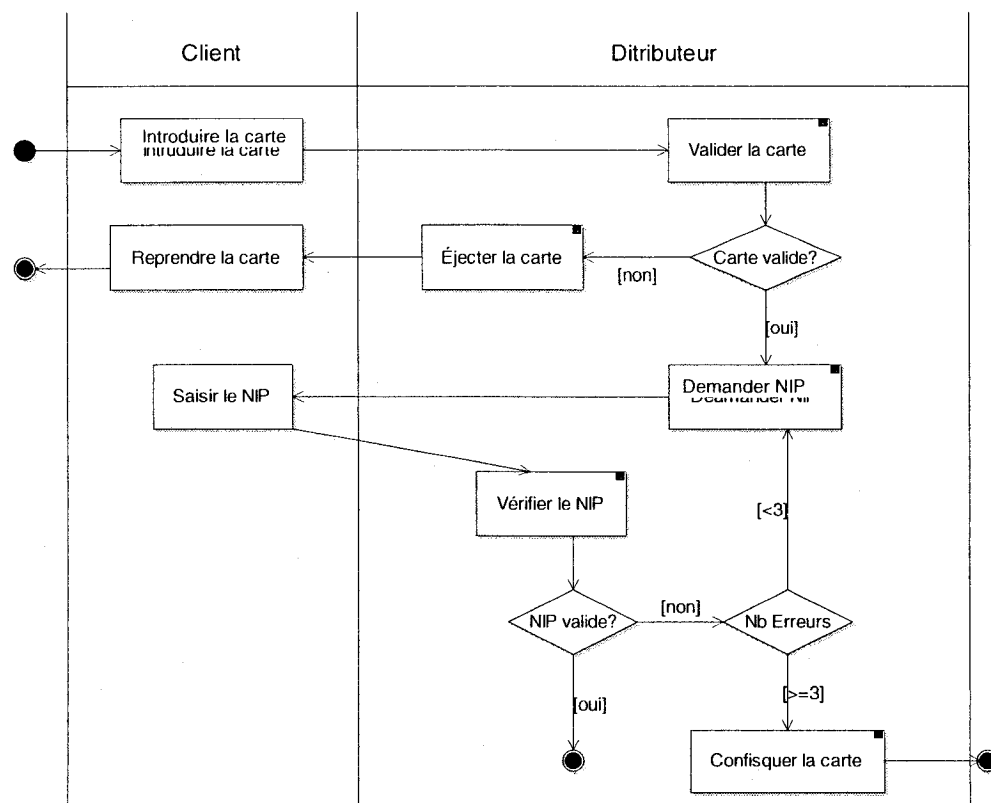


Figure 1.4: Diagramme d'activités pour le cas d'utilisation "s'identifier"

1.3.1.4 États et transitions

Ce diagramme est composé d'états, qui représentent les modes dans lesquels le système peut se trouver, ainsi que les transitions, qui sont les événements qui font passer le système d'un état à un autre.

Le modèle de SRS dans norme IEEE Std. 1233-1998 [5] dispose d'une partie (point 2.2) qui se rapporte aux états du système. Le diagramme d'états transitions est particulièrement bien adapté pour celle-ci, car il montre non seulement les états, mais également les transitions pour passer de l'un à l'autre. La Figure 1.5 montre le diagramme d'états transitions pour le système de distributeur de billets.

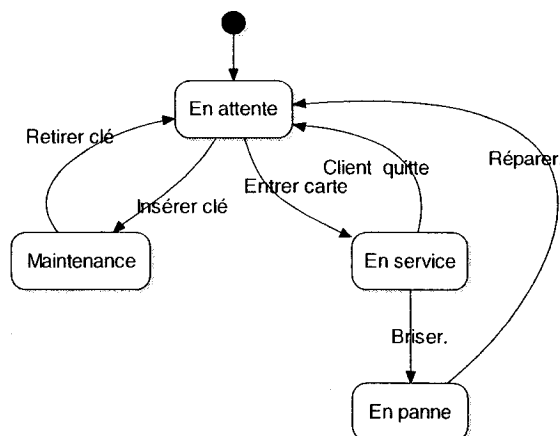


Figure 1.5: Diagramme d'états transitions d'un distributeur de billets

1.3.2 Technologies de l'information supportant le processus d'IE

Les technologies de l'information supportant le processus d'IE, plus spécifiquement les logiciels, se présentent sous la forme de bases de données dans lesquelles on peut stocker les exigences. Les exigences sont présentées dans les lignes d'un tableau et les colonnes représentent les attributs relatifs aux exigences (par exemple le numéro, la méthode de vérification, l'assignation, un commentaire, une évaluation du

risque...). Il est possible de les classer et d'appliquer une hiérarchie dans les exigences. Ce qui différencie ces outils par rapport à un simple tableur (ex : MS Excel) ou un logiciel de traitement de texte est la possibilité de créer et gérer les liens de traçabilité (qui a été expliquée dans la section 1.1.5) entre les exigences. En plus de stocker les exigences, ces logiciels proposent une foule de fonctionnalités décrites dans la partie 1.3.2.1. Les principaux logiciels disponibles en 2007 sont dans l'ordre de part de marché : DOORS de la firme Telelogic AB., RequisitePRO de IBM et CaliberRM de Borland. Dans le cadre de ce projet de maîtrise, c'est le logiciel DOORS qui fut utilisé¹.

1.3.2.1 Fonctionnalités

Dans cette partie seront décrites les principales fonctionnalités offertes par la plupart des logiciels dédiés à l'IE. Hoffman [23] les a rassemblées dans un tableau pour pouvoir évaluer les logiciels de support à l'ingénierie des exigences. Voici une liste des fonctionnalités qui y figurent :

- **Traçabilité** : les outils d'IE sont capables de créer et de gérer les liens entre les exigences, et de traiter différentes natures de traçabilité. Ils sont donc aussi capables de générer automatiquement des analyses d'impact (qui permettent d'évaluer l'impacte d'un changement dans les exigences), matrices de conformité (qui permettent de prouver que le produit développé répond à toutes les exigences formulées par le client) ou encore des matrices de traçabilité (qui permettent de représenter visuellement les liens de traçabilité entre deux documents).
- **Référentiels (*baseline*)** : pour Meinardier [24], un référentiel est « un ensemble d'exigences validées et approuvées dans le développement d'un système ». Ils permettent de prendre une image instantanée des exigences à un instant précis (en arrivant à un jalon important, lors de l'adoption du SRS, par exemple). Cela permet d'identifier les exigences qui ont changé depuis la dernière entente avec le client et de déterminer quelles exigences doivent être validées par le client. Un référentiel sert également à comparer les exigences actuelles aux exigences d'un

¹ Ce logiciel est utilisé par l'entreprise CAE (partenaire du projet) et par l'École Polytechnique. Cette dernière fait partie de l'Alliance universitaire Telelogic, ce qui facilite l'accès au produit.

jalon précédent, ou à revenir à un état antérieur des exigences. Enfin, il peut être utilisé comme base pour un autre projet. Les référentiels sont également très utiles pour la gestion de la configuration (cf. 1.5.2.1).

- **Réutilisation** : lors d'un nouveau projet, il est intéressant de pouvoir réutiliser un projet antérieur similaire, en partie ou au complet. Les logiciels d'IE permettent de copier un projet et ainsi d'en réutiliser les parties semblables. Cela est cependant possible seulement si l'organisation maîtrise déjà la réutilisation sans avoir recours à un logiciel d'IE.
- **Flux des travaux** : il s'agit d'un support automatique pour suivre le processus d'IE. Celui-ci est programmé dans le logiciel, et l'utilisateur est assisté dans chacune des tâches du processus d'IE. (exemple : lors d'un changement, la personne responsable de l'exigence modifiée reçoit automatiquement un message lui demandant d'approuver ou rejeter le changement.)
- **Droits d'administration** : ceux-ci définissent quels membres du projet sont autorisés à voir ou modifier quelles informations. Cela permet de s'assurer de la confidentialité des données. Ainsi, les utilisateurs sont authentifiés avec un mot de passe et ont un accès correspondant à leur rôle.
- **Historique** : cette fonctionnalité garde une trace de toutes les opérations qui ont été effectuées. Les outils les plus avancés permettent également de les annuler afin de revenir à un état antérieur d'une exigence, après une erreur par exemple.

Toutes les fonctionnalités présentées ici sont intéressantes pour une équipe responsable de l'IE car elles facilitent et améliorent grandement la prise en charge d'activités. Le Tableau 1 propose une catégorisation de ces fonctionnalités selon la contribution qu'elles ont au processus d'IE. Certaines fonctionnalités (colonne « support et simplification ») simplifient grandement le processus alors que d'autres (colonne « amélioration ») seraient difficiles, voire impossibles à réaliser sans ces outils logiciels.

Tableau 1.1 : Fonctionnalités des logiciels supportant l'IE

Support et simplification du processus d'IE	Amélioration du processus d'IE
Traçabilité	flux des travaux
Référentiels	Droits d'administration
Réutilisation	Gestion de l'équipe (dispersion des acteurs)
	Historique

1.4 Intégration

La modélisation des exigences et les technologies de l'information tels que présentés aux sections précédentes sont deux outils complémentaires. La Figure 1.6 montre comment ils s'intègrent. Les concepts opérationnels sont issus des diagrammes comportementaux¹ et sont décrits dans le document des concepts opérationnels (« OCD »). Les éléments du système issus des diagrammes structuraux², non traités dans cette étude, et sont décrits dans la description du design du système (« SyDD »). Structure et comportements convergent au niveau de la spécification des exigences du système (« SyRS »).

¹ Les diagrammes comportementaux sont ceux qui décrivent les fonctionnalités du système développé, par exemple le diagramme de cas d'utilisation ou le diagramme d'activité qui sont décrits dans les parties 1.3.1.2 et 1.3.1.3.

² Les diagrammes structuraux sont ceux qui décrivent l'architecture du produit développé, par exemple le diagramme de classe, le diagramme de structure composite et diagramme de déploiement.

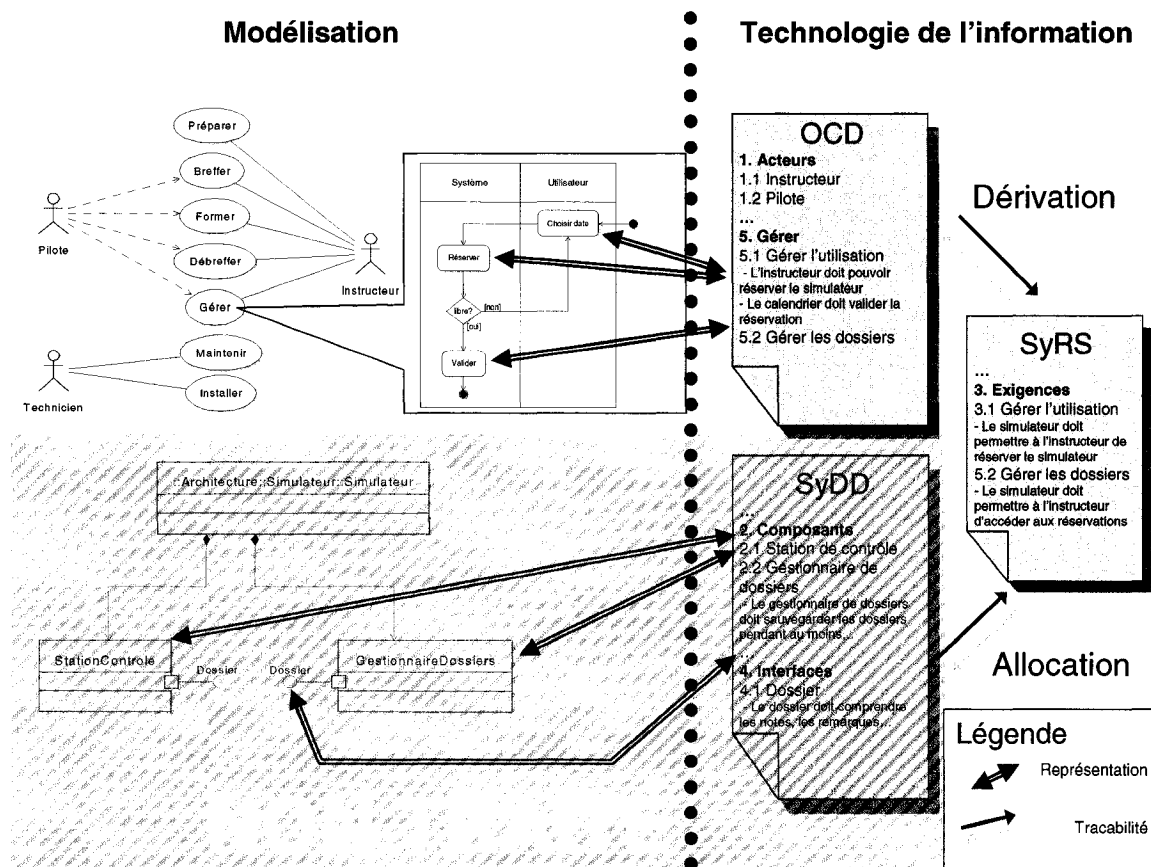


Figure 1.6 : Intégration de l'outil de modélisation et de l'outil de support pour l'IE

1.5 Liens entre l'ingénierie des exigences et d'autres domaines spécifiques aux projets de développement de systèmes

La plupart de la littérature sur l'IE porte sur des projets de développement logiciel. Certains articles décrivent cependant comment on peut étendre les principes de l'IE pour les appliquer à n'importe quel type de système en général. On peut également s'en servir dans un contexte de gestion de projet, afin de ne pas oublier de tâches. Daneva et Rolland [25, 26] montrent un exemple d'utilisation de l'IE pour l'implantation d'un ERP ou encore Atle Gulla [27] explique comment l'IE a été utilisée pour la réingénierie de processus.

Cette section identifie quelques liens importants entre l'IE et deux disciplines, soient la gestion de projets, et l'ingénierie système ainsi qu'avec deux domaines spécifiques, soient la gestion de la configuration et les systèmes de gestion de données techniques. Ces différents domaines de connaissances et d'application se trouvent souvent interreliés dans le cadre des activités d'une entreprise. Toutefois, selon la nature des systèmes développés, ou des pratiques industrielles en vigueur, le degré d'intégration des processus sous-jacents peut varier.

1.5.1 Disciplines

1.5.1.1 Lien avec la gestion de projet

Le PMI (*Project Management Institute*, le principal organisme qui établit les standards en gestion de projet) définit la gestion de projet comme « l'application de connaissances, de compétences, d'outils et de techniques aux activités du projet afin d'en respecter les exigences. » L'IE a donc un lien très fort avec la gestion de projet. Cependant, le Corpus des connaissances en management de projet [28] (*PMBOK, Project Management Body of Knowledge*, qui rassemble les bonnes pratiques en matière de gestion de projet) ne traite pas directement de l'IE, mais y fait référence dans certains chapitres :

- **Qualité** : L'ASQ (*American Society for Quality* qui est l'un des principaux organismes de promotion de la qualité en Amérique du Nord) définit la qualité comme « le degré de conformité des caractéristiques d'un produit aux exigences¹ » [29]. Une des tâches dans la gestion de la qualité est de transformer les attentes des parties prenantes en exigences grâce à l'analyse des parties prenantes (expliquée dans le point suivant).
- **Contenu (ou portée)** : pour Turner [30], la gestion du contenu d'un projet permet de s'assurer que « le projet contient tout le travail requis, et uniquement celui-ci, pour assurer la bonne fin du projet¹ ». Cela représente un des rôles de

¹ Traduction : [27]

l'IE dans le développement d'un système. Le PMBOK fait ici référence à l'IE sous le nom d'« analyse des parties prenantes » comme outil pour la définition du contenu. Une des données de sortie sont les « exigences du projet » qui sont en fait la spécification des exigences système.

Un autre point de vue est celui de Hull [10] qui considère que la gestion de projet traite des décisions relatives aux coûts et aux échéanciers, alors que l'IE s'occupe de celles qui ont trait à la qualité.

Enfin, Davis [31] relève que la mesure de critères comme le nombre d'exigences, la volatilité des exigences, les exigences allouées, les exigences vérifiées sont de bons indices mesurables facilement et permettent par exemple d'évaluer l'avancement d'un projet.

1.5.1.2 Ingénierie des systèmes

Pour définir l'ingénierie des systèmes, il faut d'abord définir ce qu'est un système. Voici la définition que l'on trouve dans la norme ISO/IEC 15288:2002 [32] :

« Un système est un ensemble composite de personnels, de matériels et de logiciels organisés pour que leur interfonctionnement permette, dans un environnement donné, de remplir les missions pour lesquelles il a été conçu. »¹

Blanchard [3] note donc qu'un système est défini par 4 caractéristiques :

- Il est constitué **d'une combinaison complexe de ressources** (humains, matériels, logiciels, données, équipement...)
- Il fait partie d'une **hiérarchie d'autres systèmes** de plus haut niveau.
- Il est constitué d'une **hiérarchie de sous-systèmes**, qui interagissent entre eux. Chacun de ces sous-systèmes peut être considéré comme un système à part.

¹ Traduction : [23]

- Enfin, le système doit avoir une **mission** à accomplir, les exigences.

Aucun des sous-systèmes ne peut à lui seul remplir la mission du système. La solution doit faire appel au système dans son ensemble. C'est donc l'interaction des sous-systèmes qui permet au système de remplir sa mission. Ainsi, la définition des interfaces internes (entre sous-systèmes) et externes (du système) avec son environnement est un des points cruciaux de la définition d'un système. Enfin, cette hiérarchie de systèmes fait que un système est un « système de systèmes ». C'est la combinaison de la hiérarchie des sous-systèmes et la définition de leurs interfaces qui permet de faire abstraction de la complexité des niveaux sous-jacents (les sous-systèmes sont considérés comme des « boîtes noires ») et qui permet de concevoir des systèmes complexes.

Pour ce qui est de l'ingénierie des systèmes, l'AFIS (L'association française d'ingénierie système, la principale association de promotion de l'ingénierie système en France) [4] donne une première définition simple :

« L'ingénierie des systèmes (IS) est une démarche méthodologique pour maîtriser la conception des systèmes et produits complexes. »

Une autre définition plus complète est donnée par l'INCOSE (*International Council On Systems Engineering*, la plus importante association de promotion de l'ingénierie système en Amérique du Nord) :

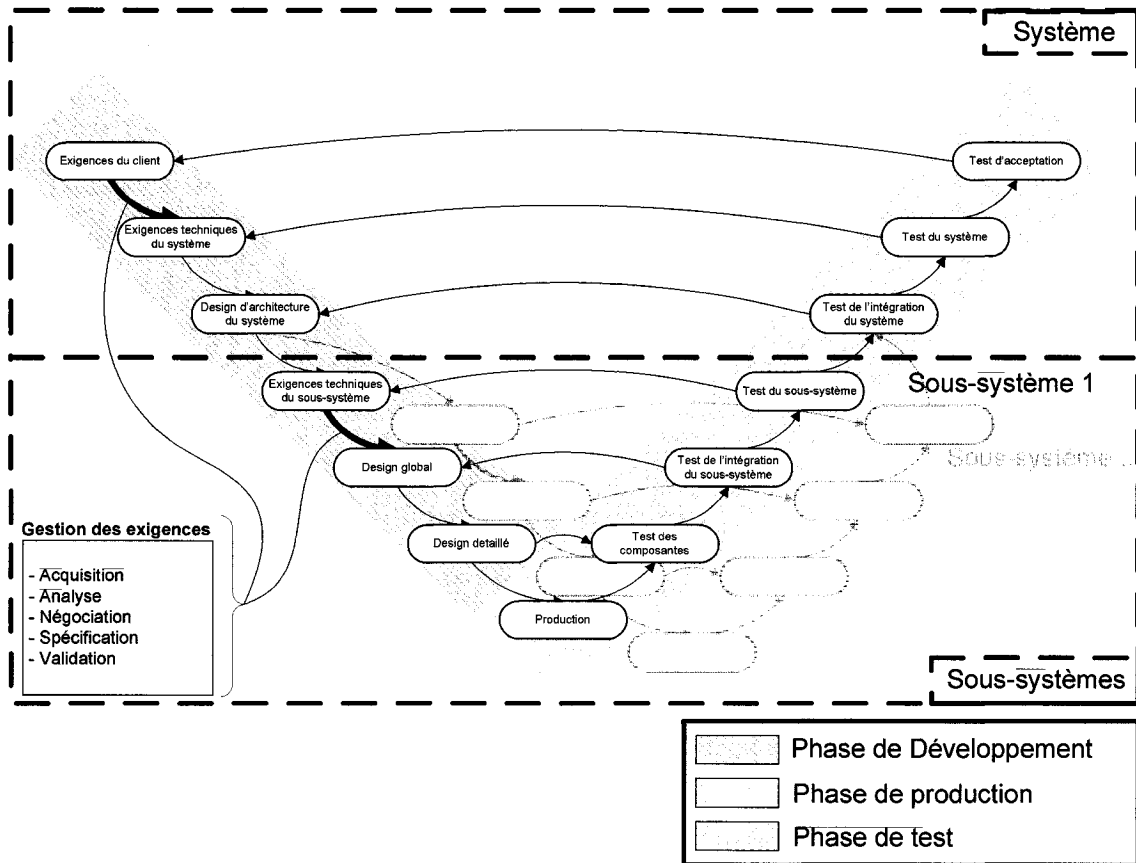
« C'est une discipline dont la responsabilité est de créer et d'exécuter un processus interdisciplinaire afin de s'assurer que les besoins des parties prenantes et du client sont satisfaits d'une manière fiable, efficace aussi bien du point de vue de l'échéancier que des coûts et qui garantit une haute qualité tout au long du cycle de vie du système¹. [...] »

1.5.1.2.1 Cycle de développement en V

Les types de cycle de développement les plus connus sont les cycles de développement en cascade, en spirale ou encore en V. Les deux premiers sont plus souvent utilisés pour développer des systèmes logiciels, alors que le dernier est plutôt utilisé pour le développement de systèmes qui ne sont pas faits que de logiciel. C'est ce dernier qui est utilisé dans cette recherche. Pour Blanchard [3], il reflète bien la stratégie *top-down* de définition du système et la stratégie *bottom-up* d'intégration et de vérification du système.

Les activités du processus d'IE présentées en détail dans la partie 1.3.1.2 interviennent à différents moments dans le cycle de développement. Comme on peut le voir dans la Figure 1.7, qui montre un cycle de développement en V, l'IE intervient au début du cycle de développement d'un système ou d'un sous-système avec les activités d'acquisition, analyse, négociation et spécification des exigences qui servent à définir le système et ce qu'il devra faire. L'IE intervient également pour définir les tests, qui seront basés sur les exigences du système. Enfin, le suivi des exigences peut intervenir à tout moment dans le cycle de développement, car il est lié aux changements apportés aux exigences.

¹ Traduction libre

Figure 1.7 : Cycle de développement en V ¹

1.5.2 Domaines spécifiques

1.5.2.1 Lien avec la gestion de la configuration

La gestion de configuration consiste à gérer les composants d'un système, notamment les modifications qui y sont apportées. Celle-ci fait partie intégrante de l'IE [6]. Cela est d'autant plus vrai dans la mesure où les systèmes d'informations supportant le processus d'IE prennent également en charge les tâches de gestion de la configuration.

¹ Traduction libre et adaptation de [3]

Une configuration est capturée dans un référentiel (*baseline*), qui est une image des exigences à un moment précis.

D'une manière générale, on peut dire que l'IE prend place lors du développement du système jusqu'à la spécification des exigences et à la création d'un référentiel (la notion de référentiel a été vue dans la section 1.2.5). Le suivi des exigences tient à proprement dit plus de la gestion de la configuration. La base de données des exigences avec les liens de traçabilité est indispensable pour la gestion de la configuration.

1.5.2.2 Lien avec les systèmes de gestion de données techniques

Les systèmes de gestion de données techniques (*Product Data Management*, PDM) permettent de stocker, gérer et contrôler toutes les informations concernant la définition, la production et la maintenance d'un produit [33]. Le principal livrable du processus d'IE, le SRS, se présente sous la forme d'une base de données. Celle-ci servira de source pour les données qui seront entrées dans le PDM. Il est souhaitable que la traçabilité soit maintenue entre les exigences et les données techniques qui sont stockées dans le PDM.

1.6 Synthèse

Les exigences sont des énoncés prescriptifs du problème des besoins des parties prenantes. La norme IEEE Std. 1233-1998 [5] qui rassemble les meilleures pratiques pour la formulation des exigences recommande que celles-ci soient abstraites, univoques, nécessaires, uniques et complètes. Les exigences sont rassemblées dans la spécification des exigences système (SRS). Il existe deux types d'exigences, les exigences fonctionnelles qui spécifient les fonctions du système, et les exigences non fonctionnelles qui représentent les exigences qui n'ont pas de lien avec une fonction du système à développer, par exemple la performance, la maintenabilité, la sécurité, etc. Une des missions de l'IE est de découvrir les exigences tacites qui sont des exigences

importantes pour le client mais qu'il n'a pas spécifiées, soit parce qu'il les considère comme acquises, soit parce qu'il en ignore l'existence. Lorsque de nouvelles exigences sont découvertes, par dérivation par exemple, un lien appelé « traçabilité » est gardé entre l'exigence parente et ces nouvelles exigences.

Le processus d'ingénierie des exigences est composé de 7 activités. L'acquisition consiste à découvrir les exigences des parties prenantes, l'analyse a pour rôle de comprendre et dériver les exigences des parties prenantes, la négociation permet de s'entendre avec le client sur l'interprétation des exigences et de proposer des alternatives, la spécification génère une spécification contractuelle de toutes les exigences du système, l'allocation assigne les exigences à des éléments de l'architecture, la génération de tests crée des cas de tests qui permettent de vérifier que les exigences sont atteintes et enfin le suivi permet de faire des changements formels aux exigences.

Les exigences fonctionnelles sont modélisées dans des diagrammes de cas d'utilisation et d'activité et sont rassemblées dans un document de concepts opérationnels (OCD).

CHAPITRE 2 :

PROBLÉMATIQUE SPÉCIFIQUE ET OBJECTIFS DE RECHERCHE

L'industrie fait de plus en plus appel à des équipes dont les membres sont distribués géographiquement pour mener des projets qui deviennent de plus en plus complexes et risqués, notamment dans des domaines comme l'aéronautique, le développement logiciel ou les télécommunications. Au fil des années, de nouveaux outils technologiques ont ouvert de nouvelles opportunités afin de pouvoir faire appel à des ressources spécialisées dans le monde entier. Cependant, la dispersion des acteurs tend à complexifier la coopération entre eux, tant du point de vue technique que non-technique. L'ingénierie des exigences (IE), sujet du chapitre précédent, se trouve elle aussi touchée par la dispersion de l'équipe.

Ce chapitre explore brièvement les raisons qui amènent les entreprises à recourir aux équipes dispersées et identifie les principaux défis associés à ce phénomène. Il traite également des facteurs déterminants propres à l'application de l'IE en contexte distribué. La dernière partie est consacrée aux outils permettant de pallier aux problèmes rencontrés par les équipes de projet qui pratiquent l'IE dans un tel contexte. Enfin, ce chapitre ouvre la voie au volet empirique de recherche qui fut réalisé.

Il est à noter que l'emploi de l'expression « IE distribuée » est utilisé pour désigner, de manière plus concise, l'ingénierie des exigences dans un environnement distribué.

2.1 Conditions menant à une IE distribuée

L'IE distribuée n'est pas toujours un choix, en effet, certaines conditions peuvent forcer une organisation à utiliser ce mode de travail, dont les suivantes, telles que mentionnées par French [34], Lloyd [11], Karolak [35], Layzell [36] et Campbell [37] :

- **Localisation** : la localisation de certaines ressources spécialisées (humaines ou matérielles) peut mener à une IE distribuée, en raison des coûts de déplacement ou de l'impossibilité de déplacer les ressources en question.
- **Partage** : les ressources spécialisées peuvent devoir travailler sur plusieurs projets à la fois. Le fait de ne pas les déplacer permet d'augmenter leur productivité.
- **Risques et coûts** : dans les projets de très grande envergure, il peut arriver qu'une seule entreprise ne puisse endosser tous les risques et fasse appel à d'autres organisations, même concurrentes, afin de faire alliance et ainsi partager les coûts et les risques.
- **Sécurité** : lorsque plusieurs organisations font équipe dans un projet, il se peut qu'un acteur ne souhaite pas avoir des membres d'une autre organisation dans ses murs pour des raisons de sécurité
- **Proximité avec le client** : dans un souci de proximité avec le client ou si le client exige un support sur place, l'équipe sera amenée à être divisée.

Toutes ces situations sont aujourd'hui assez fréquentes, particulièrement dans les grandes entreprises. Le souci d'utiliser au mieux les ressources en place (par exemple, plusieurs filiales), allié à la disponibilité des technologies de communication, tend à renforcer toujours plus ce phénomène de dispersion des activités.

2.2 Avantages

Gérer les exigences avec une équipe distribuée offre une solution aux problèmes qui viennent d'être cités. Certains auteurs comme Lloyd [11], Wiegers [6] et Campbell [37] montrent que cette situation propose en outre des avantages, notamment :

- **Créativité** : une équipe distribuée fait collaborer des personnes d'horizons et de cultures différents, ce qui présente un gros potentiel en termes de créativité.
- **Travail continu** : dans une équipe dispersée sur plusieurs continents, où les acteurs sont répartis sur plusieurs fuseaux horaires, il est possible de travailler en continu. Lorsque la journée de travail se termine dans une des localisations, la journée ne fait que commencer dans une autre et on peut prendre le relais.
- **Choix de ressources** : la possibilité de faire appel à des ressources extérieures donne au gestionnaire de projet un plus grand choix parmi les ressources qu'il peut assigner à son projet.
- **Émulation** : lorsque plusieurs sites ont une activité comparable, une concurrence bénéfique peut s'installer. L'émulation peut ainsi augmenter la productivité et la qualité, chaque site voulant avoir une bonne image auprès de l'autre. Cette émulation peut d'ailleurs être stimulée par la création de rapports portant sur des indicateurs clé de réussite.

2.3 Conditions favorisant la mise en place

Bien qu'elle puisse apporter de nombreux avantages, l'IE distribuée comporte tout de même son lot de difficultés vis-à-vis son implantation. Selon Hoffman [23], Jacobs [38] et Campbell [37], certains facteurs font que l'IE sera plus ou moins simple à mettre en place. Voici quelques-uns de ces facteurs :

- **Secteur d'activité de l'organisation** : certains secteurs d'activité sont plus propices à une approche système et ainsi à l'IE. Le secteur des services par exemple se prête moins bien à une approche système que l'industrie automobile par exemple.
- **Niveau de maturité de l'organisation** : le niveau de maturité influe sur la qualité d'adaptation des compagnies au travail distribué. En effet, deux partenaires dont les processus sont documentés, trouveront plus facilement un dénominateur commun entre leurs processus, que pour des partenaires dont les processus sont *ad hoc*.
- **Technologie** : La technologie met à disposition des outils puissants qui facilitent ou améliorent l'IE (par exemple, les outils vus dans la section 1.3.2). Cependant, là où la technologie peut être un facilitateur pour les entreprises plus matures, on

remarque qu'elle constitue un facteur de complexité supplémentaire pour les autres.

- **Composition de l'équipe** : d'une part, il est important d'avoir un coordinateur d'expérience à la tête du projet. D'autre part, la composition de l'équipe est primordiale. Souvent, les responsables de projets vont choisir les membres de leur équipe non seulement pour leurs aptitudes et leurs disponibilités, mais aussi et surtout ils vont choisir les personnes avec qui ils ont déjà travaillé et où une confiance de travail s'est développée.

2.4 Défis de l'ingénierie des exigences dans un contexte distribué

Au-delà des conditions plus générales exposées en 2.3, l'application d'un processus d'IE distribuée pose des défis importants sur le plan opérationnel. En fait, l'IE distribuée porte en elle-même deux types de défis : ceux inhérents à l'IE, et ceux liés au contexte distribué.

2.4.1 Défis inhérents à l'IE

Quelle que soit la situation du projet, distribuée ou non, celui-ci sera confronté à des difficultés liées à l'IE. Hoffman [23], le Standish Group [39] et la norme IEEE Std 1233-1998 [5] relèvent les suivantes:

- **Coût** : il peut souvent être difficile de convaincre la haute direction de l'utilité de faire une bonne gestion des exigences. En effet, comme il s'agit de prévenir des problèmes, les bénéfices ne sont pas visibles directement, en revanche, les coûts sont beaucoup plus évidents et l'on peut penser que les retombées d'une bonne gestion des exigences ne méritent pas les efforts nécessaires.
- **Complexité** : l'IE s'adresse principalement à des systèmes complexes. C'est pour cela que l'on fait appel à des outils tels que des bases de données pour gérer toutes les exigences. Ainsi, sans méthodes et discipline on ne peut parvenir à en capturer toutes les exigences et avoir une vue d'ensemble sur le projet.

- **Pression de la concurrence** : celle-ci pousse toujours et encore à réduire les temps de développement, ce qui laisse moins de temps pour gérer correctement les exigences. Cependant, utilisée à bon escient, l'IE peut aider à réduire les cycles de développement.
- **Implication du client** : elle est primordiale pour l'IE. Le processus est itératif, et c'est, entre autres, l'interaction avec le client qui permet chaque nouvelle itération. Plus le client est impliqué dans la génération du SRS, mieux celui-ci reflétera ses besoins.
- **Compréhension commune des exigences** : les langues parlées ont malheureusement le défaut de pouvoir être ambiguës. Certains mots peuvent avoir plusieurs significations. Même si celles-ci ne seront pas forcément fondamentalement différentes, cela peut créer des malentendus qu'il faut éliminer le plus tôt possible avec les parties prenantes.
- **Traçabilité** : en s'approchant des exigences atomiques (des exigences de très bas niveau), le nombre d'exigence augmente de façon exponentielle. Cela demande donc beaucoup d'effort pour maintenir la traçabilité.

2.4.2 Défis de l'IE liés au contexte distribué

Le contexte distribué amène la notion de distance. Lipnack [40] et Evaristo [41] définissent cette dernière comme n'étant pas toujours géographique, mais également pouvant également être culturelle, temporelle, fonctionnelle, hiérarchique, ou encore en termes de compétence. Damian [42] explique que la distance amène non seulement de nouvelles difficultés, mais amplifie les problèmes liés à l'IE. Sosa [43] ajoute que celle-ci constitue une barrière aux communications techniques. Il est possible de regrouper les problèmes liés à la distance, en quatre catégories principales [42]: la communication, la gestion de l'information, la culture et le décalage horaire. La Figure 2.1 montre les résultats d'une recherche menée pour étudier les problèmes rencontrés lors de l'IE dans un contexte distribué. Ceux-ci sont présentés en trois couches : les natures de problèmes, les problèmes et enfin les étapes du processus d'IE touchés par ces problèmes.

2.4.2.1 Première couche

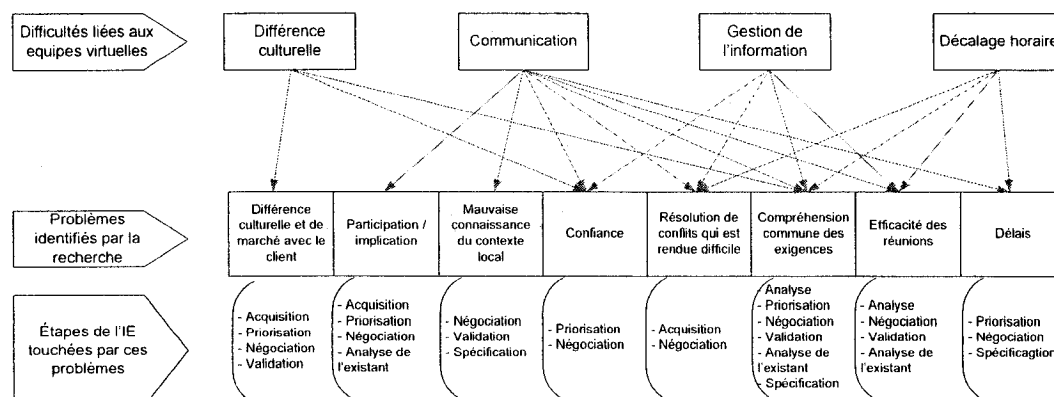


Figure 2.1: Nature des problèmes liés à la distance et implication dans le processus d'IE¹

La première couche représente les natures de problèmes qui sont liées aux projets distribués, sans égard particulier pour l'IE, mais qui y sont tout de même applicables :

- **La communication** : elle est capitale dans les projets distribués. L'IE est souvent vue comme une discipline comportant à la fois des aspects techniques et sociaux. Campbell et Damian [37, 42] identifient la qualité de la communication comme un facteur crucial dans la réussite de négociations et de discussions avec une équipe distribuée. En plus des réunions formelles, Davis et Lloyd [11, 31] insistent sur le fait que des communications informelles aident les acteurs à établir des liens qui favorisent la confiance ainsi que la diffusion de l'information.
- **La culture** : La culture est l'ensemble des connaissances acquises, l'instruction, le savoir d'un individu, mais aussi l'ensemble des structures sociales ainsi que les comportements collectifs qui caractérisent une société [33]. Il peut s'agir par exemple de la culture d'entreprise, de la langue ou des jours de congés. La culture sert de référentiel commun. Karolak [35] explique que si celui-ci n'est pas partagé, cela rend la collaboration difficile.
- **La gestion de l'information** : les intervenants doivent avoir accès aux mêmes informations (même version), afin d'obtenir une collaboration consistante et

¹ Traduction libre et adaptation de [42]

éviter le travail en double [44]. Pour Jacobs [38], un des facteurs cruciaux est que si les partenaires utilisent des systèmes d'information différents, il faut s'assurer que ceux-ci sont compatibles ou trouver des passerelles pour passer de l'un à l'autre.

- **Le décalage horaire** : il est inéluctable dans les projets qui sont répartis sur plusieurs fuseaux horaires. Si cela constitue un défi pour la coordination et la dynamique de groupe, il est cependant possible d'en tirer un avantage pour effectuer du travail en continu.

2.4.2.2 Seconde couche

La seconde couche relève les problèmes spécifiques à la conduite de l'IE dans un contexte distribué [44].

- **Différence culturelle et de marché avec le client** : la distance augmente la probabilité de différence culturelle avec le reste de l'équipe, le client ou le marché (les concurrents locaux par exemple) local [31].
- **Participation / implication** : l'implication de l'utilisateur est primordiale. Celle-ci permet une bonne communication et favorise une compréhension commune des exigences [31]. Les membres de l'équipe doivent être également intéressés et impliqués dans le projet [37].
- **Mauvaise connaissance du contexte local** : La communication des exigences passe par un canal formel, ce qui ne laisse pas de place pour les communications informelles qui informent les acteurs sur le contexte local. Cela peut mener à de mauvaises interprétations ou encore à l'usage de stéréotypes liés aux cultures [44].
- **Confiance** : la distance est un obstacle à l'instauration d'une relation de confiance, notamment en raison du manque de relations personnelles entre les membres de l'équipe (*corridor talk*) [45].
- **Résolution de conflits rendue difficile**. La recherche a montré que la négociation n'était pas touchée par la dispersion et qu'un espace électronique commun de partage des données était très utile et permettait une approche plus ciblée du problème [46].
- **Compréhension commune des exigences** : ce défi fait référence à l'univocité des exigences (cf. 1.1.2) et à l'ambiguïté inhérente aux langues parlées. Un mot

peut vouloir dire beaucoup de choses différentes, et on peut dire la même chose de plein de façons différentes. Il faut donc que le client et le fournisseur s'entendent sur la signification des exigences. Cela est critique au moment des négociations et au moment de donner des priorités aux exigences.

- **Efficacité des réunions** : d'une part, il est difficile d'organiser ces réunions du fait des emplois du temps chargés des preneurs de décision, mais également en raison du peu de chevauchement entre les fuseaux horaires des différents sites de développement. D'autre part, au cours des réunions qui se passent à distance (p.e. conférence téléphonique), les participants ne se rendent pas compte des acteurs en présence.
- **Formalité** : les membres de l'équipe distribuée doivent attendre une réunion formelle avant de discuter d'un problème qu'ils ne peuvent régler seul, ce qui peut engendrer des délais supplémentaires.

2.5 Outils de communication

Des outils de communications adaptés sont nécessaires à la bonne marche d'un projet distribué. Les technologies de l'information sont considérées comme un moyen de communication car elles permettent de transmettre de l'information. Dans le cas de l'IE, il est possible de transmettre des commentaires à propos d'une exigence. Les caractéristiques principales des outils de communication pour les équipes distribuées sont le synchronisme et la richesse du média de communication [47] :

- **Synchronisme** : on distingue les outils de communication synchrones qui permettent de mettre les acteurs dans un contexte de communication instantanée, et les outils asynchrones où il y a un temps plus ou moins long entre un message et la réponse. Le synchronisme donne une impression de proximité et un réel sentiment d'interaction. Dans un contexte de négociation, il a été constaté que dans un cadre synchrone, celles-ci étaient plus dures et plus compétitives que dans un cadre asynchrone où les acteurs étaient moins sujets à leurs émotions [48]
- **Richesse** : qui caractérise la capacité d'un média à transmettre la richesse de la communication humaine. Au-delà de la simple transmission d'un message, celle-ci représente une interaction sociale qui est le vecteur d'autres informations et la base de l'apprentissage du travail collaboratif [49]. La richesse représente donc les différents types d'information qui peuvent être communiqués (un simple

échange d'information, transmission de documents, vidéo, présence sociale). [50]

Les outils de communication courants sont par exemple [50] :

- **Téléphone (ou voix sur IP)** : il est utile dans les cas d'urgence et pour clarifier les informations [44].
- **E-mail** : un courriel peut être gardé pour servir de documentation et de support à la décision. [37] De plus, son utilisation réduit les problèmes de langue [44] Enfin, il est considéré comme un média riche pour ses capacités de transfert de fichiers ou d'envoi à plusieurs destinataires. [11]
- **Clavardage (*chat*, et autres messageries instantanées)** : L'accès est simple et permet d'avoir une réponse rapide [50]. Certains permettent le travail avec les acteurs sur un écran commun [44].
- **Vidéoconférence** : Le fait de voir les autres acteurs permet de diminuer la distance perçue. [50]

2.6 Synthèse de la recherche dans le domaine de l'IE distribuée

Jusqu'à maintenant, les concepts présentés sont liés aux trois thèmes principaux de la recherche, soient l'IE, les équipes distribuées et la modélisation. Comme il a été vu, ces thèmes de recherche disposent respectivement d'une littérature abondante. Par contre, la juxtaposition de ces thèmes est moins fréquente, bien que certains chercheurs et groupes de normalisation s'y intéressent de plus en plus. Parmi ceux-ci, on retrouve, par exemple:

- **L'IE dans un contexte distribué** : Damian (Herlea) [42], Lloyd [11], Jacobs [38], Bittner [18]
- **La modélisation dans l'IE** : SysML [7], Hull [10], Alexander [27]
- **La modélisation dans un contexte distribué** : Zhuge [51]

La Figure 2.2 présente sous forme graphique les champs de recherche touchés par cette recherche et présente une liste non exhaustive des chercheurs qui se sont déjà intéressés aux thèmes concernés et dont les travaux sont cités dans ce mémoire.

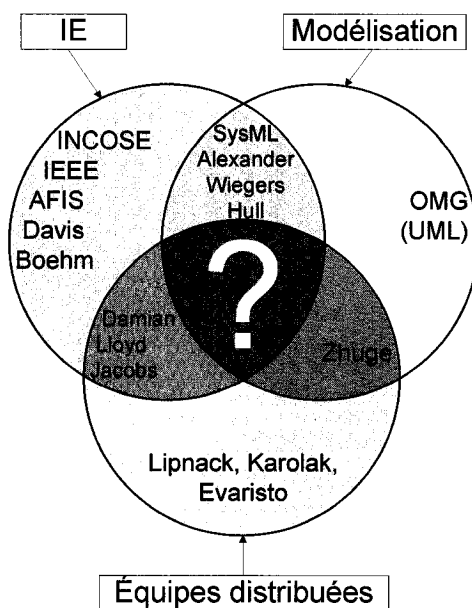


Figure 2.2 : Problématique, champs de recherche et chercheurs notables

À partir de cette représentation, on observe donc que malgré l'avancement considérable des connaissances, certaines « zones grises » existent encore, notamment à l'intersection des trois domaines considérés.

2.7 Objectifs de la recherche

La section précédente a montré qu'il existait peu ou pas de travaux à l'intersection de l'IE, de la gestion de projets distribués et de la modélisation. Cette recherche vise donc à approfondir cette « zone grise » en adoptant résolument une approche pratique, par le biais d'une étude de cas en entreprise.

En résumé, la problématique est la suivante : compte tenu que le développement de systèmes complexes s'effectue de plus en plus dans un contexte distribué, l'IE doit s'adapter à un tel contexte. Certaines technologies de l'information sont en mesure de gérer l'IE dans un contexte distribué et offrent la possibilité de modéliser des exigences. Toutefois, certains problèmes se posent pour chacun de ces domaines. Les technologies utilisées devraient contribuer à les atténuer, tout en posant elles-mêmes de nouveaux défis d'utilisation et d'implantation. Cette recherche étudie cette problématique et propose l'hypothèse suivante :

Les problèmes rencontrés pendant le processus d'IE dans un environnement distribué peuvent être amoindris, ou éliminés, d'une part grâce à la modélisation des exigences, et d'autre part grâce à l'utilisation d'un système d'information spécialisé, adapté au contexte distribué.

De façon plus concrète, les objectifs de recherche sont les suivants :

- Étudier les problèmes actuels de l'industrie dans le domaine de l'IE distribuée
- Développer et évaluer des solutions dans le but de répondre aux problèmes
- Expérimenter ces solutions sur un projet réel
- Apporter une expérience pratique sur un sujet encore peu documenté dans la littérature actuelle

Le prochain chapitre expose le contexte dans lequel s'est déroulé le volet empirique de la recherche, notamment en présentant l'entreprise où s'est déroulé le projet.

CHAPITRE 3 :

ASPECTS MÉTHODOLOGIQUES DE LA RECHERCHE

Ce chapitre présente l'approche utilisée pour atteindre les objectifs de recherche. Il décrit d'abord l'entreprise qui a servi de milieu pour la réalisation de l'étude. Plus spécifiquement, la section 3.1 expose les principales caractéristiques de « l'existant », c'est-à-dire des conditions et de l'environnement dans lesquels ce projet fut réalisé. Elle présente également l'étape préparatoire (exploratoire) qui fut menée préalablement au projet principal de recherche.

Dans un second temps, ce chapitre fournit une description des principales étapes ayant mené à la réalisation du travail d'expérimentation pour la mise en place de l'IE distribuée. La section 3.2 couvre le volet empirique du projet de recherche chez CAE en 4 grandes étapes : l'étude du processus actuel, la proposition d'un nouveau processus, l'expérimentation de ce nouveau processus dans un projet de CAE et enfin l'évaluation des résultats.

3.1 Description du milieu à l'étude

3.1.1 Choix du terrain: un projet de l'entreprise CAE

CAE est une entreprise montréalaise dont les activités se concentrent dans le domaine des technologies de simulation et de modélisation (leader mondial, 80% de part de marché) ainsi que dans les services de formation intégrés (deuxième mondial). L'entreprise a une vaste clientèle internationale, autant du côté de l'aviation civile que dans le domaine militaire. L'entreprise emploie environ 5 000 personnes réparties dans

19 pays et sur tous les continents. Il s'agit donc d'un cas typique d'entreprise où les tâches se font souvent en mode dispersé. Par exemple, lorsqu'une équipe d'une des filiales a besoin d'aide ou d'effectifs, il est fréquent que les autres filiales soient sollicitées.

Le produit le plus connu de l'entreprise est le simulateur d'avion. Il s'agit d'un système complexe devant non seulement simuler un avion avec une très haute fidélité, mais également son environnement. Il peut s'agir de d'autres avions, de stations de radio au sol ou encore, dans le cas d'avions militaires, d'unités tactiques (chars, soldats, etc.). Le simulateur de vol est un système très complexe pour lequel l'IE s'avère essentielle à sa conception et sa fabrication. La Figure 3.1 donne un aperçu des composantes d'un simulateur d'avion type de CAE.



Figure 3.1 : Composants d'un simulateur d'avion

Afin de lui donner un fondement pratique solide, cette recherche fut réalisée sur un projet réel d'entreprise. CAE constitue donc un terrain d'étude idéal, d'une part grâce au contexte international de l'entreprise, et d'autre part grâce aux technologies dont cette firme dispose (matériel de vidéoconférence, connexion Internet de très haute qualité, un intranet très développé, etc.). L'entreprise a déjà une certaine expertise dans la gestion des équipes dispersées, ce qui s'avère aussi une dimension importante pour la conduite du projet.

Le projet étudié est un projet de recherche et développement (R&D) visant à développer un nouveau simulateur¹. Ce type de projet est destiné à exploiter un créneau dans le marché, ce qui signifie qu'il n'y a pas de client spécifique rattaché au projet. C'est un chef de projet, tenu volontairement tenu à l'écart de l'équipe de projet, qui a joué le rôle de client dans tout le processus d'IE. Afin de ne pas biaiser les résultats du processus d'IE, le client et l'équipe de développement ne se sont pas rencontrés avant la revue des exigences du système prévue par le processus d'IE. L'équipe de développement s'est installée dans un lieu différent afin qu'aucune information ne s'échappe par inadvertance. Même si, en pratique, il n'est pas possible d'empêcher les personnes de se parler entre elles, la consigne de « distance » fut rigoureusement respectée. En fait, comme le projet devait également permettre l'expérimentation de l'IE distribuée, les différents intervenants ont volontairement accepté ces conditions.

Un des aspects intéressants liés à ce projet de R&D² est l'opportunité de travailler à partir de l'état initial (« la page blanche »), c'est-à-dire sans référence particulière à des projets antérieurs, contrairement aux projets habituels. En effet, par souci d'économie de temps, il arrive souvent que l'entreprise réutilise une archive d'un projet similaire (même type de simulateur, même type d'avion), pour travailler

¹ Comme il s'agit d'un projet confidentiel, la description du projet doit demeurer générale.

² Pour éviter la confusion avec le projet de recherche de l'auteur (projet de maîtrise), le terme 'projet R&D' sera utilisé pour identifier celui de l'entreprise CAE.

uniquement sur les écarts. Dans le cas présent, le processus d'IE pouvait donc être expérimenté du tout début.

3.1.2 État de l'existant : l'ingénierie des exigences chez CAE

3.1.2.1 Le processus

En 2005, CAE a entrepris de documenter tous les processus de la firme. Le processus d'IE fait partie des processus qui ont été documentés. Du nom de MST-PRS-004, il se présente sous la forme d'un document MS Word de 40 pages. Le processus est découpé en phases qui sont séparées par des jalons matérialisés par une rencontre de l'équipe de développement à l'interne ou avec le client ou encore la génération d'un document. Chaque phase du processus est représentée par un graphique et les activités de cette phase sont documentées. L'Annexe A : montre un extrait de ce processus tiré du document de CAE. Comme il est disponible sur l'intranet mondial de CAE, toutes les filiales de l'entreprise peuvent consulter ce processus. Théoriquement, une équipe d'IE d'Allemagne peut travailler avec une équipe de Montréal sans trop d'apprentissage puisqu'elles utilisent le même processus.

En pratique, les choses se passent un peu différemment. D'abord, la description du processus n'est pas mise à jour au même rythme d'évolution du processus lui-même. De plus, les filiales utilisent certes toutes le même processus, mais les différences culturelles mènent à des mises en pratiques parfois divergentes et parfois inappropriées.

3.1.2.2 Évaluation du processus d'IE (métriques)

La littérature offre une multitude de critères pour évaluer un processus d'IE. Ceux-ci se situent par ordre d'importance perçue dans trois dimensions : la qualité du service d'IE, la qualité des produits du processus d'IE et enfin le coût du processus d'IE

(ce dernier critère étant considéré comme bien moins important que les deux autres dans l'étude citée [52]).

Selon cette même recherche [52], la qualité du service est difficile à évaluer. Celui-ci ne sera donc pas mesuré dans cette recherche. On peut cependant noter à ce sujet que CAE dispose d'un département d'IE, ce qui est peu fréquent dans l'industrie. Cela a été fait dans le but de développer une expertise en IE pour CAE. Cette équipe créée il y a trois ans dispose actuellement de 5 membres. Auparavant, l'ingénierie des exigences était assumée par l'équipe de soumission, le gestionnaire de projet et l'ingénieur de projet, mais il n'existait pas de processus d'IE, celle-ci était donc réalisée *ad hoc*. Le fait d'avoir un département dédié a permis de développer un processus d'IE ce qui a grandement amélioré la qualité du service de l'IE (fiabilité, historique des projets). On peut donc dire que le département d'IE offre un service de qualité.

Les métriques utilisées par CAE sont définies dans le processus d'IE et permettent d'évaluer la qualité des livrables du processus d'IE. CAE dispose d'un outil qui permet de calculer en temps réel ces métriques et de visualiser leur évolution dans le temps. Les métriques utilisées par CAE sont les suivantes :

- **Nombre d'exigences** : CAE développe des simulateurs d'avion depuis longtemps, et sait qu'en moyenne, un SRS doit compter environ 1 200 exigences. Si ce nombre s'avère moindre, les experts jugent que les exigences ne sont peut-être pas suffisamment explicitées. Dans le cas contraire (« trop » d'exigences), on juge que plusieurs exigences peuvent être en dehors de la portée du projet (*scope creep*), trop détaillées, ou constituent déjà une solution de design
- **Volatilité des exigences** : aussi appelé « maturité » des exigences, la volatilité désigne le nombre de changements apportés aux exigences entre deux jalons. Il est normal que des changements interviennent pendant le processus d'IE, surtout au début du processus, lors qu'il faut clarifier les exigences. Plus l'on s'approche de la spécification des exigences techniques, et moins il doit y avoir de changements.

Enfin, le coût de l'IE il est en dehors de la portée de cette étude et que celui-ci est considéré de moindre importance dans l'étude citée [52].

3.1.2.3 La technologie utilisée

CAE utilise le logiciel DOORS 7.1 pour l'IE de tous ses projets. Celui-ci est considéré comme l'un des meilleurs logiciels pour l'IE [53]. De plus, grâce au module DOORS/Analyst, la modélisation des exigences est intégrée à DOORS. Il est bien adapté pour un usage dans un réseau local (LAN). Son utilisation dans un réseau étendu (WAN) comme Internet est certes possible, mais n'est pas recommandée pour des raisons de sécurité principalement, mais également de performances et d'attribution des licences.

Pour les projets distribués, CAE utilise le mécanisme de partitions de DOORS qui permet d'exporter un projet au complet dans un fichier *DOORS Project Archive*. Dépendamment du niveau de confidentialité requis par le projet, celui-ci sera transmis par FTP sécurisé (habituellement les projets commerciaux), ou gravé sur un CD-ROM et transmis par un transporteur agréé (projets militaires). Une fois la partition transmise, celle-ci est importée par le partenaire dans sa propre base de données. Ainsi, ultimement, le projet existe autant de fois qu'il y a de partenaires dans le projet.

Si tous les partenaires doivent modifier des données relatives au projet, il en résulte d'importants problèmes de synchronisation de toutes ces versions du projet. Habituellement, les partenaires s'entendent pour qu'un seul des partenaires apporte des modifications entre deux mises à jour, ce qui oblige les autres partenaires à figer leur version du projet afin qu'aucun changement n'y soit apporté. Un autre inconvénient de cette solution est que cela oblige tous les partenaires à utiliser DOORS et d'avoir des serveurs dédiés ainsi que des licences pour l'utiliser. Enfin, le recours aux partitions coûte cher en temps, notamment dans le cas des projets militaires où l'échange des données doit transiter par un transporteur agréé.

3.1.2.4 Synthèse : les difficultés du processus traditionnel

Le processus traditionnel d'IE de CAE est certes utilisable tel quel dans un contexte distribué, mais plusieurs lacunes sont observées :

- Le processus actuel ne prend pas en compte l'intervention de partenaires ou de membres d'une équipe distribuée.
- La documentation actuelle n'est pas mise à jour régulièrement et n'est pas connue d'autres filiales de CAE dans le monde
- La technologie utilisée impose des délais et un effort de gestion supplémentaires.

La recherche d'une solution à ces lacunes permettra de répondre aux objectifs de recherche : d'une part, il s'agit d'un problème concret de l'industrie qui est lié à l'IE dans les projets distribués. D'autre part, s'agissant d'un projet réel, la recherche devra proposer une solution technologique fiable, utilisable dans le contexte actuel de l'entreprise.

3.1.3 Pistes explorées dans la phase initiale de la recherche

Préalablement à la phase d'expérimentation du projet de maîtrise, une phase d'exploration technique fut entreprise afin d'examiner, de concert avec l'entreprise, certaines solutions. Voici une courte description de chacune d'entre elles et du résultat obtenu :

- **Logiciel DOORS XT** : À l'été 2005, un projet pilote fut mené au sein de la Chaire de recherche du Canada en gestion de projets technologiques, en collaboration avec CAE, afin d'évaluer la prochaine génération de DOORS (DOORS XT). Cette version était apparemment adaptée à l'utilisation sur un réseau étendu comme Internet grâce : à l'utilisation de communications cryptées et signées (HTTPS avec certificat) pour la confidentialité, à une implantation sous forme de client léger (*thin client*) pour les performances, et l'attribution des licences du côté du serveur. Enfin, une interopérabilité avec DOORS 7.1 était offerte. Malheureusement, les versions évaluées (2.0 et 2.1), bien que prometteuses, montraient de sérieuses lacunes en ce qui concerne la stabilité,

notamment au niveau de l'interopérabilité avec DOORS 7.1. Les résultats de ce projet pilote furent présentés lors de la conférence des usagers de Telelogic en 2005 [54].

- **Citrix / Remote desktop / VNC** : Une autre approche fut examinée, celle qui consiste à utiliser un ordinateur à distance. Dans ce cas, les commandes de l'utilisateur (mouvements de la souris et saisie au clavier) sont transmises au serveur, une image de l'écran du serveur est envoyée à l'utilisateur. Cette solution ne fut pas retenue, d'une part pour des raisons pratiques (par exemple, le copier-coller depuis l'ordinateur de l'utilisateur ne fonctionne pas), et d'autre part, en raisons du coût des licences (Citrix et Remote Desktop) et des performances (VNC).

3.2 Le volet empirique du projet de maîtrise chez CAE : les grandes étapes

Les grandes étapes qui ont permis de réaliser le volet empirique de cette recherche sont illustrées à la Figure 3.2. Chacune de ces étapes sont détaillées ci-après. Les résultats propres à chaque étape font l'objet d'une présentation distincte, au Chapitre 4 (« présentation des résultats »).

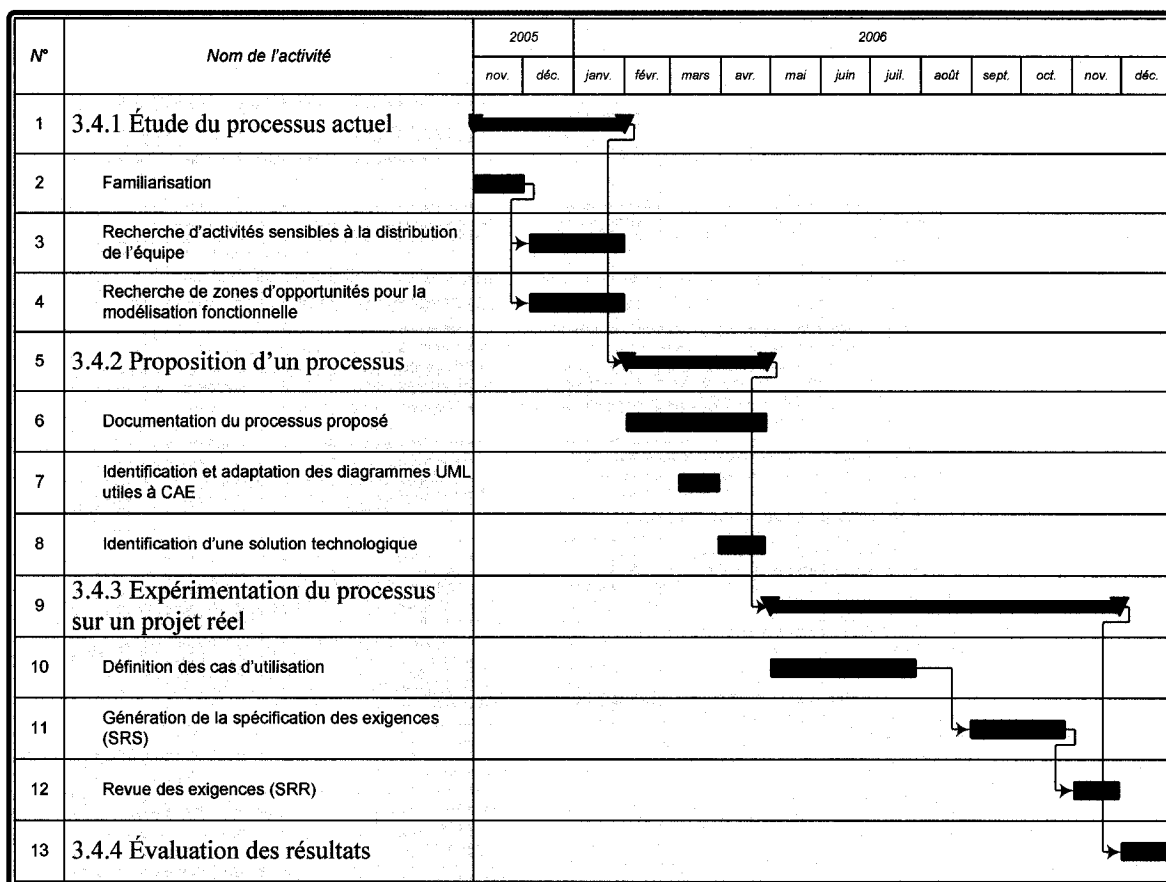


Figure 3.2 : Échéancier du volet empirique du projet de maîtrise

3.2.1 1^{ère} grande étape : étude du processus actuel

La première étape dans cette recherche a consisté à analyser le processus actuel de CAE. L'étude du processus d'IE a débuté par l'analyse du document MST-PRS-004 (version 2.1) qui décrit le processus d'IE de CAE. Cette analyse a permis une première familiarisation avec le processus de CAE. Le but principal de cette étude fut d'examiner les activités susceptibles d'être affectées par le caractère distribué des projets, puis d'identifier celles qui présentent des opportunités d'amélioration grâce à la modélisation fonctionnelle.

Ce processus est découpé en 4 phases. Celui-ci est basé sur le processus proposé par le standard IEEE Std. 1220-1994 [9], et respecte le niveau 3 de maturité du CMMI.

3.2.2 2^{ème} grande étape : proposition d'un processus

Il fut convenu avec le partenaire industriel (CAE) que le nouveau processus d'IE à définir devait répondre aux critères suivants: **(1)** répondre aux contraintes imposées par le contexte distribué de l'équipe de développement, **(2)** incorporer la modélisation des exigences et enfin **(3)** se baser sur une solution technologique qui minimise l'effort requis pour la synchronisation des données entre les partenaires.

- **(1)** Pour répondre au contexte distribué, les tâches consistant à discuter des exigences ou à organiser des réunions ont été révisées afin de décider si il fallait organiser une réunion avec tous les partenaires et / ou une réunion interne à chaque partenaire.
- **(2)** La modélisation des exigences permet d'une part de mieux couvrir la totalité des exigences du système à développer, mais selon l'hypothèse de recherche, elle permet d'améliorer la communication entre les partenaires et de ce fait réponde au point **(1)**. La modélisation intervient principalement lors de l'analyse des exigences.
- **(3)** Le processus proposé identifiera les technologies à utiliser. Les critères de succès de la technologie à utiliser sont d'une part la capacité des partenaires à pouvoir travailler sur des données à jour, et d'autre par la sécurité des données vis-à-vis de l'extérieur du projet, mais également à l'intérieur du projet entre les partenaires.

3.2.3 3^{ème} grande étape : expérimentation du processus sur un projet réel

L'expérimentation sur un projet réel avait pour but de valider le processus proposé dans un contexte de projet réel. Elle s'est déroulée en trois étapes principales, calquées sur les étapes du processus proposé:

- **Définition des cas d'utilisation** : les cas d'utilisation furent définis pour servir de méthode d'acquisition des exigences. D'autres méthodes ont pu être utilisées en parallèle.
- **Génération de la spécification des exigences du système** : la spécification est issue par dérivation des cas d'utilisation qui auront été définis et des autres méthodes d'acquisition des exigences qui auront été utilisées. Afin de s'assurer que tous les cas d'utilisation sont couverts dans la spécification des exigences du système, les liens de traçabilité entre les cas d'utilisation et les exigences ont été établis.
- **Revue des exigences système** : au cours de celle-ci, les experts ont dû évaluer les exigences. Cette rencontre s'est faite en face à face avec le client. Chaque expert devait exposer les problèmes rencontrés avec les exigences et le cas échéant proposer des solutions alternatives.

L'étape d'expérimentation s'est limitée à ces trois premières étapes du processus. Ce dernier comporte quelques étapes supplémentaires (par exemple, la vérification et la validation) mais elles sont plus ou moins pertinentes dans le cadre de la présente recherche. En effet, la plupart de ces étapes peuvent être réalisées en contexte local.

3.2.4 4^{ème} grande étape : évaluation des résultats

La dernière étape de la recherche a consisté à évaluer le processus proposé. Pour ce faire, les métriques utilisées par CAE pour mesurer l'efficacité du processus d'IE (cf. 3.1.2.2) ont été calculées. Les résultats obtenus ont été comparés aux résultats de projets comparables, la désignation de ces derniers étant basée sur l'envergure, la nature et le nombre de participants du projet.

3.3 Synthèse

Cette recherche s'appuie sur une expérience pratique menée conjointement avec CAE pour valider le processus d'IE proposé dans un contexte réel de l'industrie. Le projet étudié est un projet de recherche et développement pour un nouveau type de

simulateur. Le processus d'IE de CAE n'est pas prévu pour une équipe de projet dispersée et éprouve des difficultés pour diffuser ce processus d'une part et des complications techniques d'autre part qui imposent un effort supplémentaire pour les projets distribués. CAE a déjà exploré certaines solutions par le passé, mais aucune ne s'est montrée satisfaisante.

Le volet empirique du projet de recherche chez CAE est découpé en quatre grandes étapes, l'étude du processus actuel, la proposition d'un nouveau processus, l'expérimentation du nouveau processus sur un projet réel de CAE et enfin l'évaluation des résultats.

CHAPITRE 4 :PRÉSENTATION DES RÉSULTATS

Ce chapitre présente les résultats obtenus dans l'expérimentation décrite dans la partie précédente. Le tableau qui suit met en relation les activités de l'échéancier présenté dans le chapitre précédent (Figure 3.2) et les résultats qui sont présentés dans ce chapitre.

Tableau 4.1 : Correspondance entre les grandes étapes du volet empirique de la recherche et des résultats

Activité	Résultat
3.4.1 Étude du processus actuel	4.2 Documentation
3.4.2 Proposition d'un processus	
	4.3 Outils
3.4.3 Expérimentation du processus sur un projet réel	4.4.1 Résultats
3.4.4 Évaluation des résultats	4.4.2 Analyse des résultats

4.1 L'approche générale proposée

Le but recherché dans la proposition du nouveau processus d'IE pour CAE est double. Il doit d'une part satisfaire aux contraintes d'une équipe distribuée. D'autre part, il doit intégrer la modélisation des exigences fonctionnelles. La Figure 4.1 qui suit est un

rappel du chapitre 1 et montre la vue d'ensemble du processus d'IE qui met en valeur le lien entre la modélisation des exigences fonctionnelles et la spécification des exigences.

4.2 Documentation

Cette étude consiste à préparer trois livrables à partir du document MST-PRS-004. Cette documentation se trouve dans son intégralité dans l'Annexe C.

4.2.1 Cartographie du processus

La cartographie du processus a été entreprise afin de donner une vue graphique du processus ainsi que de la simultanéité des tâches. Elle intègre de manière consistante les informations qui composent le dictionnaire d'activités et la matrice d'assignation des responsabilités. Celle-ci a été faite à l'aide du logiciel ARIS Toolset 6.23 qui est prévu pour la cartographie et la réingénierie de processus. Habituellement, ARIS utilise les diagrammes eEPC (*Extended Event-Driven Process Chain*) pour représenter les processus. Ceux-ci sont basés sur l'enchaînement d'événements et d'actions afin de représenter le processus ainsi que le résultat de ces actions. Cette discipline est surtout nécessaire pour faire appel à la simulation de processus, qui est surtout utile dans la réingénierie de processus. Dans ce projet, seules les actions sont représentées afin d'alléger les cartographies de processus, étant donné que la simulation n'est pas pertinente. La Figure 4.2 présente un exemple de cartographie de processus et le Tableau 4.2 explique les conventions qui ont été utilisées. L'intégralité de la cartographie du processus se trouve dans l'Annexe C.

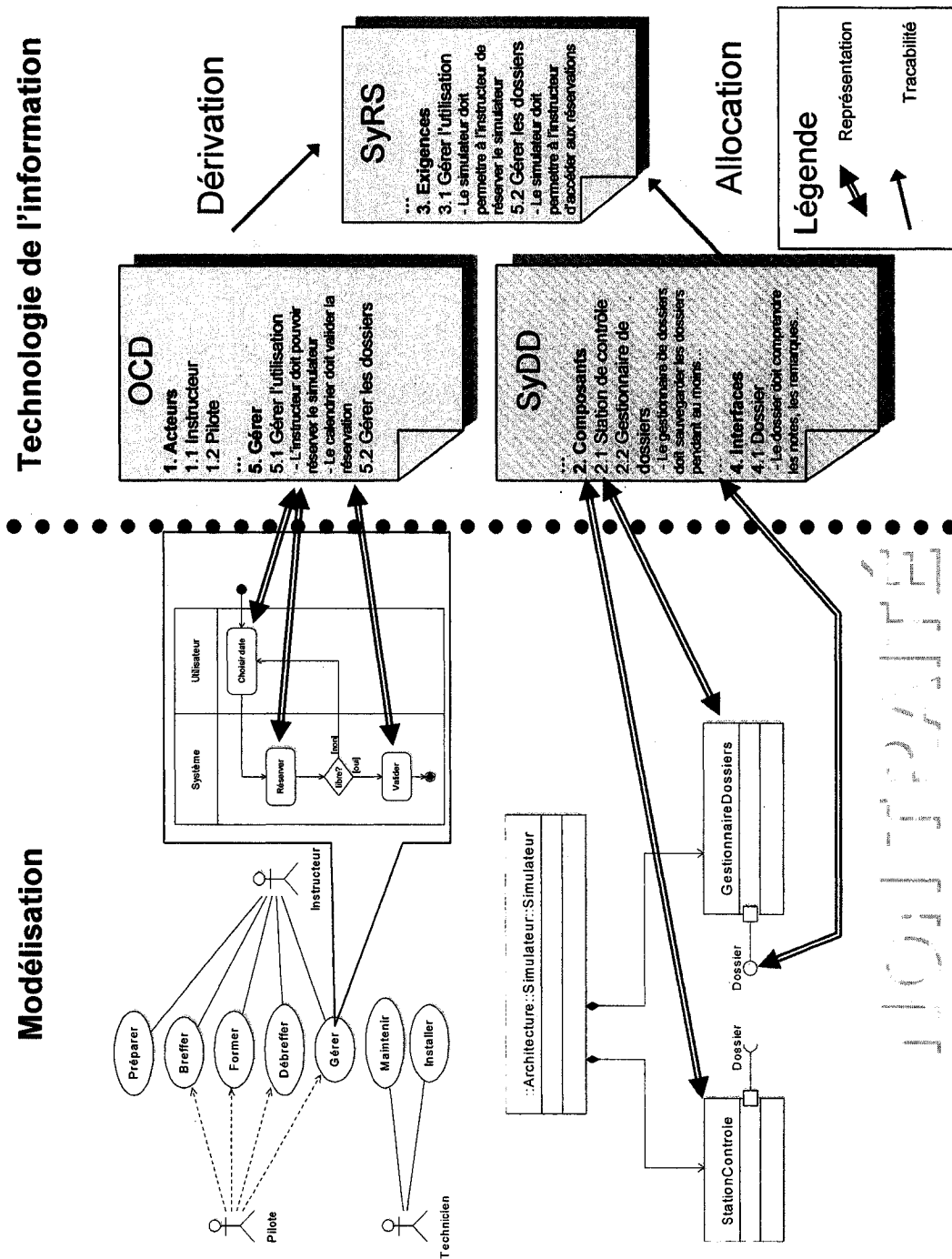


Figure 4.1 : Intégration de l'outil de modélisation et de l'outil de support pour l'IE

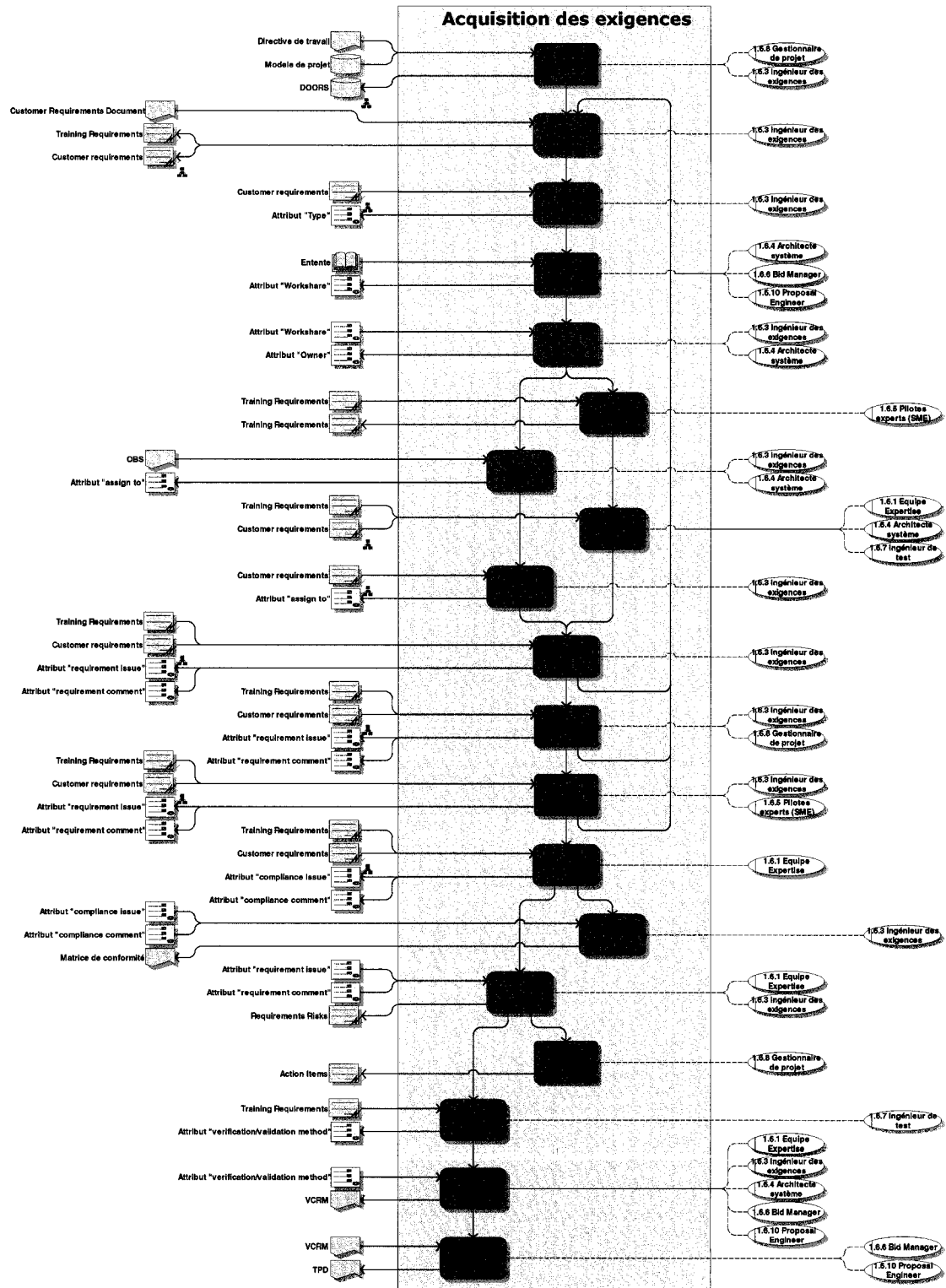

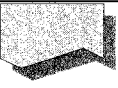

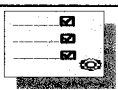
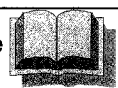

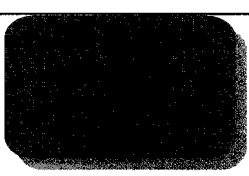
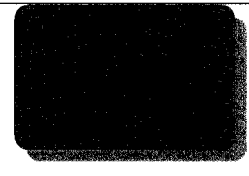


Figure 4.2 : Extrait de la cartographie du processus d'IE de CAE

Tableau 4.2 : Conventions utilisées dans la cartographie de processus

DOORS 	Un système d'information (DOORS par exemple)
Directive de travail 	Un document papier
Training Requirements 	Un document électronique
Attribut "Type" 	Un attribut dans DOORS
Entente 	Un document contractuel
1.6.3 Ingénieur des exigences 	Un acteur
	Une activité effectuée localement
	Une activité pouvant être affectée par le caractère distribuée de l'équipe

Les informations qui figurent dans la cartographie de processus sont issues du document MST-PRS-004. Celles qui n'ont pas été trouvées dans le document ont été ajoutées sur la base de la norme IEEE Std. 1233, de la littérature ou demandées au département d'IE de CAE. Enfin, ces livrables ont été validés avec le département d'IE de CAE.

4.2.2 Dictionnaire

Les descriptions des activités décrites dans le document MST-PRS-004 ont été classées dans un dictionnaire d'activités qui donne pour chacune d'entre elles les

entrants, les sortants ainsi qu'une brève description de la tâche. Ces données sont entièrement intégrées avec celles trouvées dans la cartographie du processus. La Figure 4.3 montre un exemple de description de tâche. Le dictionnaire peut être trouvé dans son intégralité en Annexe C :

1.1.1. Définir le cadre de travail

Entrants
<ul style="list-style-type: none"> • Directive de travail (« work directive ») • Modèle de projet de CAE
Description
<ol style="list-style-type: none"> 1. Le modèle de projet de CAE (Project Framework) est instancié à l'emplacement du nouveau projet, puis adapté (nom, droits d'accès, modification de modules, attributs). 2. Le cadre de travail de travail collaboratif doit être défini. Les acteurs concernés devront s'accorder sur un processus de gestion des exigences, mode d'accès, de communication et de partage des données (VPN, partitions...). Une formation des partenaires peu familiers avec les outils ou avec le processus peut s'avérer nécessaire.
Sortants
<ul style="list-style-type: none"> • Nouveau projet dans DOORS

Figure 4.3 : Un exemple de description d'une activité dans le dictionnaire d'activités

4.2.3 Matrice d'assignation des responsabilités

Les responsabilités des membres de l'équipe de projet ont été transcrites sous forme de matrice d'assignation des responsabilités (RAM). Celle-ci permet une référence rapide aux responsabilités et permet de définir deux niveaux de responsabilité. Les responsabilités primaires (1) indiquent quel membre de l'équipe doit conduire l'activité en question. Une responsabilité secondaire (2) indique que le membre de l'équipe participe à l'activité. La Figure 4.4 présente un exemple de RAM. Les chiffres devant chacune des activités correspondent au numéro d'activité retrouvé dans la cartographie et dans le dictionnaire. La RAM présente dans son intégralité en Annexe C.

Activités										
	Développeurs	Ingénieur de projet	Ingénieur des exigences	Architecte système	Équipes d'expertise	Gestionnaire de soumission	Ingénieur de test	Gestionnaire de projet	Logistique	Ingénieur de soumission
2.1 Acquisition des exigences										
2.1.1. Définir le cadre de travail			2					1		
2.1.2. Importer / acquérir et modéliser les exigences du client			1							
2.1.3. Assigner un type aux exigences du client			1							
2.1.4. Définir une répartition des exigences entre les partenaires				2		1				2
2.1.5. Identifier un responsable pour les exigences			1	2						
2.1.6. Identifier des exigences de formation tacites ou manquantes					1					
2.1.7. Identifier les exigences couteuses	1			1			1			
2.1.8. Créer les équipes d'expertise			2	1						
2.1.9. Assigner les exigences du marché aux équipes d'expertise			1							
2.1.10. Revue interne chez chaque partenaire			1		2					
2.1.11. Revue interne au projet		2	1		2			2		
2.1.12. Revue avec la direction du projet		2	1		2			2		
2.1.13. Définir la conformité pour chaque exigence	1									
2.1.14. Créer la matrice de conformité			1							
2.1.15. Identifier et classier les problèmes restants	1		2							
2.1.16. Gérer les actions à entreprendre « Rq Action Item »								1		
2.1.17. Définir la méthode de vérification							1			
2.1.18. Créer la VCRM	1		1	1		1				1
2.1.19. Créer le TPD	1		1	1		1				1

Figure 4.4 : Extrait de la matrice d'assignation des responsabilités

4.3 Outils

Cette section décrit les outils qui furent utilisés pour supporter le processus tel que proposé. La première partie décrit l'infrastructure réseau qui permet un travail collaboratif à distance. La seconde montre les moyens de communication utilisés par CAE pour travailler avec des partenaires distants. Enfin la dernière partie décrit les outils de support à l'IE.

4.3.1 VPN

La solution utilisée fait appel à un réseau privé virtuel (VPN) qui permet d'une part de créer un réseau local virtuel sur la base d'un réseau étendu, et d'autre part assure la confidentialité de l'échange des données grâce à divers algorithmes de cryptage. Les serveurs (DOORS et de licences) sont accessibles depuis le VPN. Il existe deux « couches » de VPN : l'une pour les employés de CAE qui accèdent à leurs données depuis l'extérieur du réseau CAE, l'autre qui est accessible pour les partenaires et les employés de CAE. Cela est fait pour des fins de sécurité. Seuls les projets distribués sont sur le serveur dans le VPN des partenaires (Figure 4.5).

Cette solution répond donc aux soucis de sécurité et d'attribution de licences. Les partenaires disposant tous d'une connexion rapide, les performances sont acceptables.

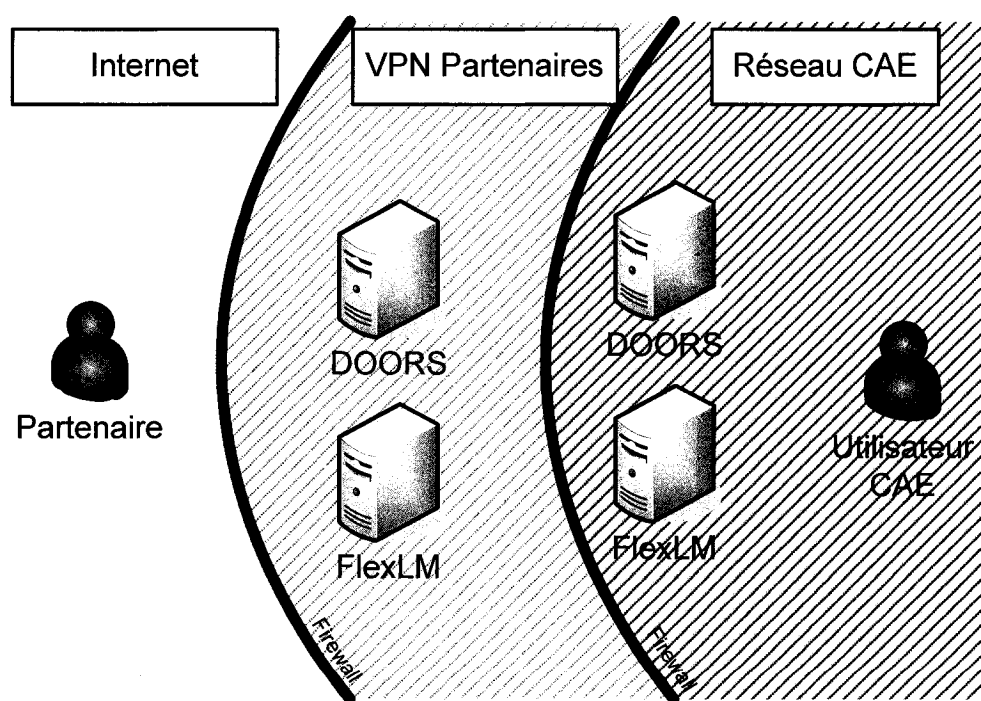


Figure 4.5: Topologie du VPN

4.3.2 Moyens de communication

CAE dispose d'une multitude de moyens de communication différents.

- **Pour les réunions** : les réunions se déroulent si possible en face à face, mais quand cela n'est pas possible, et en fonction de la technologie disponible chez les partenaires, CAE peut utiliser une vidéoconférence ou encore une conférence téléphonique.
- **Pour les discussions informelles** : le courriel est fréquemment utilisé pour celles-ci, car il permet de garder une trace. Une autre possibilité est l'utilisation du téléphone.
- **Pour la collaboration** : le travail collaboratif se fait au travers du VPN qui a été décrit plus tôt (cf. 4.3.1). Celui-ci ne constitue qu'une infrastructure commune, sur laquelle peuvent être déployés et bâtis des systèmes d'informations qui se trouveront alors virtuellement dans le même réseau. L'outil Webex qui permet de voir et contrôler un ordinateur permet une interaction directe sur un même écran à distance.

4.3.3 Choix des outils de support pour l'IE

4.3.3.1 Outil supportant l'IE

Étant donné que la technologie VPN permet d'utiliser un système d'information quel qu'il soit sur un réseau étendu comme Internet, c'est DOORS 7.1 qui a été choisi pour supporter l'IE. D'une part parce que la version la plus récente de ce même outil (DOORS XT) n'est pas encore suffisamment fiables [55], et d'autre part car CAE l'utilise depuis plusieurs années et le maîtrise donc parfaitement.

4.3.3.2 Outil de modélisation

Les outils supportant la modélisation en UML sont légion. Pour choisir parmi cette multitude de logiciels, une attention particulière fut portée aux critères suivants :

- Intégration avec DOORS

- Intégration avec d'autres outils
- Diagrammes supportés
- Simplicité d'utilisation

Le logiciel DOORS/Analyst 3.0 a été choisi pour l'étude, notamment pour sa forte intégration avec DOORS.

4.3.3.3 Standard de modélisation

Les diagrammes considérés dans cette étude (diagramme de cas d'utilisations et diagramme d'activité) sont communs aux deux standards. Ils peuvent être considérés comme étant aussi bien des diagrammes UML que SysML.

4.4 Évaluation du processus proposé

Cette section est consacrée aux résultats obtenus lors de l'expérimentation et à leur analyse.

4.4.1 Résultats

4.4.1.1 Résultats du processus proposé

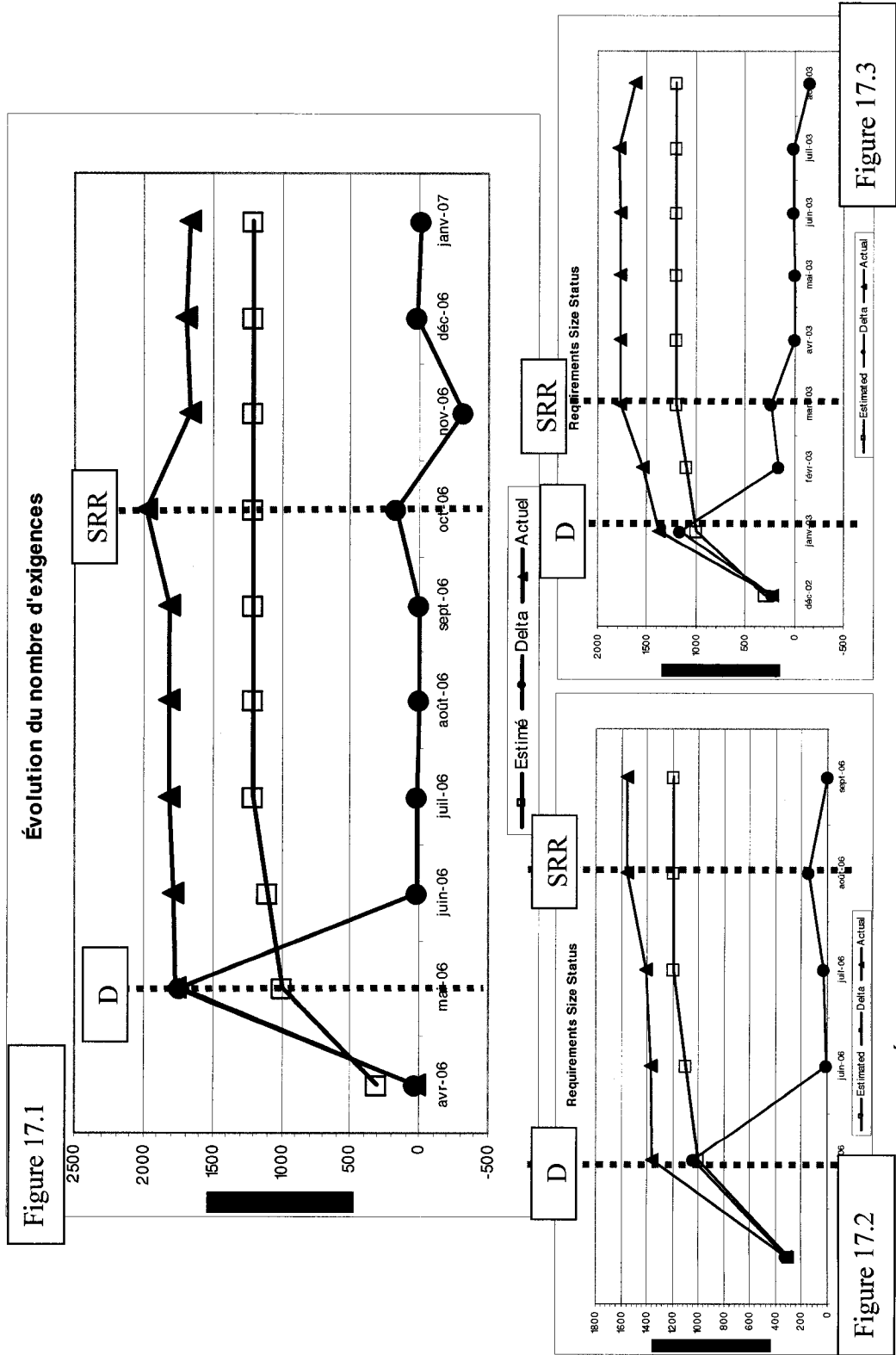
Les figures qui suivent montrent graphiquement l'évolution des métriques décrites dans la section 3.1.2.2. Il y a deux types de figure :

- Les figures 17.1, 17.2 et 17.3 fournissent les résultats des mesures qui permettent d'évaluer la première métrique, à savoir le nombre d'exigences. Elles montrent également une courbe estimée, basée sur l'expérience de CAE, ainsi que la courbe des changements (delta).
- Les figures 18.1, 18.2 et 18.3 mettent l'accent sur la deuxième métrique, la volatilité des exigences en montrant trois dimensions de changements qui peuvent arriver à une exigence, un ajout, une suppression ou encore une

modification (on considère qu'une exigence a été modifiée à partir du moment où un de ses attribut a été changé ou encore qu'elle a été déplacée dans la spécification).

Chacun de ces types de figure présente trois graphes, les figures 17.1 et 18.1 représentent les données du projet étudié dans cette recherche, les autres (17.2, 17.3, 18.2, 18.3) représentent quant à eux les résultats obtenus sur des projets de référence (projets comparables), pour fins de comparaison.

Le choix des projets de référence a été fait conjointement avec l'équipe d'IE de CAE. Il s'agit de projets de développement de systèmes personnalisés pour le ministère de la Défense nationale. À noter que les lignes vertes pointillées indiquent les principaux jalons observés dans le processus d'IE : « D » est le début du processus, i.e. le moment où l'équipe d'IE commence à travailler sur le projet, et « SRR » est la revue des exigences système avec le client (*System Requirements Review*).



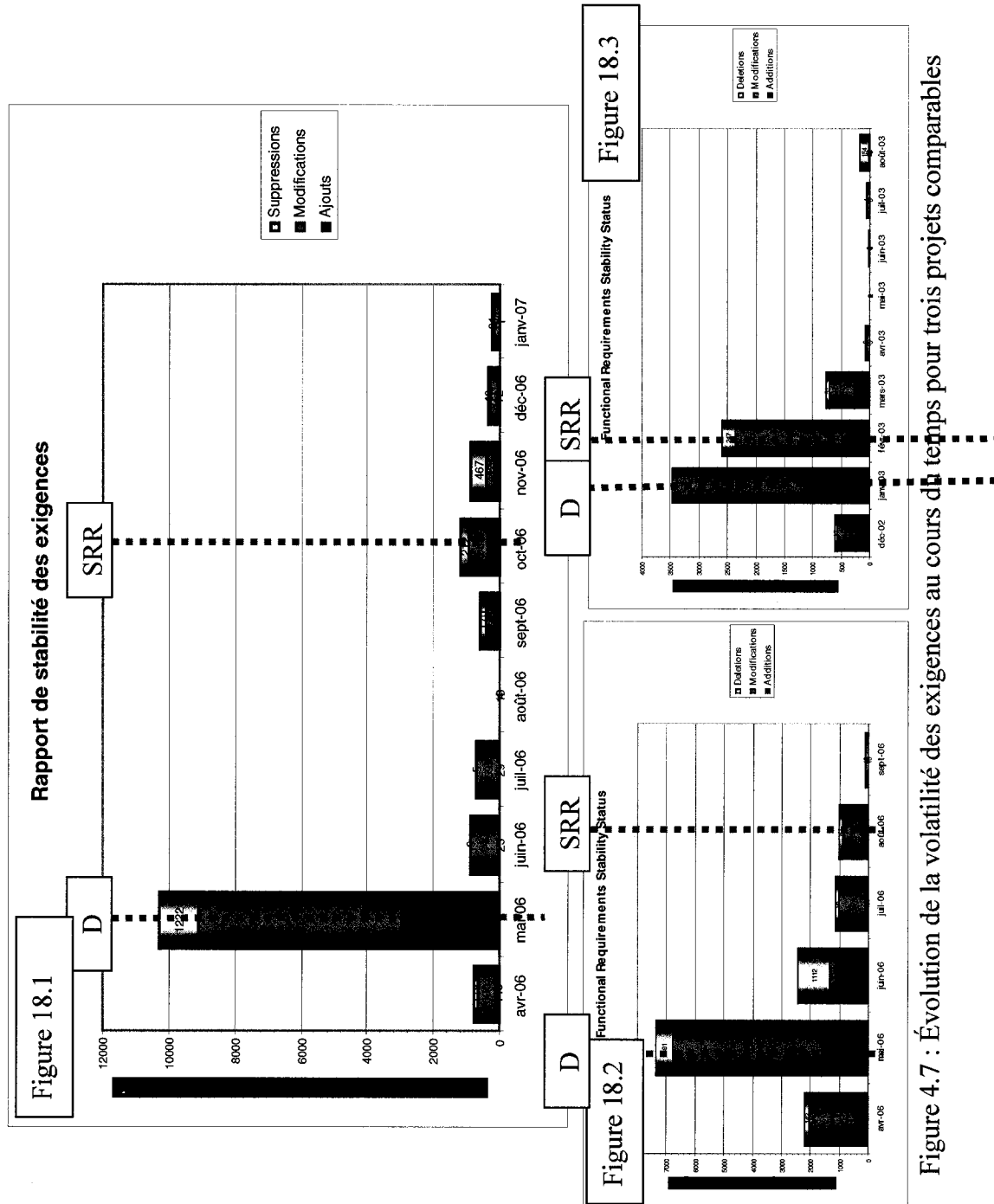
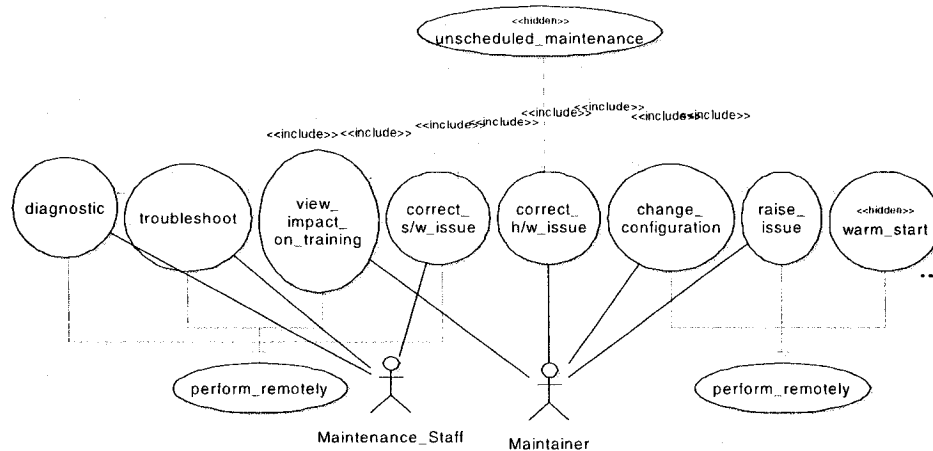


Figure 4.7 : Évolution de la volatilité des exigences au cours du temps pour trois projets comparables

4.4.1.2 Résultats pour la modélisation des exigences fonctionnelles

Dans ce projet, le gestionnaire de projet a voulu s'intéresser uniquement au cas d'utilisation de la maintenance du simulateur d'avion (maintenance corrective, maintenance préventive), car les autres sont suffisamment bien connus de CAE et ne représentaient pas de risque. En effet, le client suppose souvent que la maintenance est un dû (exigence tacite) et il spécifie des exigences souvent incomplètes pour ce qui a trait celle-ci. La Figure 4.8 montre un exemple de modélisation faite à l'aide d'un cas d'utilisation. L'intégralité de la modélisation des exigences fonctionnelles se trouve en Annexe B.

4.3.3.1 Unscheduled_Maintenance



4.3.3.2 diagnostic

The Maintenance Staff checks the status of all computing nodes (ready for load, loaded, issue, TBD).

The Maintenance Staff checks the status of the IO System.

The Maintenance Staff checks the status of mandatory softwares. (MOM server, boot manager, DCU, TBD) (loaded, unloaded, run-time)

The Maintenance Staff reproduces the issue.

The Maintenance Staff monitors labels.

The Maintenance Staff sets labels.

The Maintenance Staff runs CTS.

The Maintenance Staff collects variable values.

(The Maintenance Staff plugs a LabView)

4.3.3.3 view_impact_on_training

The Maintainer checks the impact of the non functioning systems on training.

4.3.3.4 troubleshoot

The Maintenance Staff reads documentation to find how to fix the issue. (wiring diagram)

The Maintenance Staff reads Diagnostics which includes supplemental information about corrective action.

4.3.3.5 correct_s/w_issue

The Maintenance Staff reloads a crashed process.

The Maintenance Staff changes the CAL files. (CAL files are calibration files)

4.3.3.6 correct_h/w_issue

The Maintainer replaces a defective part.

Figure 4.8: Exemple du cas d'utilisation "maintenance corrective"

4.4.2 Analyse des résultats

4.4.2.1 Performance par rapport au contexte distribué

La section 3.2.4 explique que le critère de succès du processus proposé est d'arriver à des résultats semblables à un projet local, aussi bien au niveau des résultats en eux-mêmes, qu'au niveau de leur évolution dans le temps. Cela permettrait de conclure que la distribution de l'équipe de projet n'aura pas eu d'influence sur le processus d'IE.

Pour ce qui est du nombre d'exigences, on peut constater de façon générale que la forme de la courbe d'évolution du nombre d'exigences est semblable aux projets de référence, avec une croissance rapide du nombre d'exigences au début du projet, puis une stabilisation jusqu'à la revue des exigences où l'on observe un petit sursaut du nombre d'exigences dans les trois graphes. Si l'on observe de plus près le nombre d'exigences, on peut remarquer que le projet étudié a environ 1750 exigences au début du projet, alors que les projets de référence sont environ à 1400 exigences. Cette différence est attribuable au fait que les projets ont une nature un peu différente : le projet étudié est un projet de R&D où le système doit être exploré dans son ensemble, alors que les projets de référence sont des projets personnalisés qui certes s'approchent des projets de R&D dans le sens où les éléments de l'architecture du système ne sont pas tous connus au départ.

Pour ce qui est de la volatilité des exigences, on remarque aussi que de manière générale, les trois graphes ont la même forme, avec un grand pic de changements en début de projet qui représente le travail de développement des exigences et le travail sur les attributs des exigences (par exemple assignation du type, allocation à un groupe d'expertise...). À l'étape de la revue des exigences, on peut voir que le nombre d'ajouts ou de suppressions d'exigences est devenu faible, alors que certaines modifications sont faites. Ces modifications sont le résultat des équipes d'expertise qui évaluent les

exigences en vue de la revue des exigences. Enfin, suite à la revue des exigences, on peut remarquer un sursaut d'activité¹ avec quelques ajouts et suppressions mais principalement des modifications (souvent de la formulation).

Les critères observés n'indiquent donc pas de changement majeur dans le profil de l'évolution du nombre d'exigences ou de la volatilité des exigences pour le projet observé, malgré son contexte distribué. Cela permet donc de valider le processus proposé dans sa mission de répondre à la problématique de dispersion de l'équipe de développement.

4.4.2.2 Performance de la modélisation des exigences fonctionnelles

Il est difficile d'évaluer ce que la modélisation des exigences fonctionnelles apporte comme amélioration par rapport au contexte distribué. Cependant, la modélisation a d'autres effets bénéfiques sur l'IE. Elle constitue d'une part une autre source d'exigence pendant la phase d'acquisition des exigences, ce qui permet de découvrir plus d'exigences. D'autre part, comme CAE construit des simulateurs depuis de nombreuses années, des spécifications antérieures sont utilisées comme base pour les nouvelles spécifications dans le but de gagner du temps et de l'effort. Les exigences issues de ces spécifications n'ont cependant dans la majorité des cas pas de parent, et il est difficile de savoir si l'exigence répond à une attente du client. Afin de mesurer l'apport que constitue la modélisation des exigences fonctionnelles, deux critères ont été mesurés :

- **le nombre d'exigences relatives à la maintenance** : 18 et 23 exigences de maintenance ont été comptabilisées dans les projets de référence alors que la

¹ N.B. L'outil de mesure des critères de performance du processus d'IE de CAE, faisant les mesures par mois, il est difficile d'apprécier l'activité relative à la revue des exigences pour le deuxième projet de référence car elle se confond ?? à celle du début du projet. Cependant, ces données sont accessibles à partir de l'historique de DOORS et ont été confirmées à partir de cette source.

spécification du projet étudié en comporte 60. Dans ces trois projets, les attentes du client en ce qui concerne la maintenance est la même, mais on constate que grâce à l'approche par cas d'utilisation, la spécification est plus complète. Il s'agit donc d'une amélioration très appréciable du processus.

- **la proportion d'exigences liées à un document d'exigences de haut niveau :** C'est une information pertinente pour un projet, d'une part car cela permet de savoir que l'exigence traduit bien un besoin du client, et d'autre part, cela permet de mieux comprendre le besoin du client. Dans les projets de référence, seulement 15% des exigences ont un lien de traçabilité avec des exigences de plus haut niveau, alors que ce taux atteint près de 80% dans le projet étudié. Les 20% restant étant constitués d'exigences de performance de la maintenance (par exemple le nombre d'années de disponibilités de pièces de rechange ou encore le temps moyen de réparation)

Ainsi, la modélisation a permis d'augmenter d'une part le nombre d'exigences et d'autre part d'augmenter le taux d'exigences liées à un parent dans la spécification des exigences pour le projet observé. On peut cependant remarquer que les résultats sont particulièrement bons car la maintenance était un thème mal spécifié dans les spécifications témoins. Pour des thèmes mieux connus, comme la partie sur la session d'entraînement, le nombre d'exigences final aurait vraisemblablement été similaire, mais il reste que la différence au niveau du taux d'exigences liées à une exigence de haut niveau serait restée.

4.4.2.3 Performance de la technologie utilisée

Pour ce qui est de l'évaluation de la technologie, trois critères ont été évalués :

- **Confidentialité :** CAE utilise le système SecureID de la firme RSA Security qui permet une authentification double : chaque utilisateur doit s'identifier avec un nom d'utilisateur et un mot de passe, ce mot de passe étant constitué d'une série de 12 chiffres, 6 fixes qui sont connus seulement par l'utilisateur et 6 autres qui sont donnés par un générateur qu'il porte avec lui qui change de combinaison toutes les minutes. Les informations d'authentification sont cryptées à l'aide d'un algorithme AES (*Advanced Encryption Standard*, standard de chiffrement avancé) de 256 bits qui est depuis 2000 le chiffrement standard pour les organisations du gouvernement des États-Unis. Les communications suivantes

sont cryptées à l'aide d'un algorithme triple DES (*Data Encryption Standard*, standard de cryptage des données) de 163 bits. L'étude de ces algorithmes de cryptage est au delà de la portée de cette recherche, mais à titre de comparaison, la plupart des sites commerciaux utilisent un algorithme moins performant (RC4) sur seulement 128 bits. Il reste que même cet algorithme « moins performant » reste très puissant. Burt Kaliski [56] affirme que même avec une planète constituée d'ordinateurs de la taille d'un grain de sable, cela prendrait 1000 ans pour casser une telle clé. On peut donc conclure que les algorithmes utilisés par le VPN de CAE sont largement satisfaisants.

- **Performance** : pour mesurer la performance de la technologie, le temps de latence a d'abord été évalué. Celui-ci représente le temps nécessaire entre le moment où une instruction est envoyée au serveur et le moment où le résultat est reçu. Celle-ci est d'environ 7 ms depuis le réseau de CAE, 30 ms depuis Montréal, 70 ms depuis l'Europe (France, Allemagne) et 110 ms depuis l'Australie. Toutes ces mesures ont été prises depuis des connexions à haute vitesse, en partant du principe que les partenaires qui travailleront avec CAE en disposeront. Le débit de données quant à lui varie entre 1,5 Mbit/s à Montréal et 300 KBit/s en Australie. Ces performances sont satisfaisantes, mais imposent certains délais lors de l'utilisation de DOORS, notamment à l'ouverture et à la fermeture d'un module, lorsqu'une grande quantité de données sont échangées.
- **Stabilité** : une étude rigoureuse de la stabilité de cette solution n'a pu être conduite (elle demanderait une analyse sur une longue période de temps). Aucun problème n'a cependant été rapporté.

Les trois critères d'évaluation permettent de valider la technologie mise en place. Un petit bémol reste à noter sur le plan des performances. Bien que satisfaisantes, il serait souhaitable de se débarrasser des temps de chargement des modules. Cela sera cependant possible grâce à la prochaine évolution de DOORS (version XT) qui a été conçu pour limiter les données échangées.

Cela reste néanmoins une grande avancée du point de vue collaboratif par rapport à la solution actuelle. En effet, les projets doivent être échangés par archives de projets DOORS ou des feuilles Excel, ce qui pose beaucoup de problèmes de synchronisation des données, obligeant souvent certains partenaires à geler leur version du projet. De plus, selon la confidentialité des projets, les archives de projet sont soit envoyées par FTP (*File Transfer Protocol*, protocole de transfert de fichier), soit par média (par

exemple un CD-ROM) et acheminé par transport accrédité, ce qui peut prendre plusieurs jours.

4.4.2.4 Conclusion sur les résultats

Les résultats de l'expérience sur un projet réel sont concluants et valident

- **le processus proposé** : le nombre d'exigence, la volatilité des exigences et le profil de l'évolution de ceux-ci dans le temps sont comparable aux projets de référence.
- **la modélisation des exigences fonctionnelles** : grâce aux cas d'utilisation, un nombre plus important d'exigences ont été découvertes et en outre, les exigences sont en grande partie reliées à une exigence de plus haut niveau, ce qui n'était pas le cas auparavant et qui constitue une amélioration considérable.
- **la technologie** : celle-ci assure une confidentialité satisfaisante avec un temps de latence raisonnable pour les membres de l'équipe, quelle que soit leur situation géographique. Enfin, aucun problème de stabilité n'a été relevé.

CHAPITRE 5 : DISCUSSION

Ce chapitre discute des contributions théoriques dans une première partie puis dans contributions pratiques dans la seconde partie. Ensuite, les limites de la recherche seront abordées pour enfin terminer avec les avenues de recherche.

5.1 Contributions théoriques

Cette recherche s'est concentrée sur l'utilisation de la modélisation des exigences fonctionnelles pour faciliter la communication dans les projets dont l'équipe est distribuée géographiquement. La recherche a consisté non seulement à étudier cette piste, mais elle l'a également mise en œuvre dans un projet réel de l'industrie. Cette expérimentation dans le contexte réel de l'industrie apporte un fondement pratique aux théories trouvées dans la littérature, dont on ne trouve actuellement que peu de références faisant état d'une utilisation concrète.

Cette recherche contribue également sur le plan théorique dans la mesure où les résultats permettent d'alimenter les réflexions qui se font respectivement dans les problématiques de modélisation, d'ingénierie des exigences et de gestion d'équipes dispersées, mais aussi dans la combinaison de celles-ci. Cette recherche permet également de mieux comprendre les liens possibles entre ces trois thèmes de recherche.

5.2 Contributions pratiques

L'étude des différentes solutions pour utiliser DOORS à distance en collaboration avec d'autres utilisateurs a montré que la solution du VPN répond aux

attentes les plus importantes (sécurité et distribution des licences). Cette expérimentation constitue également un pas en avant vers la prochaine étape version de DOORS, la version XT qui sera basée sur une architecture à client léger. En effet, en utilisant DOORS sur un VPN, la seule chose qui diffère entre la solution expérimentée et DOORS XT (outre les fonctionnalités) est la performance qui est restée cependant acceptable au cours de l'expérience.

La définition de convention d'utilisation pour les diagrammes comportementaux d'UML pour modéliser les exigences fonctionnelles des projets de CAE a permis de mieux définir la source des exigences qui se trouvent dans la spécification du système par rapport à des projets similaires. De plus, la modélisation donne une plus grande confiance dans la couverture des exigences et ainsi minimise le risque.

Enfin, la documentation du processus d'IE de CAE (cartographie, dictionnaire et RAM) constitue un artefact qui permet de communiquer plus efficacement le processus à des partenaires. En effet, la cartographie donne une bonne vue d'ensemble des activités de chacune des phases du processus, ainsi que des acteurs et des livrables. Le dictionnaire décrit en détail chacune des activités. La RAM est une référence rapide aux responsabilités de chacun. Ces trois façons de représenter le processus sont complémentaires et donnent ensemble une représentation plus complète du processus que la documentation actuelle. La documentation étant plus complète et plus visuelle, il est plus facile d'identifier les goulots d'étranglement de faire évoluer le processus.

5.3 Limites de la recherche

Comme toute recherche, celle-ci comporte un certain nombre de limites qu'il faut reconnaître. Une première limite est liée au fait que la délocalisation des intervenants était relativement restreinte puisque seul le client était éloigné du reste de l'équipe; cette dernière étant situé dans d'autres locaux de l'entreprise. Certes, cela est suffisant pour

pouvoir parler de distance, mais une dispersion plus importante aurait permis d'explorer certaines autres dimensions, notamment la question du décalage horaire.

Deuxièmement, il faut noter que la recherche fut réalisée que sur un seul projet; de ce fait, il demeure difficile de généraliser les résultats à un ensemble élargi de projets. En dépit de cette limite, l'étude fournit néanmoins un aperçu de certains problèmes qui sont rencontrés aujourd'hui par l'industrie. Quant aux problèmes rencontrés au cours de cette étude, des pistes de solution ont été explorées et évaluées, ce qui permettrait à un autre projet de recherche de ne pas se heurter aux mêmes difficultés.

Une grande partie de la littérature part du principe que chaque projet est nouveau et que l'on repart de zéro. Cela est peut-être vrai dans les projets de logiciel, mais dans le contexte de CAE, même pour un projet de R&D, certains éléments du projet sont déjà connus avant même de l'avoir commencé. L'architecture de haut niveau par exemple est bien connue. Pour le projet étudié, le système visuel sera le même que sur les anciens simulateurs, avec certes quelques modifications. La solution étant déjà connue, moins d'effort va être mis pour la spécification des exigences relatives au rendu visuel par exemple, ce qui limite l'analyse des exigences. Cela représente donc des contraintes de design qui amoindrissent le champ d'action de l'IE par rapport à ce qui est proposé par la littérature.

5.4 Avenues de recherche

Cette recherche s'est concentrée sur le processus d'IE, mais ce dernier n'est qu'une partie du processus d'ingénierie des systèmes. Il serait intéressant d'étudier l'impact de la distribution sur d'autres processus, comme le design ou encore la vérification et validation.

La modélisation dans cette recherche s'est concentrée sur les concepts opérationnels, mais UML permet de modéliser bien d'autres facettes du système comme la structure

(diagramme de classe, diagramme de structure composite qui mettent l'emphasis sur la composition du système et des sous systèmes). Ces diagrammes introduisent des notions de l'approche par objets qui est à la base destinée au développement logiciel, mais qui peut s'adapter à un développement de système. Cela ouvre la voie pour une gestion efficace des interfaces et vers une plus grande réutilisation entre projets grâce au mécanisme de généralisation/spécialisation.

Enfin, l'Extreme Programming est une méthode dite « agile » de développement de logiciels. Cela signifie que l'emphasis est mise sur :

- l'interaction entre personnes plutôt que sur un processus et des outils,
- la collaboration avec le client, plutôt qu'une négociation de contrat,
- accepter et compter avec le changement des exigences.

La recherche conduite et l'Extreme Programming semblent s'opposer sur bien des points, notamment en ce qui concerne l'utilisation d'un processus. Cela vient du fait que cette recherche traite de projets complexes et de large envergure avec une équipe de développement distribuée, alors que l'Extreme Programming est destiné à des projets logiciels de petite ou moyenne envergure, développés par un binôme qui travaille sur un même ordinateur. Malgré ces différences, il y a certainement certaines choses à apprendre de l'Extreme Programming pour le développement de systèmes complexes notamment pour ce qui a trait de l'IE afin de rendre le processus plus léger et réactif.

CONCLUSION

Au travers d'une étude de terrain, cette recherche a permis de trouver une solution fonctionnelle aujourd'hui à un souci que rencontre l'industrie pour le développement de produit dont l'équipe de projet est distribuée géographiquement. Cette solution permet non seulement de pallier à certains défis pour le processus d'IE introduits par la distance, mais elle permet également d'améliorer le processus actuel, même dans un contexte local grâce à la modélisation des exigences fonctionnelles qui permettent une meilleure traçabilité entre la spécification des exigences système et les documents d'exigence de haut niveau.

La présente recherche avait pour but de proposer un nouveau processus d'IE pour CAE dans le cadre de leurs projets distribués. Le processus proposé fait appel à des outils technologiques qui permettent à des partenaires distribués géographiquement de disposer d'un réseau privé virtuel (VPN) sur la base d'Internet ce qui permet à l'équipe de projet de travailler sur une seule base de données d'exigences, ce qui mène à une meilleure collaboration. D'autre part, le processus proposé fait appel à des outils de modélisation comme UML qui permet de donner une représentation graphique des exigences de haut niveau. Le fait de développer les cas d'utilisation permet de mieux couvrir les exigences du système. Une des tâches essentielles de l'expérimentation a été de produire une documentation du processus proposé.

Cette recherche a permis de contribuer à éclaircir une « zone grise » de la recherche actuelle, à l'interface de la modélisation, la gestion d'équipes distribuées et l'ingénierie des exigences grâce à une étude empirique qui pourra servir de base à des études futures. Plus spécifiquement, l'utilisation d'un réseau privé virtuel comme support à l'utilisation d'un logiciel spécialisée en ingénierie des exigences a permis d'améliorer la collaboration entre les membres de l'équipe distribuée. D'autre part, pour modéliser les exigences fonctionnelles avec UML les diagrammes de cas d'utilisation ont été utilisés et ont permis de découvrir plus d'exigences du système. Enfin, le processus

proposé permet d'intégrer les modèles UML à la spécification des exigences du système grâce à des liens de traçabilité.

Bibliographie

- [1] HARWELL, R.M., (1993). What is a requirement? in *Proceedings of the Third International Symposium of the NCOSE*.
- [2] (2005). IEEE Standard for Application and Management of the Systems Engineering Process. *IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998)*: p. 0_1-87.
- [3] BLANCHARD, B.S., (1998). *System engineering management*. 2nd ed: New York ;Toronto : John Wiley & Sons. p.488.
- [4] AFIS, (2006). Maitriser l'ingénierie des exigences. 2006 Last Update [En ligne]; Disponible:
http://www.afis.fr/upload/SDD/RECHERCHE/JPM_050414_0002.pdf.
 [Consulté le 22/11/2006].
- [5] INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, (1998). IEEE guide for developing system requirements specifications. *IEEE Std 1233, 1998 Edition*.
- [6] WIEGERS, K., (2003). *Software Requirements*. Second Edition ed, Redmond, Washington: Microsoft Press. p.516.
- [7] OMG, (2006). SysML Specification. 2006 Last Update [En ligne]; Disponible:
<http://www.sysml.org/docs/specs/OMGSysML-v1-DraftAdopted-06-05-03.pdf>.
 [Consulté le 22/11/2006].
- [8] YOUNG, R.R., (2004). *The Requirements Engineering Handbook*, Norwood: Artech House. p.254.
- [9] INSTITUTE OF ELECTRICAL AND ELECTRONICS ENGINEERS, (2005). IEEE Standard for Application and Management of the Systems Engineering Process. *IEEE Std 1220-2005 (Revision of IEEE Std 1220-1998)*: p. 87.

- [10] HULL, E., K. JACKSON, and J. DICK, (2005). *Requirements Engineering*. Second Edition ed. Practitioner Series, London: Springer Verlag. p.193.
- [11] LLOYD, W.J., M.B. ROSSON, and J.D. ARTHUR, (2002). Effectiveness of elicitation techniques in distributed requirements engineering. Essen, Germany: IEEE Comput. Soc.
- [12] BOEHM, B., et al., (1999). Requirements engineering, expectations management, and the Two.
- [13] (2000). IEEE Recommended practice for architectural description of software-intensive systems. *IEEE Std 1471-2000*: p. i-23.
- [14] KANDÉ, M.M., et al., (2002). Bridging the gap between IEEE 1471, an architecture description language, and UML. *Software and Systems Modeling*. 1(2): p. 113-129.
- [15] YU, E.S.K., (1997). Towards modelling and reasoning support for early-phase. 1997 Last Update [En ligne]; 226-235]. Disponible: <http://www.cs.toronto.edu/pub/eric/RE97.pdf>. [Consulté le 22/11/2006].
- [16] CONRAD, B., (2006). SysML and UML 2 support for activity modeling. *Systems Engineering*. 9(2): p. 160-186.
- [17] OMG, (2005). UML 2.0 Superstructure Specification. 2005 Last Update [En ligne]; Disponible: <http://www.omg.org/cgi-bin/apps/doc?formal/05-07-04.pdf>. [Consulté le 03/10/2006].
- [18] BITTNER, K. and I. SPENCE, (2002). *Use Case Modeling*, Boston: Addison-Wesley Professional. p.368.
- [19] KRUCHTEN, P., (2000). *The Rational Unified Process, An Introduction*. 2nd ed, Reading, Massachussets: Addison - Wesley. p.295.

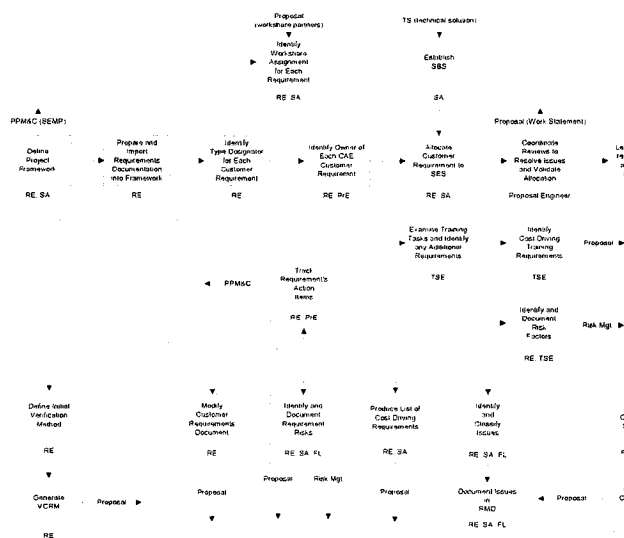
- [20] PIECHOCKI, L., (2000). UML en francais. 2000 Last Update [En ligne]; Disponible: <http://uml.free.fr/>. [Consulté le 03/12/2006].
- [21] (1998). IEEE guide for information technology - system definition - Concept of Operations (ConOps) document. *IEEE Std 1362-1998*.
- [22] ALEXANDER, I., (2002). Initial industrial experience of misuse cases in trade-off analysis. in *IEEE Joint International Conference on Requirements Engineering*. Essen.
- [23] HOFFMANN, M., et al., (2004). Requirements for requirements management tools. in *IEEE International Requirements Engineering Conference*. Kyoto.
- [24] JEAN-PIERRE, M., (1998). *Ingénierie et intégration des systèmes*, Paris: Hermes. p.543.
- [25] DANEVA, M. and R. WIERINGA, (2005). Requirements engineering for cross-organizational ERP implementation undocumented assumptions and potential mismatches. in *IEEE International Conference on Requirements Engineering*. Paris.
- [26] ROLLAND, C. and N. PRAKASH, (2001). Matching ERP system functionality to customer requirements. in *Fifth IEEE International Symposium on Requirements Engineering*. Toronto.
- [27] ATLE GULLA, J. and T. BRASETHVIK, (2000). On the challenges of business modeling in large-scale reengineering. in *4th International Conference on Requirements Engineering*. Schaumburg.
- [28] PROJECT MANAGEMENT INSTITUTE, (2004). *A guide to the project management body of knowledge (PMBOK)*, Newtown Square, PA: PMI. p.388.
- [29] AMERICAN SOCIETY FOR QUALITY, (2007). American Society for Quality Glossary. 2007 Last Update [En ligne]; Disponible: <http://www.asq.org/glossary>. [Consulté le 19/12/2006].

- [30] TURNER, J.R., (1992). *The Handbook of Project-Based Management*, New York: McGraw-Hill. p.357.
- [31] DAVIS, A.M., (1990). *Software Requirements Analysis and Specification*, Englewood Cliffs, New Jersey: Prentice Hall. p.516.
- [32] (2005). Adoption of ISO/IEC 15288:2002 Systems Engineering - System Life Cycle Processes. *IEEE Std 15288-2004 (Adoption of ISO/IEC Std 15288:2002)*: p. 0_1-67.
- [33] WIKIPEDIA, Update [En ligne]; Disponible: <http://www.wikipedia.org>. [Consulté le 01/10/2006.
- [34] FRENCH, A. and P. LAYZELL, (1998). A study of communication and cooperation in distributed software project teams. in *Proceedings. International Conference on Software Maintenance*. Bethesda.
- [35] KAROLAK, D.W., (1998). *Global Software Development, Managing Virtual Teams and Environments*, Los Alamitos, California: IEEE Computer Society. p.158.
- [36] LAYZELL, P., O.P. BRERETON, and A. FRENCH, (2000). Supporting collaboration in distributed software engineering teams. in *Proceedings of the Seventh Asia-Pacific Software Engineering Conference*. Singapore.
- [37] CAMPBELL, C.L. and B. VAN DE WALLE, (2003). Asynchronous requirements engineering: enhancing distributed software development. in *ITRE2003. International Conference on Information Technology: Research and Education*. Newark.
- [38] JACOBS, J., et al., (2005). Exploring defect causes in products developed by virtual teams. *Information and Software Technology*. 47(6): p. 399-410.

- [39] INTERNATIONAL, T.S.G., (1994). The CHAOS Report. 1994 Last Update [En ligne]; Disponible: http://www.standishgroup.com/sample_research/PDFpages/chaos1994.pdf. [Consulté le 19/11/2006].
- [40] LIPNACK, J. and J. STAMPS, (1997). *Virtual Teams: Reaching Across Space, Time, and Organizations with Technology*, New York: John Wiley & Sons. p.288.
- [41] EVARISTO, J.R., et al., (2004). A dimensional analysis of geographically distributed project teams: a case study. *Journal of Engineering and Technology Management*. 21(3): p. 175-189.
- [42] DAMIAN, D.E. and D. ZOWGHI, (2003). RE challenges in multi-site software development organisations. 8(3): p. 149-160.
- [43] SOSA, M.E., et al., (2002). Factors that influence technical communication in distributed product development: an empirical study in the telecommunications industry. *Engineering Management, IEEE Transactions on*. 49(1): p. 45-58.
- [44] DAMIAN, D.E. and D. ZOWGHI, (2003). Requirements Engineering challenges in multi-site software development organizations. *Requirements Engineering Journal*, (8): p. 149-160.
- [45] HERBSLEB, J.D. and D. MOITRA, (2001). Global software development. *Software, IEEE*. 18(2): p. 16-20.
- [46] DAMIAN, D.E., et al., (2000). The effects of communication media on group performance in. in *4th International Conference on Requirements Engineering*. Schaumburg.
- [47] DESANCTIS, G. and P. MONGE, (1998). Communication Processes for Virtual Organizations. 1998 Last Update [En ligne]; Disponible: <http://jcmc.indiana.edu/vol3/issue4/desanctis.html> [Consulté le 15/11/2005].

- [48] PESENDORFER, E.-M. and S. KOESZEGI, (2006). Hot Versus Cool Behavioural Styles in Electronic Negotiations: The Impact of Communication Mode. *Group Decision and Negotiation*. 15(2): p. 141-155.
- [49] HORMAN, J., (2005). Une exploration de l'interaction sociale en ligne lors de la réalisation d'activités d'apprentissage collaboratif dans deux espaces interactifs: un site internet et des wikis, in *Faculté des sciences de l'éducation*, Université Laval: Laval. p. 218.
- [50] SIVUNEN, A. and M. VALO, (2006). Team leaders' technology choice in virtual teams. *Professional Communication, IEEE Transactions on*. 49(1): p. 57-68.
- [51] ZHUGE, H., (2003). Workflow- and agent-based cognitive flow management for distributed team Cooperation. *Information & Management*. 40(5): p. 419-429.
- [52] EL EMAM, K. and N.H. MADHAVJI, (1995). Measuring the success of requirements engineering processes.
- [53] YPHISE, (2005). Gestion des exigences sur le cycle de vie des applications, in *Decider Volume*, Yphise, Editor, Yphise: Paris. p. 26.
- [54] HOUDE, R., M. BOURGAULT, and G. RETI, (2005). Implementing DOORS XT through a pilot project. *Telelogic UGC*: p. 24.
- [55] HOUDE, R., M. BOURGAULT, G. RETI, (2005). Implementing DOORS XT through a pilot project. in *Telelogic UGC*.
- [56] KALISKI, B., (2004). The Impending Demise of RSA? 2004 Last Update [En ligne]; Disponible: <ftp://ftp.rsasecurity.com/pub/cryptobytes/crypto1n1.pdf>. [Consulté le 12/11/2006].

For the engineering staff to properly scope and estimate the effort that will be required to develop, construct, transport, integrate, and support a training system, it is critical from the onset of the proposal preparation or an ECP that a clear understanding of the customer's needs, expectations, constraints, and interfaces be established and shared amongst all involved. This process step occurs in parallel with the "Understand Customer Requirements" step in the Proposal Process (MST-PRS-003), which also covers any ECPs. The REQ&M&D process defines the activities related to all technical and non-technical requirements. The TSE performs an analysis of the training tasks required by the customer and generates an operational concept and mission scenarios. The TSE identifies cost driving requirements resulting from the training task analysis.



2.1.1 Entry Criteria

The RFP or ECP kick-off meeting was held as per the Proposal Process (MST-PRS-003).

2.1.2 Inputs

Item	Source
Customer Requirements Document or	Customer
Customer Requirements (from a similar program) or	RE
Training Needs Analysis	TSE
SBS	SA
Workshare	Bid Manager/Proposal Engineer

2.1.1 Activities

1. **[Requirements engineer]** Receive customer contractual requirement documents from contracts organization or from project management.
2. **[Requirements engineer]** Prepare the requirements documentation for import into a requirements management database using the standard requirement database framework to document customer requirements and import the requirements documentation.
3. **[Requirements engineer, training solutions expert]** Prioritize customer requirements efficiency, if applicable.
4. **[Requirements engineer, systems architect]** Identify a type designator (mandatory, information, guidance, heading) for each requirement.

2.1.2 Outputs

Item	Destination
Project Specific Requirements Management Database	RE
Updated/Modified Customer Requirements Document (if required)	Customer
Compliance Matrix	Bid Manager/Proposal Engineer
VCRM	Bid Manager/Proposal Engineer
List of Identified Risks and Cost Driving Requirements	Bid Manager/Proposal Engineer
Updated Requirements Action Items	Project Engineer (PE)
Current Project SBS	SA
Documented Operational Concepts and Scenarios	TSE, Project Team

2.1.3 Exit Criteria


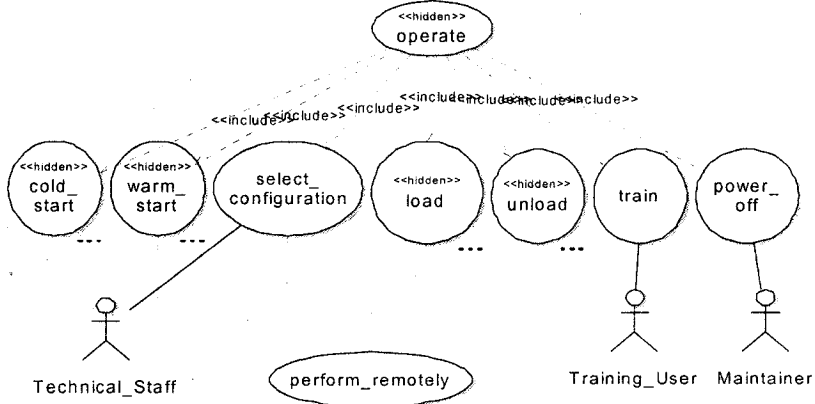
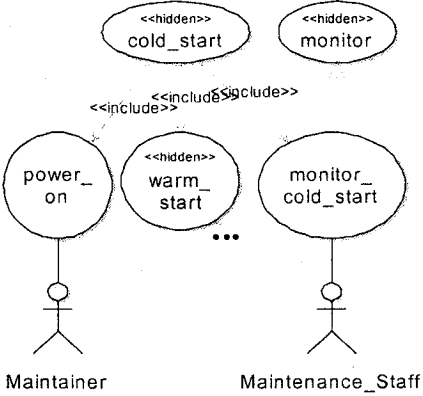
- Requirements management database has been populated.
- Preliminary VCRM is complete.
- Operational concepts and scenarios have been generated as required.

ANNEXE B :

CAS D'UTILISATIONS DE LA MAINTENANCE D'UN SIMULATEUR D'AVION

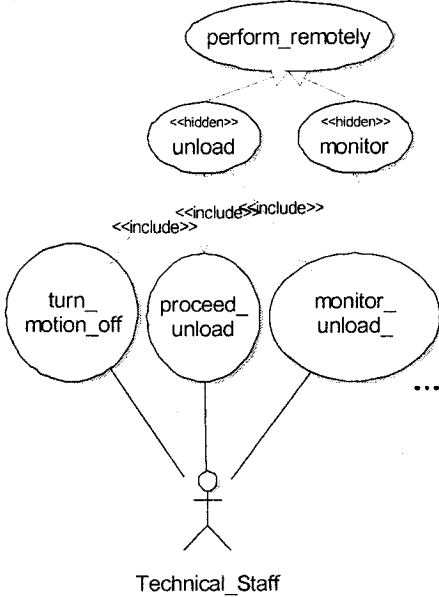
Absno	
17	1 General
18	1.1 Purpose
308	1.1.1 Purpose of the document
326	<p>The Operational Concept Description (OCD) describes a proposed system in terms of the user needs it will fulfill, its relationship to existing systems or procedures, and the ways it will be used.</p> <p>The OCD is used to obtain consensus among the acquirer, developer, support, and user agencies on the operational concept of a proposed system. Depending on its use, an OCD may focus on communicating the user's needs to the developer or the developer's ideas to the user and other interested parties. The term "system" or "Training Device" may be interpreted to apply to a portion of a system.</p>
309	1.1.2 Purpose of the system
19	1.2 Scope
310	1.2.1 Sope of the document
325	1.2.2 Scope of the system
402	<p>All operations discribed in this document are meant to be performed locally (from the Operation Server or Computer Room or from the Cockpit), unless they inherit from the "perform remotely" task (OR_362)</p>

38	2 Actors
54	2.1 Instructor
55	2.2 CAE
56	2.3 Maintainer
57	2.4 Trainee
360	2.5 Training_User
392	The Training User is either the Instructor or the Trainee
363	2.6 Maintenance_Staff
393	Maintenance Staff is either the Maintainer or staff from CAE
364	2.7 Technical_Staff
394	Technical Staff is either the Mainainer or the Instructor
366	2.8 Management
369	2.9 Users
372	<pre> graph TD MS[Maintenance_Staff] -.-> CAE[CAE] MS -.-> M[Maintainer] TS[Technical_Staff] -.-> M TS -.-> I[Instructor] TU[Training_User] -.-> I TU -.-> T[Trainee] </pre>
26	3 Conditions

32	4 Uses_Cases
76	4.1 General use cases
79	
33	4.2 Operate
62	4.2.1 Operate
65	
95	4.2.2 ColdStart
96	4.2.2.1 Cold Start
99	
100	4.2.2.2 power_on
315	The Maintainer proceeds to the power on sequence, which includes powering on :

329	- Power Distribution Cabinet (P1).
316	- all peripherals.
317	- Operation Server.
328	- Own Ship Node.
318	- Flight Compartment (if controlled by separate circuit breaker.)
319	- all other computers.
330	- TBD
209	4.2.2.3 monitor_cold_start
220	The Maintainer monitors power devices from Power Distribution Cabinet (P1).
331	The Maintainer monitors once the Operation Server is powered on and loaded.
168	4.2.3 Warm_Start
169	4.2.3.1 Warm Start
172	<pre> graph TD Remote([perform_remotely]) WarmStart([<<hidden>> warm_start]) Monitor([<<hidden>> monitor]) IncludeExclude[<<include>>] ProceedWarmStart([proceed_warm_start]) MonitorStart([monitor_start]) Staff((Maintenance_Staff)) Remote --> WarmStart Remote --> Monitor WarmStart --> IncludeExclude Monitor --> IncludeExclude IncludeExclude --> ProceedWarmStart IncludeExclude --> MonitorStart ProceedWarmStart --- Dots1[...] MonitorStart --- Dots2[...] Dots1 --- Staff Dots2 --- Staff </pre> <p>The diagram illustrates the 'Warm Start' process. It begins with a use case 'perform_remotely' which includes two hidden use cases: 'warm_start' and 'monitor'. These two use cases are grouped under an 'include/exclude' relationship. This relationship leads to two subsequent use cases: 'proceed_warm_start' and 'monitor_start'. Both of these use cases are connected to a stick figure actor labeled 'Maintenance_Staff' via three-dot lines, indicating an ongoing or repeated sequence of actions.</p>
191	4.2.3.2 proceed_warm_start
395	The Maintenance Staff proceed to the warm start sequence.
322	Warm Start is a subpart of cold start.

101	4.2.3.3 monitor_start
182	The Maintenance Staff monitors the start of the Training Device.
173	4.2.4 select_configuration
323	The Technical Staff selects the visual configuration.
333	The Technical Staff selects the loader configuration.
110	4.2.5 Load
111	4.2.5.1 Load
114	<pre> graph TD TS[Technical_Staff] --- PL((proceed_load)) TS --- MLP((monitor_load_progress)) PR((perform_remotely)) --> HL("<<hidden>> load") PR --> HM("<<hidden>> monitor") HL --> Dots[...] HM --> Dots Dots --- Include("<<include>>include>>") </pre>
115	4.2.5.2 proceed_load
311	The Technical Staff initiates the loading sequence.
117	4.2.5.3 monitor_load_progress
159	The Technical Staff monitors the loading sequence, including :
223	- loading state of each computing node
334	- TBD

118	4.2.6 Unload
119	4.2.6.1 Unload
122	 <pre> graph TD TS((Technical_Staff)) TS --- UC1((turn_motion_off)) TS --- UC2((proceed_unload)) TS --- UC3((monitor_unload_...)) UC1 --> UC4((perform_remotely)) UC2 --> UC4 UC3 --> UC4 UC4 -.-> UC5((<<hidden>> unload)) UC4 -.-> UC6((<<hidden>> monitor)) UC5 -.-> UC7((<<include>>)) UC6 -.-> UC7 UC7 -.-> UC8((<<include>>)) UC8 -.-> UC9((<<include>>)) UC9 -.-> UC10((<<include>>)) </pre> <p>The diagram illustrates the 'Unload' process. A stick figure actor labeled 'Technical_Staff' is connected to three use cases: 'turn_motion_off', 'proceed_unload', and 'monitor_unload_...'. These three use cases are all connected to a central use case 'perform_remotely'. From 'perform_remotely', dashed arrows point to two hidden use cases: '<<hidden>> unload' and '<<hidden>> monitor'. These hidden use cases are further connected to a series of nested include relationships, represented by '<<include>>' labels, indicating a complex sub-process.</p>
124	4.2.6.2 turn_motion_off
230	The Technical Staff turns the motion off.
190	4.2.6.3 proceed_unload
234	The Technical Staff initiates the unload sequence.
185	4.2.6.4 monitor_unload_progress
163	The Technical Staff monitors the unload progress, including :
224	- loading state of each computing node
336	- TBD
187	4.2.7 log_issues
202	The Training User writes issues encountered during the training session in the Issue Log.
128	4.2.8 train
238	Not developped
174	4.2.9 power_off

337	The Technical Staff proceeds to the power off sequence, which includes powering off :
343	- all computers other than the Own Ship node and the Operation Server.
342	- the Flight Compartment (if controlled by separate circuit breaker.)
341	- Own Ship Node.
340	- Operation Server.
339	- all peripherals.
338	- Power Distribution Cabinet (P1).
344	- TBD
34	4.3 Maintain
39	4.3.1 Maintain
42	<pre> graph TD Maintain((<<hidden>> maintain)) Scheduled((<<hidden>> scheduled_maintenance)) Unscheduled((<<hidden>> unscheduled_maintenance)) Monitor((<<hidden>> monitor)) Scheduled -.-> <<include>> Maintain Unscheduled -.-> <<include>> Maintain Monitor -.-> <<include>> Maintain Scheduled -.-> Ellipsis1[...] Unscheduled -.-> Ellipsis2[...] Monitor -.-> Ellipsis3[...] </pre> <p>The diagram shows a use case 'maintain' (labeled <<hidden>>) at the top. Below it are three use cases: 'scheduled_maintenance', 'unscheduled_maintenance', and 'monitor', all also labeled <<hidden>>. Each of these three use cases has a dashed arrow pointing to the 'maintain' use case, with the label <<include>> on the arrow. Below each of the three bottom use cases is an ellipsis (...).</p>

35	4.3.2 Scheduled_Maintenance
48	4.3.2.1 Scheduled Maintenance
51	<pre> graph TD Management((Management)) Maintainer((Maintainer)) consult_service_bulletin((consult_service_bulletin)) read_maintenance_schedule((read_maintenance_schedule)) morning_readiness((morning_readiness)) maintain_databases((maintain_databases)) maintain_CAE_s_w((maintain_CAE_s/w)) maintain_vendor_s_w((maintain_vendor_s/w)) maintain_h_w((maintain_h/w)) backup((backup)) restore((restore)) update_maintenance_schedule((update_maintenance_schedule)) perform_remotely_1((perform_remotely)) perform_remotely_2((perform_remotely)) scheduled_maintenance((scheduled_maintenance)) Management --- consult_service_bulletin Maintainer --- read_maintenance_schedule Maintainer --- morning_readiness Maintainer --- maintain_databases Maintainer --- maintain_CAE_s_w Maintainer --- maintain_vendor_s_w Maintainer --- maintain_h_w Maintainer --- backup Maintainer --- restore Maintainer --- update_maintenance_schedule Maintainer --- perform_remotely_1 Maintainer --- perform_remotely_2 consult_service_bulletin -.-> <<include>> read_maintenance_schedule read_maintenance_schedule -.-> <<include>> morning_readiness morning_readiness -.-> <<include>> maintain_databases maintain_databases -.-> <<include>> maintain_CAE_s_w maintain_CAE_s_w -.-> <<include>> maintain_vendor_s_w maintain_vendor_s_w -.-> <<include>> maintain_h_w maintain_h_w -.-> <<include>> backup backup -.-> <<include>> restore restore -.-> <<include>> update_maintenance_schedule update_maintenance_schedule -.-> <<include>> scheduled_maintenance scheduled_maintenance -.-> <<include>> perform_remotely_1 perform_remotely_1 -.-> <<include>> perform_remotely_2 perform_remotely_2 -.-> <<include>> update_maintenance_schedule </pre> <p>The diagram illustrates the 'Scheduled Maintenance' use case. It features two actors: 'Management' and 'Maintainer'. 'Management' is associated with the 'consult_service_bulletin' use case. 'Maintainer' is associated with 'read_maintenance_schedule', 'morning_readiness', 'maintain_databases', 'maintain_CAE_s/w', 'maintain_vendor_s/w', 'maintain_h/w', 'backup', 'restore', 'update_maintenance_schedule', 'perform_remotely', and 'scheduled_maintenance'. A series of 'include' relationships (dashed arrows) connect the use cases in a sequence: 'consult_service_bulletin' includes 'read_maintenance_schedule', which includes 'morning_readiness', which includes 'maintain_databases', which includes 'maintain_CAE_s/w', which includes 'maintain_vendor_s/w', which includes 'maintain_h/w', which includes 'backup', which includes 'restore', which includes 'update_maintenance_schedule', which includes 'scheduled_maintenance'. Additionally, 'scheduled_maintenance' includes 'perform_remotely', which includes 'update_maintenance_schedule'.</p>
88	4.3.2.2 consult_service_bulletin
204	The Maintainer or the Management reads release notes from the CAE Service Bulletin.
205	The Maintainer or the Management reads reports documenting known issues from the CAE Service Bulletin.
192	4.3.2.3 read_maintenance_schedule
206	The Maintainer reads the Maintenance Schedule to know which scheduled maintenance tasks is due.
89	4.3.2.4 maintain_databases
243	The Maintainer uploads a new database.
367	The Maintainer modifies a database (add, freeze... TBD).
90	4.3.2.5 maintain_CAE_s/w
305	The Maintainer downloads software updates from CAE.
306	The Maintainer uploads CAE software updates to the Training Device.
188	4.3.2.6 maintain_vendor_s/w
401	The Maintainer downloads software updates from the Vendor.
307	The Maintainer uploads vendor software updates to the Training Device,

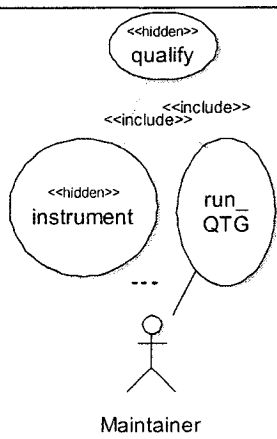
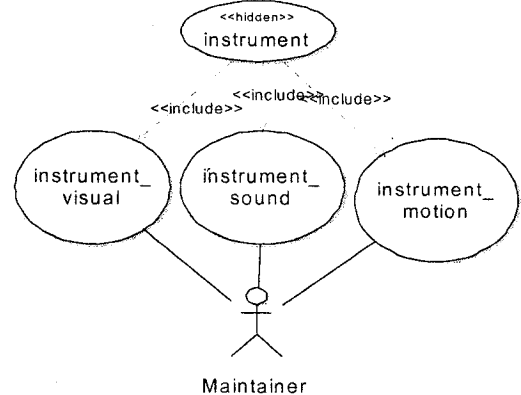
189	4.3.2.7 maintain_h/w
207	The Maintainer replaces parts (filters, fuse, etc.), as stated in Maintenance Guide LSF-0738 CAE Maintenance Manual.doc
208	The Maintainer calibrates hardware (IG, sensors, etc.) , as stated in Maintenance Guide LSF-0738 CAE Maintenance Manual.doc
92	4.3.2.8 backup
239	The Maintainer saves the data of the Operation Server and computing nodes to an external data storage.
93	4.3.2.9 restore
240	The Maintainer restores a previously saved set of data on the Host or computing nodes.
193	4.3.2.10 update_maintenance_schedule
246	The Maintainer indicates in the Maintenance Schedule which maintenance has been done.
247	The Maintainer adds new maintenance tasks to perform.
143	4.3.2.11 Morning_Readiness
144	4.3.2.11.1 Morning Readiness

147	<pre> graph BT Maintainer((Maintainer)) --- UC1([initiate_morning_readiness]) Maintainer --- UC2([monitor_morning_readiness]) UC1 -- include --> UC3([morning_readiness]) UC2 -- include --> UC4([monitor]) UC3 --- UC5([perform_remotely]) UC4 --- UC5 style UC3 stroke-dasharray: 5 5 style UC4 stroke-dasharray: 5 5 style UC5 stroke-dasharray: 5 5 </pre>
148	4.3.2.11.2 initiate_morning_readiness
373	The Maintainer initiates the Morning readiness sequence, which includes checks on :
374	- light bulbs
375	- flight controls
376	- sound
377	- motion
378	- visual
380	- database check
381	- computer nodes check
379	- TBD
368	4.3.2.11.3 monitor_morning_readiness
256	The Maintainer monitors the Morning Readiness.
382	The Maintainer checks the Morning Readiness results.

383	4.3.2.11.4 check_flight_controls
384	4.3.2.11.5 check_sond
385	4.3.2.11.6 check_motion
386	4.3.2.11.7 check_visual
387	4.3.2.11.8 check_morning_readiness_results
36	4.3.3 Unscheduled_Maintenance
43	4.3.3.1 Unscheduled_Maintenance
46	<p>The diagram illustrates the 'unscheduled_maintenance' use case, which is a container use case labeled '<<hidden>>'. It includes several sub-use cases: 'diagnostic', 'troubleshoot', 'view_impact_on_training', 'correct_s/w_issue', 'correct_h/w_issue', 'change_configuration', 'raise_issue', and 'warm_start' (also labeled '<<hidden>>'). The 'diagnostic' and 'troubleshoot' use cases are associated with the 'Maintenance_Staff' actor. The 'view_impact_on_training', 'correct_s/w_issue', 'correct_h/w_issue', 'change_configuration', and 'raise_issue' use cases are associated with the 'Maintainer' actor. The 'warm_start' use case is also associated with the 'Maintainer' actor. There are two 'perform_remotely' use cases, one associated with 'Maintenance_Staff' and one with 'Maintainer'. The diagram also shows several include relationships: 'diagnostic' includes 'troubleshoot'; 'troubleshoot' includes 'view_impact_on_training'; 'view_impact_on_training' includes 'correct_s/w_issue'; 'correct_s/w_issue' includes 'correct_h/w_issue'; 'correct_h/w_issue' includes 'change_configuration'; 'change_configuration' includes 'raise_issue'; 'raise_issue' includes 'warm_start'. Additionally, 'diagnostic' includes 'perform_remotely', 'troubleshoot' includes 'perform_remotely', 'view_impact_on_training' includes 'perform_remotely', 'correct_s/w_issue' includes 'perform_remotely', 'correct_h/w_issue' includes 'perform_remotely', 'change_configuration' includes 'perform_remotely', 'raise_issue' includes 'perform_remotely', and 'warm_start' includes 'perform_remotely'.</p>
184	4.3.3.2 diagnostic
389	The Maintenance Staff checks the status of all computing nodes (ready for load, loaded, issue, TBD).
390	The Maintenance Staff checks the status of the IO System.
391	The Maintenance Staff checks the status of mandatory softwares. (MOM server, boot manager, DCU, TBD) (loaded, unloaded, run-time)
278	The Maintenance Staff reproduces the issue.
262	The Maintenance Staff monitors labels.
263	The Maintenance Staff sets labels.
264	The Maintenance Staff runs CTS.
313	The Maintenance Staff collects variable values.
312	(The Maintenance Staff plugs a LabView)

154	4.3.3.3 view_impact_on_training
265	The Maintainer checks the impact of the non functioning systems on training.
155	4.3.3.4 troubleshoot
266	The Maintenance Staff reads documentation to find how to fix the issue. (wiring diagram)
349	The Maintenance Staff reads Diagnostics wich includes supplemental information about corrective action.
86	4.3.3.5 correct_s/w_issue
268	The Maintenance Staff reloads a crashed process.
314	The Maintenance Staff changes the CAL files. (CAL files are calibration files)
365	4.3.3.6 correct_h/w_issue
267	The Maintainer replaces a defective part.
186	4.3.3.7 change_configuration
270	The Maintainer enables or disables features in the configuration.
271	Available features are :
272	- cockpit type and version
273	- engine type and version
274	- database version
275	- number of visual projectors to be used
276	- (TBD)
175	4.3.3.8 raise_issue
279	The Maintainer raises an issue to CAE.
211	4.3.4 Monitor
212	4.3.4.1 Monitor
161	The User sees fatal errors in red, information and warnings in yellow, and successful operations in green.
345	The Instructor gets operation level informations.
347	The Maintainer gets maintenance level informations.

348	The User sees operational level informations by default.
346	4.3.4.1.1
215	<pre> graph TD monitor("<<hidden>> monitor") check_progress((check_progress)) abort_process((abort_process)) check_state((check_state)) check_errors((check_errors)) check_results((check_results)) perform_remotely((perform_remotely)) monitor -.-> <<include>> check_progress monitor -.-> <<include>> abort_process monitor -.-> <<include>> check_state monitor -.-> <<include>> check_errors monitor -.-> <<include>> check_results monitor -.-> perform_remotely </pre>
216	4.3.4.2 check_progress
280	The User checks the state of advancement in percentage of accomplished job and estimation of the remaining time.
282	The User checks activity of the process every 5 seconds.
217	4.3.4.3 check_errors
283	The User checks errors that occur during the task that is monitored.
218	4.3.4.4 check_results
284	The User checks a report with the results of the current task including following informations :
286	- success of the task
287	- time taken
288	- warnings
289	- (TBD)
219	4.3.4.5 check_state
290	The User checks the state of the task as follows :
292	- running
293	- stalled
294	- failed
295	- finished
291	4.3.4.6 abort_process
296	The User can abort current task.

83	4.4 Qualify
129	4.4.1 Qualify
132	 <pre> graph TD Qualify((<<hidden>> qualify)) Instrument((<<hidden>> instrument)) RunQTG((run_QTG)) Maintainer((Maintainer)) Qualify -- "<<include>>" --> Instrument Qualify -- "<<include>>" --> RunQTG Maintainer -.- Instrument Maintainer -.- RunQTG </pre> <p>UML Use Case Diagram for 4.4.1 Qualify. The diagram shows a stick figure actor labeled 'Maintainer' connected to two use cases: '<<hidden>> instrument' and 'run_QTG'. Above these is a third use case '<<hidden>> qualify'. There are two '<<include>>' relationships: one from 'qualify' to 'instrument', and another from 'qualify' to 'run_QTG'. A dashed line connects the 'Maintainer' actor to the 'instrument' use case.</p>
133	4.4.2 run_QTG
297	The Maintainer runs QTG
135	4.4.3 Instrument
136	4.4.3.1 Instrument
139	 <pre> graph TD Instrument((<<hidden>> instrument)) Visual((instrument_visual)) Sound((instrument_sound)) Motion((instrument_motion)) Maintainer((Maintainer)) Instrument -- "<<include>>" --> Visual Instrument -- "<<include>>" --> Sound Instrument -- "<<include>>" --> Motion Maintainer --- Visual Maintainer --- Sound Maintainer --- Motion </pre> <p>UML Use Case Diagram for 4.4.3.1 Instrument. The diagram shows a stick figure actor labeled 'Maintainer' connected to three use cases: 'instrument_visual', 'instrument_sound', and 'instrument_motion'. Above these is a fourth use case '<<hidden>> instrument'. There are three '<<include>>' relationships: one from 'instrument' to 'instrument_visual', one from 'instrument' to 'instrument_sound', and one from 'instrument' to 'instrument_motion'. Solid lines connect the 'Maintainer' actor to each of the three lower use cases.</p>
140	4.4.3.2 instrument_visual
299	The Maintainer instruments the visual system.
300	The Maintainer callibrates qualification instruments.
141	4.4.3.3 instrument_sound
301	The Maintainer instruments the sound system.
302	The Maintainer callibrates qualification instruments.

142	4.4.3.4 instrument_motion
303	The Maintainer instruments the motion system.
304	The Maintainer calibrates qualification instruments.
362	4.5 perform_remotely
396	The user can perform this task locally
398	The user can perform this task through internet
397	The user can perform this task from a laptop
400	The user can perform this task from the cockpit
399	The user can perform this task from the computer room

ANNEXE C :

PROCESSUS D'INGÉNIERIE DES EXIGENCES PROPOSÉ.

1. Introduction

Ce processus est destiné aux projets distribués et locaux de CAE. Il est fait de telle sorte qu'il puisse s'adapter aussi bien aux projets de produits existants pour des clients, que des projets de nouveaux produits qui répondent à une demande du marché. Dans ce dernier cas, le « client » sera le marché.

La différence fondamentale entre ces deux situations est que dans le cas d'un projet d'un produit déjà existant, certains documents d'un projet antérieur pourront être réutilisés (p.e. TPD). De plus, la solution étant déjà connue (on dispose de l'architecture dès le début du projet), facilitera certaines étapes.

Il est également prévu pour pouvoir s'adapter aux situations où le développement du système est confié à plusieurs partenaires. Dans ce cas, les tâches correspondantes sont marquées en bleu.

2. Processus de développement et de gestion des exigences

2.1. Acquisition des exigences

Le but de l'acquisition des exigences est de recueillir et de comprendre toutes les attentes du client et de les documenter dans DOORS afin de préparer le TPD qui fera partie de la soumission.

Les exigences peuvent être recueillies sous forme d'un document électronique ou papier fourni par le client, ou encore découvertes interactivement avec le client.

Idéalement, cette étape ne devrait s'occuper que des exigences opérationnelles, mais il arrive souvent que le client spécifie des exigences techniques dont il faudra tenir compte lors de la soumission.

Après une évaluation des exigences par les partenaires, le TPD pourra être généré.

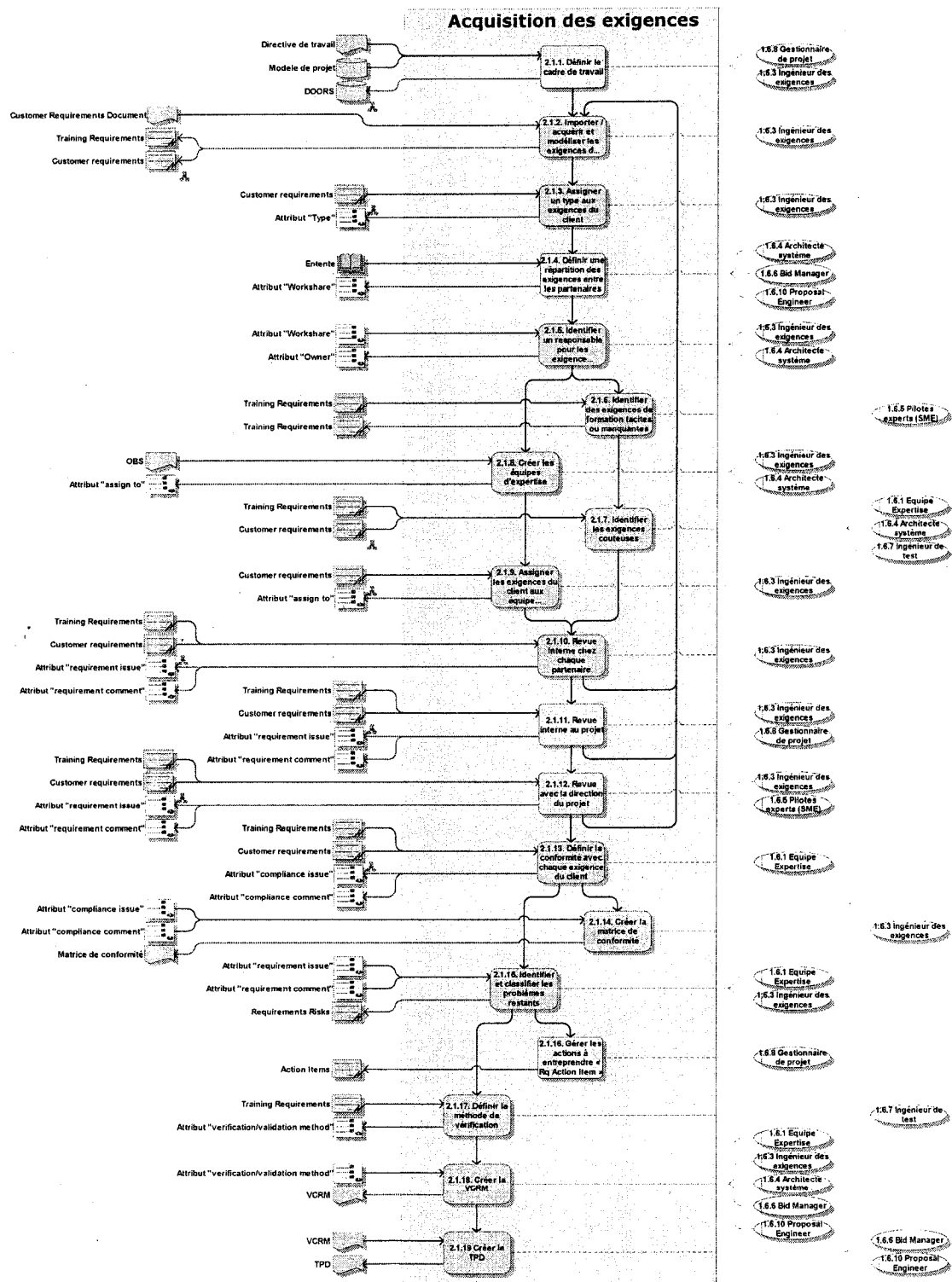


Figure 5.1: Cartographie du processus "Acquisition des exigences"

2.1.1. Définir le cadre de travail

Entrants
<ul style="list-style-type: none"> • Directive de travail (« work directive ») • Modèle de projet de CAE
Description
<ol style="list-style-type: none"> 1. Le modèle de projet de CAE (Project Framework) est instancié à l'emplacement du nouveau projet, puis adapté (nom, droits d'accès, modification de modules, attributs). 2. Le cadre de travail de travail collaboratif doit être défini. Les acteurs concernés devront s'accorder sur un processus de gestion des exigences, mode d'accès, de communication et de partage des données (VPN, partitions...). Une formation des partenaires peu familiers avec les outils ou avec le processus peut s'avérer nécessaire.
Sortants
<ul style="list-style-type: none"> • Nouveau projet dans DOORS

2.1.2. Importer / acquérir et modéliser les exigences du client

Entrants
<ul style="list-style-type: none"> • Document du client / Étude de marché
Description
<ol style="list-style-type: none"> 1. Les exigences du client sont importées dans DOORS, dans le document (1-1) Customer requirements. Les exigences ne seront alors plus changées, car celles-ci représentent les attentes du client en ses propres mots. 2. Créer des diagrammes de cas d'utilisations qui permettront de modéliser les besoins du client ou du marché. En fonction du niveau de détail nécessaire, il est possible de créer des diagrammes de cas d'utilisation et des diagrammes d'activités (liés à chaque cas d'utilisation). Les cas d'utilisations et les activités représentent des exigences de haut niveau. Les activités de modélisation et d'acquisition sont faites itérativement tant que de nouvelles exigences sont trouvées.
Sortants
<p>(1-1) Customer Requirements</p> <p>(1-2) Training Requirements (cas d'utilisation général)</p>

2.1.3. Assigner un type aux exigences du client

Entrants
<ul style="list-style-type: none"> • (1-1) Customer Requirements
Description
1. Assigner un type à chacune des exigences du client (obligatoire, recommandation, information, en-tête).
Sortants
Attribut « type » dans le document (1-1) Customer Requirements

2.1.4. Définir une répartition des exigences entre les partenaires

Entrants
<ul style="list-style-type: none"> • Entente signée entre les partenaires
Description
<ol style="list-style-type: none"> 1. Il est important d'obtenir un consensus entre les partenaires sur la répartition des exigences. Des règles claires de répartition peuvent être établies (p.e. par domaine d'expertise, par sous-système, par cas d'utilisation...). Il est cependant souhaitable d'éviter de confier un même sous-élément du projet à plusieurs partenaires différents. Si un consensus n'est pas trouvé, des négociations auront lieu. 2. Une fois la répartition effectuée, les droits d'accès dans la base de données sont adaptés en conséquence.
Sortants
<ul style="list-style-type: none"> • Attribut « workshare » dans le document (1-1) Customer Requirements

2.1.5. Identifier un responsable pour les exigences

Entrants
Attribut « workshare » dans le document (1-1) Customer Requirements
Description
1. Pour chaque exigence qui lui a été assignée lors du 2.1.4, chaque partenaire doit identifier les responsables et obtenir leur approbation.
Sortants
<ul style="list-style-type: none"> • Attribut « owner » dans le document (1-1) Customer Requirements

2.1.6. Identifier des exigences de formation tacites ou manquantes

<u>Entrants</u>
(1-2) Training Requirements
<u>Description</u>
1. Examiner les exigences de formation et vérifier s'il y a des exigences qui n'ont pas été exprimées par le client ou qui sont manquantes.
<u>Sortants</u>
Mise à jour de (1-2) Training Requirements

2.1.7. Identifier les exigences couteuses

<u>Entrants</u>
<ul style="list-style-type: none"> • (1-1) Customer Requirements • (1-2) Training Requirements
<u>Description</u>
1. Identifier les exigences qui entraînent le plus de coûts par rapport à la valeur pour le client. 2. Dans cette décision stratégique sera également évaluée la possibilité d'utiliser cette exigence sous forme d'option pour des projets futurs.
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut « CAE comment »

2.1.8. Créer les équipes d'expertise

<u>Entrants</u>
Organizational Breakdown Structure (OBS)
<u>Description</u>
1. Création des équipes d'expertise en fonction des domaines d'expertise requis par le projet.
<u>Sortants</u>
Attribut « assign to » dans le document (1-1) Customer Requirements

2.1.9. Assigner les exigences du marché aux équipes d'expertise

Entrants
(1-1) Customer Requirements
Description
1. Assigner les exigences du client aux équipes d'expertise créées en 2.1.8
Sortants
<ul style="list-style-type: none"> • Attribut « assign to » dans le document (1-1) Customer Requirements

2.1.10. Revue interne chez chaque partenaire

Entrants
(1-1) Customer Requirements
(1-2) Training Requirements
Description
<ol style="list-style-type: none"> 1. Chaque partenaire doit organiser une revue interne des exigences avec les équipes d'expertise afin de traiter les problèmes rencontrés et valider l'allocation. Les problèmes liés à des exigences mal formées sont marqués par l'attribut « Requirements issue », alors que les autres problèmes seront capturés dans l'attribut « comment » et seront discutés lors du 2.1.11 2. De nouvelles exigences peuvent être découvertes lors de cette revue. Dans ce cas ces exigences devront suivre le présent processus à partir du 2.1.2.
Sortants
<ul style="list-style-type: none"> • Attribut « Requirement issue » dans le document (1-1) Customer Requirements. • Attribut « CAE comment¹⁷ » dans le document (1-1) Customer Requirements. • Attribut « compliance comment » dans le document (1-1) Customer Requirements. • Attribut « compliance statement » dans le document (1-1) Customer Requirements.

¹⁷ Cet attribut doit être adapté au partenaire en question. Chaque partenaire dispose d'un attribut privé de commentaire.

2.1.11. Revue interne au projet

Entrants
(1-1) Customer Requirements (1-2) Training Requirements
Description
<ol style="list-style-type: none"> 1. Organiser une revue interne avec tous les partenaires du projet afin de traiter les problèmes rencontrés au niveau de l'organisation. La revue est faite en face à face lorsque cela est possible, sinon par vidéoconférence. Les problèmes non résolus seront discutés avec le client lors du 2.1.12. 2. De nouvelles exigences peuvent être découvertes lors de cette revue. Dans ce cas ces exigences devront suivre le présent processus à partir du 2.1.2.
Sortants
<ul style="list-style-type: none"> • Attribut « Requirement issue » dans (1-1) Customer Requirements. • Attribut « CAE comment¹ » dans le document (1-1) Customer Requirements. • Attribut « compliance comment » dans (1-1) Customer Requirements. • Attribut « compliance statement » dans (1-1) Customer Requirements.

2.1.12. Revue avec la direction du projet

Entrants
(1-1) Customer Requirements (1-2) Training Requirements
Description
<ol style="list-style-type: none"> 1. Cette revue doit permettre de s'assurer que les exigences comprises par l'organisation représentent bien les besoins du client. Elle sert également à résoudre les problèmes encourus et établir un consensus sur la solution à adopter. Dans le cas d'un projet pour le marché, le client peut être remplacé par un groupe d'experts. Cette réunion a lieu en face à face si cela est possible, sinon, par vidéoconférence. 2. De nouvelles exigences peuvent être découvertes lors de cette revue. Dans ce cas ces exigences devront suivre le présent processus à partir du 2.1.2.
Sortants
<ul style="list-style-type: none"> • Attribut « Requirement issue » dans (1-1) Customer Requirements. • Attribut « CAE comment¹ », « compliance comment » dans (1-1) Customer Requirements. • Attribut « compliance statement » dans (1-1) Customer Requirements.

2.1.13. Définir la conformité pour chaque exigence

Entrants
<ul style="list-style-type: none"> • (1-1) Customer Requirements • (1-2) Training Requirements
Description
<ol style="list-style-type: none"> 1. Pour chaque exigence de type « obligatoire » est établi le degré de conformité (« compliant », « partial compliant », « non compliant »). 2. Dans le cas où la conformité n'est pas totale, une justification doit être apportée avec le client dans l'attribut « compliance comment ».
Sortants
<ul style="list-style-type: none"> • Attribut « compliance statement » dans (1-1) Customer Requirements • Attribut « compliance comment » dans (1-1) Customer Requirements

2.1.14. Créer la matrice de conformité

Entrants
<ul style="list-style-type: none"> • Attribut « compliance statement » (1-1) Customer Requirements • Attribut « compliance comment » (1-1) Customer Requirements
Description
<ol style="list-style-type: none"> 1. Créer la matrice de conformité par rapport aux attentes du client.
Sortants
<ul style="list-style-type: none"> • Matrice de conformité

2.1.15. Identifier et classer les problèmes restants

Entrants
<ul style="list-style-type: none"> • Attribut « Requirements issue » (1-1) Customer Requirements. • Attribut « comment » dans le document (1-1) Customer Requirements.
Description
<ol style="list-style-type: none"> 1. Les problèmes qui subsistent après les différentes revues (interne, intra-organisation et avec le client) sont documentés et classifiés. Ceux-ci seront traités comme des risques dans le processus de gestion des risques (MST-PRS-010)
Sortants
<ul style="list-style-type: none"> • (0-2) Requirements Risks

2.1.16. Gérer les actions à entreprendre « Rq Action Item »

Entrants
Description
<ol style="list-style-type: none"> 1. Si les partenaires ont des choses à faire avant de passer à l'étape suivante, celles-ci seront marquées dans ce document. Les actions à entreprendre sont caractérisées par 3 données : quoi, qui et quand. 2. Les partenaires doivent indiquer leur état d'avancement sur ces actions dans ce même document (date de commencement, clôture).
Sortants
Le document (0-1) Action Items

2.1.17. Définir la méthode de vérification

Entrants
<ul style="list-style-type: none"> • (1-1) Customer Requirements
Description
<ol style="list-style-type: none"> 1. Définir une méthode de vérification pour les exigences techniques. (test, inspection, analyse, démonstration)
Sortants
Attribut « verification / validation method » dans le document (1-1) Customer Requirements

2.1.18. Créer la VCRM

Entrants
<ul style="list-style-type: none"> • Attribut « verification / validation method » dans le document (1-1) Customer Requirements
Description
<ol style="list-style-type: none"> 1. Créer la matrice de vérification (« verification cross reference matrix »), qui sera incluse dans la description technique du produit (TPD)
Sortants
<ul style="list-style-type: none"> • VCRM

2.1.19. Créer le TPD

<u>Entrants</u>
<ul style="list-style-type: none">• VCRM
<u>Description</u>
<ol style="list-style-type: none">1. Créer le TPD2. Dans le cas d'un projet pour le marché, le TPD servira à prendre la décision de continuer le projet ou non, et si ou, il servira de gabarit lors d'une commande.
<u>Sortants</u>
<ul style="list-style-type: none">• TPD

2.2. Analyse des exigences

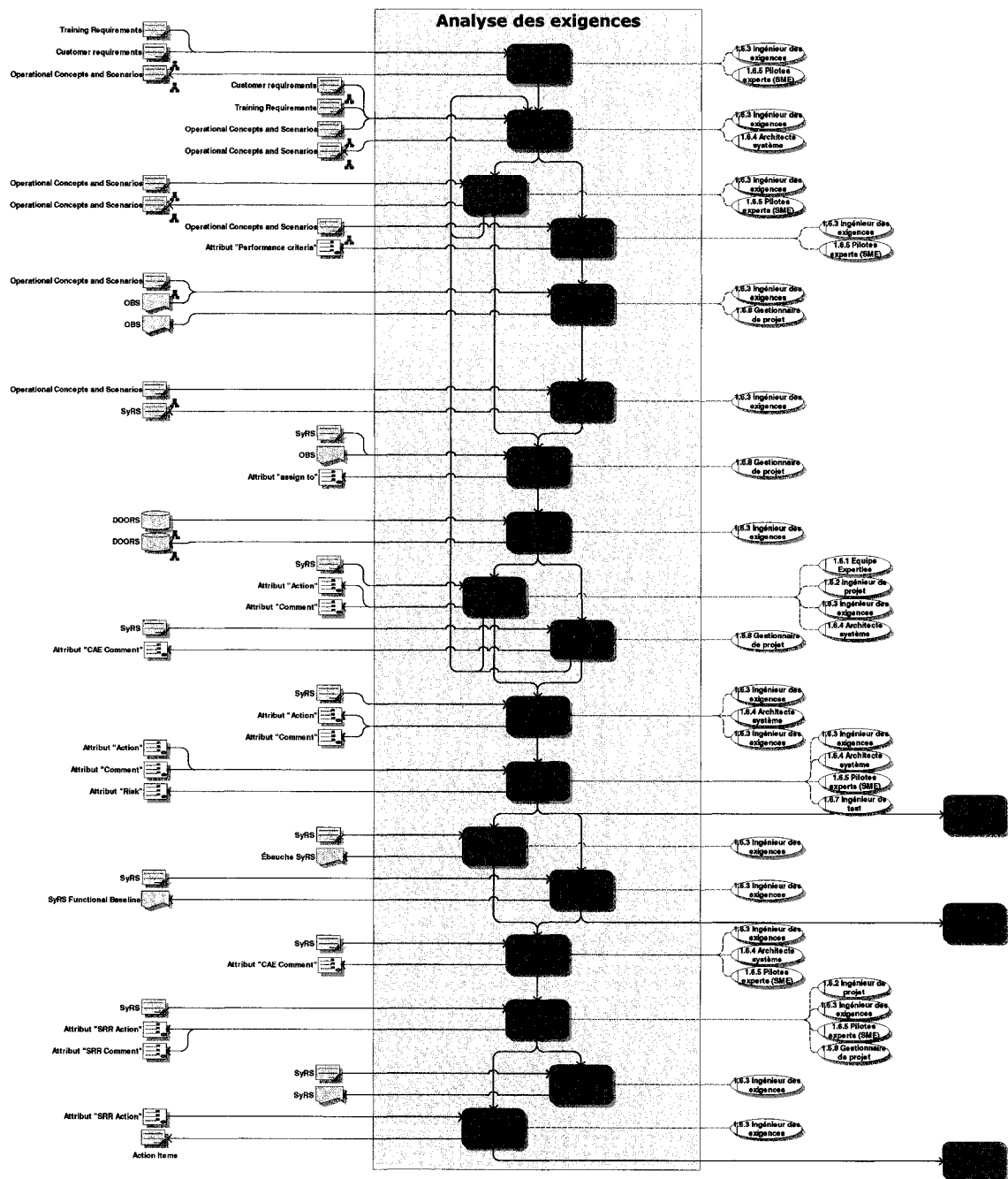


Figure 5.2: Cartographie du processus "Analyse des exigences"

2.2.1. Définir l'environnement opérationnel et de formation

Entrants
<ul style="list-style-type: none"> • (1-2) Training Requirements (Cas d'utilisations généraux) • (1-1) User requirements
Description
<ol style="list-style-type: none"> 1. Définir les limites ainsi que les contraintes du système. 2. Identifier les acteurs externes qui interagissent avec le système et définir les interfaces.
Sortants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios (Actors) • (2-2) Non-technical Requirements • (2-3) Interfaces Requirements Document

2.2.2. Développer les concepts opérationnels et les scénarios

Entrants
<ul style="list-style-type: none"> • (1-2) Training Requirements (Cas d'utilisations généraux) • (2-1) Operational Concepts and Scenarios • (1-1) User requirements
Description
<ol style="list-style-type: none"> 1. Développer les cas d'utilisations et les décomposer en scénarios (d'autres cas d'utilisations plus ciblés et des cas alternatifs) et/ou les détailler dans des diagrammes d'activités.
Sortants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios

2.2.3. Revue des concepts opérationnels et des scénarios

Entrants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios
Description
<ol style="list-style-type: none"> 1. Cette revue vise à détecter des scénarios manquants et de fait les exigences manquantes. Ces tâches (2.2.2 et 2.2.3) sont itératives et est répétées jusqu'à ce que le niveau de détail requis soit atteint. (celui-ci dépend du risque associé aux exigences en question)
Sortants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios

2.2.4. Analyser et quantifier les fonctionnalités requises

Entrants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios
Description
<ol style="list-style-type: none"> 1. Analyser et quantifier (s'assurer que les fonctionnalités demandées soient vérifiables, cf. IEEE 1233 Std. 1998) les fonctionnalités requises par et client et l'utilisateur.
Sortants
<ul style="list-style-type: none"> • Attribut "Performance criteria" dans (2-1) Operational Concepts and Scenarios

2.2.5. Analyser les exigences et identifier des équipes d'expertise manquantes

Entrants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios • OBS
Description
<ol style="list-style-type: none"> 1. Analyser les cas d'utilisations et les diagrammes d'activités pour identifier des domaines d'expertise manquants 2. Le cas échéant, créer les équipes d'expertise requises.
Sortants
<ul style="list-style-type: none"> • OBS

2.2.6. Dériver les scénarios en exigences système

Entrants
<ul style="list-style-type: none"> • (2-1) Operational Concepts and Scenarios
Description
<ol style="list-style-type: none"> 1. Dériver les exigences issues de la modélisation (cas d'utilisations et diagrammes d'activités) en exigences système.
Sortants
<ul style="list-style-type: none"> • (2-4) SyRS

2.2.7. Assigner les exigences système aux équipes d'expertise

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS • OBS
<u>Description</u>
1. Assigner les exigences système aux équipes d'expertise et obtenir leur accord.
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut "Assign to" dans (2-4) SyRS

2.2.8. Maintenir la traçabilité vers les exigences du client

<u>Entrants</u>
<ul style="list-style-type: none"> • Base de données DOORS
<u>Description</u>
1. Maintenir la traçabilité entre les exigences du client et les exigences système
<u>Sortants</u>
<ul style="list-style-type: none"> • Base de données DOORS

2.2.9. Valider les exigences système

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS
<u>Description</u>
1. S'assurer que les exigences système ont la même signification pour toutes les personnes concernées en interne
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut « CAE Comment » dans (2-4) SyRS

2.2.10. Examiner les exigences

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS
<u>Description</u>
<ol style="list-style-type: none"> 1. S'assurer que les exigences système expriment les mêmes fonctionnalités, limites, contraintes et implications que les exigences du client.
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut « CAE Comment » dans (2-4) SyRS

2.2.11. Conduire la revue interne des exigences système (ISRR)

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS
<u>Description</u>
<ol style="list-style-type: none"> 1. Conduire la revue des exigences système interne à chaque partenaire du projet avec les équipes d'expertises, fournisseurs externes, afin de s'assurer de l'accord des partenaires dans la signification des exigences 2. Discuter, le cas échéant, des problèmes existants avec les exigences afin d'identifier les actions à entreprendre pour les résoudre et identifier les nouveaux problèmes.
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut « Action » dans (2-4) SyRS • Attribut « Comment » dans (2-4) SyRS

2.2.12. Identifier les risques des exigences

Entrants
<ul style="list-style-type: none"> • Attributs « Comment » • « CAE Comment » dans (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Identifier les risques des exigences en se basant sur les problèmes qui subsistent. La réunion a lieu de préférence en face-à-face ou par vidéo conférence si ce n'est pas possible. 2. La liste des risques est transmise au gestionnaire de projet pour qu'il l'ajoute comme sujet pour le groupe de discussion, conformément au processus de gestion des risques (MST-PRS-010)
Sortants
<ul style="list-style-type: none"> • Attributs « Risk » dans (2-4) SyRS • Risk Management

2.2.13. Générer une ébauche de SyRS

Entrants
<ul style="list-style-type: none"> • (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Générer une ébauche de SyRS qui servira de base de discussion lors des différentes revues. 2. Envoyer l'ébauche de SyRS à toutes les parties prenantes qui seront présentes lors du SRR.
Sortants
Ébauche de SyRS papier

2.2.14. Générer une ébauche de référentiel fonctionnel

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS
<u>Description</u>
1. Générer un référentiel fonctionnel qui sera intégré dans l'ébauche de SyRS.
<u>Sortants</u>
<ul style="list-style-type: none"> • Functional Baseline

2.2.15. Revue de l'ébauche de SyRS chez chaque partenaire

<u>Entrants</u>
<ul style="list-style-type: none"> • (2-4) SyRS
<u>Description</u>
1. Organiser une revue de l'ébauche de SyRS chez chaque partenaire en s'assurant que toutes les parties prenantes soient présentes, conformément au processus de revue des partenaires (MST-PRS-020).
<u>Sortants</u>
<ul style="list-style-type: none"> • Attribut « CAE Comment » dans (2-4) SyRS

2.2.16. Mettre à jour les actions à entreprendre

<u>Entrants</u>
<ul style="list-style-type: none"> • Attribut « SRR action » dans (2-4) SyRS
<u>Description</u>
1. Mettre à jour la base de données d'actions à entreprendre.
<u>Sortants</u>
<ul style="list-style-type: none"> • Le document (0-1) Action Items

2.2.17. Générer le SyRS

Entrants
<ul style="list-style-type: none"> • (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Générer la version finale du SyRS qui sera figée dans un référentiel. 2. Envoyer le SyRS à toutes les parties prenantes.
Sortants
<ul style="list-style-type: none"> • SyRS papier

2.3. Revue des exigences système

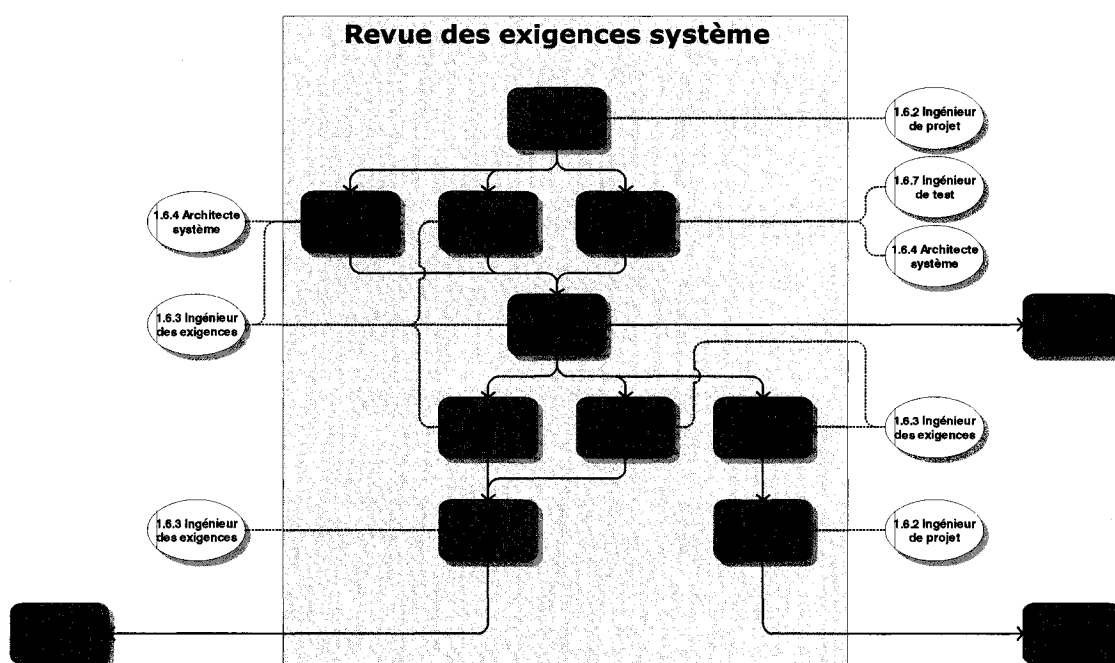


Figure 5.3: Cartographie du processus “revue des exigences système”

2.3.1. Tenir la revue des exigences système (SRR)

Entrants
<ul style="list-style-type: none"> • (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Préparer et tenir le SRR. Au cours de celui-ci seront exposés les exigences systèmes et les problèmes subsistants aux revues internes pour trouver des actions à entreprendre pour les résoudre et obtenir l'accord du client. Tous les partenaires doivent être présents à cette revue. Elle a lieu de préférence en face-à-face, ou si ce n'est pas possible, par vidéo conférence.
Sortants
<ul style="list-style-type: none"> • Attributs « SRR Comment » et « SRR action » dans (2-4) SyRS

2.3.2. Revue des problèmes d'exigence restants avec le client

Entrants
<ul style="list-style-type: none"> • (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Les problèmes restants sont exposés et revus avec le client afin de négocier une solution acceptable pour tout le monde.
Sortants
<ul style="list-style-type: none"> • Trade-off document

2.3.3. Revue de la traçabilité avec le client

Entrants
<ul style="list-style-type: none"> • (1-1) Customer Requirements • (1-2) Training Requirements • (2-1) Operational Concepts Document • (2-4) SyRS
Description
<ol style="list-style-type: none"> 1. Montrer au client la traçabilité du SyRS jusqu'aux exigences qu'il a formulées, afin de lui prouver que le système développé répond à tous ses besoins.
Sortants
Contrat

2.3.4. Présenter la VCRM et la philosophie de test au client

<u>Entrants</u>
VCRM Attribut « test method » dans (2-4) SyRS
<u>Description</u>
1. Présenter au client la philosophie des tests.
<u>Sortants</u>

2.3.5. Mettre à jour les exigences du client

<u>Entrants</u>
<u>Description</u>
1. Au cours des étapes précédentes, le client peut avoir fait des concessions sur certaines de ses exigences de ses exigences et le document des exigences du client est modifié en conséquence.
<u>Sortants</u>
(1-1) Customer requirements

2.3.6. Générer un SyRS mis à jour

<u>Entrants</u>
(2-4) SyRS
<u>Description</u>
1. Générer un nouveau SyRS.
<u>Sortants</u>
(2-4) SyRS

2.3.7. Mettre à jour la liste des risques

<u>Entrants</u>
(2-4) SyRS
<u>Description</u>
1. La liste des risques est mise à jour et transmise à la gestion du risque (MST-PRS-010)
<u>Sortants</u>

2.3.8. Créer un referential fonctionnel

<u>Entrants</u>
(2-4) SyRS
<u>Description</u>
1. Un nouveau référentiel fonctionnel est créé. A partir de maintenant, les changements apportés au SyRS devront être approuvés par le client suivant le processus de gestion du changement (MST-PRS-030).
<u>Sortants</u>
(2-4) SyRS

2.3.9. Mettre à jour les actions à entreprendre

<u>Entrants</u>
<u>Description</u>
1. Mise à jour de la liste des actions à entreprendre.
<u>Sortants</u>
(0-1) rq db items

2.3.10. Production des minutes du SRR avec le client.

<u>Entrants</u>
<u>Description</u>
1. Production des minutes du SRR en accord avec le client. Celles-ci serviront à améliorer les prochaines revues des exigences.
<u>Sortants</u>
SRR minutes

	Ingenieur de soumission
	Logistique
	Gestionnaire de projet
	Ingenieur de test
	Gestionnaire de soumission
	Équipes d'expertise
	Architecte système
	Ingenieur des exigences
	Ingenieur de projet
	Développeurs
Activities	

2.1 Acquisition des exigences

2.1.1. Définir le cadre de travail			2					1		
2.1.2. Importer / acquérir et modéliser les exigences du client			1							
2.1.3. Assigner un type aux exigences du client			1							
2.1.4. Définir une répartition des exigences entre les partenaires				2		1				2
2.1.5. Identifier un responsable pour les exigences			1	2						
2.1.6. Identifier des exigences de formation tacites ou manquantes					1					
2.1.7. Identifier les exigences couteuses	1			1			1			
2.1.8. Créer les équipes d'expertise			2	1						
2.1.9. Assigner les exigences du marché aux équipes d'expertise			1							
2.1.10. Revue interne chez chaque partenaire			1		2					
2.1.11. Revue interne au projet		2	1		2			2		
2.1.12. Revue avec la direction du projet		2	1		2			2		
2.1.13. Définir la conformité pour chaque exigence	1									
2.1.14. Créer la matrice de conformité			1							
2.1.15. Identifier et classier les problèmes restants	1		2							
2.1.16. Gérer les actions à entreprendre « Rq Action Item »								1		
2.1.17. Définir la méthode de vérification							1			
2.1.18. Créer la VCRM	1		1	1		1				1

2.1.19. Créer le TPD	1		1	1		1			1
----------------------	---	--	---	---	--	---	--	--	---

2.2 Analyse des exigences

2.2.1. Définir l'environnement opérationnel et de formation					1					
2.2.2. Développer les concepts opérationnels et les scénarios			2		1					
2.2.3. Revue des concepts opérationnels et des scénarios			2		1					
2.2.4. Analyser et quantifier les fonctionnalités requises			1							
2.2.5. Analyser les exigences et identifier des équipes d'expertise manquantes		2	1	1						
2.2.6. Dériver les scénarios en exigences système		2	1	2						
2.2.7. Assigner les exigences système aux équipes d'expertise			1	2						
2.2.8. Maintenir la traçabilité vers les exigences du client			1							
2.2.9. Valider les exigences système						1				
2.2.10. Examiner les exigences			1	2						
2.2.11. Conduire la revue interne des exigences système (ISRR)	1	1	1	1						
2.2.12. Identifier les risques des exigences			2	1						
2.2.13. Générer une ébauche de SyRS			1							
2.2.14. Générer une ébauche de référentiel fonctionnel			1							
2.2.15. Revue de l'ébauche de SyRS chez chaque partenaire			1							
2.2.16. Mettre à jour les actions à entreprendre		2	1							
2.2.17. Générer le SyRS			1							

2.3. Revue des exigences système

2.3.1. Tenir la revue des exigences système (SRR)		1	1	1	2			1		
2.3.2. Revue des problèmes d'exigence restants avec le client		1	2					1		
2.3.3. Revue de la traçabilité avec le client			1					1		
2.3.4. Présenter la VCRM et la philosophie de test au client		2	1				1			
2.3.5. Mettre à jour les exigences du client			1							
2.3.6. Générer un SyRS mis à jour			1							
2.3.7. Mettre à jour la liste des risques		2	1					2		
2.3.8. Créer un référentiel fonctionnel		2	1							
2.3.9. Mettre à jour les actions à entreprendre		2	1							
2.3.10. Production des minutes du SRR avec le client.			1							