Intégration de la Sûreté de Fonctionnement dans les Processus d'Ingénierie Système

Romaric Guillerm

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par *l'Université Toulouse III – Paul Sabatier*
Discipline ou spécialité : *Systèmes Industriels*

Présentée et soutenue par *Romaric GUILLERM*
Le *15 juin 2011*

Titre :

Intégration de la Sûreté de Fonctionnement dans les Processus d'Ingénierie Système

---

JURY

M. Antoine RAUZY, Chargé de Recherches à l'Ecole Polytechnique de Paris (Président)
M. Jean-Pierre BOUREY, Professeur à l'Ecole Centrale de Lille (Rapporteur)
M. Frédéric KRATZ, Professeur à l'ENSI de Bourges (Rapporteur)
M. Jean-Claude PASCAL, Professeur à l'UPS de Toulouse (Examinateur)
M. Hamid DEMMOU, Maître de Conférences à l'UPS de Toulouse (Directeur)
M. Nabil SADOU, Professeur-assistant à Supélec de Rennes (Co-directeur)

Ecole doctorale : *Systèmes*
Unité de recherche : *LAAS-CNRS*
Directeur(s) de Thèse : *M. Hamid DEMMOU et M. Nabil SADOU*

Remerciements

Le travail présenté dans ce mémoire a été réalisé au Laboratoire d'Analyse et d'Architecture des Systèmes (L.A.A.S.) du C.N.R.S. de Toulouse, au sein du groupe de recherche Ingénierie Système et Intégration (I.S.I.).

Mes remerciements vont en premier lieu à Monsieur Raja CHATILA, directeur du LAAS, pour m'avoir accueilli dans ce laboratoire pendant ces années de travail.

Je remercie également Monsieur Hamid DEMMOU, Maître de Conférence HDR à l'Université Paul Sabatier de Toulouse, de m'avoir accueilli au sein de son groupe de recherche I.S.I. et d'avoir encadré mon travail en m'apportant nombres de recommandations.

Ce travail a également été dirigé par Monsieur Nabil SADOU, Professeur-assistant à l'Ecole Supélec de Rennes, à qui j'exprime ma gratitude pour ses conseils, son soutien et sa confiance.

Je tiens à remercier Monsieur Antoine RAUZY, Chargé de Recherches au Laboratoire d'informatique de l'Ecole Polytechnique de Paris, de m'avoir fait l'honneur de présider mon jury de soutenance.

Je suis très reconnaissant envers Monsieur Jean-Pierre BOUREY, Professeur à l'Ecole Centrale de Lille, et Monsieur Frédéric KRATZ, Professeur à l'Ecole Nationale Supérieure d'Ingénieurs de Bourges, d'avoir accepté d'étudier mon travail et d'en être les rapporteurs ainsi que pour l'intérêt et l'attention qu'ils ont accordés à cette étude.

J'exprime toute ma gratitude à Monsieur Antoine RAUZY et à Monsieur Jean-Claude PASCAL, Professeur à l'Université Paul Sabatier de Toulouse, d'avoir accepté d'examiner ce travail.

Je voudrais également exprimer une profonde gratitude à Monsieur Abd-El-Kader SAHRAOUI, Professeur à l'Université de Toulouse Le Mirail, pour les nombreuses discutions que nous avons pu avoir et qui m'ont permis d'avancer dans mon travail, ainsi que pour son amitié.

Je remercie enfin l'ensemble des personnels du laboratoire LAAS, et plus particulièrement l'ensemble des membres (ou anciens membres) du groupe I.S.I., pour leur accueil. Je ne saurais oublier non plus mes collègues doctorants ou stagiaires qui m'ont tenu compagnie et avec qui j'ai passé de bon moment : Jean VERRIES, Jacqueline KONATE, Mourad MESSAADIA, Vincent ALBERT, Carlos GOMEZ, Vikas SHUKLA, Amar CHAALANE, Hassen GHARBI, Oumar KONE, Fallou GUEYE, Bernat GACIAS,

Frédérique BANIEL, Maria Alejandra AYALA PEREZ, Touria BEN RAHHOU, Samia OURARI, Mariem TROJET, Boadu Mensah SARPONG, Kata KIATMANAROJ et Panwadee TANGPATTANAKUL.

Le dernier mot s'adresse à mes parents, mon beau-père, mes sœurs et toute ma famille, mes amis et mes proches (que je ne pourrai malheureusement pas tous citer car il me manquerait de la place et je crains d'en oublier, je m'en excuse) pour leurs soutiens permanents à tous les niveaux et pour leur fidélité dans leur engagement à mes côtés, notamment dans les moments les plus difficiles.

Le 1 juillet 2011, à Toulouse

Romaric GUILLERM

Je dédie ce travail de thèse, aboutissement de mes études,

à ma Mère, sans qui rien n'aurait était possible.

Je le dédie également à une personne qui m'est très chère,

à laquelle je souhaite une même prochaine réussite.

Table des Matières

Table des Figures

Introduction Générale

L'intégration de diverses technologies, notamment celles de l'informatique et l'électronique, fait que les systèmes conçus de nos jours sont de plus en plus complexes. Ils ont des comportements plus élaborés et plus difficiles à prévoir, ont un nombre de constituants en interaction plus important et/ou réalisent des fonctions de plus haut niveau. Parallèlement à cette complexification des systèmes, la compétitivité du marché mondial impose aux développeurs de systèmes des contraintes de coût et de délais de plus en plus strictes. La même course s'opère concernant la qualité des systèmes, notamment lorsque ceux-ci mettent en jeu un risque de vies humaines ou un risque financier important.

Ainsi, les développeurs sont contraints d'adopter une approche de conception rigoureuse pour répondre aux exigences du système souhaité et satisfaire les diverses contraintes (coût, délais, qualité, sûreté de fonctionnement,…). Pour prendre en compte ces contraintes, un cadre a été défini. Il s'agit de l'Ingénierie Système (IS) qui est une démarche méthodologique pour maîtriser la conception des systèmes et des produits complexes. Cette démarche est définie à travers différentes normes visant à guider la conception de système.

La sûreté de fonctionnement, qui prend une place de plus en plus importante dans la conception des systèmes, doit être intégrée dans les processus d'ingénierie système. En effet, les propriétés de sûreté sont des propriétés émergentes. Elles résultent d'interdépendances qui existent dans le système et dans l'interaction de ce dernier avec son environnement. L'analyse de la sûreté de fonctionnement nécessite donc une démarche globale.

Le travail présenté dans ce mémoire a pour objectif la proposition d'une approche globale pour la prise en compte de la sûreté de fonctionnement. Il s'appuie sur la norme EIA-632, qui est largement employée, en particulier dans les domaines aéronautique et militaire. Il consiste à améliorer les processus d'ingénierie système décrits par l'EIA-632, afin d'intégrer une prise en compte globale et explicite de la sûreté de fonctionnement. En effet, jusqu'à présent la sûreté de fonctionnement était obtenue par la réutilisation de modèles génériques après avoir étudié et développé chaque fonction indépendamment. Il n'y avait donc pas de prise en compte spécifique des risques liés à l'intégration de plusieurs technologies. L'intégration de la sûreté de fonctionnement dans les processus de l'IS offre un cadre structurant pour mener les analyses dans un contexte plus large que celui traditionnellement rencontré dans le milieu fiabiliste.

Un des processus les plus critiques de l'IS est celui de l'ingénierie des exigences. Pour cette raison, nous proposons de nous intéresser aux exigences de sûreté de fonctionnement au niveau global et le plus tôt possible dans la phase de développement, pour ensuite les décliner aux niveaux inférieurs, ceci en s'appuyant sur les processus de la norme EIA-632

que nous étoffons. Nous proposons également une méthode originale de déclinaison d'exigences de sûreté de fonctionnement à base d'arbres de défaillances et d'AMDEC.

Pour une meilleure gestion de la complexité des systèmes, la tendance actuelle est d'utiliser des approches se basant sur l'Ingénierie Système Basée sur les Modèles (ISBM), qui propose de s'appuyer fortement sur les modèles, ceci pour toutes les étapes du cycle de développement. Ainsi, une autre facette du travail présenté propose un modèle d'information basé sur SysML pour appuyer notre approche.

Survol de la thèse

Ainsi, trois axes principaux se dégagent des travaux présentés dans ce mémoire de thèse. Ceux-ci concernent :

1. **Une démarche d'Ingénierie Système intégrant la sûreté de fonctionnement**, qui explicite au niveau processus d'ingénierie système comment prendre en compte les aspects « sûreté de fonctionnement » pour la conception.
2. **Une méthodologie de déclinaison d'exigences de sûreté de fonctionnement**, qui aide à la définition et la déclinaison d'exigences de sûreté niveau système en exigences de sûreté allouées aux sous-systèmes (à différents niveaux).
3. **Un modèle d'information en SysML**, qui permet de collecter toutes les informations produites lors de la conception, d'appuyer et de guider cette conception.

La démarche d'Ingénierie Système intégrant la sûreté de fonctionnement

La démarche d'ingénierie système intégrant la sûreté de fonctionnement explique les activités à accomplir à chaque étape du travail de conception (définition des exigences, de la solution logique, de la solution physiques, etc.) pour concevoir un système sûr de fonctionnement. Le niveau de vision adopté ici est celui du processus. Nous ne considérons pas les méthodes, encore moins les outils. Cette vision processus n'impose pas de cycle de conception particulier (par exemples : cycle en V ou cycle en spirale). Néanmoins, les « allers-retours » entre définition des exigences et définition de la solution (logique et physique) semblent inévitables, d'autant plus que nous traitons des systèmes complexes. En fait, des choix de conception, des conflits entre exigences, voire même des difficultés techniques à satisfaire certaines exigences, ceci lors de la définition de la solution, amènent à définir de nouvelles exigences (ou modifier d'anciennes) et à les « réinjecter » dans la spécification. C'est particulièrement le cas pour un certain nombre d'exigences concernant les propriétés de sûreté de fonctionnement, qui ne peuvent être identifiées, définies et/ou déclinées complètement qu'une fois une architecture fonctionnelle décidée, ou bien un choix d'architecture physique fait.

La méthodologie de déclinaison d'exigences de sûreté de fonctionnement :

Notre démarche de conception sûre, qui s'attache donc à traiter les aspects de sûreté de fonctionnement d'un système, implique entre autres une traçabilité rigoureuse. Tout élément de la conception doit justifier sa présence par l'existence d'exigences lui correspondant. Cela signifie que la source de la conception est fondée sur l'ensemble des exigences dans notre démarche. Ainsi, rendre le système sûr de fonctionnement consiste

entre autres à être capable de définir les exigences de sûreté de fonctionnement, de les traiter, de les décliner, et de les allouer. La méthodologie de déclinaison d'exigences de sûreté de fonctionnement que nous proposons permet d'aider à cette définition d'exigences de sûreté, tout d'abord au niveau du système. Puis, elle permet la déclinaison nécessaire de ces exigences de sûreté au niveau plus bas : celui des sous-systèmes. La méthodologie combine à la fois des AMDEC niveau système, des Arbres de Défaillances (AdD), et des AMDEC niveau sous-systèmes. L'AMDEC niveau système nécessite la solution logique du système pour pouvoir être réalisée. Elle est à l'origine d'actions correctives qui conduiront à des exigences de Sûreté de Fonctionnement (SdF) de niveau système. Ces actions correctives s'attachent à réduire soit la probabilité d'apparition du mode de défaillance, soit la gravité des effets du mode de défaillance. Concernant les actions correctives qui s'intéressent à la probabilité, l'utilisation d'un AdD, une fois la solution physique disponible, et d'AMDEC sous-système permettent de décliner les exigences niveau système en exigences niveau sous-systèmes. Pour les actions correctives qui s'occupent de la gravité, le niveau de spécification est aussi le niveau système. L'utilisation d'arbres d'événements (Event Tree) peut faciliter l'identification ou la compréhension de ces actions correctives.

Le modèle d'information en SysML

Nous souhaitons également appuyer la conception, avec sa génération de données d'ingénierie (exigences, solutions logique et physique), par un modèle d'information. Celui-ci permet de regrouper toutes les données de la conception au sein d'une base commune à tous les participants (ingénieurs système, ingénieurs de sûreté de fonctionnement, …). Bien entendu, doivent être inclus les différentes exigences, les fonctions identifiées, ainsi que les composants choisis, mais également un ensemble de liens entre ces éléments afin de rendre cohérent le modèle et d'expliquer l'utilité des diverses informations. Il s'agit là encore de l'aspect de traçabilité, très important pour une conception cohérente et sûre. Le langage bien adapté que nous avons choisi pour réaliser ce modèle est SysML (une extension d'UML à l'ingénierie système). Il permet de définir aussi bien les exigences, les fonctions et les composants, que les différents liens de traçabilité évoqués plus haut. L'intérêt d'un tel modèle d'information qui regroupe au sein d'un même modèle les aspects de spécification (les exigences) et de solution de conception (fonctions, composants et architectures associées) est qu'il permet une prise en compte rapide et efficace des évolutions des deux aspects en parallèle, définition des exigences et définition de la solution. La présence de ces deux aspects (exigences et solution) au sein du même modèle ne contrarie pas pour autant les recommandations des autorités de sûreté qui sont de ne pas mêler les exigences à la modélisation de la solution. En effet, SysML permet de clairement distinguer et séparer les deux aspects. Egalement pris en compte dans notre modèle, les « cas de tests » du langage SysML sont utilisés et liés aux exigences qu'ils vérifient. Devant être prévus dès l'étape de définition des exigences, ils permettent d'intégrer les aspects de vérification et validation dans le modèle.

Plan du manuscrit

Suite à cette introduction générale, le premier chapitre présente le cadre de travail qui est celui de l'ingénierie système. Il présente entre autres la gestion des exigences, la gestion des risques, ainsi que le domaine de la sûreté de fonctionnement. Ce chapitre finit par la

présentation de notre problématique, sur la base d'un constat fait sur les origines des problèmes de sûreté de fonctionnement des systèmes complexes.

Le deuxième chapitre du mémoire expose la démarche d'ingénierie système à base de processus qui prend en compte la sûreté de fonctionnement. Ce chapitre propose un état de l'art sur le sujet et présente les visions processus et normes d'ingénierie système. La norme EIA-632, sur laquelle nous nous appuyons, est présentée avant l'exposition de notre démarche processus proposée.

Le troisième chapitre concerne la méthodologie de déclinaison d'exigences de sûreté de fonctionnement. Le bien-fondé de cette méthodologie est justifié et celle-ci est également resituée dans les processus d'ingénierie système. Un état de l'art sur la déclinaison d'exigences de sûreté est développé, une formalisation UML est proposée et un cas d'étude simple illustre son application.

Le quatrième chapitre souhaite orienter la démarche de prise en compte de la sûreté de fonctionnement vers une démarche d'Ingénierie Système Basée sur les Modèles (ISBM). Le modèle d'information développé dans le cadre de ces travaux de thèse et basé sur le langage SysML sera exposé.

Le cinquième et dernier chapitre du mémoire présente un exemple issu du monde aéronautique qui sert d'illustration aux propositions des chapitres précédents. Il s'agit d'un avion de ligne fictif, sur lequel nous appliquons la méthodologie de déclinaison d'exigences de sûreté à différents niveaux et pour lequel nous fournissons une modélisation SysML utilisant notre modèle d'information.

Pour finir, une conclusion générale fait le bilan des contributions apportées, ceci par rapport à la problématique identifiée. Il s'agit également de présenter les perspectives des travaux afin de les améliorer et de les poursuivre.

Chapitre 1 : Ingénierie des systèmes complexes et sûreté de fonctionnement

1. Introduction

Ce premier chapitre du mémoire introduit le cadre de travail : l'ingénierie système. Il correspond au domaine de recherche du groupe ISI (Ingénierie Système et Intégration) du LAAS-CNRS au sein duquel cette thèse a été réalisée. Spécifiquement, nous traitons des systèmes complexes, dont nous donnons notre propre définition. Nous discernons quelles sont les principales caractéristiques de ces systèmes, et expliquons en quoi cette complexité influe sur les activités de développement.

Un aspect important de la conception système sur lequel se focalise le travail de thèse est la sûreté de fonctionnement [Laprie, 2004]. Nous présentons cette notion et décrivons comment elle intervient dans l'ingénierie des systèmes complexes.

Ainsi, dans ce chapitre, après la définition de ce qu'est un système complexe et après avoir exposé ses principales caractéristiques, nous abordons la discipline de l'ingénierie système. Les notions de cycles de vie, cycles de développement et approche de conception sont ensuite exposées.

Parmi les différents processus de l'ingénierie système, nous faisons un focus sur la gestion des exigences et la gestion des risques, qui importent énormément pour notre problématique. Puis nous présentons la sûreté de fonctionnement, avec ses concepts, quelques-unes de ses méthodes et son enjeu. Pour finir, nous exposons un constat sur les origines des problèmes de sûreté de fonctionnement des systèmes complexes, appuyé par quelques exemples de systèmes qui ont défaillis. La problématique de notre travail qui s'appuie sur ce constat est présentée à la fin de ce premier chapitre.

2. Systèmes complexes

2.1. Définition d'un système

Depuis toujours, l'homme invente et conçoit des instruments, des matériels, des équipements, … pour améliorer ou faciliter sa vie. Ces fabrications ont un intérêt direct ou indirect pour l'homme en apportant un service. La motivation de toutes ces constructions vient des fonctions apportées par le système, qui se manifestent par des interactions entre le système lui-même et l'environnement. Concernant les premières réalisations de l'homme,

celles-ci étaient relativement simples. Ainsi, la « structure » interne de la conception était très facilement compréhensible. Mais malgré leur simplicité, ces éléments constituaient déjà les premiers « systèmes ». Puis, ils ont sans cesse évolué en devenant de plus en plus élaborés et compliqués, surtout depuis le début de l'ère industrielle. Cependant, quelque soit le système, une caractéristique demeure. En effet, ils font tous l'objet d'association d'éléments (constituants, composants, …) en interaction dans le but de faire émerger de nouvelles fonctionnalités qui permettent de rendre un ou plusieurs services attendus.

Ainsi, nous donnons dès à présent une définition d'un système, extraite du livre « Ingénierie et Intégration des systèmes » de Jean-Pierre MEINADIER [Meinadier, 1998], qui est la suivante :

> ***Définition :*** *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.*

Nous constatons alors que par une généralisation, l'homme peut lui-même faire partie d'un système, à la manière d'un « constituant ».

L'AFIS (Association Française d'Ingénierie Système) [AFIS] a également sa propre définition du système, dans laquelle l'intérêt des interactions est clairement exprimé :

> ***Définition :*** *Un* ***système*** *est décrit comme un* ***ensemble d'éléments en interaction*** *entre eux et avec l'environnement, intégré pour rendre à son* ***environnement*** *les services correspondants à sa* ***finalité****. Un système présente donc des propriétés nouvelles résultant des interactions entre ses constituants : si l'on intègre des éléments pour faire un système, c'est bien pour bénéficier des effets de synergie résultant de leurs interactions.*

Une autre définition d'un système donnée par l'INCOSE (*International Council on Systems Engineering*) est la suivante :

> *Un* ***système*** *est un ensemble d'éléments interdépendants qui interagissent entre eux de façon organisée et forment un ensemble unique [INCOSE 2004].*

Finalement, nous pouvons conclure que les interactions constituent l'essence même des systèmes. Grâce à elles, de nouvelles propriétés apparaissent, dépassant celles propres aux simples constituants.

2.2. Complexité

Alors que bon nombre d'ouvrages fournissent une définition claire du mot « système », il n'en va pas de même pour l'expression « système complexe », qui s'impose pourtant largement aujourd'hui [Magee & al., 2004].

En revenant à l'origine de l'adjectif « complexe », il apparait que ce mot nous provient du latin *cum plexus*, qui signifie qu'il y a beaucoup d'intrications, que « tout est lié ». Le lien avec les interactions et leur nombre est alors acquis. Mais est-ce nécessaire et/ou suffisant pour former un système complexe ?

En se référant au dictionnaire, nous trouvons :

***Complexe**, adjectif*
Sens 1 : Qui comprend plusieurs éléments ayant de nombreux rapports entre eux. Synonyme : emmêlé. Anglais : complex.
Sens 2 : Difficile à appréhender, à analyser et en saisir le sens.
Synonyme : difficile. Anglais : complex, complicated.

Le sens 1 de la définition ci-dessous reprend l'idée d'une interaction importante entre les constituants du système, mais ajoute à cela le fait que ces éléments sont un certain nombre. Le sens 2 fait supposer que le fonctionnement du système n'est pas compréhensible immédiatement, il est difficile à suivre et à étudier. S'ajoute donc une difficulté de compréhension et d'analyse.

Nous avons alors tenté de donner notre propre définition d'un « système complexe », en tenant compte de ce qui précède mais aussi d'autres écrits que nous avons pu rencontrer pendant nos recherches.

***Système Complexe :** Système capable de fournir des fonctions de haut niveau, dont le comportement global est difficile à prévoir et dont la structure présente un graphe d'interaction non-trivial, souvent pourvu de boucles de rétroactions, et associant la plupart du temps plusieurs technologies de par l'implication de nombre de constituants.*

La complexité de ces systèmes a forcé les concepteurs à adopter et à s'adapter à une méthode de développement bien plus organisée et rigoureuse [Bar-Yam, 2005]. La section 3 présente cette nouvelle façon d'aborder la conception, nécessaire pour maîtriser cette complexité grandissante : l'Ingénierie Système.

2.3. Propriétés émergentes

Un concept important qui a fait son apparition avec les systèmes complexes est celui de « propriétés émergentes » [Leveson, 2004], [Black & al., 2009]. Elles désignent les caractéristiques du système qui apparaissent du fait de la mise en interaction des constituants. En effet, ces propriétés émergentes ne peuvent pas être attribuées à des composants seuls. Elles sont basées sur le fait que l'ensemble fait plus que la somme de ses parties.

L'émergence est un domaine de recherche à part entière, que nous n'explorons pas ici. Seulement, comme annoncé en introduction, nous allons nous intéresser aux propriétés de sûreté de fonctionnement des systèmes complexes qui sont fortement liées à un caractère émergent.

3. Ingénierie Système

Après avoir revu les concepts de système et système complexe, nous nous intéressons à la conception de ces systèmes. Plus particulièrement, nous exposons le domaine qu'est l'Ingénierie Système, à travers son historique, sa définition et quelques concepts comme le cycle de vie d'un système ou le cycle de développement. Nous terminons la section par deux

aspects inclus dans l'ingénierie système sur lequel l'essentiel du travail va s'appuyer : l'ingénierie des exigences et la gestion des risques.

3.1. Historique et définition

L'Ingénierie Système a fait son apparition aux Etats-Unis dans les années 50 pendant les courses à l'espace et au développement de missiles équipés de tête nucléaire. Pour ces derniers, considérés comme une nécessité absolue pour la sécurité du pays, les services militaires et leurs sous-traitants civils subissaient une pression énorme pour développer, tester et mettre en service ces missiles nucléaires le plus rapidement possible, de même que pour mettre en orbite des satellites d'espionnage. Ils se sont donc attachés à trouver des outils et des techniques pour améliorer la performance du système, et surtout celle de la gestion du développement (en termes de performance technique, d'échéance calendaire ou encore de contrôle du coût). A partir de ce moment, le management de l'ingénierie a alors beaucoup évolué, en standardisant l'utilisation de spécifications, de documents d'interfaces, de revues de conception ou encore de gestion de configuration formelle. Les progrès du monde informatique ont, également, énormément contribué à l'expansion de l'Ingénierie Système, facilitant les travaux de gestion et d'enregistrement de documents et de configurations. Les outils informatiques ont aussi permis un net progrès grâce à la possibilité d'effectuer toutes sortes de simulation.

En France, il existe depuis 1999 une association regroupant des entreprises concernées par l'Ingénierie Système : l'Association Française d'Ingénierie Système [AFIS]. On y trouve entre autre des entreprises comme Airbus, Alcatel Space, Thales, Giat Industrie, Renault, PSA Peugeot Citroën, France Télécom, EDF, RATP, etc. Il existe aussi une association internationale d'Ingénierie Système, implantée aux Etats-Unis : l'International Council on Systems Engineering (INCOSE). L'AFIS est d'ailleurs affiliée à l'INCOSE en tant que « chapitre français de l'INCOSE ».

L'Ingénierie Système est donc le domaine qui va aider le concepteur et lui permettre de mener son développement de la meilleure manière possible. Cette nouvelle science est finalement le résultat d'un retour d'expérience de la part de grandes entreprises industrielles impliquées dans le développement de systèmes technologiques complexes. En fait, deux raisons principales sont à l'origine de cette science. La première provient d'échecs industriels et commerciaux, car ils ont grandement motivé l'introduction d'une méthode d'aide, d'assistance ou de gestion afin de les éviter. Ils vont d'ailleurs continuellement être analysés pour améliorer l'Ingénierie Système. La seconde vient du constat qu'une grande part des activités à mener pour les développements est un invariant vis-à-vis des projets et des secteurs d'application. Les problèmes à affronter sont en effet souvent de même nature :

- des besoins insuffisamment exprimés,
- des spécifications imprécises ou incomplètes,
- des solutions non justifiées ou non validées,
- une formation des utilisateurs insuffisante,
- des ressources et compétences mal planifiées,
- etc.

Après ce rapide historique de l'avènement de l'Ingénierie Système, voyons la définition qu'en donne l'AFIS :

> *L'**Ingénierie Système** (ou ingénierie de systèmes) est une démarche méthodologique générale qui englobe l'ensemble des activités adéquates pour concevoir, faire évoluer et vérifier un système apportant une solution économique et performante aux besoins d'un client tout en satisfaisant l'ensemble des parties prenantes.*

Pour résumé, l'Ingénierie Système est donc une approche méthodologique qui a été développée et est en amélioration continue pour permettre au concepteur de système complexe d'aboutir à une solution équilibrée dans les meilleurs coûts et délais et répondant au mieux aux besoins des différentes entités concernées par l'utilisation du système (les clients, les agents de maintenance, d'autres systèmes,…).

3.2. Cycle de vie et cycle de développement

Lorsque l'on traite de l'Ingénierie Système, il est incontournable de considérer le cycle de développement du produit et son cycle de vie.

Le cycle de développement le plus connu, apparu d'abord dans le domaine informatique, est incontestablement le cycle en V [Forsberg & al., 1991a]. Celui-ci est composé de 2 branches. La branche descendante correspond à une démarche de raffinements successifs qui répond à la phase de conception, partant du général (l'expression des besoins souvent à travers un Cahier des Charges Fonctionnel (CdCF)) pour déboucher sur le particulier. La branche ascendante, quant à elle, détaille les phases d'intégration et de validation du système.

La Figure I.1 ne donne qu'un exemple de cycle de développement en V, car le nombre d'étapes et les noms des phases changent souvent d'un ouvrage à l'autre. Mais le principe reste toujours le même. On constate ici qu'une première phase de spécification démarre avec le CdCF en donnée d'entrée, produit le référentiel des exigences, encore appelé spécification technique des besoins, et prévoit d'ores et déjà le plan de validation du système qui sera utilisé en phase finale. Ensuite, l'étape de conception démarre pour aboutir au dossier de conception, composé d'une spécification de l'architecture du système et des spécifications techniques des besoins des constituants, et du plan et tests d'intégration. Peut alors commencer le développement des différents constituants qui vont composer le système final. Cette étape de développement peut d'ailleurs être vue comme la juxtaposition de multiples sous-cycles en V qui peuvent se dérouler en parallèle, à hauteur d'un cycle par constituants à concevoir. Si le constituant n'est pas réalisable par un seul métier (ou génie), le sous-cycle associé reprend celui du système. Pour finir, se succèdent les étapes d'intégration du système, pendant lesquelles les constituants sont assemblés entre eux en suivant le plan d'intégration préalablement établi, et de validation du système, pour vérifier que le système répond bien aux besoins initiaux (le CdCF) et en utilisant le plan de validation.

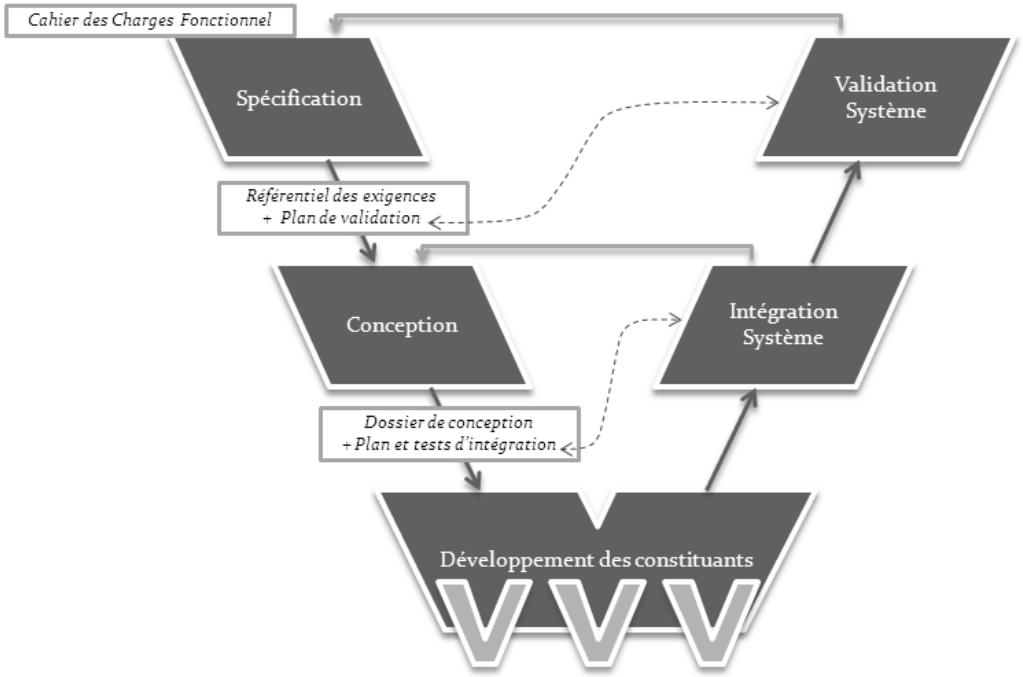


Figure I.1 : Cycle de développement en V d'un système

Le piège de ce cycle en V est la linéarité apparente d'une succession séquentielle des opérations. Car il n'en est rien. En pratique, il y a de nombreuses rétroactions des phases aval vers les phases amont, suite à des découvertes tardives d'erreurs obligeant à remettre en question des phases préalablement validées ou suite à une demande de modification d'exigences ou une évolution du marché par exemple. C'est pourquoi d'autres cycles de développement existent, avec notamment le modèle incrémental, le modèle à versions successives ou encore le modèle en spirale [Boehm, 1986].

Mais le champ d'action de l'Ingénierie Système ne se limite pas à la seule étape du développement du produit. En effet, elle agit sur tout le cycle de vie du système, depuis la phase de conceptualisation jusqu'à celle de retrait de service, comme le montre la Figure I.2.

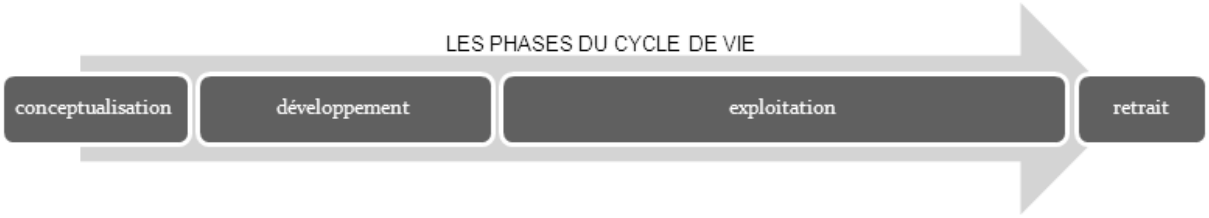


Figure I.2 : Cycle de vie d'un produit

3.3. Approche générale de conception des systèmes

L'Ingénierie Système fournit donc une démarche méthodologique qui suit le produit tout au long de sa vie, depuis ses premières conceptualisations, jusqu'à son retrait définitif du service, en passant par les phases de développement et d'exploitation. Le paragraphe qui suit explique de manière générale comment et sous quelles contraintes est conçu un système.

Tout d'abord, un nombre important d'acteurs peut être impliqué dans la réalisation du produit. Cela induit déjà un premier besoin en termes d'organisation pour gérer au mieux les communications entre tous ces acteurs. On trouve évidemment le développeur en charge de la conception (aussi appelé concepteur ou maître d'œuvre) et le ou les clients (ou utilisateurs). Mais bien souvent, on a également affaire à un maître d'ouvrage (pour le compte duquel est réalisé le système), à des sous-traitants ou des fournisseurs. Tous ces acteurs, mis à part le développeur lui-même, vont constituer ce que l'on appelle des parties prenantes du système (voir Figure I.3).

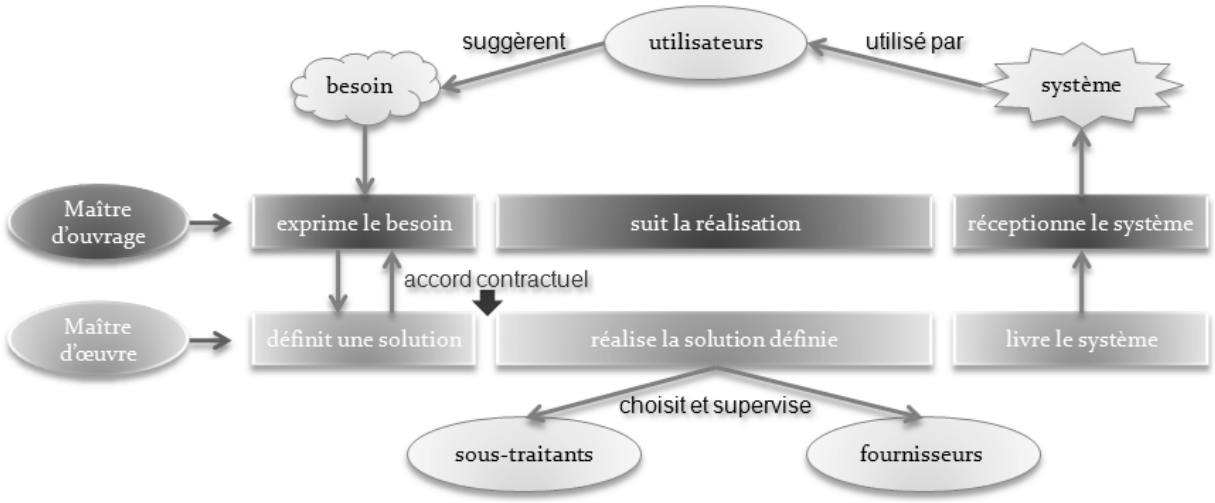


Figure I.3 : Vue globale de la conception d'un système

Le concepteur est donc en premier lieu informé des besoins des utilisateurs, que ce soit par l'intermédiaire d'un maître d'ouvrage ou non. A cet ensemble, il doit adjoindre également ceux des autres parties prenantes. Tous ces besoins vont alors être étudiés et analysés, puis progressivement transformés en une spécification faite d'exigences fonctionnelles et non-fonctionnelles qui ne sont rien d'autre que des énoncés plus ou moins formels des besoins de l'ensemble des parties prenantes.

Peut alors démarrer la phase d'étude fonctionnelle du système en commençant par une analyse externe, pour ensuite procéder à une analyse interne qui consiste à décomposer les fonctions en sous-fonctions et débouche sur l'architecture fonctionnelle du système.

Puis, la conception du système proprement dit commence par l'approche organique, qui consiste à découper le système en autant de sous-systèmes que nécessaire et à faire des choix d'organes matériels ou de modules logiciels. On obtient alors l'architecture organique du système, qui peut être vue comme une solution logique (voir Figure I.4).

Pour finir, il reste l'approche physique qui consiste à faire des choix d'organes physique pour répondre à l'architecture organique préalablement établie. Il s'agit ici de produire ce que l'on nomme : architecture physique du système. L'ensemble composé de l'architecture organique et l'architecture physique réunies forme ce que l'on appelle la synthèse de la solution.

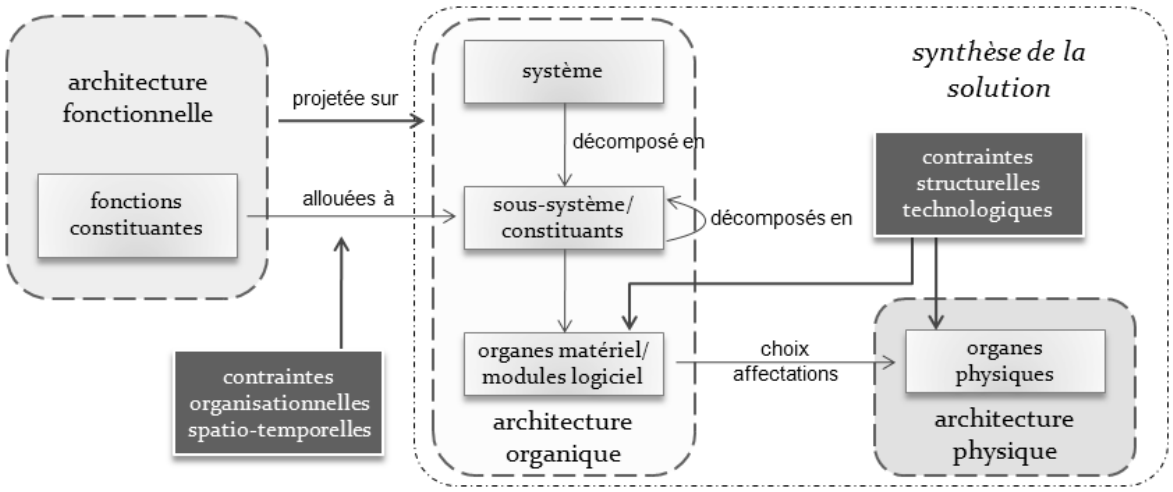


Figure I.4 : Démarche générale de conception

Lors du développement d'un système complexe, de nombreux domaines doivent travailler ensemble, coopérer et collaborer en vue d'obtenir une solution satisfaisante (voir Figure I.5). Nous distinguons 4 domaines principaux : le domaine du besoin, le domaine des génies, celui des spécialités et celui de la logistique. Typiquement, le domaine du besoin inclus le maître d'ouvrage et les multiples points de vue fournis par les manageurs, les opérateurs ou encore les utilisateurs. Le domaine des génies est certainement celui qui nous vient le plus rapidement à l'esprit lorsque nous pensons à la construction d'un système technique complexe, car il rassemble les différentes compétences : mécanique, thermique, électrique, logiciel, télécommunications, etc. Celui des spécialités dites transverses s'occupent de toutes les compétences interdisciplinaires, tel que la sûreté de fonctionnement, la sécurité ou encore l'ergonomie. Le dernier domaine à ne pas oublier est celui de la logistique. Il est absolument nécessaire si l'on souhaite respecter des délais ou encore une bonne qualité de service. Il traite notamment des aspects d'approvisionnement, de production, d'exploitation, de maintien en condition opérationnel ou de retrait du service.

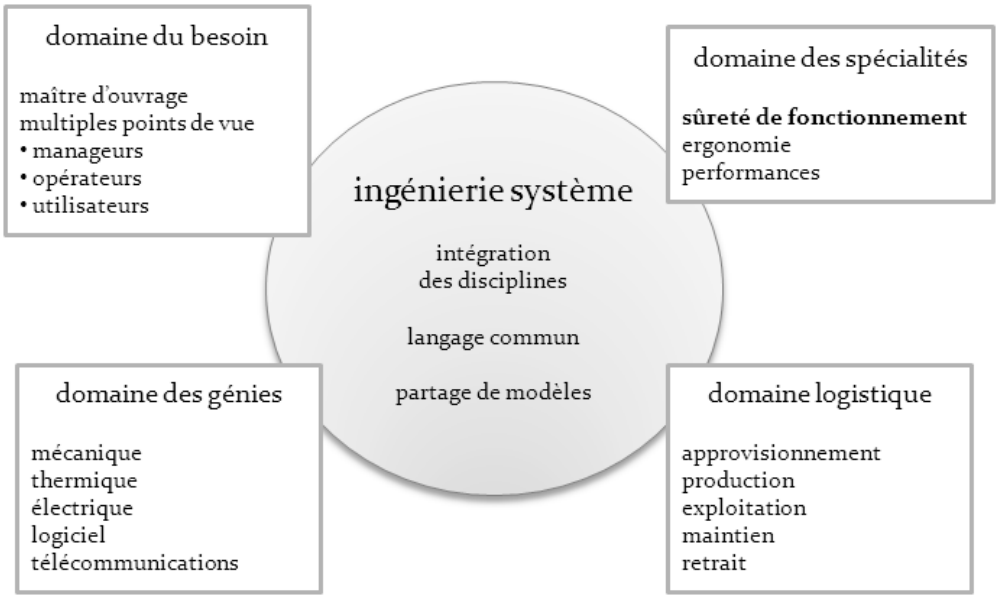


Figure I.5 : L'Ingénierie Système fédératrice de domaines

L'Ingénierie Système a alors pour rôle de faire cohabiter toutes ces disciplines en les intégrant au sein d'un même ensemble de processus. La tâche n'est pas évidente, surtout lorsqu'il s'agit de faire communiquer tous les acteurs. Pour cela, il semble nécessaire de s'appuyer sur un langage commun. Le moyen le plus sûr et qui permet le moins d'ambigüité et de divergence de compréhension reste sans aucun doute l'utilisation et le partage de modèles [Estefan, 2008]. Le concept de modèle reste en fait indissolublement lié à celui de système, du fait de la complexité du système, de son hétérogénéité et de sa pluridisciplinarité, car l'esprit humain n'appréhende la complexité qu'en la modélisant. En effet, comprendre ou concevoir ne revient qu'à créer des modèles mentaux et agir ou réaliser n'est rien d'autre que de se conformer aux modèles que l'on a construits. On peut citer 3 types de modèles [Verries, 2010] que l'AFIS classifie en fonction du rôle qu'ils jouent dans la conduite des activités d'ingénierie système (voir Figure I.6) :

- les modèles **cognitifs** : pour l'analyse et l'exploration du problème ou du besoin à l'origine du système et pour la validation des concepts opérationnels.
- les modèles **normatifs** :
  - de type **prescriptif** : pour la définition des exigences, la formalisation du besoin.
  - de type **constructifs** : pour la définition des solutions apportées ou envisagées au niveau de la conception architecturale du système.
- les modèles **prédictifs** : pour prévoir, estimer et valider le comportement du système vis-à-vis de ses exigences, à l'aide de modèles **formels** ou de modèles **analytiques** selon les techniques utilisées.

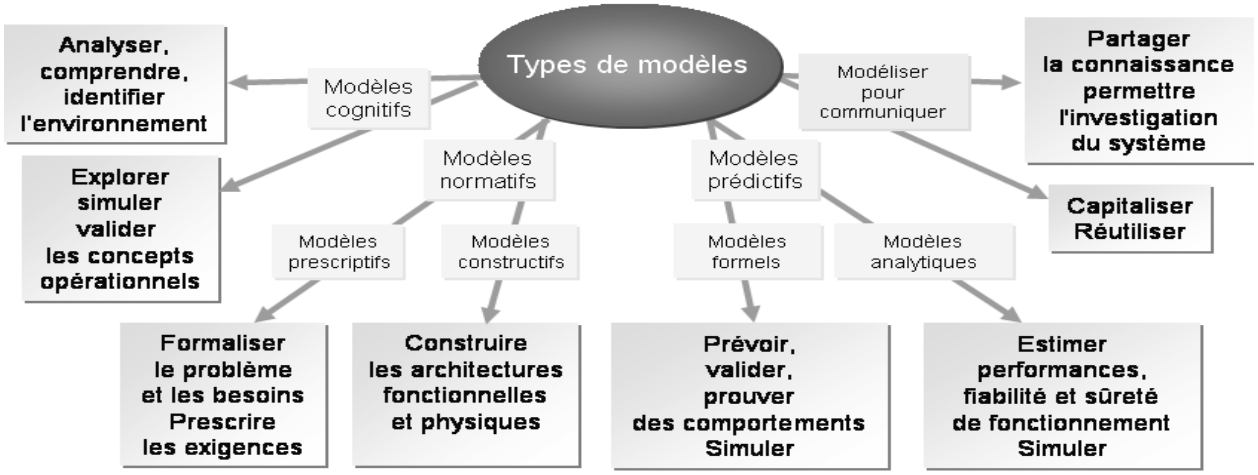


Figure I.6 : Typologie des modèles en IS (AFIS)

3.4. Ingénierie des exigences

Un point de l'Ingénierie Système sur lequel se concentre une partie de notre travail concerne l'ingénierie des exigences. C'est une part de l'ingénierie système très importante, qui est en charge de s'occuper de toutes les activités liées aux exigences telles que leur définition, leur traçabilité, leur modification, leur gestion en terme de maturité, etc. Nous verrons plus tard en quoi elle est importante pour notre problématique concernant la conception de systèmes sûrs de fonctionnement.

La section qui vient s'attache donc à présenter les principaux concepts et intérêts de cette ingénierie des exigences et commence, en premier lieu, par la définition d'une exigence.

3.4.1. Définition d'une exigence

Une exigence correspond à une expression de besoin bien formulée émanant du client ou de toutes autres parties prenantes en lien avec le système à développer. Elle transmet un besoin en fonctionnalité (exigence fonctionnelle) ou en qualité (exigence non-fonctionnelle) que doit satisfaire le produit qui est en train d'être conçu. Concernant les exigences non-fonctionnelles, celles-ci peuvent représenter :

- des contraintes globales de qualité de service,
- des aptitudes du système (fiabilité, opérabilité, convivialité, …),
- des contraintes opérationnelles (conformité à des normes d'utilisation),
- des contraintes de conception (réutilisation d'existant).

L'intérêt principal de transcrire les besoins en exigences réside dans la non-ambiguïté qui doit en découler à travers leurs formulations. De plus, cela apporte un bon support de communication entre les différentes parties prenantes du projet qui doivent collaborer.

3.4.2. Rôle et intérêt de l'ingénierie des exigences

Gérer les exigences [Buren & al., 1998] dans un projet est une activité fondamentale à son bon déroulement. En effet, un nombre important de documents peut être produit lors de la conception d'un système. Sans l'ingénierie des exigences, il serait quasiment impossible de garantir la cohérence et la qualité nécessaire au succès du projet. Effectivement, des études statistiques ont montré que les exigences interviennent pour environ 40% des succès ou échecs d'un projet, d'où leur importance vis-à-vis de notre préoccupation.

Ainsi, l'ingénierie des exigences permet de [Sommerville & al., 1997] :

- collecter les exigences,
- faciliter leur expression,
- détecter les incohérences entres elles,
- les valider,
- gérer leur priorité (les hiérarchiser),
- gérer les changements d'exigences,
- gérer la qualité,
- faire le lien avec le reste du projet et/ou avec le contexte,
- et encore assurer leur traçabilité.

L'ingénierie des exigences doit aussi veiller à ce que chaque exigence soit correctement déclinée, allouée, suivie, satisfaite, vérifiable, vérifiée et justifiée.

Nous saisissons bien l'importance de l'ingénierie des exigences dans un projet, pour sa réussite, et donc pour garantir que le système conçu répondra bien aux besoins et fonctionnera comme prévu. Tout écart vis-à-vis du respect des exigences pourra être à l'origine d'un fonctionnement non-souhaité, d'où le lien avec notre problématique. En

particulier, nous avons constaté l'importance des exigences liées aux interfaces, qui sont encore trop souvent la cause de problèmes de conception, et donc de retards et de surcoûts.

Ci-après, nous allons faire un focus sur trois aspects majeurs de l'ingénierie des exigences : l'expression des exigences, la traçabilité et le changement d'exigences.

3.4.3. Expression des exigences

Une bonne expression des exigences constitue un point clé de la réussite d'un projet. Toute ambiguïté ou tout oubli à ce niveau sera source de complication ultérieure, pouvant bien entendu s'en suivre de retard, de surcoût, de pénalité, etc. Il faut veiller à ce que les interprétations que peuvent se faire les différentes parties prenantes soient les mêmes. Pour cela, il convient d'utiliser des termes simples et d'éviter ceux ambigus ou vagues. Il est aussi recommandé de joindre schémas ou modèles normalisés qui peuvent éclairer l'exigence dès que cela est possible. (Nous rappelons ici l'adage bien connu : « un bon schéma vaut mieux qu'un long discours ».)

Outre la formulation même du « besoin » exprimé à travers l'exigence, à celle-ci peut également être associée d'autres attributs comme : le type (primaire ou dérivée), le niveau de conformité (obligatoire, conseil ou information), la priorité, la portée (exigence sur le système lui-même ou le programme) ou encore l'état (vérifiée, validée, …). Toutes ces informations ont leurs importances et doivent être tenues à jour.

3.4.4. Traçabilité

La traçabilité des exigences [Ramesh & al., 2001] est sans doute le concept le plus important dans l'ingénierie des exigences. Elle permet de connaitre facilement l'origine des exigences, ainsi que tous les liens entre les exigences elles-mêmes ou entre les exigences et le reste du projet ou le contexte (besoins utilisateur, réalisation, tests, …).

Dans la littérature [Ramesh, 1998], la traçabilité est citée comme facteur de qualité d'une bonne conception. Tout d'abord dans le but de décrire les connexions entre les différents niveaux d'exigences, elle présente un ensemble d'avantages qui permettent :

- de montrer plus facilement que la conception satisfait les exigences et d'aider à identifier rapidement quelles exigences ne sont pas satisfaites par la solution (autrement dit : de vérifier/valider la solution proposée),
- de ne perdre aucune justification vis-à-vis de choix de conception,
- de faciliter et maîtriser l'évolution du système dans le futur,
- de comprendre l'impact d'un changement d'exigence et de faciliter la prise en compte de l'évolution.

3.4.5. Changement d'exigences

Le changement d'exigences fait partie intégrante de l'ingénierie des exigences. Il est alors nécessaire de garantir une traçabilité, comme nous venons de l'expliquer ci-dessus. Mal suivi, le changement d'exigences a souvent été source de graves problèmes de conception. Ceux-ci entraînent des retards et des surcoûts néfastes à la survie de

l'entreprise, voire même des dysfonctionnements important du système portant atteintes à la sécurité de biens et/ou de personnes.

Une bonne ingénierie des exigences, avec une traçabilité complète où toutes les informations nécessaires sont présentes, doit permettre d'analyser et de prendre en compte l'impact de changements d'exigences. Changements qui deviennent d'ailleurs de plus en plus inévitables, d'une part, dû à la complexité des systèmes qui impose de faire des hypothèses initiales à revoir et ajuster plus tard dans le développement, d'autre part, dû à l'arrivée de nouvelles technologies ou plus simplement à cause de la rapidité de l'évolution du marché actuel.

Un autre aspect important qui complique encore la tâche d'ingénierie des exigences provient du fait que plusieurs parties prenantes et les différents acteurs de la conception interviennent pour définir et traiter les exigences. Derrière cette idée, il y a la notion d'un travail collaboratif, et donc d'une gestion collaborative des exigences. Ceci fait l'objet de nombreux travaux et constitue un domaine de recherche à part entière. Pour plus d'information, le lecteur peut par exemple se référer à [Konate, 2009] ou [Coulin, 2007].

3.4.6. Quelques outils d'ingénierie des exigences

Pour finir cette partie sur l'ingénierie des exigences, nous donnons quelques exemples d'outils d'ingénierie des exigences utilisés dans le monde industriel.

Le plus connu est incontestablement l'application DOORS d'IBM. C'est en effet le leader de ce marché, utilisé par les plus grandes entreprises dans tous les domaines (avionique, spatial, médical, défense, …). DOORS permet une structuration des exigences, en associant autant de champs que désiré à la définition des exigences. Il permet aussi de gérer des liens de traçabilité entre les exigences, dont la sémantique est à définir par l'utilisateur. En résumé, DOORS fournit un environnement complet et collaboratif de gestion des exigences.

Mais il existe en fait de très nombreux autres outils qui permettent de faire de la gestion des exigences, ayant chacun leurs avantages ou leurs inconvénients, en étant plus adapté à certains domaines qu'à d'autres. L'INCOSE propose d'ailleurs une liste (non-exhaustive) de ces outils, reprise en Figure I.7 (cf. http://www.incose.org/ProductsPubs/products/rmsurvey.aspx).

OUTIL	NOM DU VENDEUR
Accept Requirements (Accept 360)	Accept Software
Acclaro DFSS Version 5	Axiomatic Design Solutions, Inc.
Aligned Elements Version 1.5 (AE 1.5)	Aligned AG
Avenqo PEP Version 1.2	Avenqo
Blueprint Requirements Center 2010	Blueprint Software Systems, Inc.
Cameo Requirements+ Version 4.0	No Magic Inc.
CASE Spec Version 8.15	Goda Software
Cognition Cockpit (Cockpit) Version 5.1	Cognition Corporation
Contour by Jama Software (Contour) Version 2.9	Jama Software
CORE Version 5.1.5	Vitech Corporation
Cradle Version 5.7	3SL, Inc.
Dimensions RM (DimRM) Version 10.1.4	Serena Software
Enterprise Architect Version 7.1	Sparx Systems

Envision VIP Version 9	Future Tech Systems, Inc.
IBM Rational DOORS Version 9.2	IBM
IBM Rational RequisitePro Version 7.1	IBM
inteGREAT Version 4.7	eDev Technologies
IRQA Version 4	Visure Solutions
Kovair Global Lifecycle (Kovair) Version 5.5	Kovair Software, Inc.
MagicDraw and SysML Plugin Version 16.5	No Magic Inc.
MKS Integrity 2009	MKS Inc.
PACE Version 3	Viewset Corporation
Polarion Requirements Version 2	Polarion Software
Project & Test Engineering System (PTESY) Version 5.4	Andromeda s.r.l.
Psoda Version 5.02.1	e-LM.com Limited
RaQuest Version 3.0	SparxSystems Japan Co., Ltd
ReqMan Version 2.0	RequirementOne Inc.
Reqtify Version 2010-1A	Geensoft
Requirements Manager (ReMa)	Accord Software and Systems Pvt. Ltd.
RTIME Version 5	QAVantage
Teamcenter Requirements (Tc RM) Version 8	Siemens
TraceCloud	TraceCloud
What To Do Next (WTDN)	4SQ Solutions LLC
workspace.com Requirements Management	workspace.com

Figure I.7 : Outils de gestion d'exigences (INCOSE)

3.5. Gestion des risques

La gestion des risques est une activité importante pour le bon déroulement du projet de conception. C'est une activité d'ingénierie système que nous nous devons de présenter, vu notre problématique de conception sûre de fonctionnement, qui impose une gestion des risques.

Ainsi, dans cette section, nous donnons tout d'abord la définition d'un risque et nous justifions l'intérêt de l'étude des risques. Puis nous présentons les étapes, les stratégies et les outils de la gestion des risques.

3.5.1. Définition d'un risque

Un risque correspond à une perte ou une dégradation potentielle, identifiée et souvent quantifiable. C'est un danger éventuel plus ou moins prévisible qui peut affecter l'issue du projet. Il est nécessairement lié à une situation ou une activité et est associé à la probabilité de l'occurrence d'un événement ou d'une série d'événements. Il peut être caractérisé par une grandeur à deux dimensions : sa **criticité**. La Figure I.8 illustre cette caractéristique avec :

- en abscisse : la **sévérité** (ou **gravité**) des effets et des conséquences du risque considéré,
- en ordonnée : la **probabilité** d'occurrence du risque.

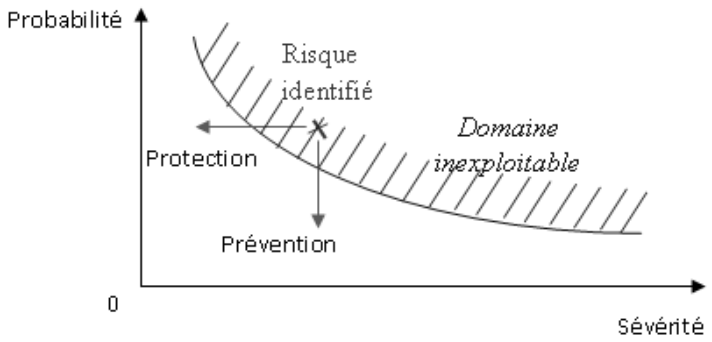


Figure I.8 : Criticité des risques

Sur cette Figure I.8, on peut d'ores et déjà identifier la stratégie de réduction du risque qui consiste à réduire sa probabilité d'occurrence (prévention) et/ou à diminuer sa sévérité (protection) pour rendre le risque acceptable.

En fait, la première définition scientifique du risque a été donnée en 1738 par Daniel Bernoulli dans "*Specimen theoriae novae de mensura sortis"* [Bernoulli, 1738] : « le risque est l'espérance mathématique d'une fonction de probabilité d'événements ». Plus simplement, il s'agit de la valeur moyenne des conséquences des événements pondérés par leurs probabilités.

Dans cette définition, nous retrouvons bel et bien la notion de probabilité et celle de sévérité par l'intermédiaire de la valeur représentative des conséquences des événements. Mais d'autres éléments sont à associer aux risques comme : les conséquences elles-mêmes, la stratégie de gestion du risque, l'outil employé pour appliquer la stratégie, et bien entendu le ou les facteurs de l'apparition du risque.

3.5.2. Pourquoi s'intéresser aux risques ?

La prise en compte des risques est essentielle pour une bonne conduite de projet. En effet, ils peuvent être responsables de l'échec du projet ou, tout du moins, d'une augmentation non négligeable du coût ou du délai de développement. S'intéresser aux risques permettrait donc d'éviter ce genre de problèmes. Cela permettrait de diminuer le nombre et le coût des accidents, d'empêcher les accidents handicapants ou mortels, ou plus simplement d'éviter l'insatisfaction du client.

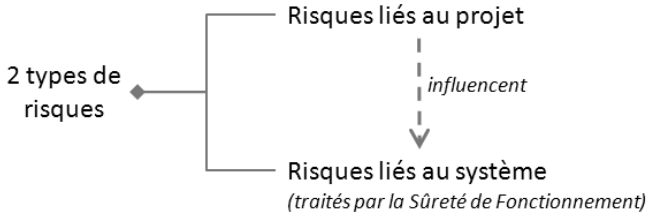


Figure I.9 : 2 familles de risques

D'un point de vue global, on peut identifier deux grandes familles de risques :

- Les risques présents lors de l'exploitation du système et directement liés au système et à son fonctionnement. Typiquement, il s'agit des risques que la sûreté de

fonctionnement va traiter. Par exemple, on retrouve ceux relatifs à une défaillance d'un sous-système ou d'un composant, ou encore ceux en rapport avec la protection des personnes utilisant le système.

- Les risques liés au projet ou au développement lui-même. Ils n'interviennent pas directement dans la sûreté de fonctionnement du système. En revanche, ils peuvent être à l'origine de risques du premier type. Comme exemple, nous pouvons citer le risque dû au retard d'un fournisseur ou bien celui causé par la rencontre d'une difficulté technique plus importante que prévue dans la phase de développement. Ce qui dans les deux cas pourra se traduire par un retard de l'achèvement de la conception, accompagné éventuellement d'une pression plus importante sur les chargés de développement qui peut être un facteur d'augmentation du nombre de fautes de conception (d'où le lien avec les risques liés au système et la sûreté de fonctionnement).

Finalement, nous comprenons bien pourquoi la gestion des risques est une activité importante pour la réussite des projets. Vis-à-vis de notre problématique, l'intérêt est évident concernant le premier type de risques (ceux liés au système), mais l'étude des autres risques semble également d'à propos à la vue de leurs possibles influences. Dans les paragraphes suivants, nous allons étudier plus en détail cette gestion des risques, en présentant notamment ses étapes, les stratégies possibles de traitement des risques et les outils appuyant certaines de ces stratégies.

3.5.3. Etapes de la gestion des risques

Comme nous venons de le voir, la gestion des risques est fondamentale, puisque c'est elle qui permet de prévoir la survenue du problème, de l'incident, de l'accident, et de mettre en œuvre les actions pour le prendre en compte. Elle se décompose en grandes étapes, encore appelées phases de la gestion des risques.

Identification des risques

Dans cette première étape, il s'agit de procéder à l'inventaire des risques. Tous les risques doivent être répertoriés, que ceux-ci soient de type financier, organisationnel, technique ou encore humain. Les sources d'information possibles pour réaliser cette identification des risques peuvent être des consultations, des archives ou encore des mémoires de projets.

Evaluation et classement des risques

Après la phase d'identification des risques, ces derniers doivent être évalués en tenant compte des conséquences possibles. C'est à cette étape qu'il est question d'évaluer la probabilité du risque et sa sévérité, pour en déduire sa criticité. Après l'évaluation de tous les risques, un classement peut être établi en fonction de la criticité associée à chaque risque. Il permettra de données des priorités aux risques à étudier dans l'étape suivante.

Traitement ou acceptation des risques

A cette étape, il s'agit de traiter ou d'accepter les risques qui ont été identifiés et évalués lors des phases précédentes. Typiquement, les risques ayant une forte criticité vont sans

doute être traités, tandis que certains risques ayant une criticité faible pourront être simplement acceptés, sans aucune modification. Le paramètre de jugement le plus important qui permettra de choisir entre le traitement ou l'acception d'un risque correspond très souvent à la différence entre le coût des dégâts si le risque survient et le coût du traitement du risque.

En fait, dans cette phase de traitement ou acceptation, différentes stratégies de gestion des risques peuvent être employées, stratégies qui elles-mêmes disposent d'outils pour leurs mises en œuvre. Nous détaillons ces stratégies et ces outils dans les paragraphes suivants.

Analyse des conséquences et suivi

En dernière étape, il convient de refaire une analyse en vue de réexaminer les éléments impactés par les traitements de risque, principalement pour vérifier si de nouveaux risques ne sont pas apparus. Il s'agit également de commencer un suivi des risques, et plus particulièrement un suivi de l'évolution de leurs criticités au cours du projet.

3.5.4. Stratégies de la gestion des risques

Comme nous l'avons annoncé ci-dessus, il existe diverses stratégies pour traiter les risques. Par ordre croissant de coût, on distingue en effet quatre manières de gérer les risques :

L'évitement

Cette stratégie consiste à ne pas faire l'activité qui présente le risque considéré. Elle est certes la moins chère et la moins risquée du point de vue des décideurs, mais elle constitue un réel frein au développement de l'entreprise. Sans compter que la plupart du temps, elle ne fait que reporter le risque sur d'autres entreprises ou le remettre à plus tard.

L'acceptation

La deuxième stratégie de gestion des risques correspond tout simplement à l'acceptation du risque tel qu'il est. Ce choix sera fait principalement pour des risques à faible criticité, c'est-à-dire dont la probabilité d'occurrence est faible et/ou les conséquences du risques sont limitées. Malgré tout, cette approche ne permettra jamais de protéger le personnel ou les outils de production dans le cas où le risque se produirait. Pour cela, il est nécessaire qu'une volonté de réduction de risque se manifeste.

La réduction du risque

La démarche la plus classique de la gestion des risques correspond à cette troisième stratégie : la réduction du risque. L'objectif est de réduire la criticité du risque, ceci en abaissant sa probabilité d'occurrence et/ou sa sévérité.

Le transfert

Lorsque l'entreprise ne souhaite pas gérer un risque (sans qu'il soit question de l'accepter ou non), elle a la possibilité, dans certain cas, de transférer le risque incriminé. A titre financier, le transfert de risque s'effectue lorsque l'entreprise contracte une assurance

ou toute autre forme de couverture de risque financier ou garantie financière. A titre opérationnel et économique, il a lieu lorsque l'entreprise sous-traite l'activité présentant le risque sous une forme ou une autre (sous-traitance directe, en cascade, cotraitance, externalisation ou outsourcing). Dans ce cas, un sous-traitant sérieux et qualifié pourra faire payer très cher sa prestation ou au contraire démontrer qu'il gère mieux le risque pour un prix équivalent voire inférieur. En revanche, le recours à un sous-traitant non qualifié ou dédaigneux du risque fera courir un risque encore plus grand.

3.5.5. Outils de la gestion des risques

Pour être effectivement accomplies, certaines stratégies énoncées ci-dessus disposent d'outils de la gestion des risques. Il s'agit principalement des stratégies de réduction du risque et de transfert, dont nous donnons ici quatre exemples d'outils :

La prévention

La prévention consiste à *diminuer la probabilité d'occurrence* du risque en diminuant ou supprimant certains des facteurs du risque. Cet outil correspond à une stratégie de réduction du risque et il constitue d'ailleurs la meilleure manière de gérer les risques associés à ses propres ressources.

Les actions correctives

Les actions correctives cherchent à *diminuer les effets du risque* lorsque celui-ci survient. Par exemple, un harnais de sécurité sur un échafaudage n'a aucune influence sur le risque de chute lui-même, mais diminue fortement les effets de la chute (blessures, fractures, traumatismes,…). Lorsqu'il est impossible d'agir sur les facteurs du risque (et donc la probabilité d'occurrence), les actions correctives constituent un outil efficace de minimisation d'impact en modifiant les conséquences.

La diversification

Un outil intéressant dans certain cas pour la diminution du risque est la diversification. Il s'agit de diviser le risque en plusieurs risques plus ou moins indépendant pour ainsi réduire les conséquences globales du risque initial. Par exemple, si un retard dans la livraison d'un composant par un fournisseur présente un risque important pour l'entreprise, la solution de diversification consiste à passer commande non plus auprès d'un unique fournisseur, mais auprès de plusieurs fournisseurs.

Le palliatif

Le dernier outil présenté ici est celui du palliatif, qui permet de conduire des stratégies de transfert. Il consiste à « profiter de l'occurrence du risque », en utilisant à son profit l'événement, sans pour autant modifier la probabilité ou les conséquences du risque. Un exemple typique de cet outil est celui de l'assurance qui couvre le risque en proposant un dédommagement, mais n'empêche en aucun cas l'accident.

3.5.6. Synthèse

Pour résumer, la gestion des risques est une activité importante pour une bonne réussite d'un projet de conception. Sans elle, atteindre le but du projet dans les conditions de coûts et délais souhaités devient peu probable. La Figure I.10 reprend les différentes notions importantes à la gestion des risques sous la forme d'un arbre, rappelant les différentes phases de la gestion des risques, les principales stratégies possibles de traitement des risques et les outils appuyant ces stratégies.

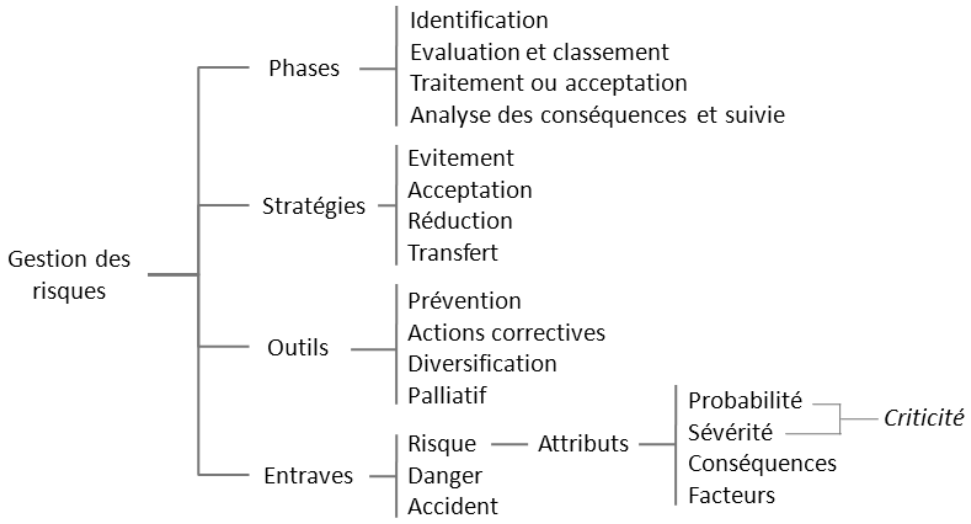


Figure I.10 : Arbre de la gestion des risques

L'étendue de la gestion des risques permet aussi bien de s'occuper des risques en rapport avec le déroulement de la conception que ceux présents lors de l'utilisation du système. Ce sont d'ailleurs ces derniers risques, liés au système, qui pourront devenir des données d'entrée pour des études de sûreté de fonctionnement du système.

Les attributs des risques sont également représentés sur la Figure I.10.

Parmi les entraves de la gestion des risques figurent aussi les dangers et les accidents, où :

- Un **danger** définit une propriété intrinsèque à une substance, à un système technique, à une disposition (élévation d'une charge), …, à un organisme, etc., de nature à entraîner un dommage sur un élément vulnérable.
- Un **accident** est un événement aléatoire, fortuit, qui entraîne des dommages vis-à-vis des personnes, des biens ou de l'environnement ou qui entraîne un engagement de responsabilité.

Ainsi, un risque fait le lien entre un danger et un accident dans le sens où il correspond à la possibilité de survenance d'un accident résultant d'une exposition aux effets d'un phénomène dangereux. Par analogie avec les entraves de la trilogie de la sûreté de fonctionnement (fautes, erreurs et défaillances), la chaîne d'apparition est ici la suivante : danger → risque → accident.

Dans la partie suivante, nous abordons un domaine de l'ingénierie qui est la sûreté de fonctionnement, s'attachant à l'étude des risques associés au système à concevoir.

4. Sûreté de fonctionnement

4.1. Historique

Les problèmes de Sûreté de Fonctionnement [Lannoy, 2008] existent depuis très longtemps, dès qu'un système a pu défaillir ou tomber en panne. Mais au départ, aucune étude spécifique de Sûreté de Fonctionnement n'était appliquée.

C'est dans les années 1930 que la première collecte d'informations statistiques sur des moteurs et des accidents d'appareils a été conduite dans le secteur du transport aérien. Entre 1939 et 1942, on a vu apparaître les tous premiers objectifs quantifiés donnés par le capitaine A.F. Pugsley de la 7ème brigade d'infanterie canadienne. Il évalua des taux de défaillances à hauteur de 10^{-5}/h pour les avions et 10^{-7}/h pour leurs structures.

Puis dans les années 1940, des techniques de fiabilité commencèrent à se développer, avec notamment la conception des V1 en Allemagne ou encore celle des moteurs de traction des locomotives aux Etats-Unis.

Dans les années 1950, le concept de maintenance [Jardine, 1987] fait son apparition. On assiste également aux toutes premières études sur la fiabilité humaine pour les nouvelles centrales nucléaires. A la même époque, des travaux de recueil de données de fiabilité électronique sont entamés.

A partir de 1960, les industries aéronautiques et spatiales firent des analyses relatives aux défaillances de composants. Le département de la défense américain (DoD) promulgua les premières vraies exigences de Sûreté de Fonctionnement suite à des accidents sur missiles. En 1961, les laboratoires Bell utilisent le nouveau concept d'arbre des causes sur le projet du missile Minuteman [Watson, 1961]. Cette technique va directement être reprise par Boeing. En France, la méthode des combinaisons de pannes est utilisée sur le projet Concorde, puis sur Airbus.

En 1962, l'Académie des Sciences accueille le mot « fiabilité » dans sa terminologie.

A partir de 1970, les premiers travaux sur la fiabilité des logiciels [Ledoux & al., 2007] commencent et de nombreuses études sont menées dans le domaine du nucléaire. Nous pouvons citer, par exemple, le rapport américain Rasmussen [Rasmussen, 1975] sur les risques nucléaires des centrales de Surry 1 et Peach Bottom 2. En 1979, la catastrophe nucléaire de Three Miles Island [Rogovin, 1979] motive encore plus le développement d'outils de Sûreté de Fonctionnement. Puis, progressivement, les techniques de Sûreté de Fonctionnement vont largement se diffuser et s'étendre à de plus en plus de domaines : la chimie, le ferroviaire, l'automobile, le traitement et l'épuration de l'eau, et l'ensemble des grands secteurs industriels.

Aujourd'hui, des réglementations et des certifications imposent l'utilisation d'outils dédiés à la sûreté de fonctionnement (cf. chapitre 2 pour des exemples de normes de sûreté).

Elles induisent aussi une certaine idée de la couverture du risque. En effet, on préfère dorénavant parler de tort ou de responsabilité dans les procès qui font suite à des accidents, à la place de la notion de risque. Pourtant, le risque existe toujours.

Parallèlement, la compétitivité croissante force les entreprises à adopter la productivité la meilleure possible, et donc réduisent au maximum les arrêts de production et maximisent la disponibilité de leurs équipements.

Pour conclure ce rapide historique de la sûreté de fonctionnement, nous pouvons dire qu'aujourd'hui la sécurité des biens et des personnes n'a jamais été aussi importante, phénomène accru par la pression médiatique et écologique autour des accidents majeurs.

4.2. Concepts et définitions

La Sûreté de Fonctionnement est, comme nous l'avons aperçu dans le paragraphe qui traite de l'intégration de domaines par l'ingénierie système, une spécialité dite transverse. Elle concerne tous les domaines techniques, tous les génies. Elle a pour finalité le maintien du bon fonctionnement d'un système, d'un produit ou d'un de ses constituants, dans le temps, tout au long de son cycle de vie. La sûreté de fonctionnement est perçue à travers différents attributs : la fiabilité, la maintenabilité, la disponibilité, ou encore la sécurité. Son étude est devenue fondamentale pour les systèmes critiques, où un dysfonctionnement peut engendrer un risque humain ou financier important. Elle s'applique également aux systèmes d'information, où là encore un dysfonctionnement peut causer un risque financier conséquent, mais également un risque social.

J.C. Laprie [Laprie & al., 1995] a défini la sûreté de fonctionnement d'un système comme étant « *la propriété qui permet à ses utilisateurs de placer une confiance justifiée dans le service qu'il leur délivre* », où un utilisateur est un autre système (humain ou physique) qui interagit avec le système considéré et où le service délivré est le comportement du système tel que perçu par ses utilisateurs.

Un bon moyen pour présenter la sûreté de fonctionnement est de s'appuyer sur l'arbre de la sûreté de fonctionnement [Avizienis & al., 2001], [Avizienis & al., 2004], présenté en Figure I.11.

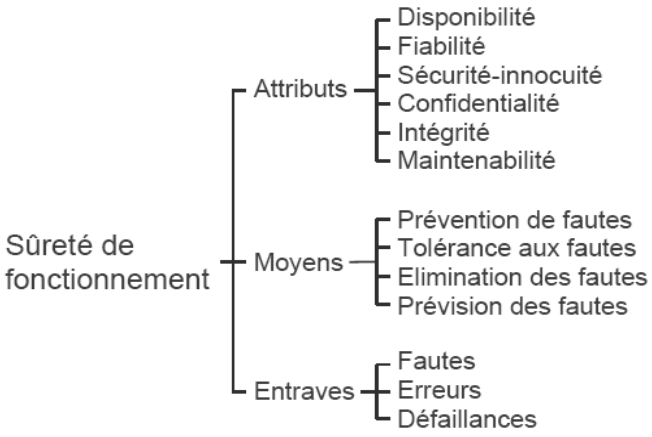


Figure I.11 : Arbre de la Sûreté de Fonctionnement

Ainsi, la sûreté de fonctionnement possède différents aspects : ses **attributs**, sur lesquels un concepteur mettra plus ou moins l'accent suivant le type d'utilisation du système :

- le fait d'être prêt à l'utilisation conduit à la **disponibilité**,
- la continuité du service conduit à la **fiabilité**,
- la non-occurrence de conséquences catastrophiques pour l'environnement conduit à la **sécurité-innocuité**,
- la non-occurrence de divulgations non-autorisées de l'information conduit à la **confidentialité**,
- la non-occurrence d'altérations inappropriées de l'information conduit à l'**intégrité**,
- l'aptitude aux réparations et aux évolutions conduit à la **maintenabilité**.

La Sûreté de Fonctionnement vise principalement à réduire le plus possible le nombre de **défaillances** d'un système. Autrement dit, elle fait en sorte que le système délivre au maximum les services normalement prévus, sans qu'ils dévient de leurs objectifs respectifs.

Or, il est possible que des **erreurs** viennent perturber le bon fonctionnement du système et fassent défaillir le système. On définit d'ailleurs une erreur comme la partie de l'état du système susceptible d'entraîner une défaillance.

Ces erreurs proviennent de ce que l'on appelle des **fautes**, lorsque ces dernières ont été activées. Ces fautes peuvent être de nature très diverses :

- suivant la phase de conception : fautes *de développement* ou fautes *opérationnelles*,
- suivant la frontière du système : fautes *internes* ou fautes *externes*,
- suivant leur localisation : fautes *matérielles* ou fautes *logicielles*,
- suivant leur cause phénoménologique : fautes *naturelles* ou fautes *dues à l'homme*,
- suivant les intentions en jeu : fautes *non malveillantes* ou fautes *malveillantes*,
- suivant les capacités en jeu : fautes *accidentelles*, fautes *délibérées* ou fautes *d'incompétence*,
- suivant leur persistance : fautes *permanentes* ou fautes *temporaires* (voire *furtives*).

La sûreté de fonctionnement met à disposition différents **moyens** pour limiter la présence de fautes, et donc éviter l'occurrence de défaillances :

- **la prévention des fautes** : qui s'attache à empêcher l'occurrence ou l'introduction de fautes,
- **la tolérance aux fautes** : qui fait en sorte que le système délivre toujours un service acceptable, même en présence de fautes,
- **l'élimination des fautes** : qui veut réduire la présence (nombre, sévérité) des fautes,
- **la prévision des fautes** : qui cherche à estimer la présence, la création et les conséquences des fautes.

Pour résumer, les **attributs** de la sûreté de fonctionnement – disponibilité, fiabilité, sécurité-innocuité, confidentialité, intégrité et maintenabilité – correspondent aux propriétés du système liées à la sûreté de fonctionnement. Ils permettent d'apprécier la

qualité du système vis-à-vis des services délivrés. Les **entraves** à la sûreté de fonctionnement – fautes, erreurs et défaillances (cf. Figure I.12) – sont les circonstances indésirables causes ou résultats de la non-sûreté de fonctionnement du système. Pour combattre ces entraves et atteindre les niveaux souhaités des attributs, les **moyens** de la sûreté de fonctionnement vont être sollicités : la prévention des fautes, la tolérance aux fautes, l'élimination des fautes et la prévision des fautes.

Figure I.12 : Relation faute-erreur-défaillance

Bien souvent, la sûreté de fonctionnement fait dorénavant l'objet d'une validation par un organisme extérieur et indépendant de la structure qui a développé le système, surtout lorsque des vies humaines sont en jeu. Dans certains secteurs d'activité (comme dans l'aéronautique par exemple), il est question de **certification**, qui est un contrôle de qualité basé sur des normes. L'AFNOR défini d'ailleurs la certification comme la « *reconnaissance, par un organisme indépendant du fabricant ou du prestataire de service, de la conformité d'un produit, service, organisation ou personnel à des exigences fixées dans un référentiel.* »

4.3. Quelques méthodes de sûreté de fonctionnement

Les principales méthodes de sûreté de fonctionnement consistent en des analyses des défaillances du système, de ses sous-systèmes et de ses composants pour déterminer leurs causes et estimer leurs conséquences sur le service rendu par le système. Ces analyses peuvent être qualitatives ou quantitatives, et elles se conforment à l'une des deux approches suivantes :

- L'approche **inductive** : qui correspond à un raisonnement du particulier vers le général, où l'on recherche les effets d'une défaillance sur le système ou son environnement.
- L'approche **déductive** : qui correspond à un raisonnement du général vers le particulier, où l'on recherche les causes possibles d'une défaillance.

Nous présentons ci-dessous quelques méthodes classiques de la sûreté de fonctionnement. L'objectif n'est pas d'être exhaustif. Pour une revue complète des différentes méthodes de sûreté de fonctionnement, le lecteur est invité à consulter [Villemeur, 1988], [Aldemir & al., 2006] ou encore [M2OS, 2010].

Blocs – Diagrammes de Fiabilité (BDF) : méthode graphique servant à visualiser les sous-ensembles d'un système pour en faire apparaître la façon dont ils contribuent aux différentes fonctions du système. Elle montre notamment les redondances, les éléments qui contribuent à une même fonction et les éléments de secours nécessaires. Elle sert de base à des analyses de fiabilité, de sécurité, de maintenabilité et de disponibilité.

Allocation de fiabilité : pour décliner les objectifs de fiabilité spécifiés au niveau d'un produit complexe (par exemple le système) en objectifs applicables à différents niveaux de l'arborescence technique ou fonctionnelle du produit.

Evaluation prévisionnelle de fiabilité : pour évaluer et affiner, avec un degré de précision croissant avec l'avancement du projet, la fiabilité du produit à l'aide de techniques et de données évoluant avec les progrès de la conception et l'acquisition de résultats d'essais. Elle constitue une base importante pour l'orientation des choix de conception.

FIDES : Pour estimer la fiabilité des composants, cartes électroniques ou sous-ensembles sur étagères (*COTS : Component Of The Shelf*).

Estimations de fiabilité à partir d'essais ou de retour d'expérience : pour déterminer les paramètres de lois de fiabilité à partir de données traitées de retour d'expérience ou d'essais.

Fiabilité prévisionnelle en mécanique : pour estimer la fiabilité des composants mécaniques en prenant en compte les modes de défaillance particuliers (rupture, déformation, grippage, bruyance…) liés à la fatigue, à l'usure et au vieillissement.

Fiabilité en mécanique – la méthode contrainte-résistance : pour évaluer la fiabilité d'une pièce mécanique soumise à des contraintes.

Méthode d'analyse de la SdF du logiciel : par les normes, par les spécifications, par les moyens ou par l'analyse. Il s'agit de construire la sûreté de fonctionnement (fiabilité, sécurité…), analyser et évaluer l'occurrence de bogues d'un logiciel et si possible le plus tôt dans le développement d'un nouveau logiciel.

Analyse préliminaire de risques (APR) (*Preliminary hazard analysis (PHA)*) : pour identifier les points du système qui peuvent être critiques pour la sécurité, évaluer les risques correspondants, les scénarios associés et définir les critères de conception à appliquer [Desroches & al., 2009], [Mortureux, 2002].

Analyse des Modes de Défaillance et de leurs Effets (AMDE) (*Failure Modes and Effects Analysis (FMEA)*) et Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité (AMDEC) (*Failure Modes, Effects and Criticity Analysis (FMECA)*) : méthodes d'analyse inductive et rigoureuse ayant pour buts d'identifier les défaillances dont les conséquences peuvent affecter le fonctionnement d'un système, et – pour l'AMDEC – de les hiérarchiser selon leur niveau de criticité afin de les maîtriser [Faucher, 2004], [Faucher, 2009], [Landy, 2007].

Méthode de l'Espace des Etats: pour évaluer les principales caractéristiques de fiabilité et de disponibilité d'un système réparable.

Analyse par arbre de défaillances (AdD) (*Fault Tree Analysis (FTA)*) : permet une analyse déductive des causes techniques ou opérationnelles pouvant provoquer des situations contraires à un objectif spécifié, en particulier, de sécurité (situation redoutée) ou de disponibilité (événement indésirable) [Lee & al., 1985], [Vesely & al., 1987].

Analyse par arbre d'événements (AAE) (*Event Tree Analysis (ETA)*) : permet d'identifier et d'évaluer les conséquences possibles d'un événement initiateur selon les circonstances ou dysfonctionnements avec lesquels il se combine.

Analyse par arbre de causes : pour réunir dans une représentation synthétique et logique tous les éléments ayant contribué à un incident avéré.

Arbres de Maintenance et d'Aptitude à la Maintenance : fournit le moyen de définir, d'optimiser et d'actualiser la politique de maintenance des outils de production.

Hazard and Operational Study (HAZOP) : pour examiner en détail les composants d'un système et déterminer ce qui se produirait si ce composant devait fonctionner en dehors de son mode normal d'utilisation.

Analyse des Risques, PoInts Critiques pour leur Maîtrise (ARPIC-M) (*Hazard Analysis Critical Control Point (HACCP)*) : est un moyen de garantir la salubrité des aliments. Il repose sur la prévision et la prévention des dangers biologiques, chimiques et physiques.

La simulation de Monte Carlo : La simulation de Monte Carlo [Batut, 1986], [Dubi, 2000] est une technique utilisée pour estimer la probabilité de résultats en répétant un grand nombre de fois une expérience à l'aide de la simulation et en utilisant des nombres aléatoires. C'est une méthode qui a pour but d'imiter un système réel.

Analyse de zone : pour mettre en évidence les problèmes résultant des interactions physiques entre éléments voisins ou de flux perturbateurs générés par des sources externes.

Maintenance Basée sur la Fiabilité (MBF) (*Reliability Centered Maintenance (RCM)*) : pour optimiser la maintenance tout en maîtrisant la sécurité, la disponibilité et la durée de vie d'un équipement.

Intégration Conception et Soutien (ICS) : pour aider à la décision et la sélection d'une solution préférentielle pour la conception d'un matériel durable et réparable, compte tenu de critères de coût, de disponibilité et d'efficacité.

Plans d'expériences : pour permettre aux concepteurs de maîtriser les paramètres de conception, à l'aide d'un nombre minimal d'essais. Le réglage de ces paramètres permet d'optimiser les performances du produit ou des procédés et/ou de réduire leur sensibilité aux différentes sources de variabilité.

Essais accélérés de durée de vie : pour prédire, de manière économique et sur un laps de temps réduit, l'évolution dans le temps d'une (ou plusieurs) performance(s) fonctionnelle(s) ainsi que la durée de vie d'une entité matérielle utilisée dans ses conditions normales d'emploi, à partir d'essais réalisés sous des valeurs de contraintes supérieures aux niveaux spécifiés en utilisation normale.

Essais aggravés : pour explorer les marges de fonctionnement d'un produit en développement et déceler au plus tôt, pour pouvoir les corriger, les défauts inhérents à la conception (produit et procédés) qui réduisent ces marges à des valeurs jugées insuffisantes.

Epreuves de déverminage : pour faire apparaître les défauts de jeunesse d'un produit afin de les corriger avant livraison. Elles consistent à soumettre les exemplaires d'un matériel en sortie de production à des cycles de contraintes adaptées (électriques, mécaniques, thermiques...) visant à précipiter les défauts latents en défauts patents (observables).

Logique de traitement des incidents et actions correctives (LTI-AC) (*Failure report and corrective Action System (FRACAS)*) : pour fournir toutes les informations requises pour identifier les causes de dysfonctionnements d'un produit manifestées au cours de son développement ou de son utilisation en vue d'apporter, en temps et en heures, les actions correctives appropriées.

Coût de Cycle de Vie (CCV), Coût global de possession (CGP) (*Lyfe Cycle Cost (LCC)*) : vise l'optimisation prévisionnelle et la maîtrise du coût global de possession d'un produit, d'une machine ou d'une installation. C'est une donnée économique destinée à faciliter les choix stratégiques de conception et de développement du produit ainsi que le contrôle de sa gestion. Cette analyse permet d'orienter les études de sûreté de fonctionnement, les études de maintenance et de soutien logistique intégré.

L'ouvrage [Villemeur, 1988] dresse un état de l'art complet des méthodes de sûreté de fonctionnement. Des techniques plus récentes d'évaluation de la sûreté de fonctionnement des systèmes dynamiques peuvent être consultées dans [Khalfaoui, 2003], [Sadou, 2007] et [Aldemir & al., 2006].

4.4. Enjeu de la sûreté de fonctionnement

Plus une erreur de conception est découverte tardivement, plus le risque technique induit peut être lourd et entraîner des surcoûts et des retards considérables pour le projet. L'apparition du risque peut notamment conduire à la mise en cause de la sécurité des personnes et des biens, à la dégradation de l'environnement, à la perte de fonctions ou tout simplement à la dégradation de l'image de marque.

L'enjeu de la sûreté de fonctionnement est donc d'identifier les risques au plus tôt dans la phase de développement du produit.

Comme nous l'avons déjà vu, la sûreté de fonctionnement est une activité d'ingénierie système. Elle peut être qualitative ou quantitative. La part qualitative correspond à l'optimisation des études et elle représente environ 70% de l'activité totale. Les 30% restants représentent la partie quantitative consacrée à la maîtrise des risques avant fabrication à partir des architectures déjà élaborées. C'est donc une phase d'optimisation des architectures des systèmes et de leur mise en œuvre de façon à maximiser, à moindre coût, leur robustesse aux aléas.

En résumé, l'analyse de la sûreté de fonctionnement est une action de réduction des risques et donc du coût à l'achèvement. Elle s'exerce essentiellement pendant les premières phases des projets, jusqu'à la mise en production.

5. Sûreté de fonctionnement des systèmes complexes

La sûreté de fonctionnement des systèmes complexes ne va pas de soi et nécessite une prise en compte particulière. Dans ce paragraphe, nous citons quelques exemples de catastrophes ou d'échecs de systèmes qui n'ont pas fonctionné correctement. Puis nous résumons les origines des problèmes de sûreté de ces systèmes complexes, pour déboucher sur le constat qu'une approche de prise en compte globale de la sûreté est nécessaire.

5.1. Exemples d'échecs

5.1.1. Navette spatiale Challenger

Le 28 janvier 1986, la navette spatiale américaine Challenger de la NASA se désintégra environ 70 secondes après son décollage. L'origine du problème qui conduisit à la catastrophe venait d'un joint d'étanchéité du propulseur d'appoint à poudre droit, adjacent au réservoir externe de la navette, qui n'était pas conçu pour les conditions climatiques du jour du lancement.

Plusieurs ingénieurs avaient pourtant exprimé leur préoccupation quant à l'effet de la température sur la résistance des joints toriques en caoutchouc qui permettaient de sceller les joints du propulseur à poudre. En effet, s'il faisait plus froid que 11,7°C, il n'y avait plus aucune garantie d'étanchéité. Cette considération était importante, car les joints toriques étaient repérés comme des composants d'un niveau « critique 1 ». C'est-à-dire que s'ils ne fonctionnaient pas de manière optimale, cela détruirait la navette et son équipage.

Ce risque, pourtant connu des dirigeants de la NASA depuis 1977, a été négligé pour ce décollage, car jugé non suffisamment établi au regard des enjeux politiques de ce vol en particulier (une enseignante choisie après une sélection sur l'ensemble du pays faisait partie de l'équipage). Ainsi, alors que la température était de -13°C avant le décollage, l'autorisation de mise à feu a tout de même été accordée.

Le joint défailli. Une fuite de gaz brûlant provoqua un départ de flamme qui endommagea le réservoir principal rempli d'hydrogène, la structure du réservoir cédant sous l'effet de la chaleur. Le dôme inférieur du réservoir d'hydrogène se sépara et heurta la partie supérieure du réservoir d'oxygène. Au même moment, le propulseur d'appoint à poudre pivota et percuta la structure inter-réservoir. S'en suivi la désintégration complète du réservoir, puis de la navette qui subit un facteur de charge d'environ 20g, bien supérieur à ses tolérances.

5.1.2. Ariane 5

Le 4 juin 1996 a eu lieu le vol inaugural du lanceur Ariane 5 de la société Arianespace. Mais 40 secondes après le décollage, le lanceur dévie de sa trajectoire et explose. En fait, l'erreur qui a conduit à la catastrophe provient de la réutilisation d'un composant logiciel d'Ariane 4 pour Ariane 5 : le programme de contrôle de la vitesse horizontale de la fusée. Mais comme la trajectoire d'Ariane 5 est différente de celle d'Ariane 4, une des variables du programme qui avait un domaine de variation fixé a dépassé sa plage de valeurs avec Ariane 5. Ce qui a entrainé une succession d'erreurs qui a conduit à la destruction du

lanceur. Cette erreur aura coûté 500 millions de dollars uniquement en matériel, sans parler des retards engendrés et des conséquences sur l'image de marque d'Arianespace.

5.1.3. Mars Polar Lander

Mars Polar Lander était une sonde spatiale destinée à l'observation et l'analyse du sol martien. Elle faisait partie du programme *Mars Surveyor* de la NASA. La sonde fut perdue lors de son entrée dans l'atmosphère martienne le 3 décembre 1999. Elle était équipée de pieds pour l'atterrissage, sur lesquels étaient disposés des capteurs. Liés à un calculateur, ceux-ci permettaient de détecter l'impact avec le sol afin de couper les moteurs.

Mais lors du déploiement des pieds de la sonde, les vibrations engendrées et perçus par les capteurs furent mal interprétées par le logiciel de bord, qui considéra que la soude était arrivée sur le sol martien. Les moteurs destinés à freiner la descente de la sonde furent coupés prématurément, alors qu'elle était encore à une quarantaine de mettre du sol. Mars Polar Lander impacta alors la surface à une vitesse de 22 mètres par seconde et fut détruite.

5.1.4. Plate-forme pétrolière BP

Une plateforme pétrolière de *British Petroleum (BP)*, appelée *Deepwater Horizon*, a explosé le 20 avril 2010. Cette catastrophe a causé une marée noire de très grande ampleur dans le Golfe du Mexique, ainsi que 11 décès, 17 blessés et le naufrage de la plateforme.

Concernant les raisons de cet accident, les spécialistes de BP pensent qu'une dizaine de causes majeures sont entrées en jeu.

Le premier problème concerne un anneau en ciment qui devait empêcher les infiltrations indésirables d'hydrocarbures dans le tubage du puits. Selon la BP, le parapétrolier Halliburton chargé de la confection de cet anneau aurait coulé un ciment non-conforme aux caractéristiques du puits Macondo. Ainsi, le coulis n'a pas rempli son rôle d'obstacle et de l'huile et du gaz ont pu remonter vers la plateforme.

Ensuite, peut-être sous l'effet d'un relâchement d'azote, le verrou suivant (appelé Shoe Track) n'a pas fonctionné non plus. Les raisons de ce dysfonctionnement ne sont toujours pas expliquées à la date où ces lignes sont écrites.

Parallèlement, dans la matinée du 20 avril, des techniciens réalisent des tests de pression afin de confirmer l'intégrité du puits. Après l'analyse des données, BP estime qu'au moment où ces tests sont réalisés, cette intégrité du puits n'est absolument pas démontrée, ce qui aurait dû suffire pour interrompre les opérations. Mais cela n'a pas été le cas, car les résultats des tests ont été déclarés « positifs » et, à bord, les techniciens de BP ont approuvés.

Faute toute aussi grave, les techniciens et ingénieurs de Transocean et Sperry n'ont pas vu, sur leurs appareils de mesure, que le flexible reliant le puits à la plateforme était envahi d'hydrocarbures. Or, le guide de contrôle des puits de Transocean stipule que ce flexible doit faire l'objet d'une surveillance de tous les instants.

Suite à cette accumulation de fautes et d'erreurs, les réactions correctives tardives n'ont pu empêcher la catastrophe.

Pour tenter de faire retomber la pression des hydrocarbures, les techniciens avaient orienté le flux d'huile et de gaz vers un dégazeur situé sur la plateforme. Le souci est que ce dispositif qui sépare le liquide du gaz n'est pas conçu pour traiter d'importants volumes. En effet, le gaz s'échappe dans l'atmosphère, un trop important nuage d'hydrocarbure se forme alors au-dessus de Deepwater Horizon. La plateforme n'ayant pas été conçue pour une telle situation, les installations électriques ont provoqué les explosions.

En dehors des pertes humaines importantes causées par ces déflagrations, les explosions déconnectent aussi les câbles électriques qui relient la plateforme à un dispositif de fermeture du puits. Les hydrocarbures continuent donc d'alimenter l'incendie qui ravage la plateforme. Tous les espoirs restants résident alors dans le système de fermeture automatique qui doit s'activer de lui-même en cas d'incident sérieux. Hélas, aucun des deux systèmes de commande n'a fonctionné : le premier à cause de sa batterie à plat et le second en raison d'une valve défectueuse.

Fautes de conceptions, non-respect des règles de sécurité, maintenance approximative, la liste des fautes commises par les opérateurs de Deepwater Horizon a, semble-t-il, été bien longue. Selon BP, ce furent les pratiques de ses prestataires qui constituaient les causes principales...

5.2. Origines des problèmes de sûreté de fonctionnement

Dans les accidents du paragraphe précédent, il s'avère que les composants n'étaient pas défaillants en termes de non-satisfaction des exigences pour lesquelles ils ont été conçus. Les composants ont fonctionné exactement comme cela était prévu. Les problèmes proviennent des effets imprévus ou mal compris des comportements des composants sur le système dans son ensemble. Ces erreurs sont des erreurs dans la conception du système plutôt que dans la conception des composants (y compris l'analyse de la sûreté de ces composants). Notamment, il s'agit d'erreurs dans l'attribution et la traçabilité des fonctions globales du système au niveau des composants individuels.

En fait, l'augmentation de la taille et de la complexité des systèmes actuels rend les processus de conception de plus en plus difficiles. Les changements apportés aux systèmes au fil des années font apparaître des limites des approches et techniques relatives à la sûreté. Ils concernent :

- Des évolutions technologiques rapides,
- Des changements dans la nature des accidents,
- De nouveaux types de dangers,
- Une augmentation de la complexité et de l'hétérogénéité,
- Une relation plus complexe entre l'humain et l'automatisation,
- Un changement de la vision des organismes de régulation et des simples utilisateurs sur la sûreté.

Ces changements représentent un challenge pour tous les acteurs académiques et/ou industriels pour la définition de nouveaux processus, méthodes et outils d'analyse et d'évaluation de la sûreté.

Rasmussen [Rasmussen, 1997] a par exemple montré que la majorité des accidents sont souvent causés non pas par la composition de défaillances indépendantes mais reflète plutôt d'une migration systématique du comportement organisationnel vers les limites d'un comportement sûr. Ceci est principalement dû aux pressions sur les coûts et sur l'efficacité exigée dans un environnement de plus en plus compétitif.

Somme toute, les faiblesses des processus actuels d'analyse et d'évaluation de la sûreté peuvent être résumées par les points suivants (liste non exhaustive) :

- L'analyse de la sûreté comporte un certain degré intrinsèque d'incertitude. Ainsi, il existe un degré de subjectivité dans l'identification des problématiques de sûreté.
- Différents groupes ont besoin de travailler avec différentes vues du système (vue d'ingénieur système et vue d'ingénieur de sûreté par exemple). Généralement c'est un atout, mais cela peut être une barrière quand les vues sont incohérentes.
- Mauvaise définition des exigences de sûreté et de leur formalisation.
- Absence de traçabilité des exigences de sûreté.
- Les méthodes existantes (traditionnelles) sont insuffisantes vu la complexité des systèmes actuels.
- La description textuelle des modes de défaillance est souvent ambigüe.
- Absence de langage commun entre les différents métiers concernés par le système.
- Besoins mal spécifiés ou exigences mal formulées.
- Evolution des besoins/exigences dans le temps et mauvaise gestion du changement.
- Modification spontanée, parfois faite avec de bonnes intentions.
- Non accumulation de savoir-faire et manque de retour d'expérience.
- Pari technologique trop important.
- Définition erronée d'interface.
- Forte pression de la concurrence.
- Extension d'exigences fonctionnelles.

Nous constatons que les problèmes de sûreté de fonctionnement des systèmes ne proviennent pas nécessairement d'une mauvaise analyse ou mauvaise estimation des attributs de sûreté (fiabilité, maintenabilité, disponibilité, sécurité, confidentialité, intégrité), mais de nombre d'autres aspects liés aux projets et à la démarche de conception. Comme exemple nous pouvons citer : la gestion des données, les mécanismes de traçabilité, la gestion de configuration, la gestion des risques, la définition d'interface, la formulation d'exigence, etc.

Ainsi, la sûreté de fonctionnement passe certes par l'ensemble de ces attributs, mais est également sous l'influence d'un certain nombre d'aspects « contributeurs ». Ces éléments participent indirectement à la sûreté de fonctionnement du système, et c'est d'ailleurs en grande partie à eux que le travail présenté dans ce mémoire s'adresse. La Figure I.13 ci-dessous présente une vue conceptuelle de la sûreté de fonctionnement, de ses attributs, ainsi que de quelques-uns des aspects contributeurs, ceci en adoptant une représentation

s'inspirant du « building-block » d'une norme d'Ingénierie Système, l'EIA-632, que nous verrons plus loin (Système = Système produit + produits contributeurs).

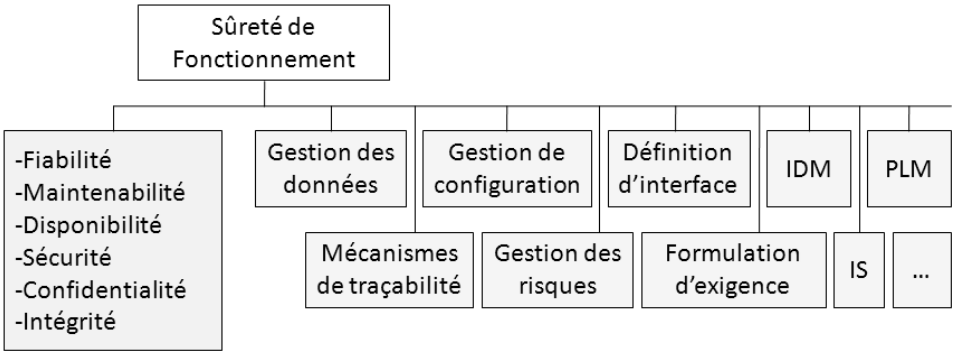


Figure I.13 : Sûreté de fonctionnement et aspects contributeurs

5.3. Nécessité d'une approche globale

En analysant les faiblesses répertoriées dans le paragraphe précédent, nous constatons que certains points sont dus à l'absence d'une approche globale d'évaluation de la sûreté. En effet, les propriétés de sûreté sont des propriétés émergentes qui résultent d'interdépendances existant dans le système et dans l'interaction avec son environnement. Il est absolument nécessaire que ces propriétés soient étudiées globalement au niveau du système complet si l'on souhaite qu'elles soient respectées. De surcroit, ces propriétés importantes du système doivent être considérées dès le début de la conception. En effet, elles ne peuvent pas être introduites ou seulement mesurées *a posteriori*, et doivent être traitées le plus tôt possible pour limiter leurs impacts sur les délais et les coûts de conception.

Une solution pour prendre en compte efficacement les propriétés de sûreté est donc la proposition d'une approche globale pour l'analyse et la gestion de ces dernières. L'ingénierie système qui est un cadre méthodologique pour la conception des systèmes complexes constitue un cadre cohérent et complet pour cette fin. Ce cadre est donc considéré dans notre travail et une approche de prise en compte de manière efficace de la sûreté est développée.

Nous avons vu plus haut que l'un des processus de l'ingénierie système est l'ingénierie des exigences [Sommerville, 2006]. L'ingénierie des exigences est souvent considérée dans la littérature comme le processus le plus critique dans le développement d'un système complexe [Juristo & al., 2002], [Komi-Sirvio & al., 2003]. Elle peut être divisée en deux groupes principaux d'activités [Parviainen & al., 2004] :

- Le développement des exigences : cette activité inclut les processus d'élicitation [Goguen & al., 1993], de documentation, d'analyse et de validation des exigences.
- La gestion des exigences : cette activité inclut les processus de gestion de la maintenance, de gestion des changements et de gestion de la traçabilité des exigences [Gotel & al., 1994], [Sahraoui, 2005].

C'est aussi sur cet aspect d'ingénierie des exigences que notre travail s'appuie fortement. Il faut effectivement accorder une attention particulière à la définition et au développement des exigences de sûreté, mais aussi à leur gestion.

6. Conclusion

Suite à la définition des systèmes complexes et de la notion de propriété émergente, l'ingénierie système a été présentée dans ce premier chapitre, en détaillant l'ingénierie des exigences et la gestion des risques. Nous avons poursuivi par la présentation du domaine de la sûreté de fonctionnement, qui fait partie de nos domaines d'intérêt. Pour finir le chapitre, nous nous sommes penchés sur la sûreté de fonctionnement des systèmes complexes. Cette dernière section a permis d'exprimer la problématique générale du travail de thèse en se basant sur l'origine des problèmes de sûreté de fonctionnement des systèmes complexes. Nous avons alors conclu qu'il était nécessaire de définir une approche globale pour la prise en compte de la sûreté de fonctionnement en l'intégrant dans les processus d'ingénierie système. Le chapitre suivant sera consacré à la présentation de cette approche.

Chapitre 2 : Démarche processus proposée

1. Introduction

Le premier chapitre nous a permis d'identifier la nécessité d'une approche globale pour la prise en compte de la sûreté de fonctionnement pour l'ingénierie des systèmes complexes. Ce deuxième chapitre est dédié à la proposition d'une démarche globale de prise en compte de la sûreté de fonctionnement répondant à cette problématique.

Pour cela, nous faisons dans un premier temps un état de l'art sur le sujet, incluant tant des normes de sûreté et des projets, que des travaux de recherche académiques. Nous résumons et synthétisons dans cette section les différents travaux, dont la plupart utilise une vision processus, puis en faisons un bilan. Ensuite, nous posons la base de notre travail en introduisant la vision processus des activités d'ingénierie système et les normes d'ingénierie système. En particulier, nous détaillons la norme EIA-632 que nous utilisons. Enfin, nous terminons le chapitre par la présentation de notre démarche processus pour prendre en compte la sûreté de fonctionnement.

2. Etat de l'art pour une conception de systèmes sûrs

Pour commencer, cette section propose un état de l'art relatif à la problématique d'une conception de systèmes complexes sûrs de fonctionnement. Tout d'abord nous évoquerons quelques normes de sûreté, puis nous citerons les grands projets en rapport avec le travail et finirons par deux exemples de travaux de recherche dans le domaine.

2.1. Normes de Sûreté

2.1.1. CEI-61508 et ses dérivées

La norme CEI-61508 [CEI-61508, 2010] est une norme générique de sûreté de fonctionnement du CEI (*International Electrotechnical Commission*). Elle est utilisée comme référentiel par tous les grands secteurs industriels. Elle traite de la sécurité fonctionnelle des systèmes électriques/électroniques et électroniques programmables (E/E/PE).

En fait, cette norme a révolutionné le monde de la sûreté de fonctionnement, car elle a su amener des nouveautés dans la façon d'intégrer et de réaliser les activités de sûreté de fonctionnement dans le cycle de développement d'un système E/E/PE. Entre autres, la norme a permis de définir des niveaux d'intégrité pour des systèmes E/E/PE qui prennent

en compte aussi bien les aspects quantitatifs que qualitatifs dans la gestion du risque. La norme intègre les activités de sécurité en les adaptant en fonction des niveaux d'intégrité de la sécurité souhaités (connus sous le nom de « **SIL** » : ***S**afety **I**ntegrated **L**evel*).

Par son aspect générique, la norme CEI 61508 reste brève sur la description des outils, méthodes et les techniques à mettre en œuvre. Mais depuis sa création, plusieurs dérivés de cette norme ont vu le jour dans le but de la rendre applicable pour les différents secteurs concernés (voir Figure II.1).

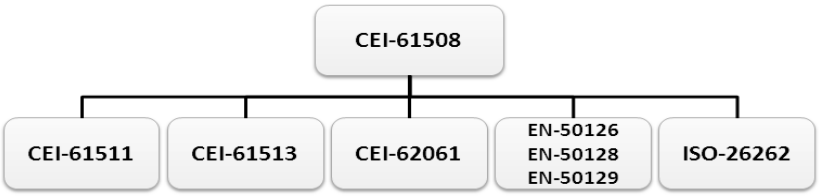


Figure II.1 : Norme CEI-61508 et ses dérivées

Ces normes dérivées sont les suivantes :

- La norme **CEI 61511**, créée en 2003, est adaptée pour les procédés industriels.
- La norme **CEI 61513**, créée en 2001, est adaptée pour le secteur du nucléaire.
- La norme **CEI 62061**, créée en 2005, est adaptée pour la sécurité des machines.
- Les normes **EN 50126/EN 50128/EN 50129**, créées respectivement pour les dernières versions, en 1999/2001/2003, sont adaptées pour le secteur du ferroviaire.
- La norme **ISO 26262**, qui devrait être publiée en tant que standard en 2011, sera adaptée pour le secteur de l'automobile.

2.1.2. ARP-4754

La norme de sûreté [ARP-4754, 1996], dont l'intitulé est « *Certification Considerations for Highly-Integrated or Complex Aircraft Systems* » et dont une nouvelle version est sortie en décembre 2010, est un standard de la *Society of Automotive Engineers* (SAE). Elle traite de processus de développement de systèmes aéronautiques en se focalisant sur les aspects de sûreté. Elle s'intéresse également à des aspects concernant la certification.

La norme fait référence à d'autres standards bien connus, comme le [DO-178B, 1992] « *Software Considerations in Airborne Systems and Equipment Certification* » pour le développement de logiciel dans le domaine aéronautique, ou encore le [DO-254, 2000] « *Design Assurance Guidance for Airborne Electronic Hardware Considerations in Airborne Systems and Equipment Certification* » pour le développement de matériel.

La Figure II.2 montre les activités et leurs organisations préconisées par la norme ARP-4754, qui fait intervenir des :

- FHA : Functional Hazard Assessment,
- PSSA : Preliminary System Safety Assessment,
- CCA : Common Cause Analysis,
- SSA : System Safety Assessment.

Cet ensemble d'activités débute par une analyse préliminaire des risques au niveau du système, qui a pour objectif d'identifier les risques potentiels qui pourraient porter atteinte au système. Eventuellement, ces risques peuvent être pondérés par une criticité (fonction de la probabilité d'apparition et de la gravité), puis classés suivant cette criticité.

Sur cette base de risques, les objectifs, les exigences ou encore la politique de SdF sont identifiés et définis. Ils doivent être intégrés directement pour la conception du système.

Puis des analyses de risques approfondies sont effectuées à partir de la conception déjà établie. Elles sont éventuellement complétées par des analyses de causes communes.

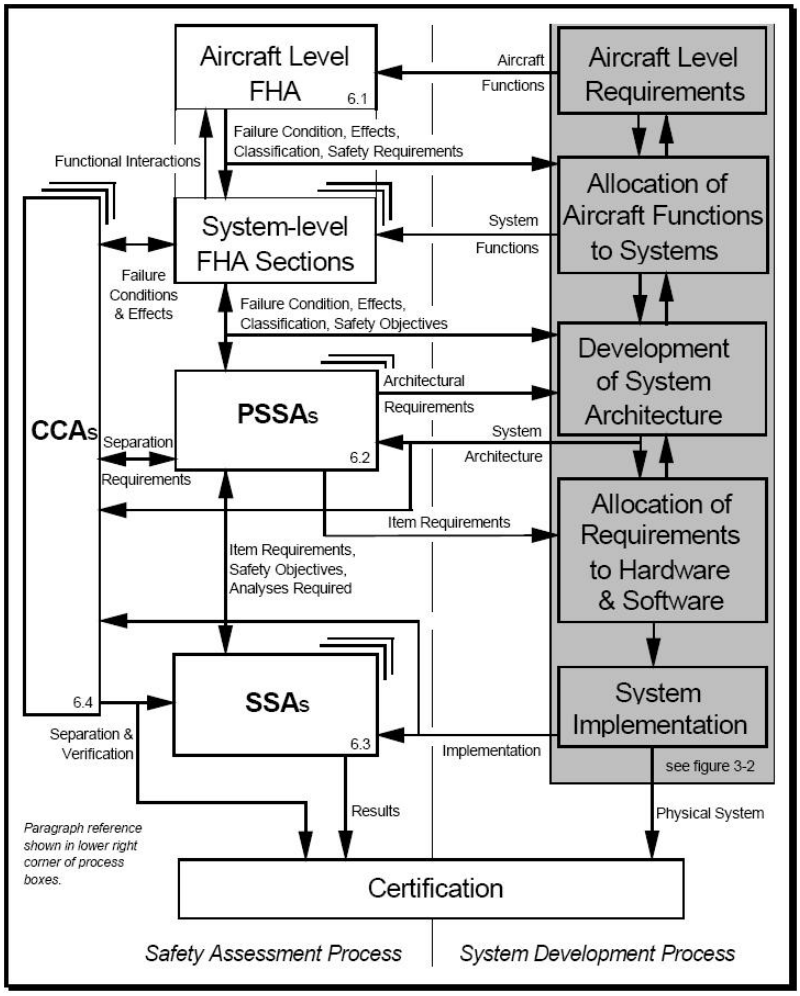


Figure II.2 : Les processus de l'ARP-4754

Enfin, un ensemble d'activités concerne l'évaluation de la sûreté de fonctionnement du système, où il s'agit de trouver des indications qui montrent que le système satisfait bien les exigences de sûreté de fonctionnement.

Une fois la réalisation terminée ou en partie achevée, des activités de tests et d'essais relatifs à la sûreté de fonctionnement sont également réalisés, ceci pour valider le respect effectif des contraintes et exigences de sûreté de fonctionnement.

Pour beaucoup de techniques d'ingénierie pour la sûreté, l'ARP-4754 fait aussi référence à un autre standard de la SAE : l'ARP-4761 présenté dans la section suivante.

2.1.3. ARP-4761

L'ARP-6761 [ARP-4761, 1996] « *Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment* » est un standard de la SAE. Il est destiné à être utilisé conjointement avec l'ARP-4754 pour démontrer la conformité du système en court de conception avec des articles de certification, tel que le 14 CFR 25.1309 concernant la FAA (Federal Aviation Administration) ou encore le CS–25.1309 concernant l'EASA (European Aviation Safety Agency).

Dans ses 30 premières pages, le standard définit un ensemble organisé de processus couvrant l'évaluation de la sûreté, avec principalement la description de méthodes d'analyse courantes pour cette évaluation de la sûreté, dont :

- FHA : Functional Hazard Assessment,
- FTA : Fault Tree Analysis,
- FMEA : Failure Mode and Effects Analysis,
- FMES : Failure Modes and Effects Summary,
- CCA : Common Cause Analysis,
- ZSA : Zonal Safety Analysis,
- PRA : Particular Risks Analysis,
- CMA : Common Mode Analysis.

Les 140 pages suivantes de ce standard détaillent plus en profondeur les différentes techniques, avec leur modélisation et explique comment elles doivent être appliquées. Les 160 dernières pages présentent un exemple fictif et l'application de ces techniques sur ce cas d'étude. Cet exemple sera d'ailleurs repris en partie dans le chapitre 5 pour illustrer notre travail.

2.2. Quelques projets

2.2.1. SQUALE

Le projet SQUALE (*Security, Safety and Quality Evaluation for Dependable Systems*) [Deswarte & al., 1999], [Deswarte, 1998] est un projet européen dont les principaux participants étaient le CR2A-DI, Admiral, IABG, le LAAS-CNRS et Matra Transport International. C'est un projet ACTS (Advanced Communications Technologies and Services) qui débuta en mars 1996 et s'acheva vers la fin 1998. Le rapport final du projet est consultable dans [SQUALE, 1999b].

Le but de ce projet était d'élaborer et de fournir un cadre pour l'évaluation de la sûreté de fonctionnement des systèmes à technologie d'information (IT System). Des critères d'évaluation de la sûreté de fonctionnement ont été définis. Ils ont un caractère générique, dans le sens où ils peuvent s'appliquer à tous les secteurs d'activités. L'intérêt de ses critères par rapport à d'autres critères comme ceux fourni par TCSEC (*Trusted Computer System Evaluation Criteria*), ITSEC (*Information Technology Security Evaluation Criteria*) [TCSEC, 1985], [Deswarte & al., 1998] ou encore d'autres normes propres à l'avionique, le ferroviaire, le nucléaire, etc. provient du fait qu'ils couvrent à la fois les aspects de sécurité-innocuité et de sécurité-confidentialité, ainsi que tous les autres attributs de la sûreté de

fonctionnement (fiabilité, disponibilité, maintenabilité,…). Pour élaborer les critères proposés par SQUALE, les participants au projet se sont d'ailleurs en partie appuyés sur les Critères Communs [Common Criteria, 1998]. (Note : il existe maintenant des versions plus récentes des Critères Communs.)

Ainsi, les différents collaborateurs du projet ont défini et proposé un panel d'activités à réaliser dans des conditions données, selon les différents attributs de sûreté de fonctionnement et leurs niveaux de confiance souhaités (déclinés en niveaux de détail, de rigueur et d'indépendance). Fondamentalement, SQUALE définit des activités à mettre en œuvre pour valider la sûreté de fonctionnement d'un système, paramétrables en fonction des niveaux de confiance voulus. Le document alors établi par le projet SQUALE [SQUALE, 1999a] se veut être applicable pour la validation ou la certification de systèmes qui sont critiques au niveau de la sûreté ou de la sécurité, ou qui ont un niveau requis élevé de disponibilité, de fiabilité ou de maintenabilité. De plus, il couvre toutes les phases du cycle de vie des systèmes.

2.2.2. ESACS

ESACS (Enhanced Safety Assessment For Complex Systems) [ESACS], [Bozzano & al., 2003] est un projet européen RTD en réponse à l'appel de Growth 2000 traitant de « nouvelles perspectives en aéronautique ». Le projet démarra en février 2001 et s'acheva en décembre 2003.

Les objectifs du projet étaient :

- de définir une méthodologie pour améliorer la réalisation des analyses de sûreté au cours du développement de systèmes complexes,
- d'établir un environnement partagé basé sur les outils supportant la méthodologie,
- de valider la méthodologie à travers son application sur un cas d'étude.

L'environnement entre la conception et la sûreté est composé d'outils pour générer des parties d'analyses de sûreté en utilisant les informations extraites directement du modèle du système et d'une base incluant toutes les informations de sûreté liées au système.

Le constat de base fait lors des débuts du projet ESACS rejoint des points identiques aux nôtres concernant les faiblesses des processus de sûreté de fonctionnement. Plus de détails sont disponibles dans [ESACS, 2001].

Les défaillances et leurs effets proviennent souvent du simple jugement des ingénieurs et sont vérifiés par des tests. Or ces jugements deviennent de plus en plus difficiles à cause de la complexification des systèmes. Principalement, le nombre d'états possibles des systèmes augmentent considérablement du fait de l'intégration de calculateurs toujours plus sophistiqués et de l'augmentation des interactions entre les différents systèmes avion (note : on parle de « système avion » en aéronautique pour désigner les premiers « sous-systèmes » de l'avion) qui implique un risque d'interaction entre des défaillances.

L'autre point concerne la manière dont les informations sont gérées et échangées par les ingénieurs. En fait, actuellement les informations entre les ingénieurs système et les ingénieurs de sûreté sont échangées sous forme papier. Ainsi, l'information concernant le

comportement du système est transmise par texte ou par modèle et elle doit être en premier lieu interprétée par l'analyste de sûreté. Dans un second temps seulement, des AMDEC ou des arbres de défaillances (par exemple) sont réalisés, étant supportés par d'autres outils. Ce type de traitement de l'information est source d'erreurs et implique la duplication des efforts de modélisation (pour la conception, puis pour la sûreté) et aussi retarde l'identification des problèmes.

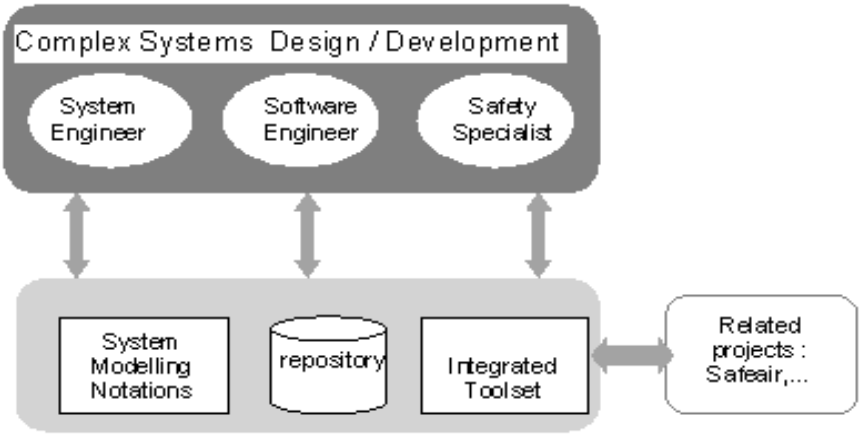


Figure II.3 : Proposition du projet ESACS

Le projet ESACS a répondu à ces faiblesses actuelles en expliquant que les analyses d'évaluation de la sûreté doivent être conduites à partir de quelques moyens automatiques directement appliqués sur le modèle fonctionnel du système (voir Figure II.3). Cela signifie que les modèles utilisés pour la conception du système doivent être enrichis avec les informations ou exigences de sûreté de fonctionnement. Egalement, il s'agit d'utiliser des méthodes de simulation ou des méthodes formelles (model-checking) pour aider aux analyses de sûreté.

Finalement, les principales recommandations issues du projet sont :

- identifier et développer un modèle commun du système à la fois pour la conception et pour la sûreté,
- développer une façon d'exprimer les exigences de sûreté plus précise et plus rigoureuse, et dans le même cadre que celui pour la conception du système,
- Enrichir les modèles du système pour inclure les comportements défaillants,
- Fournir le moyen d'utiliser le modèle du système enrichi pour la génération d'analyse des modes de défaillance et de leurs effets, ainsi que pour la génération d'arbres de défaillance,
- Fournir une classification des modes de défaillance.

2.2.3. ISAAC

Le projet européen ISAAC (*Improvement of Safety Activities on Aeronautical Complex systems*) [ISAAC], [Akerlund & al., 2006] est une continuité au projet ESACS qui avait réussi à montrer le bénéfice engendré par l'utilisation de méthodes formelles pour l'évaluation de la sûreté des avions. Ce projet se déroula de 2003 à 2006. L'objectif était d'aller encore plus loin dans l'amélioration et l'intégration des activités de sûreté des

systèmes complexes aéronautiques. Les bénéfices potentiels vont d'une meilleure confiance en la sûreté des systèmes à un accroissement de la compétitivité des industries européennes.

Entre autre, il s'agissait d'étendre les techniques formelles pour prendre en compte les erreurs humaines, les analyses de causes communes, les analyses de mission et la testabilité. Il était aussi question d'améliorer les notations d'ESACS pour représenter les exigences de sûreté. Un troisième volet s'occupait de l'aspect d'intégration, à travers des recommandations générales et de librairies et interfaces partagé. C'est en fait un des concepts clés qui permet d'améliorer l'efficacité des processus industriels, qui le plus souvent reposent sur des outils différents.

2.2.4. ASSERT

Le projet ASSERT (*Automated proof-based System and Software Engineering for Real-Time systems*) [Conquet, 2008] [Conquet & al., 2005] est un projet coordonné par l'Agence Spatiale Européen (European Space Agency – ESA), plus précisément par la division des systèmes logiciels. Il a réuni près de 28 partenaires, dont des industries spatiales et des laboratoires de recherche, ainsi que des entreprises de développement de logiciels et d'outils. Il commença le 1er septembre 2004 et se termina le 31 décembre 2007.

Ce projet avait comme objectif de changer la manière de procéder à l'ingénierie des systèmes et des logiciels. Notamment, une nouvelle approche est proposée, plus fiable car basée sur la modélisation, la préservation de propriétés système et la transformation de modèle vers le code final. Un processus est défini, accompagné d'un ensemble d'outils, et des projets concrets ont permis de valider l'approche. Parmi les nombreux langages utilisés dans les processus d'ASSERT, on trouve notamment AADL, UML ou encore Altarica.

Sur un cas d'étude d'un budget total de 50 M€, avec une partie avionique représentant 30 M€ dont la part logicielle est de 4 M€, il a été estimé un gain de 1,8 M€ sur ces 4 M€ en suivant l'approche proposée par le projet ASSERT.

2.3. Autres travaux de recherche

2.3.1. Modèle de développement à sûreté de fonctionnement explicite

Dans [Laprie & al., 1995], un modèle de développement de système est proposé. Il est caractérisé de *modèle de développement à Sûreté de Fonctionnement explicite*. En effet, les moyens de la SdF sont directement identifiables à travers les processus proposés.

La vision processus est adoptée pour la description de ce nouveau modèle, qui voulait regrouper au sein de processus fondamentaux les activités nécessaires à la construction d'un système et celles relatives à la sûreté de fonctionnement. Le processus principal est alors celui dit de *création du système*, qui inclut les sous-processus d'expression des exigences, de conception, de réalisation et d'intégration. Autour de ce processus de base vont se greffer les processus spécifiques à la SdF que sont les processus d'élimination des fautes, de prévision des fautes, de tolérance aux fautes et de prévention de fautes. A cela

s'ajoute encore tous les autres processus classiques d'assurance qualité, de certification, etc. (voir la Figure II.4).

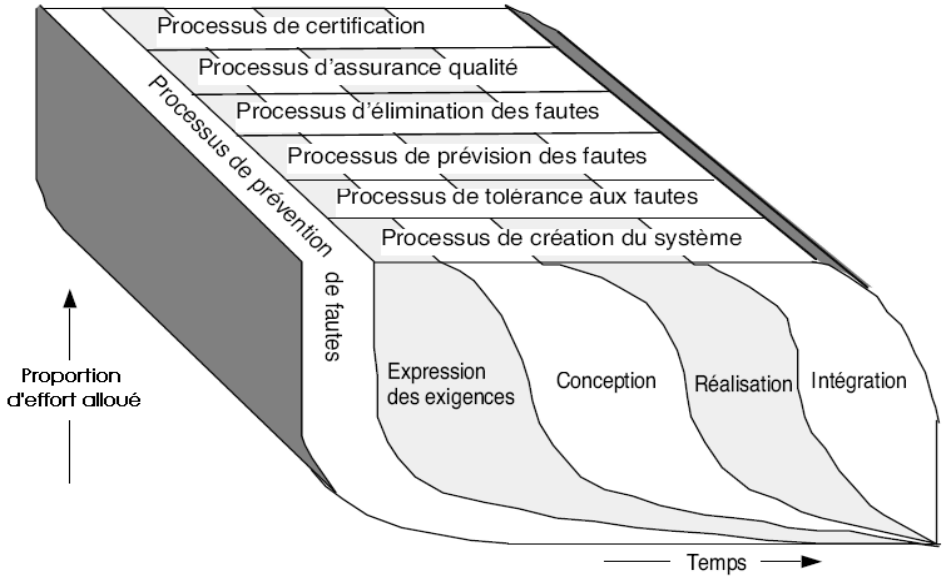


Figure II.4 : Modèle de développement à SdF explicite

Le **processus de création** fait donc apparaître quatre activités de base : l'expression des exigences, la conception, la réalisation et l'intégration. *L'expression des exigences* correspond aux activités de spécifications fonctionnelles, où il est question de définir les fonctions et les services que le système devra rendre. Une description de l'environnement d'utilisation doit également être faite et toutes les contraintes liées au développement, à la validation ou encore à l'exploitation doivent être identifiées. Cet ensemble d'activités doit aboutir à une spécification (formelle ou informelle) du système à concevoir. La *conception* correspond aux activités qui permettent de définir l'architecture du système, activités souvent itérées jusqu'à la définition de composants élémentaires. La *réalisation* consiste en la réalisation des composants ou à l'adaptation des composants qui sont réutilisés. Enfin, *l'intégration* correspond à l'assemblage des composants pour fabriquer le système et à l'intégration du système ainsi construit dans son environnement d'utilisation.

Le **processus de prévention de fautes** va définir et coordonner les activités de création du système avec les autres processus, ceci à travers trois activités principales : *définition des formalismes et des langage*s, *organisation du projet* (allocation des tâches aux équipes et gestion des ressources), et *planification et évaluation des risques liés au développement*.

Le **processus de tolérance aux fautes** comporte trois classes d'activités : la définition des classes de fautes pour lesquelles le système devra se prémunir et la description du *comportement du système en présence de ces fautes*, le *partitionnement* du système en zone de confinement d'erreur et d'indépendance de fautes, et la définition d'algorithmes et de mécanismes de *traitement des erreurs et des fautes*.

Le **processus d'élimination des fautes** est en fait le processus qui va se charger de toutes les *vérifications* ou les *tests* nécessaires au développement, à la cohérence de la

progression de la conception ou à la conformité de ce qui est produit. Il pourra être à l'origine de demande de modification de la conception, si nécessaire.

Le **processus de prévision des fautes** est composé des activités qui permettent *d'exprimer les objectifs* de sûreté de fonctionnement du système, *d'allouer ces objectifs* aux différents constituants du système et *d'évaluer la sûreté de fonctionnement*.

Toutes ces activités sont en forte interaction, que ce soit entre les processus de sûreté de fonctionnement et le processus de création, ou entre les processus de sûreté de fonctionnement eux-mêmes. Ceci provient du fait que les exigences de sûreté de fonctionnement et l'élaboration de mécanismes de tolérance aux fautes nécessitent une **analyse globale**. Cela rejoint le constat que nous avions fait de la nécessité d'une prise en compte au niveau global des exigences de sûreté de fonctionnement du système.

Les détails des processus de sûreté de fonctionnement et du processus de création ont été proposés dans [Laprie & al., 1995] sous la forme d'un ensemble de mots-clés, ceci pour chaque étape du processus de création (expression des exigences, conception, réalisation et intégration). Ils sont repris en annexe A de ce mémoire.

Critique du modèle

Le modèle ainsi proposé offre un cadre général pour le développement de systèmes sûrs de fonctionnement, ceci en regroupant et faisant apparaître clairement les activités relatives à la prévention de fautes, la tolérance aux fautes, l'élimination des fautes et la prévision des fautes. Les points importants exprimés à travers les différentes listes de mots-clés constituent une base très complète et intéressante pour l'ingénierie du système. Néanmoins, l'approche reste vraiment orientée « systèmes informatiques ». De plus, il a été question lors de la construction de ce modèle de distinguer et regrouper en blocks les activités en fonction de leur appartenance soit à la prévention de fautes, soit à l'élimination de fautes, soit à la prévision des fautes ou encore à la tolérance aux fautes. Mais vouloir absolument faire cette séparation rend le modèle, et surtout une instanciation de ce modèle, bien difficile à saisir. Vouloir absolument distinguer et regrouper les activités pour faire apparaître en quatre blocs les moyens de la sûreté de fonctionnement casse la logique de conception. Les concepteurs auront plus de difficulté à comprendre les flux entre activités et leurs enchaînements à travers ce modèle qui semble devenir trop artificiel. Les activités n'étant donc pas regroupées en fonction de leurs interactions, il est normal d'aboutir à des schémas d'interaction avec une intrication importante de flèches représentant les interactions en question.

2.3.2. Méthodologie d'IS basée sur les modèles et guidées par la SdF

Une méthodologie d'ingénierie système a été développée en collaboration entre le MIT (*Massachusetts Institute of Technology*) et le JPL (*Jet Propulsion Laboratory*). Présentée dans [Leveson & al., 2007], cette méthodologie se base sur l'utilisation de modèles et est guidée par les aspects de sûreté de fonctionnement.

Le travail réalisé se base sur le constat que, notamment dans le domaine spatial, de sérieux problèmes ou des pertes de missions résultent de l'augmentation de la complexité

des systèmes, conjuguée avec l'utilisation de nouvelles technologies, dont l'informatique et le logiciel (voir [Leveson, 2003]). Ainsi, pour contrer cette complexité, de nouvelles méthodologies sont nécessaires afin de concevoir la sûreté dans les systèmes spatiaux depuis le début du processus de conception, en utilisant des techniques basées sur des modèles pour trouver tôt les fautes de conception (c'est-à-dire lorsque les coûts de modification sont encore faibles).

L'idée de la méthodologie est de placer l'analyse des risques dans le processus de conception nominal, au lieu d'être considérée comme une activité séparée. La méthode intègre aussi plusieurs aspects développés par le MIT et/ou le JPL. Coté MIT, nous retrouvons le STAMP (pour *Systems-Theoretic Accident Model and Processes* - [Leveson & al., 2003] et [Leveson & al., 2005]), la STPA (pour *STAMP-Based Hazard Analysis* - [Leveson & al., 2005]) et *l'Intent Specification* (un cadre de spécification d'ingénierie système structuré et basé sur des contraintes). L'apport du JPL vient par son approche d'ingénierie système basé sur des modèles, avec des analyses d'états.

Pour revenir sur *l'Intent specification*, elle correspond à une nouvelle forme de spécification améliorée car conçue pour :

- Faciliter la traçabilité,
- Appuyer la vérification des différentes propriétés du système (dont la fiabilité),
- Réduire les coûts d'une modification et d'une ré-analyse du système.

La méthodologie est composée d'étapes qui sont les suivantes :

- **Etape 1** : Identifier les objectifs de la mission, les exigences et les contraintes,
- **Etape 2** : Définir les accidents système ou les pertes inacceptables,
- **Etape 3** : Définir les risques haut-niveau,
- **Etape 4** : Définir les contraintes de sûreté haut-niveau,
- **Etape 5** : Identifier les contraintes d'environnement et d'utilisateur,
- **Etape 6** : Procéder à la décomposition fonctionnelle haut-niveau,
- **Etape 7** : Concevoir la structure du système de contrôle haut-niveau,
- **Etape 8** : Procéder à l'analyse préliminaire des risques,
- **Etape 9** : Définir les spécifications des éléments du système,
    - **Etape 9.1** : Définir les objectifs, les hypothèses, les exigences et les contraintes,
    - **Etape 9.2** : Développer les modèles du système sous contrôle,
    - **Etape 9.3** : Définir et concevoir le comportement opérationnel du système de contrôle,
    - **Etape 9.4** : Développer des modèles formels du système de contrôle,
    - **Etape 9.5** : Continuer à procéder à la STPA,
    - **Etape 9.6** : Raffiner itérativement la conception du système,
- **Etape 10** : Procéder aux tests de validation,
- **Etape 11** : Générer la conception et le code logiciel.

2.4. Synthèse

Ainsi, nous venons de parcourir un état de l'art pour la conception de systèmes complexes sûrs de fonctionnement. Les normes de sûreté aident énormément à comprendre quels sont les objectifs et les activités pour obtenir un système sûrs. Par exemple, l'ARP-4754 fournit une très bonne vision des activités de sûreté de fonctionnement. Mais en soit, ces normes ne définissent pas une approche globale et unifiée avec les activités de conception nominales, c'est-à-dire qu'elles ne sont pas intégrées complètement aux processus d'ingénierie système.

Les quelques projets que nous avons cité n'offrent pas non plus une réelle approche globale d'ingénierie système. Ils se focalisent plus sur les méthodes et les outils à utiliser.

Quant aux deux travaux de recherche présentés (Laprie et Leveson), le modèle de développement à sûreté de fonctionnement explicite est très orienté pour le domaine logiciel et propose, selon nous, une trop grande séparation entre les activités « classiques » de conception et les aspects de sûreté de fonctionnement. Concernant la méthodologie d'IS basée sur les modèles et guidées par la SdF, l'idée de placer l'analyse des risques dans le processus de conception nominal correspond tout à fait à notre objectif. Mais l'approche proposée est très spécifique et pas suffisamment générique. Il est en effet question d'utiliser le *STAMP*, la *STPA* ou encore l'*Intent Specification* (cf. paragraphe 2.3.2), sans une réelle séparation entre les processus et les méthodes.

Constat

Finalement, le constat que nous faisons est que toutes les normes de sûreté de fonctionnement existantes se basent plus ou moins sur un même ensemble fondamental d'activités à mettre en œuvre. Un exemple a été vu en Figure II.2, qui montre les activités et leurs organisations préconisées par la norme ARP-4754 [ARP-4754, 1996].

Cet ensemble d'activités, synthétisé Figure II.5, débute par une analyse préliminaire des risques au niveau du système. Elle a pour objectif d'identifier les risques potentiels qui pourraient porter atteinte au système. Eventuellement, ces risques peuvent être pondérés par une criticité (fonction de la probabilité d'apparition et de la gravité), puis classés suivant cette criticité.

Sur cette base de risques, les objectifs, les exigences ou encore la politique de SdF sont identifiés et définis. Ils doivent être intégrés directement pour la conception du système.

Puis des analyses de risques approfondies sont effectuées à partir de la conception déjà établie. Elles sont éventuellement complétées par des analyses de causes communes.

Enfin, un ensemble d'activités concerne l'évaluation de la sûreté de fonctionnement du système, où il s'agit de trouver des indications qui montrent que le système satisfait bien les exigences de sûreté de fonctionnement.

Une fois la réalisation terminée ou en partie achevée, des activités de tests et d'essais relatifs à la sûreté de fonctionnement sont également réalisés, ceci pour valider le respect effectif des contraintes et exigences de sûreté de fonctionnement.

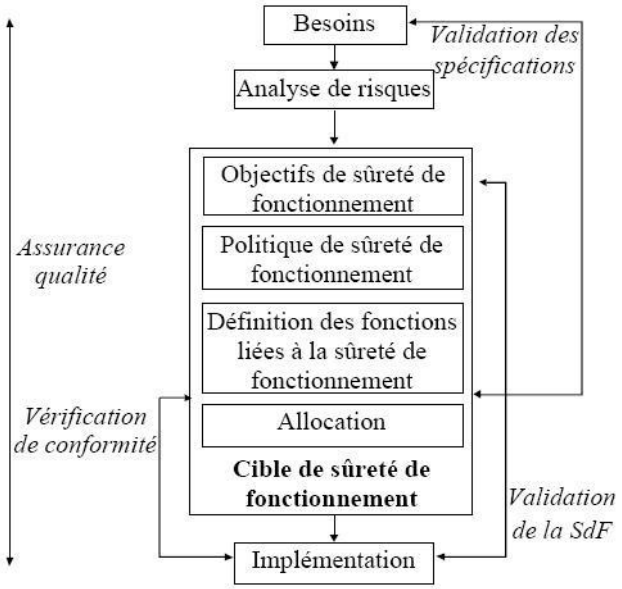


Figure II.5 : Processus génériques de base relatifs à la SdF

Selon les secteurs, les systèmes ou les criticités des risques à traiter, les activités devront être réalisées avec des niveaux de rigueur, de détail ou encore d'indépendance (d'autres facteurs étant possibles) bien déterminés. En fonction de ces niveaux, les activités seront en quelque sorte « paramétrées » et devront par exemple faire appel tantôt à des méthodes informelles, tantôt à des méthodes formelles.

Ainsi, il est utile de définir des niveaux de confiance et de les associer à chaque caractéristique d'un système ou sous-système. Ces niveaux définiront alors ceux des activités à mettre en œuvre. Comme exemples de documents spécifiques à la SdF définissant des niveaux, nous retrouvons les critères communs [Common Criteria, 1998], le rapport du projet SQUALE [SQUALE, 1999a] ou encore la norme CEI-61508 [CEI-61508, 2010] (avec les SIL).

Notre proposition

Comme l'établit le constat à la base de notre problématique, la sûreté de fonctionnement doit être prise en compte dès le début de la conception du système, afin d'éviter des dépassements non-souhaités des coûts et des délais du projet de conception. L'ingénierie système est un cadre de travail approprié pour la conception de systèmes complexes. Dans notre travail, nous montrons comment la sûreté de fonctionnement peut être explicitement gérée dans ce cadre.

Une approche d'ingénierie système pour la sûreté de fonctionnement se base sur l'hypothèse que les propriétés de SdF ne peuvent être traitées entièrement qu'en prenant en compte toutes les variables et tous les aspects (du social au technique) [Kotovsky & al., 1985]. Cette base pour l'ingénierie système a été établie à partir du principe qu'un système est plus que la simple somme de ses constituants. La gestion de la sûreté de fonctionnement doit alors suivre toutes les étapes de l'ingénierie système, depuis la définition des exigences jusqu'à la vérification et la validation du système, et même son retrait de service. Si nous considérons, par exemple, une exigence de fiabilité définie pour le système complet, sa

formalisation et son analyse doit permettre d'assurer que les solutions techniques sélectionnées lors de la progression de la conception répondent bien à cette exigence de fiabilité au niveau des sous-systèmes et après leur intégration.

En fait, les activités relatives à la sûreté de fonctionnement peuvent être classées selon trois catégories principales.

- Les activités de **prévision des dysfonctionnements** [Sadou & Demmou, 2009] regroupent un ensemble d'analyses de sûreté de fonctionnement, telle que la recherche de scénarios [Guillerm & al., 2009a], [Guillerm & al., 2010a] conduisant aux défaillances ou l'analyse des conséquences de défaillances. Ces activités sont à réaliser en parallèle à la conception du système.
- Les activités de **tolérance aux fautes et aux pannes** [Arlat & al., 1999] s'intéressent à la mise en place de mécanismes tels que les redondances pour recouvrir certaines pannes, ou encore prennent en compte les erreurs possibles des opérateurs. Elles font partie intégrante des activités de conception du système.
- La dernière famille d'activités est celle **des démonstrations d'obtention de la sûreté** [Sheard, 2006]. Elle regroupe l'ensemble des activités de vérification et validation (V&V), de tests et d'essais relatifs à la sûreté de fonctionnement du système [Cohn, 1989], [Beizer, 1990]. Ces activités sont réalisées une fois certaines étapes de conception achevées (pour la V&V) ou à la fin d'étapes clés de la réalisation (pour les tests et les essais).

Nous proposons ainsi une nouvelle approche à l'aide de processus qui doivent être définis indépendamment des méthodes et des outils, qui sont par ailleurs l'intérêt principal d'autres projets (ESACS [ESACS, 2001], ISAAC [Akerlund & al., 2006] et MISSA[1] par exemple).

Mais avant d'exposer notre propre démarche processus, nous nous devons de définir les processus et normes d'ingénierie système.

3. Processus et normes d'Ingénierie Système

Deux principales approches de conception ont fait l'objet de nombreux travaux et ont été définies. Le cycle en V et ses variantes [Forsberg & al., 1991b] , [Forsberg & al., 1995], [Boehm & al., 1994], [Boehm, 1988], et l'approche processus. Le cycle en V a déjà été aperçu au premier chapitre.

Quant à l'approche processus, elle est basée sur le fait que quelque soit la stratégie utilisée pour le développement d'un système, les activités restent les mêmes [Evrot, 2008]. Les processus techniques représentant ces différentes activités d'ingénierie système sont groupés en deux catégories principales : les processus de définition du système et les processus de vérification et de validation du système. Ils sont définis par des normes d'ingénierie système ([IEEE-1220, 2005], [EIA-632, 1999], [ISO-15288, 2008]).

[1] More Integrated Systems Safety Assessment (http://www.missa-fp7.eu/)

L'approche processus est adoptée dans notre travail car elle est plus flexible que l'approche de développent en V. En effet, la vision processus ne contraint pas la séquence des activités de développement, contrairement aux approches basées sur des cycles [Meinadier, 2002]. Cette différence est la motivation pour le choix de l'approche processus, d'autant plus qu'il s'agit de systèmes complexes.

3.1. Vision processus

Une manière de transcrire l'Ingénierie Système passe donc par la définition de processus d'Ingénierie Système. Ces processus informent des tâches ou des activités à réaliser par des acteurs du projet (équipes de conception, de management,…) pour réaliser la mission souhaitée. Ils possèdent des données d'entrée et de sortie et sont activés dès que des entrées respectent certaines contraintes préétablies, si tant est que les ressources nécessaires sont disponibles (voir la schématisation d'un processus ci-après).

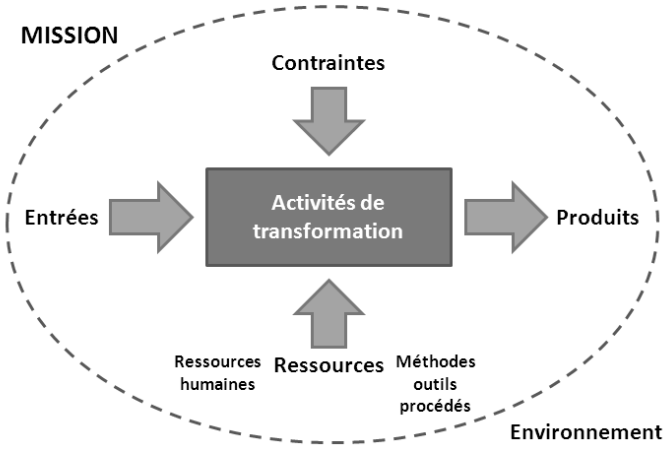


Figure II.6 : Processus d'Ingénierie Système

L'AFIS fournit une définition du processus qui est :

***Processus** : Un processus est un ensemble d'activités interactives coordonnées pour transformer progressivement des éléments d'entrée en produits.*

En fait, différents niveaux de définition des processus existent [AFIS, 2005a] :

- Le processus **normalisé** : il est défini dans des normes (cf. le paragraphe suivant), a souvent un caractère générique et de recommandation, et présente des types d'activités à réaliser et de résultats attendus. Ce genre de processus provient souvent d'un consensus entre des experts d'entreprises les plus avancées.
- Le processus **défini** ou **institutionnalisé** dans un organisme : il résulte de l'ajustement d'un processus normalisé aux besoins de l'organisme, en fonction des pratiques et des méthodes choisies par l'organisme pour réaliser les activités. D'ailleurs, la conception et l'amélioration des processus institutionnalisés relève de l'ingénierie des processus.
- Le processus **ajusté** à un projet : il provient d'une adaptation d'un processus institutionnalisé aux justes besoins du projet, en prenant en compte les contraintes et les ressources du projet en question.

Comme nous l'avons vu dans le premier chapitre, un modèle transmet de manière plus efficace et plus sûre une information qu'une explication textuelle qui peut comporter des ambiguïtés. Ainsi, les processus sont eux aussi modélisés, par exemple en utilisant la notation graphique de la *Business Process Modeling Notation* (BPMN), anciennement proposée par les membres de la *Business Process Modeling Initiative* (BPMI) et désormais sous la coupe de l'OMG.

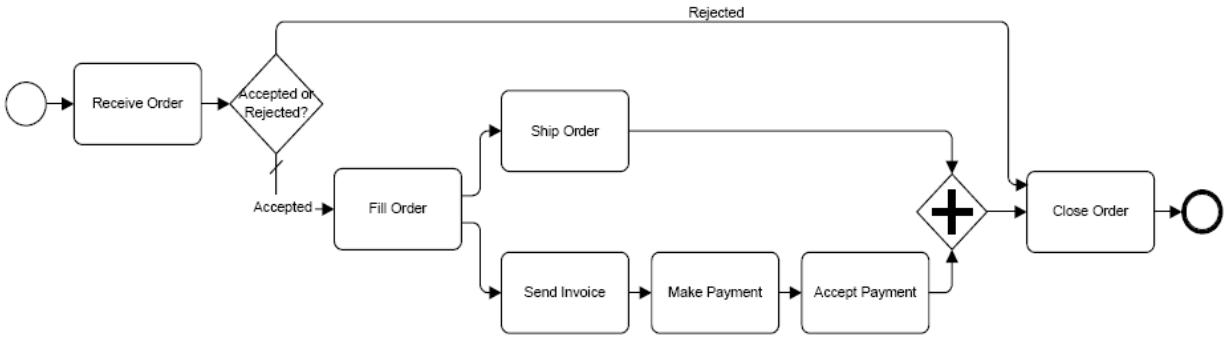


Figure II.7 : Exemple de modélisation en processus

Un autre exemple de notation possible pour la modélisation de processus est donné par le méta-modèle SPEM (Software & Systems Process Engineering Meta-Model) de l'OMG (Object Management Group) [SPEM, 2008]. Effectivement, ce méta-modèle peut être utilisé par toutes les activités de modélisation de méthodes et de processus, ceci en utilisant des notations d'UML [UML, 2010]. Dans ses travaux [Rochet, 2007], Samuel Rochet a par exemple modélisé les processus d'une norme d'IS qui est l'EIA-632 (cf. section suivante) à l'aide du langage SPEM.

3.2. Normes d'Ingénierie Système (IEEE-1220, EIA-632, ISO-15288)

Les normes d'ingénierie système sont définies en termes de processus (normalisés). Ces processus permettent de transmettre facilement, rapidement et sans ambiguïté les enchaînements et les imbrications d'activités à mettre en œuvre. Ces normes sont donc des standardisations de processus pour l'Ingénierie Système. Tantôt elles sont destinées à un domaine bien précis, tantôt elles ont un caractère beaucoup plus générique et peuvent s'appliquer dans des organismes de natures très variées. Tantôt elles ne couvrent qu'une partie du cycle de vie du produit (par exemple la phase de développement uniquement), tantôt leurs portées s'étendent à la totalité du cycle de vie (c'est-à-dire de la conceptualisation au retrait de service).

L'intérêt de concevoir, puis d'utiliser une norme, vient du fait qu'elle naît d'une capitalisation d'expérience et de savoir-faire de grandes entreprises, en tous cas pour celles qui concernent l'ingénierie système. Elle reprend donc les bonnes pratiques d'Ingénierie couramment constatées, mais se base aussi sur des échecs industriels pour s'améliorer et apporter de nouvelles activités censées éviter ces échecs. Une norme procure donc les meilleurs pratiques possibles connues pour une bonne conduite et une bonne gestion d'un programme (ou projet) industriel.

Actuellement, on dénote 3 grandes normes d'Ingénierie Système qui sont : l'IEEE-1220 [IEEE-1220, 2005], l'EIA-632 [EIA-632,1999] et l'ISO-15288 [ISO-15288, 2008]. Elles définissent donc les types d'activités à réaliser et les résultats produits. Elles recouvrent des champs différents, de manière d'autant plus approfondie que leur champ est plus limité, et ainsi se complètent.

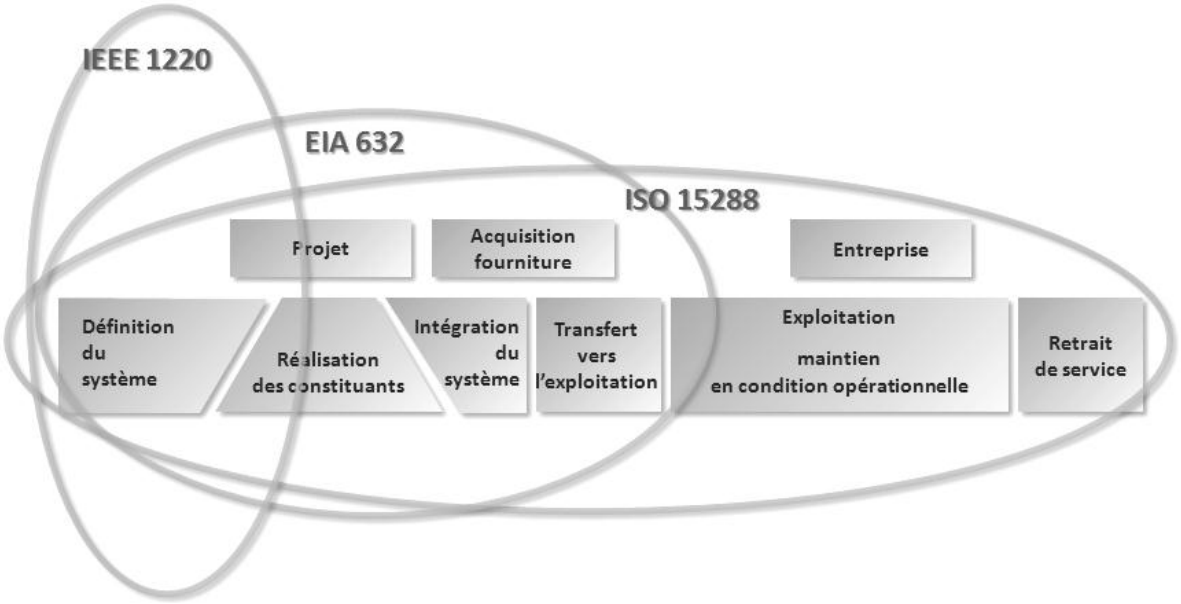


Figure II.8 : Couverture des normes IEEE-1220, EIA-632 et ISO-15288

Les définitions données par l'AFIS pour les normes IEEE-1220 et ISO-15288 sont :

- **Norme IEEE 1220 :** Issue du standard militaire MIL STD 499B, cette norme de l'IEEE, dont la version initiale date de 1994, se focalise sur les processus techniques d'ingénierie système allant de l'analyse des exigences jusqu'à la définition physique du système. Les trois processus d'analyse des exigences, d'analyse fonctionnelle et allocation et de synthèse, finement détaillés, comprennent chacun leur sous-processus de vérification ou de validation. Le processus d'analyse système a pour but d'analyser dans un cadre pluridisciplinaire les problèmes (conflits d'exigences ou solutions alternatives) issus des processus principaux afin de préparer les décisions. Le processus de maîtrise de l'IS (control) concerne tout particulièrement la gestion technique de l'ingénierie système et la maîtrise de l'information tant du système que du projet.
- **Norme ISO 15288** : Inspirée sur le plan et la forme par la norme ISO/CEI 12207 – AFNOR Z 67- 150 (Typologie des processus du cycle de vie du logiciel), cette norme de l'ISO, datant de 2003 et revue en 2008, étend les processus techniques à tout le cycle de vie du système (elle couvre ainsi les processus d'exploitation, de maintien en condition opérationnelle et de retrait de service). La norme s'applique à l'ingénierie des systèmes contributeurs qui ont leur propre cycle de vie (systèmes de fabrication, de déploiement, de soutien logistique, de retrait de service) : que l'on songe par exemple à l'ingénierie des systèmes de démantèlement et de traitements des déchets d'une installation nucléaire. Elle complète les processus s'appliquant aux projets par des processus, dits d'entreprise, qui ont pour objectif de développer le potentiel de

l'organisme d'IS en manageant les domaines communs au profit des projets d'IS. Les processus de cette norme (version 2003) sont rappelés en appendice au glossaire de base AFIS.

La norme que nous avons choisie d'utiliser dans nos travaux et qui sera détaillée plus loin est l'EIA-632, ceci pour plusieurs raisons. Tout d'abord, elle offre un bon compromis de couverture du cycle de vie par rapport à l'IEEE-1220 qui est trop spécifique aux toutes premières phases du projet et par rapport à l'ISO-15288 qui englobe la gestion de l'entreprise que nous n'abordons pas dans le travail présenté dans ce mémoire. Ensuite, l'EIA-632 s'avère être une norme très populaire et efficace, particulièrement utilisée dans les domaines aéronautiques et militaires. Le dernier point intéressant qui ressort au travers de cette norme est la définition et la prise en compte de « produits contributeurs ». Nous cernerons ce qu'ils représentent dans la section consacrée à la norme EIA-632.

3.3. Paradigme processus-méthodes-outils

Les normes d'ingénierie système proposent donc un ensemble de processus à réaliser, comme nous venons de le voir. Elles offrent un guide méthodologique fait de tâches, mais en ne s'intéressant principalement qu'aux résultats à obtenir en fonction des entrées. C'est-à-dire qu'un processus ne décrit pas la manière de réaliser le travail correspondant.

En fait, une méthode est bien souvent nécessaire pour remplir l'action demandée par un processus. Elle sert à « implémenter » le processus en question et ne doit pas être choisie pour sa flexibilité ou sa popularité, mais uniquement parce qu'elle reflète la sémantique du processus. Les méthodes peuvent bien entendu être sélectionnées parmi celles existantes, ou éventuellement être développées spécifiquement.

Pour faciliter l'application d'une méthode, des outils sont utilisés. Il est possible que plusieurs outils existent pour répondre à une méthode, de même que plusieurs méthodes peuvent satisfaire un processus.

Ainsi, dans cette approche résumée par la Figure II.9, un processus est implémenté par une méthode qui peut être appliquée en utilisant un outil. Nous pouvons noter qu'il est impossible d'utiliser un outil implémentant un processus sans avoir préalablement identifié la méthode appropriée.

Figure II.9 : Approche Processus-Méthodes-Outils

3.4. EIA-632

Dans cette section, nous présentons la norme choisie pour appuyer notre travail : il s'agit de l'EIA-632.

3.4.1. Historique et généralité

En juin 1994, un groupe de travail composé d'industriels, de membres de l'INCOSE et du *Department of Defense* (DoD) américain s'est réuni et a commencé de développer un

standard pour l'ingénierie des systèmes. Ce travail a été réalisé par le comité G-47 d'Ingénierie Système de l'*Electronic Industries Alliance* (EIA). Il en a été dégagé une ébauche de la norme EIA-632, qui deviendra un standard destiné à être utilisé par des entreprises commerciales, mais aussi par des agences gouvernementales et leurs sous-traitants.

En avril 1995, un groupe plus formel de travail, établi sous le projet PN-3537 et composé de représentants de l'EIA et de l'INCOSE, a produit une première version « finale » du standard. La version de la norme avec laquelle nous avons travaillé date elle de 1998 (et a été approuvé le 7 janvier 1999).

Ainsi, la norme EIA-632 définie une approche d'ingénierie ou de réingénierie des systèmes, incorporant les bonnes pratiques constatées dans la seconde partie du XX^{ème} siècle. Elle propose au développeur système un ensemble cohérent de processus organisés en 5 groupes :

- Les processus d'acquisition et de fourniture,
- Les processus de management technique,
- Les processus de conception système,
- Les processus de réalisation du produit,
- Les processus d'évaluation technique.

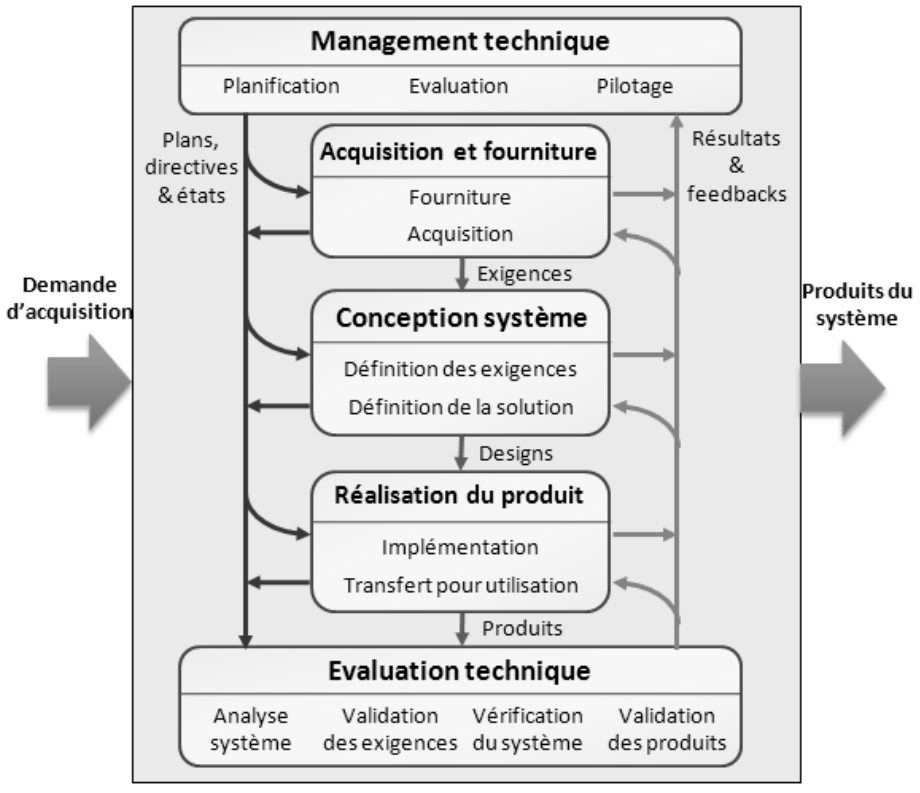


Figure II.10 : Les processus de l'EIA-632 (traduction française)

3.4.2. Interaction entre les différents processus

De manière synthétique, la Figure II.10 présente les interactions entre les différents groupes de processus, qui sont les suivantes :

1) Une demande d'acquisition survient et est traitée par le processus de *Fourniture* (car il s'agit de fournir l'objet de la demande au client) par l'établissement d'un contrat,
2) Les exigences de l'acquéreur (provenant du contrat) sont transmisses aux processus de *Conception Système* en charge de l'élaboration de la solution logique, puis physique, ainsi que de plusieurs ensembles d'exigences techniques spécifiées, dont chacun est associé à un sous-système,
3) Le processus d'*Acquisition* s'occupe d'acheter (si disponible sur le marché) ou de faire fabriquer les sous-systèmes répondant aux différents ensembles d'exigences spécifiées,
4) Une fois les sous-systèmes reçus, la réalisation du produit final peut commencer (processus d'*Implémentation*), basée sur la solution de conception préalablement établie et choisie,
5) Pour finir, le système final est transmis à l'utilisateur (processus de *Transfert pour Utilisation*), juste après avoir subi des tests et la validation finale.

En parallèle, tous les processus précédents sont gérés, cadencés et contrôlés par les processus de *Management Technique*. Les processus d'*Evaluation Technique* permettent quant à eux d'effectuer des analyses système (par exemple des analyses de risques), des validations d'exigences ou des vérifications système, ceci durant le développement et si nécessaire.

3.4.3. Les 33 sous-processus de l'EIA-632

Les 5 groupes de processus de l'EIA-632 organisent en fait 13 processus d'ingénierie système, qui eux même se décomposent en 33 sous-processus en tout. Le développeur doit alors décider lesquels de ces 33 sous-processus utiliser. La Figure II.11 fournit l'ensemble de ces sous-processus.

SOUS-PROCESSUS DU PROCESSUS DE FOURNITURE 1-Fourniture des produits **SOUS-PROCESSUS DU PROCESSUS D'ACQUISITION** 2-Acquisition des produits 3-Performance des fournisseurs **SOUS-PROCESSUS DU PROCESSUS DE PLANIFICATION** 4-Stratégie de mise en œuvre du processus 5-Définition l'effort technique 6-Ordonnancement et organisation 7-Plans techniques 8-Directives de travails **SOUS-PROCESSUS DU PROCESSUS D'EVALUATION** 9-Progrès par rapport aux plans et aux calendriers 10-Progrès par rapport aux exigences 11-Revues techniques **SOUS-PROCESSUS DU PROCESSUS DE PILOTAGE** 12-Gestion des résultats 13-Diffusion de l'information	**SOUS-PROCESSUS DU PROCESSUS DE DEFINITION DES EXIGENCES** 14-Exigences de l'acquéreur 15-Exigences des autres parties prenantes 16-Exigences techniques du système **SOUS-PROCESSUS DU PROCESSUS DE DEFINITION DE LA SOLUTION** 17-Représentations de la solution logique 18-Représentations de la solution physique 19-Exigences spécifiées **SOUS-PROCESSUS DU PROCESSUS DE REALISATION** 20-Réalisation **SOUS-PROCESSUS DU PROCESSUS DE TRANSFERT POUR UTILISATION** 21-Transfert pour utilisation	**SOUS-PROCESSUS DU PROCESSUS D'ANALYSE SYSTEME** 22-Analyse d'efficacité 23-Analyse des compromis 24-Analyse des risques **SOUS-PROCESSUS DU PROCESSUS DE VALIDATION DES EXIGENCES** 25-Validation de la déclaration des exigences 26-Validation des exigences de l'acquéreur 27- Validation des exigences des autres parties prenantes 28- Validation des exigences techniques du système 29- Validation des représentations de la solution logique **SOUS-PROCESSUS DU PROCESSUS DE VERIFICATION DU SYSTEME** 30-Vérification de la solution de conception 31-Vérification des produits finals 32-Disponibilité des produits capacitants **SOUS-PROCESSUS DU PROCESSUS DE VALIDATION DES PRODUITS FINALS** 33-Validation des produits finals

Figure II.11 : Les 33 sous-processus de l'EIA-632

Remarque : *Le terme « sous-processus » que nous employons ici remplace celui de « requirement » définis dans la norme EIA-632. En effet, nous avons décidé de ne pas traduire « requirement » par « exigence », mais plutôt par « sous-processus » afin d'éviter toutes confusions vis-à-vis des exigences proprement dites du système (c'est-à-dire les exigences d'acquéreur ou des autres parties prenantes, les exigences techniques spécifiées, les exigences techniques dérivées, etc.).*

3.4.4. Structure d'un système selon l'EIA-632

Pour pallier le fait que la négligence des produits qui contribuent à avoir le système désiré est une cause d'échecs de projets industriels [Messaadia, 2008], l'EIA-632 adopte une décomposition originale et très intéressante du système dont la brique de base est le bloc de construction (traduction de « *building block* »). Le système complet est décomposé en un (voire plusieurs) produit final auquel est associé un ensemble de produits contributeurs (traduction choisie pour « *enabling products* »).

Ces produits contributeurs [Martin, 2001] représentent tous les éléments ou les systèmes annexes nécessaires au développement, à la production, au test, au déploiement et au support du produit final, ainsi qu'à la formation des opérateurs et des agents de maintenance et au retrait du service des systèmes qui ne sont plus utilisables. Ils devront être disponibles au moment opportun, en étant soit achetés, soit fabriqués.

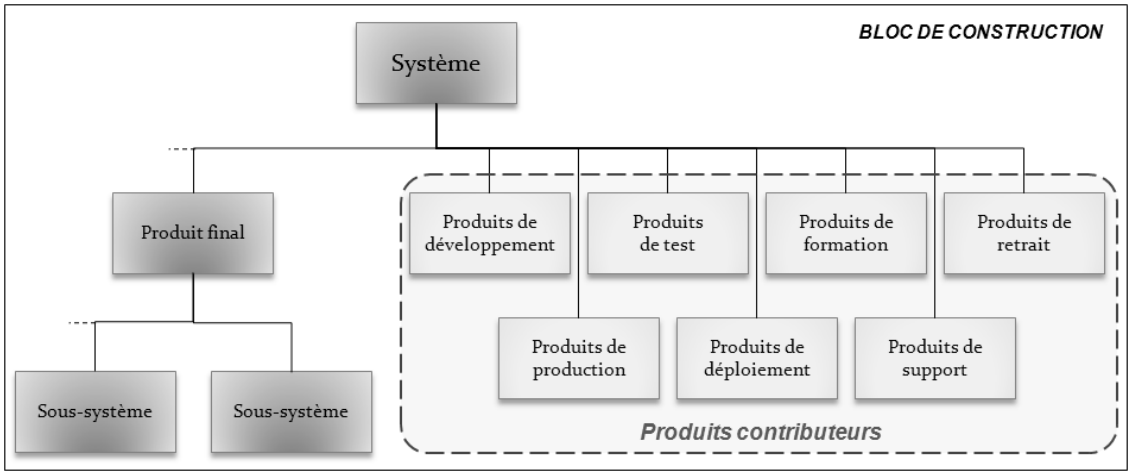


Figure II.12 : Un bloc de construction (traduction française)

Le système final est quant à lui décomposé en une hiérarchie de sous-systèmes définis en tant que modules. Le développement des modules de niveau inférieur est lancé dès que le module de niveau supérieur est complètement spécifié. Chaque module est alors considéré comme un produit qui a des caractéristiques et des exigences identifiées et fait l'objet d'un développement de même nature que le système global.

La décomposition se poursuit jusqu'à l'identification d'une des trois catégories de produits finals :

- Produits finals sur étagère,
- Produits finals pouvant être implémentés directement,
- Produits finals pouvant être fournis par un sous-traitant.

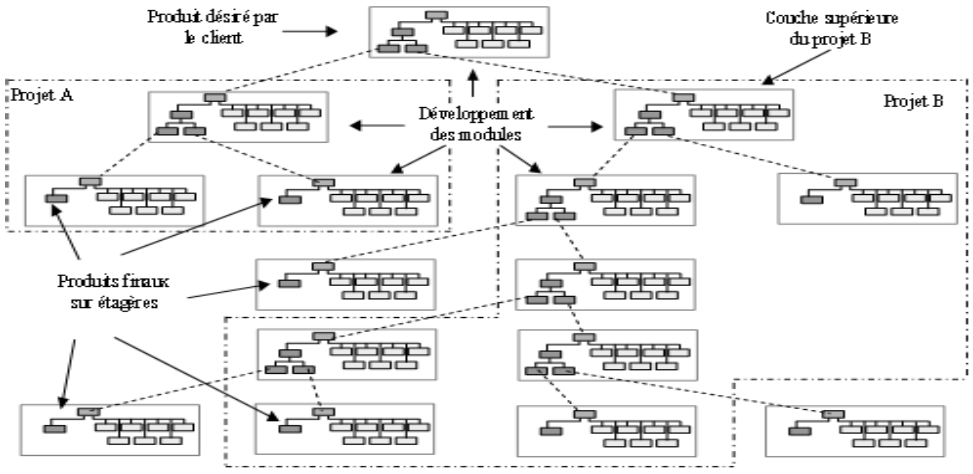


Figure II.13 : Exemple de décomposition d'un système avec l'EIA-632

Suite à la présentation de ses processus et de sa vision d'un système, nous abordons un point de l'EIA-632 substantiel pour notre travail qui est la gestion des exigences. En effet, nous avons évoqué dans la problématique l'importance d'une gestion efficace des exigences pour une conception sûre de fonctionnement (cf. chapitre 1).

3.4.5. Vision des exigences de l'EIA-632

Dans le texte de la norme EIA-632, il est question de plusieurs catégories d'exigences (revoir le chapitre 1 paragraphe 3.4.1. pour la définition d'une exigence). En effet, il apparait des exigences d'acquéreur, des exigences des autres parties prenantes, des exigences techniques du système, des exigences techniques dérivées ou encore des exigences spécifiées. Plusieurs types de solution sont également cités : la solution logique, la solution physique et la solution de conception. La gestion des exigences, dont un schéma à caractère non-normatif est présenté en annexe de la norme (annexe G) et est repris en Figure II.14, permet de bien comprendre l'articulation entre toutes ces exigences et solutions.

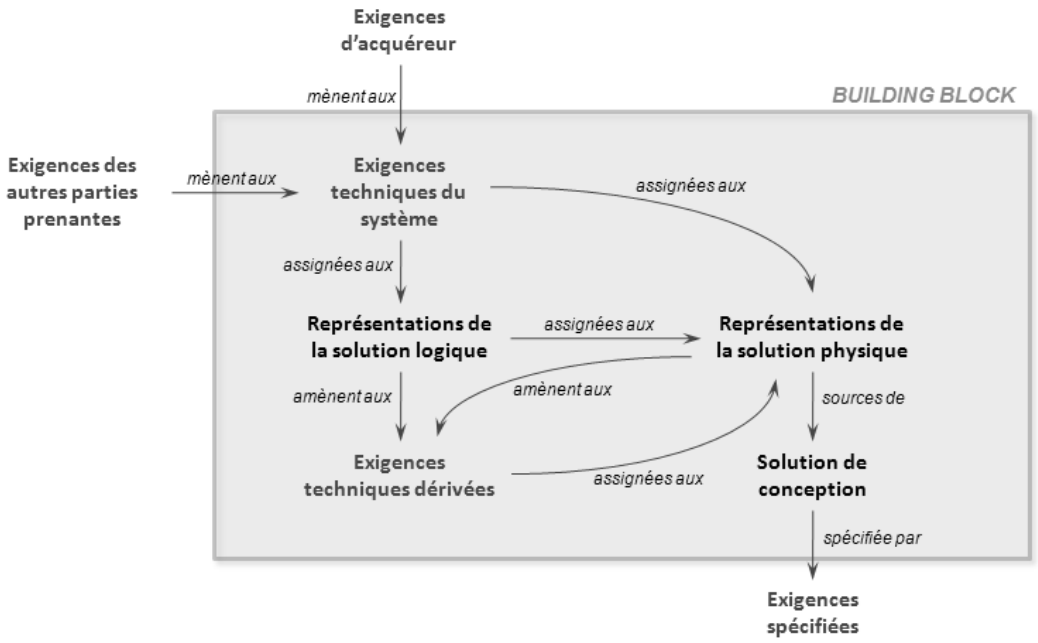


Figure II.14 : Relations entre exigences

Sur ce schéma, relatif à la construction d'un *building block*, les exigences d'acquéreur et celles des autres parties prenantes conduisent aux exigences techniques du système. Cette transformation d'exigences est réalisée par le processus de définition des exigences et le but est d'aboutir à une spécification en exigences dites techniques du système qui doit être non ambigüe, complète, consistante, réalisable, vérifiable et nécessaire et suffisante pour concevoir le système.

Ensuite, une partie des exigences techniques du système permet d'aboutir à des représentations de la solution logique. Des exigences techniques dérivées peuvent alors émerger, suite à un affinement d'exigences techniques ou à des choix de solution logique qui imposent des contraintes pour la suite du développement.

Puis la solution physique est étudiée de façon à convenir à la solution logique et à intégrer les exigences techniques du système qui n'ont pas encore été allouées. Là aussi, des exigences techniques dérivées peuvent surgir et s'ajouter aux précédentes. Elles devront être directement satisfaites par les représentations de la solution physique.

Enfin, une solution de conception émerge des représentations de la solution physique et elle conduit à plusieurs ensembles d'exigences spécifiées.

Chaque ensemble d'exigences spécifiées va donner les exigences d'acquéreur assignées pour le développement des blocs de construction du niveau inférieur (celui des sous-systèmes) comme le montre la Figure II.15 ci-dessous.

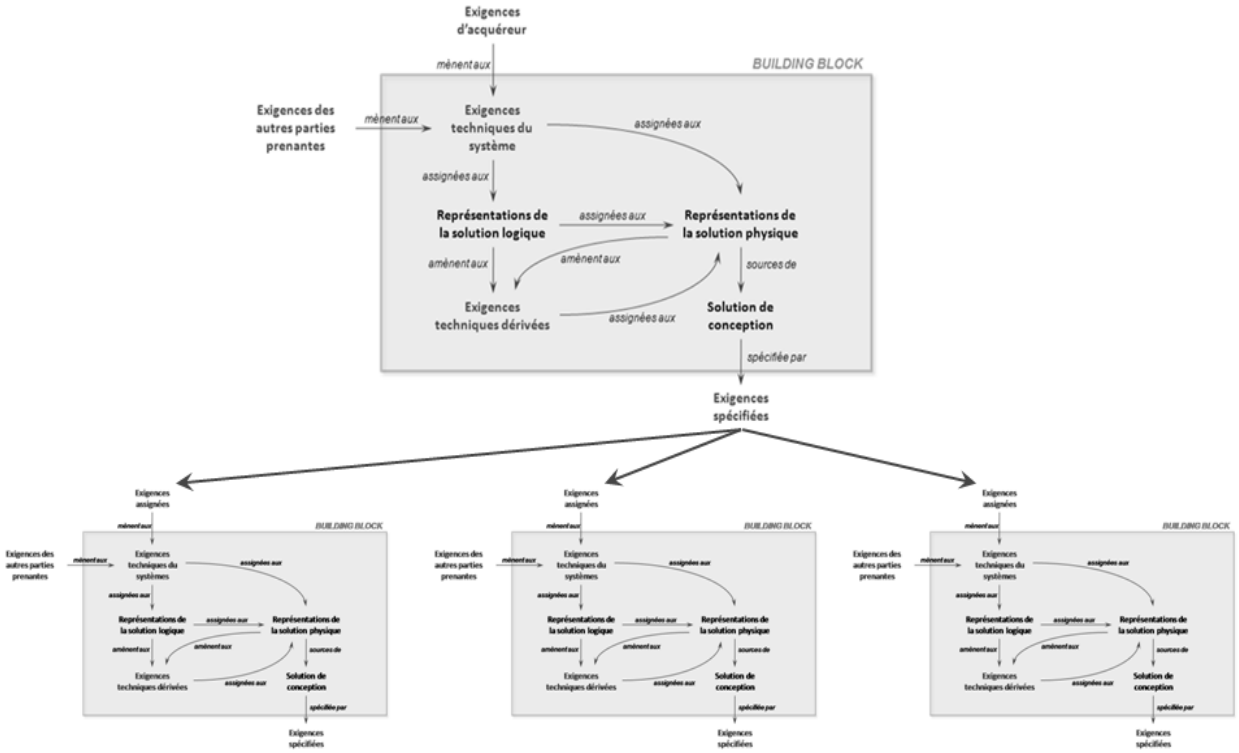


Figure II.15 : Rôle des exigences spécifiées

Cette gestion des exigences intègre un aspect de traçabilité, qui est effectivement cité à plusieurs endroits dans la norme. La traçabilité est d'ailleurs un garant fondamental au bon déroulement de la conception et est un facteur déterminant du point de vue de la

qualité et de la sûreté de fonctionnement du système, comme nous l'avons constaté dans l'analyse faite à la fin du premier chapitre concernant les faiblesses des activités d'ingénierie système.

Un exemple d'application de la norme EIA-632 pour la conception système peut être consulté dans [Esteban et al., 2009].

4. Approche globale proposée

4.1. Principe d'intégration de la SdF

L'intégration de la sûreté de fonctionnement doit concerner tous les processus d'ingénierie système. Il s'agit de pouvoir implanter dans l'approche générale de conception d'un système (défini par les processus de l'EIA-632 dans notre cas) le schéma de traitement de la sûreté de fonctionnement identifié dans la synthèse de la section précédente.

Dans un premier travail [Guillerm & al., 2009b], [Guillerm & al., 2010b], l'accent est mis principalement sur les *processus de conception système* et les *processus d'évaluation technique*.

Les exigences de sûreté de fonctionnement doivent être prises en compte dans le processus de définition des exigences, qui permet la formulation, la définition, la formalisation et l'analyse de ces exigences. Puis, un modèle de traçabilité [Gotel & al., 1994] doit être établi pour assurer la prise en compte des exigences à travers le cycle de développement du système. Ces exigences de sûreté de fonctionnement influencent les exigences d'acquéreur, les exigences des autres parties prenantes, les exigences techniques du système, les représentations de la solution logique et les représentations de la solution physique.

Les processus d'évaluation technique définissent 12 types de sous-processus allant de la validation de la déclaration des exigences jusqu'à la vérification de disponibilité des produits contributeurs. Les tâches associées à chaque sous-processus peuvent être consultées dans [EIA-632, 1999]. Les sous-processus considérés sont :

- validation de déclaration des exigences,
- validation des exigences de l'acquéreur,
- validation des exigences des autres parties prenantes,
- validation des exigences techniques dérivées,
- validation des représentations de la solution logique et vérification de la solution de conception.

L'implémentation de l'approche consiste à identifier et indiquer de quelle manière la sûreté de fonctionnement doit être considérée par chaque sous-processus de l'EIA-632. Autrement dit, les sous-processus du standard EIA-632 sont traduits ou raffinés en termes de sûreté de fonctionnement.

4.2. Les processus de conception système

Les processus de conception système sont utilisés pour convertir les exigences de toutes les parties prenantes (acquéreur et autres parties prenantes) en un ensemble de spécifications, plan ou modèles. Cet ensemble forme la solution de conception qui doit être réalisable et doit satisfaire et apporter une réponse à toutes les exigences.

Pour se faire, deux processus sont impliqués : le processus de définition des exigences et le processus de définition de la solution. Les relations entre ces processus sont visibles sur la Figure II.16.

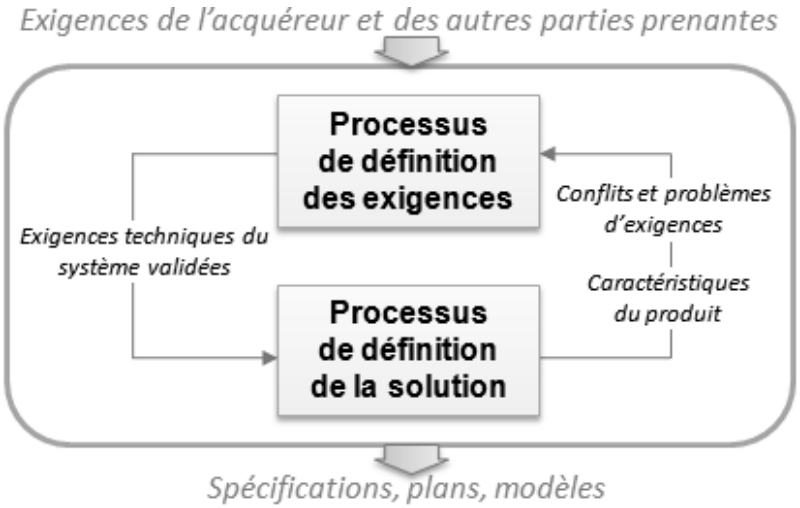


Figure II.16 : Processus de conception système

4.2.1. Le processus de définition des exigences

Le rôle du processus de définition des exigences est de transformer les exigences des parties prenantes en un ensemble d'exigences techniques du système. Trois sous-processus sont associés à ce processus, ils traitent respectivement des exigences de l'acquéreur, des exigences des autres parties prenantes et des exigences techniques du système (voir Figure II.17). Le processus de définition des exigences est ré-accompli si nécessaire.

Les exigences de sûreté de fonctionnement doivent être prises en compte dans ce processus. Cela permet la formulation, la définition, la formalisation et l'analyse de ces exigences. Puis, un modèle de traçabilité doit être construit pour assurer la considération de ces exigences à travers le cycle de développement du système.

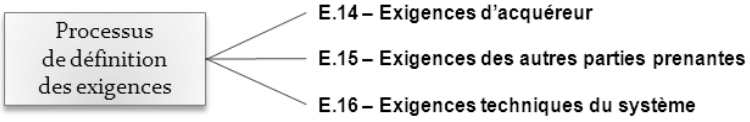


Figure II.17 : Processus de définition des exigences

4.2.1.1. E.14 : Exigences d'acquéreur

Concernant les exigences d'acquéreur, le développeur doit définir un ensemble validé d'exigences d'acquéreur pour le système, ou une partie du système. Pour cela, il doit :

1. Identifier, collecter et hiérarchiser les exigences pour le système, incluant également toutes exigences concernant le développement, la production, le test, le déploiement/l'installation, l'entrainement, l'utilisation, le support/la maintenance, et le retrait de service.
2. Assurer que les exigences répondent aux besoins et attentes de l'acquéreur, en faisant appel au sous-processus E.26 de validation des exigences d'acquéreur.

Si la distinction entre exigence fonctionnelle ou non-fonctionnelle n'a pas été possible au niveau du processus d'élicitation des exigences, le concepteur doit la faire afin de catégoriser les exigences.

Dans le cadre de la sûreté de fonctionnement, les exigences d'acquéreur correspondent généralement à des contraintes sur le système. Il est donc nécessaire d'identifier toutes les contraintes imposées par l'acquéreur, qui concerneront principalement la sécurité et la disponibilité (la fiabilité et la maintenabilité viennent dans un second temps). Une organisation hiérarchique doit être effectuée, en associant des poids aux exigences de sûreté de fonctionnement suivant leurs criticités. Par exemple, pour un avion de ligne, l'organisation hiérarchique consiste à associer à une exigence de fiabilité du système du train d'atterrissage une priorité plus élevé qu'à une exigence de fiabilité du système de climatisation.

4.2.1.2. E.15 : Exigences des autres parties prenantes

Concernant le deuxième sous-processus, le développeur doit définir un ensemble validé d'exigences des autres parties prenantes pour le système ou une partie du système. L'approche reste la même que celle des exigences d'acquéreur en termes d'identification, de collecte, de validation, etc.

Pour des exigences de sûreté provenant d'autres parties prenantes, elles peuvent, entre autres, provenir du respect d'une certification ou de contraintes de qualité. D'ailleurs, quelques standards guident le concepteur pour définir les exigences de sûreté de fonctionnement. Les systèmes critiques dans le secteur aérospatial civil sont par exemple développés en suivant les recommandations présentes dans [ARP-4754, 1996] et [ARP-4761, 1996]. Ces standards fournissent des conseils pour la « détermination » des exigences, incluant la capture des exigences, les types des exigences et les exigences dérivées.

Mise à part la source des exigences qui est différente, la même approche que pour les exigences d'acquéreur est appliquée pour les exigences des autres parties prenantes.

4.2.1.3. E.16 : Exigences techniques du système

Au niveau du sous-processus traitant des exigences techniques, le développeur doit définir un ensemble validé d'exigences techniques du système à partir des ensembles d'exigences d'acquéreur et d'exigences des autres parties prenantes.

Le résultat associé à l'accomplissement de ce sous-processus doit correspondre à un ensemble d'exigences techniques qui sont non-ambigües, complètes, consistantes, faisables, vérifiables, et nécessaires et suffisantes pour la conception du système.

Dans le cadre de la sûreté de fonctionnement, les exigences techniques du système traduisent des performances système. Cela consiste à définir des attributs de sûreté de fonctionnement tel que des MTTF[2] (temps moyen de fonctionnement avant panne), des MTBF[3] (temps moyen entre pannes), des MTTR[4] (temps moyen avant réparation), des taux de défaillances, ou encore des niveaux de confiance (SIL, ITSEC, TCSEC, SQUALE, …).

Parmi toutes les exigences formulées par l'acquéreur ou les autres parties prenantes, certaines d'entre elles sont des exigences de sûreté de fonctionnement. Elles seront une des sources d'exigences techniques de sûreté de fonctionnement.

Les études des évènements critiques définissent également des exigences de sûreté, dans le sens où leurs prises en comptes doivent conduire à une conception du système capable de les éviter. En fait, des appels au processus d'analyse des systèmes sont à faire pour procéder à des analyses (préliminaires) de risques, afin de définir, entre autres, les objectifs, la politique, les fonctions et les exigences de sûreté de fonctionnement. Il peut alors en découler des exigences de sûreté afin de réduire la probabilité des risques identifiés (par exemple à l'aide d'exigences de fiabilité) ou afin de réduire la conséquence des effets des risques (par exemple en ajoutant des barrières ou des mesures de protection). Les résultats d'analyses de risques constituent alors la seconde source d'exigences de sûreté de fonctionnement.

Par ailleurs, il est recommandé de définir des attributs pour les exigences dans le but de faciliter leur gestion, par exemple en exprimant les exigences avec un langage comme SysML [SysML, 2008]. De plus, SysML permettra de lier les exigences à la solution de conception.

4.2.2. Le processus de définition de la solution

Le processus de définition de la solution est en charge de l'élaboration des spécifications, représentations, plans et modèles correspondant à la solution de conception, ceci à partir de l'ensemble des exigences techniques du système. La solution de conception ainsi définie doit en fait satisfaire :

1. L'ensemble des exigences techniques du système qui proviennent du processus de définition des exigences,
2. Ainsi que l'ensemble des exigences techniques dérivées qui sont élicitées lors du processus même de définition de la solution.

Ce processus de définition de la solution dispose de trois sous-processus générant les représentations de la solution logique, les représentations de la solution physique et les exigences spécifiées (voir Figure II.18).

[2] Mean Time To Failure
[3] Mean Time Between Failures
[4] Mean Time To Repair

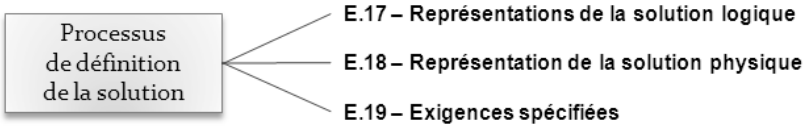


Figure II.18 : Processus de définition de la solution

4.2.2.1. E.17 : Représentations de la solution logique

Le développeur doit définir un ou plusieurs ensembles de représentations de la solution logique qui sont conformes avec les exigences techniques du système. Pour cela, il doit en particulier :

1. Faire des analyses de traçabilité,
2. Identifier et définir les interfaces, les états et les modes, les temps de réponse, et les flux de données et de contrôle,
3. Analyser les comportements,
4. Analyser les modes de défaillances et définir les effets des défaillances.

Notre recommandation est d'utiliser des modèles semi-formels ou formels (UML, SysML, réseaux de Petri, machine à états finis…) pour les représentations de la solution logique. L'utilisation de modèles formels facilite les vérifications et les analyses de par leur automatisation possible. Dans ce processus, les techniques d'analyse de la sûreté de fonctionnement sont utilisées pour le choix de la meilleure solution logique.

4.2.2.2. E.18 : Représentations de la solution physique

Le développeur doit définir un ensemble de représentations de la solution physique, qui satisfait les représentations de la solution logique, les exigences techniques dérivées et les exigences techniques du système.

Les représentations de la solution physique sont donc dérivées des représentations de la solution logique et doivent respecter toutes les exigences, en particulier les exigences de sûreté de fonctionnement. Les mêmes genres d'analyses de sûreté que pour la solution logique doivent être appliqués pour la solution physique. Les recommandations proposées pour la solution logique restent valables. Cependant, quand on approche de la solution physique, la sémantique de SysML semble ne pas être complètement suffisante. Il est alors possible d'utiliser des langages d'architecture comme AADL [Aer, 2004], VHDL-AMS [Verries and al, 2008], [Vhd, 1999], SystemC-AMS, etc.

4.2.2.3. E.19 : Exigences spécifiées

Pour ce dernier sous-processus de conception système, le développeur doit spécifier les exigences pour la solution de conception finale.

L'objectif est de caractériser entièrement la solution de conception. Le concepteur doit s'assurer que la solution de conception est cohérente avec ses exigences sources, des analyses de sûreté de fonctionnement permettant la validation de ces exigences (concernant la sûreté). Les exigences spécifiées (incluant les exigences fonctionnelles et de

performances, les caractéristiques physiques et les exigences de test) permettent alors une description complète du système final ainsi que de tous ses sous-systèmes.

Au fur et à mesure que les sous-processus de conception E.17 et E.18 progressent, le système se voit de plus en plus détaillé. Les risques sont alors affinés, et de nouveaux risques peuvent même apparaître. Des solutions de réduction de certains risques peuvent être à l'origine de nouveau problème de sûreté. Ainsi, de nouvelles actions de réductions de risques doivent être conduites et les exigences de sûreté doivent être affinées une nouvelle fois. Le processus évolue ainsi jusqu'à ce que tous les risques identifiés soient traités et atténués suffisamment pour les rendre acceptables (voir l'organisation des processus de conception système Figure II.16).

En fait, pendant la conception, il est nécessaire d'effectuer des activités de tolérance aux fautes et aux pannes qui vont permettre l'établissement de mécanismes de redondances, de reconfigurations ou de tolérance aux fautes. Il est également indispensable de procéder à des analyses de prévisions des dysfonctionnements (telles qu'une recherche de scénarios conduisant aux défaillances ou une analyse des conséquences des défaillances) qui doivent se faire en parallèle à la conception.

A travers toutes les étapes du développement, de nombreux éléments de modèle de différents niveaux d'abstraction sont créés, ce qui nécessite un réel suivi pour garantir la cohérence et l'intégrité de la conception. La traçabilité y contribue fortement, en liant les divers ensembles d'exigences (techniques, techniques dérivées, ou encore spécifiées), des solutions logique et physique. Là encore, l'utilisation du langage SysML peut apporter une aide.

4.3. Les processus d'évaluation technique

Les processus d'évaluation technique sont destinés à être invoqués par les processus de conception système. Ils forment un groupe de quatre processus, avec : le processus d'analyse des systèmes, le processus de validation des exigences, le processus de vérification du système et le processus de validation des produits finals.

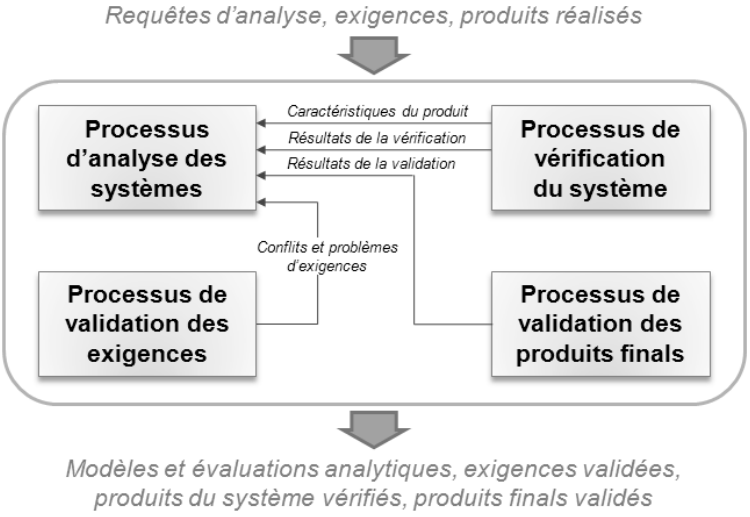


Figure II.19 : Processus d'évaluation technique

4.3.1. Le processus d'analyse des systèmes

Le processus d'analyse des systèmes est utilisé pour :

1. Fournir une base rigoureuse pour prendre les décisions techniques, résoudre les conflits d'exigences, et évaluer les alternatives de solutions physiques,
2. Déterminer les progrès vis-à-vis de la satisfaction des exigences techniques et techniques dérivées,
3. Supporter la gestion des risques,
4. Assurer que les décisions sont prises seulement après avoir évalué les coûts, les délais, les performances, et les effets des risques sur l'ingénierie ou la réingénierie du système.

4.3.1.1. E.24 : Analyse des risques

Un des sous processus appartenant au processus d'analyse système est celui d'analyse des risques. Dans ce sous-processus, le développeur doit procéder à des analyses de risques sur le système pour prévoir une stratégie de gestion des risques, supporter cette gestion des risques et supporter les prises de décisions qui en découlent. Plusieurs techniques peuvent être employées pour analyser les risques, par exemple : les arbres de défaillances [Lee & al., 1985] ou l'Analyse des Modes de Défaillance, de leur Effet et de leur Criticité (AMDEC) [Buzzatto, 1999]. Cette étape est très importante concernant la sûreté de fonctionnement, car elle détermine les risques du système qui est la base de réflexions qui apporteront de nouvelles exigences pour la sûreté du système.

Le sous-processus d'analyse des risques peut donc générer des exigences de sûreté autres que celles définies par l'acquéreur ou les autres parties prenantes. Ces nouvelles exigences doivent alors être prises en compte dans la conception du système.

4.3.2. Le processus de validation des exigences

La validation des exigences est une étape critique pour la réussite du développement et de l'implémentation du système. Les exigences sont validées quand il est certain qu'elles décrivent les exigences et objectifs d'entrée de telle sorte que le système résultant puisse les satisfaire. En fait, ce processus doit permettre de s'assurer que les exigences sont nécessaires et suffisantes pour la création de la solution de conception appropriée.

Durant ce processus, une attention particulière est portée sur l'analyse de la traçabilité, qui permet la vérification de tous les liens entre les exigences d'acquéreur, les exigences des autres parties prenantes, les exigences techniques et techniques dérivées, et les représentations des solutions logique et physique.

Comme les autres exigences, les exigences de sûreté doivent être validées. La validation doit permettre la conception de systèmes sûrs de fonctionnement. Pour faciliter cette étape, des solutions semi-formelles, telles qu'UML [Booch & al., 2005] ou SysML [SysML, 2008], peuvent être utilisées pour une bonne formulation des exigences. En effet, la diversité des personnes concernées par le projet de conception système qui peuvent avoir des connaissances limitées concernant la structure du futur système, rend les projets

d'ingénierie des exigences très difficile à l'échelle de l'entreprise. C'est pourquoi UML et SysML, à travers la diversité de leurs diagrammes, peuvent être vraiment utiles en offrant un langage commun et en présentant un ensemble de vues intégrées dans un même modèle général.

4.3.3. Le processus de vérification du système

Le processus de vérification du système est utilisé pour s'assurer que :

1. la solution de conception système générée est cohérente avec sa source d'exigences,
2. les produits finals remplissent leurs exigences spécifiées,
3. le développement ou l'obtention des produits contributeurs progressent,
4. les produits contributeurs requis seront prêts et disponibles lorsqu'ils devront l'être.

En particulier, il faut vérifier la conformité de la solution de conception vis-à-vis des exigences, dont les exigences de sûreté de fonctionnement.

La vérification formelle et la simulation [Zeigler & al., 2000] sont des méthodes couramment utilisées pour faire de la vérification système. D'autres méthodes telles que le test, le prototypage virtuel, la vérification de modèle peuvent être employées.

5. Conclusion

Ainsi, dans ce deuxième chapitre, nous avons passé en revue les différents travaux liés à notre problématique d'intégration de la sûreté de fonctionnement dans les processus d'ingénierie système dans la première partie. Le constat est que la majorité de ces travaux se situent dans le domaine informatique, d'une part, et s'intéressent plus à l'aspect méthodes et outils, d'autre part. A partir de là, dans la deuxième partie, nous avons présenté une approche globale de prise en compte de la sûreté de fonctionnement. Celle-ci s'appuie sur les processus définis dans la norme EIA-632 et définit les activités spécifiques à mettre en œuvre et qui concernent la sûreté de fonctionnement. Parmi les différents processus de l'ingénierie système, un processus spécifique est reconnu comme étant critique. Il s'agit du processus d'ingénierie des exigences, en particulier celles liées à la sûreté. Le chapitre suivant propose une approche de déclinaison des exigences de sûreté à différents niveaux d'un système complexe.

Chapitre 3 : Méthodologie de déclinaison d'exigences de sûreté de fonctionnement

1. Introduction

Après la présentation du deuxième chapitre sur l'approche globale de prise en compte de la sûreté de fonctionnement par des processus d'ingénierie système, ce troisième chapitre propose une méthodologie spécifique pour la déclinaison des exigences de sûreté de fonctionnement. Nous verrons que cette méthodologie s'intègre parfaitement dans l'approche générale du précédent chapitre.

L'objectif de cette méthodologie est d'aider à la définition d'exigences de sûreté de fonctionnement au niveau « système », mais surtout de décliner ces exigences en d'autres exigences de sûreté de fonctionnement associées aux différents sous-systèmes. Ainsi, la satisfaction des exigences aux niveaux des sous-systèmes permettra la satisfaction des exigences systèmes. L'intérêt de la méthodologie réside également dans la traçabilité générée entre les différentes exigences de sûreté de fonctionnement, à différents niveaux (système et sous-systèmes). Nous verrons l'importance de cette déclinaison et de cette traçabilité induite pour la gestion des exigences.

Ainsi, ce chapitre débute par la justification d'une telle méthodologie de déclinaison et par resituer sa place dans les processus d'ingénierie système. Puis, suit un bref état de l'art avant de présenter notre méthodologie. Nous donnons ensuite une formalisation à base d'UML de cette dernière. Puis, un cas d'étude simple permet d'illustrer la méthodologie. Pour finir le chapitre, nous dressons un bilan de cette nouvelle méthodologie en offrant quelques compléments.

2. Déclinaison d'exigences de sûreté de fonctionnement

Avant de présenter la méthodologie de déclinaison d'exigences de sûreté de fonctionnement, il convient de justifier le besoin d'une telle méthode, ainsi que de situer cette déclinaison parmi l'ensemble des processus d'ingénierie système.

2.1. Pourquoi cette déclinaison ?

Dans la problématique, nous avons vu que la complexification des systèmes impose une évolution des études des propriétés de sûreté, afin d'assurer et de permettre les niveaux de confiance nécessaires. Pour une considération effective de la sûreté de fonctionnement par

le processus de conception, il est impératif de considérer cette sûreté à travers des études globales dans les processus d'ingénierie système.

Une fois que les propriétés de sûreté ont été identifiées globalement (c'est-à-dire élicitées), celles-ci doivent être déclinées localement pour être effectivement réalisées par le système. Les propriétés locales associées aux sous-systèmes ou aux composants doivent être établies afin d'assurer les propriétés globales, ce qui implique un travail de traçabilité et des activités d'ingénierie des exigences.

2.2. Déclinaison vis-à-vis des processus d'ingénierie système

Nous avons vu au chapitre 1 l'importance de l'ingénierie des exigences dans l'ingénierie système. L'ingénierie des exigences est considérée comme un processus crucial dans la conception des systèmes complexes.

Concernant les exigences de sûreté de fonctionnement, celle-ci peuvent être classifiées comme des exigences non-fonctionnelles et sont liées aux propriétés émergentes du système. Définies au niveau du système, elles ne peuvent pas être attribuées à un seul constituant du système. Bien entendu, ces exigences non-fonctionnelles sont fondamentales pour le succès d'un projet de conception.

Deux principales activités sont définies dans l'ingénierie des exigences. La première concerne le développement des exigences, incluant les processus d'élicitation, de documentation, d'analyse et de validation des exigences. La seconde s'intéresse au management des exigences, incluant le management de la maintenance, la gestion des changements et la traçabilité des exigences.

La méthodologie de déclinaison proposée permet de prendre en compte les exigences de sûreté de fonctionnement en facilitant la gestion de leur traçabilité. En fait, elle concerne les deux processus principaux de la gestion des exigences : à la fois les activités de développement et celles de management. D'une part, la méthode aide à l'identification (c'est-à-dire l'élicitation) des exigences de sûreté de fonctionnement au niveau système. Puis, à l'aide de différentes analyses, elle permet de décliner les exigences globales de sûreté de fonctionnement en exigences locales, élicitées à leur tour. D'autre part, la traçabilité engendrée par la déclinaison répond parfaitement à un besoin des activités de management des exigences.

Positionnement par rapport à l'EIA-632

Pour bien situer la position de la méthodologie de déclinaison des exigences de sûreté de fonctionnement, la Figure III.1 reprend les principaux processus de l'EIA-632 qui entrent en jeu.

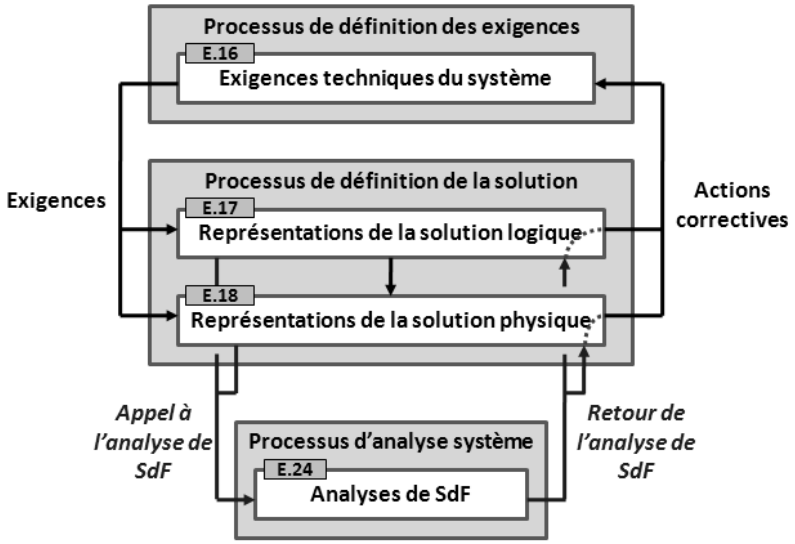


Figure III.1 : Evolution parallèle des processus de *Définition des Exigences* et de *Définition de l'architecture*

Nous avons vu dans la démarche méthodologique proposée à la fin du chapitre 2, qu'il existe différentes sources d'exigences de sûreté de fonctionnement. Celles si peuvent provenir :

1. directement de l'acquéreur ou d'une partie prenante (autre que norme ou certification),
2. de norme à respecter ou de certification à satisfaire,
3. ou encore d'analyse de sûreté de fonctionnement.

Dans la méthodologie, des analyses de sûreté (E.24) sont appelées par les processus de définition de la solution (E.17 et E.18) pour identifier les exigences de sûreté, ce qui correspond à la troisième source d'exigences citée plus haut. Ces exigences sont reprises par les processus de définition des exigences techniques du système (E.16) où elles seront bien définies et formalisées. C'est aussi à cette étape que la déclinaison doit être visible à l'aide de la traçabilité qui est gérée par le processus E.16. Toutes les nouvelles exigences sont ensuite transmises aux processus de définition de la solution pour être intégrées à la conception.

3. Travaux sur la déclinaison d'exigences de sûreté de fonctionnement

Dans la littérature, il existe quelques travaux qui traitent de déclinaison d'exigences de sûreté de fonctionnement. Principalement, des travaux ont été réalisés dans les universités de York (Grande-Bretagne) et de Queensland (Australie).

3.1. Travaux de Peter Lindsay, John McDermid et David Tombs

Les travaux de Peter Lindsay, John McDermid et David Tombs, des universités de York et de Queensland, [Lindsay & al., 1999], [Lindsay & al., 2000] et [Lindsay & al., 2002], sont plutôt orientés vers les systèmes logiciels. Ils partent du constat qu'une grande variété de

techniques d'analyse de sûreté de fonctionnement existe, mais que individuellement, ces techniques sont limitées pour faire face aux systèmes complexes, ou pour dériver et prioriser les exigences de sûreté des composants. Ils ont aussi constaté une confusion ou une ambigüité entre les techniques à utiliser pour évaluer les risques et celles plutôt destinées à assigner des objectifs de sûreté.

En fait, chaque composant individuel d'un système complexe a son propre comportement de défaillance, et les défaillances de ces composants mènent aux défaillances du système de façon plus ou moins prévisible. Un système complexe typique possède alors de multiples modes de défaillance. Dans leurs travaux, ils ont donc développé un processus pour dériver les exigences de sûreté de fonctionnement pour tous les composants du système. Ils ont notamment identifié deux aspects concernant la sûreté des composants :

- Un aspect *fonctionnel*, relatif à ce que le composant est supposé faire ou ne pas faire,
- Un aspect *quantitatif*, relatif au degré avec lequel le système peut tolérer la défaillance d'une fonction et continuer à fonctionner avec un risque acceptable.

Ainsi, pour traiter ces deux aspects, ils proposent un processus qui permet de classer les accidents potentiels et les risques systèmes, de montrer comment les défaillances systèmes conduisent aux risques, d'analyser les défaillances systèmes, et d'identifier les combinaisons de fautes de composants qui peuvent les causer. En utilisant un modèle d'erreur quantifié et basé sur les taux de défaillances tolérés des différentes classes d'accidents, et en exploitant les taux de défaillance des composants connus, ils dérivent des exigences quantifiées sur les composants restants.

Leur processus intègre des techniques existantes d'analyse de sûreté de fonctionnement pour une évaluation complète et quantifiée des risques du système. Il évalue la robustesse du système par rapport aux défaillances des composants et dérive des exigences de sûreté de fonctionnement pour les composants. En résumé, les étapes principales du processus sont les suivantes :

1. Construction d'un modèle de conception du système, qui définit le cadre du système et les fonctions de ses composants.
2. Identification et classification des risques du système. Une analyse de défaillance du type *Functional Failure Analysis* est suggérée.
3. Identification des fonctions de sûreté système et des mesures de protection à travers la considération de séquences d'accidents. L'utilisation d'arbre d'événements (*Event Tree*) est suggérée.
4. Construction d'un modèle détaillé de causes et effets, pour enregistrer comment les fautes se propagent dans le système du niveau composant au niveau système, et pour identifier les causes communes possibles. L'utilisation d'arbre de défaillance (*Fault Tree*) est suggérée.
5. Allocation d'un budget d'exigences de sûreté, en utilisant les modèles des étapes précédentes pour justifier les objectifs quantitatifs. C'est dans cette dernière étape que se concrétise la déclinaison des exigences de sûreté.

De manière très succincte, les étapes traitent respectivement les questions suivantes :

1. Comment fonctionne le système ?
2. Qu'est ce qui peut arriver de mal ?
3. Comment ce mal peut arriver ?
4. Pourquoi ce mal peut arriver ?
5. Quelle est la probabilité pour que ce mal arrive ?

Plus de détails sur ce processus sont disponible dans le rapport [Lindsay & al., 1999].

Pour appuyer la proposition de ce processus, des exemples sont proposés : celui d'une presse industrielle dans [Lindsay & al., 2000] et [Lindsay & al., 1999], ou celui d'un missile dans [Lindsay & al., 2002].

3.2. Travaux de Tim Kelly et al.

A l'université de York, les travaux sur la dérivation d'exigences de sûreté de fonctionnement ont continué. En particulier, dans [Wu & al., 2006] il est proposé un processus semblable à celui vu dans le paragraphe précédent, ainsi qu'une approche à base de Use Case Maps. L'objet de la proposition concerne aussi la génération d'exigences de sûreté de fonctionnement, et ce le plus tôt possible dans la phase de conception.

La problématique du travail se base sur un constat inclus dans celui que nous avions synthétisé à la fin du chapitre 1, qui est que des exigences inadaptées ou mal-comprises sont la cause majeur de problème de sûreté de fonctionnement. Elle s'appuie aussi sur le fait que des études ont montré l'énorme intérêt en termes de coût à régler les erreurs d'exigences tôt dans le processus de développement.

Le processus proposé s'inspire en fait du modèle *Twin Peaks* vu dans l'introduction générale, qui explicite clairement l'évolution parallèle et de manière évolutive des processus de définition des exigences et de définition des architectures (voir Figure III.2). Le principe de base de ce modèle est qu'un choix d'exigences peut déterminer un espace d'architectures, et qu'un choix d'architectures candidates peut à son tour contraindre les concepteurs à des exigences particulières.

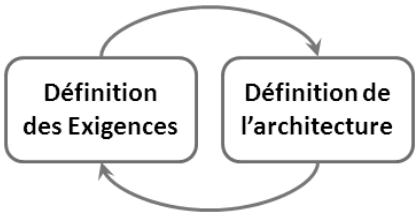


Figure III.2 : Evolution parallèle des processus de *Définition des Exigences* et de *Définition de l'architecture*

Ainsi, de manière condensée, le processus peut se résumer de la manière suivante :

Les risques sont identifiés par des analyses de risques faites à partir des descriptions disponibles de l'architecture du système et de son environnement opérationnel. Les risques identifiés sont évalués en termes de probabilité et de sévérité. Des décisions de conception architecturelle sont ensuite faites pour choisir les actions ou stratégies appropriées de réduction des risques, et des exigences de sûreté sont définies en réponse à ces choix de mécanismes de réduction de risques. Pendant que le processus de conception progresse et

que des détails supplémentaires sur le système sont obtenus, des risques vont être redéfinis et de nouveaux risques peuvent émerger. Les choix de mécanismes d'atténuation de risques peuvent eux-mêmes être à l'origine de nouveaux problèmes de sûreté. Ainsi, de nouvelles actions de réduction des risques doivent être identifiées et les exigences de sûreté doivent être affinées. Le processus évolue jusqu'à ce que tous les risques soient suffisamment atténués.

Un exemple de système de véhicule guidé automatiquement est également traité dans [Wu & al., 2006], afin d'illustrer le processus.

Dans des travaux précédents, T. Kelly, avec Karen Allenby, s'était également penché sur la possibilité d'utiliser les diagrammes de cas d'utilisation UML et des scénarios associés au cas d'utilisation pour conduire les analyses de risques. Plus de détails sur cette proposition sont disponible dans [Allenby & al., 2001]. Notamment, cette publication présente un exemple relatif au contrôle des reverses (ou inverseurs de poussée) d'un avion de ligne. Il illustre bien le fait que de nouvelles exigences de sûreté de fonctionnement sont créées suite à une analyse de risques, ceci pour différentes conditions de défaillance. Ces exigences de sûreté sont alors soit des contraintes de performance (comme un taux de défaillance ou un temps de réponse), soit des exigences fonctionnelles additionnelles.

Finalement, on s'aperçoit que le processus de dérivation d'exigences de sûreté de fonctionnement reste le même, quelques soit les techniques utilisées. C'est exactement ce qui est expliqué dans [Alexander & al., 2009], où les trois phases fondamentales des processus de dérivation d'exigences de sûreté ont été identifiées :

1. Identification des risques,
2. Analyse des risques,
3. Dérivation des exigences de sûreté de fonctionnement.

Il existe de nombreuses méthodes différentes, qui, en les combinant astucieusement, permettent la réalisation de chacune de ces étapes, et donc l'accomplissement du processus de dérivation. La même publication, [Alexander & al., 2009], cite un certain nombre de ces méthodes.

Les différents travaux que nous venons de présenter ne s'intéressent pas explicitement à la gestion de la traçabilité lors de la génération des exigences à différents niveaux. A notre connaissance, ce sont les seuls travaux traitant de la problématique de déclinaison des exigences de sûreté à différents niveaux du système.

4. Présentation de la méthodologie

Dans cette section, nous allons présenter la méthodologie de déclinaison d'exigences de sûreté de fonctionnement proposée [Guillerm & al., 2011]. Nous présentons aussi les deux méthodes de sûreté de fonctionnement utilisées : l'AMDEC et l'analyse par arbres de défaillances.

4.1. Aperçu de la méthodologie

L'objectif de la méthodologie est de pouvoir lier les exigences de sûreté de fonctionnement définies au niveau du système avec celles définies au niveau des sous-systèmes. Ces liens correspondent alors à la déclinaison des exigences.

La méthodologie se décompose en 4 étapes, organisées et définies de la manière suivante :

- **Etape 1** : analyse des défaillances au niveau du système complet qui peuvent conduire à une situation catastrophique ou, tout du moins, à un événement non-souhaité.
- **Etape 2** : analyses des causes de ces défaillances en s'appuyant sur l'architecture du système. Il s'agit de trouver les origines des défaillances système au niveau des sous-systèmes.
- **Etape 3** : analyses des défaillances au niveau des sous-systèmes.
- **Etape 4** : synthétiser la déclinaison des exigences de sûreté de fonctionnement à l'aide des informations des précédentes étapes.

La Figure III.3 résume le procédé de la méthode en montrant la séquence des différentes étapes et en spécifiant les informations d'entrées/sorties.

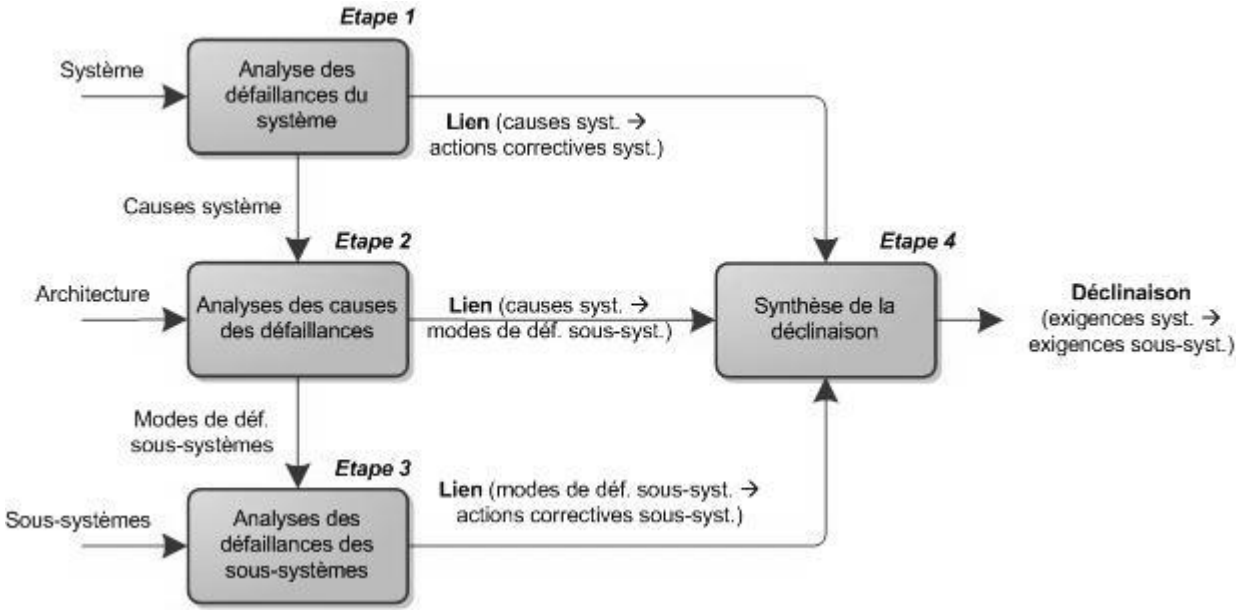


Figure III.3 : Vue générale de la méthode combinant AMDEC et Arbres de Défaillances

4.2. Méthodes utilisées

Pour appliquer cette méthodologie, nous avons choisi d'utiliser des méthodes existantes pour les étapes 1 à 3. Il s'agit de méthodes de sûreté de fonctionnement bien connues et très utilisées que sont : l'Analyse des Modes de Défaillance, de leurs Effets et de leurs Criticités (AMDEC) et l'analyse par Arbre de Défaillances (AdD). L'AMDEC a été sélectionnée car elle est bien adaptée à l'étape 1 et à l'étape 3. L'analyse par AdD est, quant à elle, un bon

choix pour réaliser l'étape 2. Il est à noter que d'autres méthodes peuvent être utilisées, du moment qu'elles remplissent l'objectif de l'étape à réaliser.

4.2.1. AMDEC

L'AMDEC (Analyse des Modes de Défaillance, de leurs Effets et de leurs Criticités) [Buzzatto, 1999], [Faucher, 2004], [Landy, 2007], [Faucher, 2009] est une analyse inductive de recherche des effets des pannes des composants sur les sous-systèmes et le système. C'est un procédé systématique pour identifier les modes potentiels de défaillances avant qu'elles ne surviennent, avec l'intention de les éliminer ou de minimiser les risques associés.

Historiquement, l'AMDEC fut développée dans le secteur aéronautique par l'armée américaine en 1949. Elle fut utilisée pour la première fois dans les années 60 pour des analyses de sécurité des avions et elle prit son essor en Europe au cours des années 70 dans les secteurs automobile, chimique et nucléaire. Initialement, la méthode était appelée AMDE (Analyse des Modes de Défaillance et de leurs Effets). Aujourd'hui, elle a ajouté l'estimation de la criticité des risques, d'où son nom : AMDEC.

Finalement, cette méthode spécifique à la sûreté de fonctionnement des systèmes permet de prioriser les interventions afin de réduire les risques les plus grands, mais aussi de formaliser la documentation. Elle aide à « prévoir » pour ne pas être obligé de « revoir ». Elle peut être utilisée pour un objet, un procédé de fabrication ou encore un processus de production. On parle alors respectivement d'AMDEC Produit, d'AMDEC Processus ou d'AMDEC Moyen.

Classiquement, les résultats d'une AMDEC sont consignés dans un tableau semblable à celui de la Figure III.4.

Système	Fonction	Mode de défaillance	Cause	Effet	Criticité/ Classification	Actions Correctives
1	*2*	*3*	*4*	*5*	*6*	*7*

Figure III.4 : Exemple de structure d'un tableau d'AMDEC

Réalisée sur un système donné (colonne n°1), l'AMDEC répertorie pour chaque fonction du système (colonne n°2) tous les modes de défaillance (colonne n°3). Puis il s'agit d'identifier les causes possibles de ces modes de défaillance (colonne n°4), sachant que plusieurs causes différentes peuvent être à l'origine d'un même mode de défaillance. Ensuite, les effets des modes de défaillances doivent être identifiés (colonne n°5). Après, une classification de chaque couple {cause + effet} de tous les modes de défaillance doit être effectuée (colonne n°6). Souvent, il s'agit de choisir une criticité parmi les 5 classes : *catastrophique*, *hasardeux*, *majeur*, *mineur* ou *sans importance*. A ce stade, l'analyse proprement dite est en principe finie. Mais une ultime étape est ajoutée. Il s'agit, pour finir, de définir les actions correctives à mettre en œuvre pour rendre les risques acceptables (colonne n°7).

Identifier les modes de défaillance :

Pour aider à l'identification des modes de défaillance, une liste de modes génériques peut être utilisée. Par exemple la norme CEI-60812 [CEI-60812, 2006] propose dans son tableau II la liste des modes génériques de la Figure III.5.

1	Défaillance structurelle (rupture)	18	Mise en marche erronée
2	Blocage physique ou coincement	19	Ne s'arrête pas
3	Vibrations	20	Ne démarre pas
4	Ne reste pas en position	21	Ne commute pas
5	Ne s'ouvre pas	22	Fonctionnement prématuré
6	Ne se ferme pas	23	Fonctionnement après le délai prévu (retard)
7	Défaillance en position ouverte	24	Entrée erronée (augmentation)
8	Défaillance en position fermée	25	Entrée erronée (diminution)
9	Fuite interne	26	Sortie erronée (augmentation)
10	Fuite externe	27	Sortie erronée (diminution)
11	Dépasse la limite supérieure tolérée	28	Perte de l'entrée
12	Est en dessous de la limite inférieure tolérée	29	Perte de la sortie
13	Fonctionnement intempestif	30	Court-circuit (électrique)
14	Fonctionnement intermittent	31	Circuit ouvert (électrique)
15	Fonctionnement irrégulier	32	Fuite (électrique)
16	Indication erronée	33	Autres conditions de défaillance exceptionnelles suivant les caractéristiques du système, les conditions de fonctionnements et les contraintes opérationnelles
17	Ecoulement réduit		

Figure III.5 : Exemple de modes de défaillances génériques extrait de l'CEI-60812

Identifier les causes :

Il existe aussi des méthodes pouvant aider à l'identification des causes possibles d'un mode de défaillances. Une possibilité est de se servir du diagramme d'Ishikawa avec ces 5 M : Main d'œuvre, Matériels, Milieu, Matières, Méthodes (voir Figure III.6).

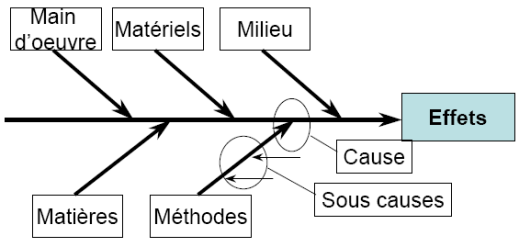


Figure III.6 : Diagramme d'Ishikawa avec les 5 M

4.2.2. Arbre de défaillances

L'analyse par arbre de défaillance [Lee & al., 1985], [Chatelet, 2000], [Sinnamon & al., 1996], [Vesely, 1987] est une méthode de type déductif, qui a été mise au point au début des

années 1960 par la compagnie américaine Bell Téléphone. Elle fut ensuite expérimentée pour l'évaluation de la sécurité des systèmes de tir de missiles, puis reprise dans de nombreux domaines tels que l'aéronautique, le nucléaire, l'industrie chimique, etc.

L'analyse par arbre de défaillance est largement répandue dans les études de sûreté de fonctionnement car elle caractérise de façon très claire les liens de dépendance, du point de vue du dysfonctionnement, entre les différents constituants d'un système. Cette technique est capable de gérer des combinaisons complexes de défaillances. Elle est donc bien adaptée à l'étude des défaillances multiples, même si elle ne s'intéresse pas à l'impact que peut avoir l'ordre dans lequel les événements sont considérés.

La méthode de l'arbre de défaillances part d'un événement indésirable et cherche à en connaître les causes possibles. L'objectif de la méthode est de déterminer les diverses combinaisons possibles d'événements qui entrainent la réalisation d'un événement indésirable donné. Le résultat est formé d'une représentation graphique des combinaisons au moyen d'une structure arborescente, comme celle montrée en Figure III.7. Un arbre de défaillances fourni alors une vue synthétique qui représente des interactions entre les composants d'un système du point de vue de la sûreté de fonctionnement.

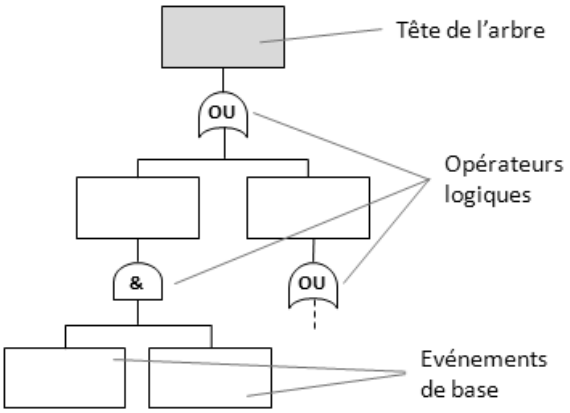


Figure III.7 : Exemple d'Arbre de Défaillances

La tête de l'arbre de défaillance (qui, par analogie, correspond à la racine) représente la plupart du temps un événement unique mettant sérieusement en question la sécurité du système étudié. Afin de faciliter l'analyse, cet événement indésirable (*top event*) doit être précisément défini. L'arbre de défaillance lui-même est alors formé de niveaux successifs d'événements tels que chaque événement est généré à partir des événements du niveau inférieur par l'intermédiaire de divers opréateurs (ou portes) logiques. Ce processus déductif est poursuivi jusqu'à ce que l'on obtienne des événements dits événements de base : les feuilles de l'arbre.

Présentation de la symbolique utilisée :

Les portes logiques utilisées sont essentiellement les portes ET et OU (voir Figure III.8). D'autres types de fonction logique existent, mais leur utilisation reste encore anecdotique.

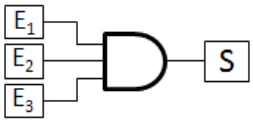
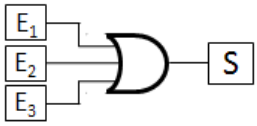
Symbole	Nom	Signification
	porte ET	L'événement de sortie (S) de la porte ET est généré si tous les événements d'entrée (E_1, E_2, … E_n) sont réalisés simultanément.
	porte OU	L'événement de sortie (S) de la porte OU est généré si au moins un des événements d'entrée (E_1, E_2, … E_n) est réalisé (OU INCLUSIF).

Figure III.8 : Les portes logiques

Concernant la représentation des événements, il existe en fait toute une série de représentations possibles, qui ont chacune leur propre signification. Le tableau de la Figure III.9 en répertorie un certain nombre.

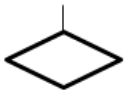
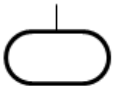
Symbole	Nom	Signification
	Rectangle	Evénement (intermédiaire) résultant de la combinaison d'autres événements par l'intermédiaire d'une porte logique
	Losange	Evénement qui ne peut être considéré comme élémentaire mais dont les causes ne seront pas développées (temps, moyens limités, etc.)
	Double losange	Evénement dont les causes ne sont pas encore développées mais le seront ultérieurement (analyse en cours)
	Cercle	Evénement élémentaire qui ne nécessite pas d'être développé plus avant.
	Maison	Evénement de base qui se produit normalement pendant le fonctionnement du système.
	Ovale	Evénement conditionnel ; utilisé avec certaines portes logiques.

Figure III.9 : Les représentations des événements

Le dernier type de symbole présenté ici est celui qui permet la connexion entre des sous-arbres. Il est bien utile dans le cas de redondance dans un arbre ou pour scinder un arbre trop complexe en arbres plus petits dans le but de faciliter sa lecture. Deux symboles de ce type sont donnés en Figure III.10.

Symbole	Nom	Signification
	Triangle	La partie de l'arbre qui devait se trouver à l'emplacement de ce symbole →  est déplacée à la suite du symbole → 
	Triangle inversé	Une partie semblable, mais non identique à celle qui devrait se trouver à l'emplacement de ce symbole →  est donnée à la suite du symbole → 

Figure III.10 : Les symboles de transfert de sous-arbres

Analyse quantitative :

L'arbre de défaillance offre une aide non-négligeable à des analyses quantitatives de sûreté de fonctionnement, en termes de fiabilité. En effet, il est possible de pondérer les événements de l'arbre par des probabilités, afin d'obtenir par exemple la probabilité de l'événement indésirable principal en fonction des probabilités des événements de base. Pour le calcul des probabilités, il s'agit d'utiliser les formules associées aux portes logiques (cf. Figure III.11).

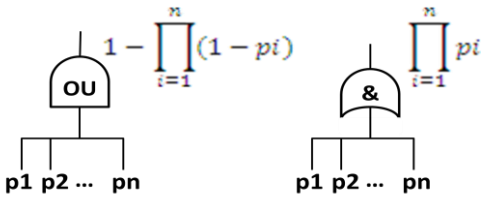


Figure III.11 : Formules pour le calcul des probabilités des fonctions OU et ET

4.3. Méthodologie de déclinaison

La méthode s'organise en quatre étapes, supposant que le système complexe est composé de plusieurs sous-systèmes. A partir de la définition du système, de ses fonctions, de son architecture, de ses sous-systèmes et de leurs fonctions respectives, la méthode combine des analyses AMDEC et des arbres de défaillances pour définir les exigences de sûreté de fonctionnement système et les informations sur la déclinaison de ces exigences globales en exigences locales associées aux sous-systèmes.

4.3.1. Etape 1 : Analyse des défaillances du système

Dans cette première étape de la méthode, il s'agit de conduire une AMDEC au niveau système. Pour chaque fonction du système, il s'agit d'identifier les modes de défaillances, leurs causes et leurs effets sur le système (éventuellement en fonction de la phase, l'état ou le mode du système). Puis il faut classifier ces effets et proposer les actions correctives nécessaires appropriées.

Une fois cette première étape achevée, l'AMDEC niveau système aura donc permis d'identifier, d'une part, un ensemble de causes système, ainsi que les criticités liées aux effets de ces causes. D'autre part, en fonction de ces criticités, des actions correctives relatives à ces causes système ont été définies. Ces actions correctives sont alors sources d'exigences de sûreté de fonctionnement à intégrer dans la conception et que le système doit satisfaire pour garantir un niveau de sûreté de fonctionnement suffisant. Notamment, elles peuvent se traduire en exigences de sûreté de fonctionnement au travers d'objectifs à respecter en termes de fréquence acceptable (il s'agit dans ce cas d'une action corrective qui porte sur les causes).

On note que des actions correctives peuvent aussi agir sur les effets d'un mode de défaillance, afin de réduire la gravité.

4.3.2. Etape 2 : Analyses des causes des défaillances

A partir des causes système identifiées dans la première étape, la seconde phase consiste à construire les arbres de défaillances qui vont lier les causes systèmes à des modes de défaillances des sous-systèmes. Ainsi, le « *top-event* » (c'est-à-dire la racine) de chaque arbre est une cause système. Le but est alors de déterminer les causes au niveau sous-système du *top-event*, en utilisant les opérateurs logiques tel que ET et OU, et sachant que les feuilles des arbres doivent correspondre à des modes de défaillances de sous-systèmes.

A ce stade, la distinction entre un **mode de défaillance** et une **cause** est bien visible. Un mode de défaillance correspond à l'effet par lequel une défaillance est reçue ou observée depuis un point de vue **externe**. Une cause représente quant à elle un événement qui conduit à un (ou plusieurs) mode de défaillance selon un point de vue **interne**.

C'est pourquoi ici, lorsque l'on considère le système et sa composition en sous-systèmes, on essaye de lier les modes de défaillance des sous-systèmes (point de vue externe sur les sous-systèmes), pour arriver, en passant éventuellement par des événements intermédiaires, à des causes système (point de vue interne sur le système). Ces dernières (les causes système) étant liées aux modes de défaillance système à travers l'AMDEC système.

Ainsi, ces arbres (dont un exemple générique est visible Figure III.12) permettent de lier une cause système à un ensemble de modes de défaillance de sous-systèmes.

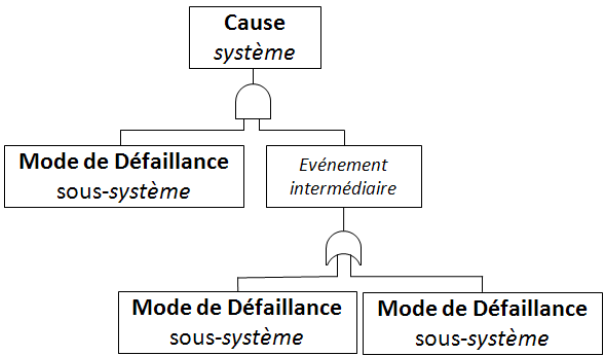


Figure III.12 : Exemple d'arbre de défaillances

Ultérieurement, il s'agit d'associer des probabilités aux éléments des arbres d'après les données de fiabilité disponibles et en respectant les objectifs de fiabilité que peuvent imposer les actions correctives de l'AMDEC niveau système. Les données d'entrée pour le calcul des autres probabilités peuvent donc être une combinaison entre des objectifs de fiabilité (niveau système) et des données ou hypothèses concernant les sous-systèmes (c'est-à-dire provenant des feuilles de l'arbre). L'essentiel est de rester cohérent dans la pondération de l'arbre, notamment en respectant les formules basiques de calcul des probabilités vis-à-vis des fonctions logiques (revoir Figure III.11). On utilise le terme de « budget » de fiabilité [Lindsay & al., 2000] pour désigner la marge de fiabilité disponible restant à attribuer aux sous-systèmes dont on n'a aucune donnée *a priori*.

4.3.3. Etape 3 : Analyses des défaillances des sous-systèmes

La troisième étape de la méthode consiste en la réalisation de plusieurs AMDEC, appliquées au niveau sous-système. Les modes de défaillances des sous-systèmes utilisés à l'étape 2 réapparaissent à travers ces AMDEC (par principe des analyses AMDEC). En fait, pour une cause système donnée, il est nécessaire de réaliser une AMDEC pour chaque sous-système dont au moins un mode de défaillance intervient dans l'arbre de défaillances de la cause système en question.

Une fois que cette phase est terminée, l'information utile pour notre méthode provient des relations entre les modes de défaillances des sous-systèmes et les actions correctives du niveau sous-système.

4.3.4. Etape 4 : Synthèse

La quatrième et dernière étape de la méthode est la synthèse qui fournit la déclinaison des exigences de sûreté de fonctionnement niveau système en exigences niveau sous-système.

Dans les étapes précédentes, des liens de 3 types différents ont été créés :

- Entre les causes système et les actions correctives système,
- Entre les causes système et les modes de défaillances sous-système,
- Entre les modes de défaillances sous-système et les actions correctives sous-système.

Ces différents liens permettent de relier les actions correctives système aux actions correctives sous-système. En traduisant les actions correctives par des exigences de sûreté de fonctionnement, nous obtenons des relations entre des exigences de sûreté de fonctionnement niveau système et des exigences de sûreté de fonctionnement niveau sous-système.

5. Formalisation UML

Nous allons, dans la section qui suit, proposer une formalisation possible de la méthodologie à l'aide d'UML. Cela revient à définir un modèle conceptuel liant les différents concepts utilisés, tels que les modes de défaillances, les effets, les causes, les actions correctives, etc.

L'intérêt de cette formalisation est double. Tout d'abord, elle permet de bien poser et montrer la relation entre les différentes notions, ce qui permet de clarifier la compréhension de la méthode. Le deuxième intérêt serait pour une implémentation éventuelle de la méthode dans un outil informatique. Le modèle faciliterait alors l'implémentation de l'outil, notamment concernant la base de données à manipuler.

Nous allons donc présenter le modèle conceptuel réalisé pour la méthode. Mais pour arriver au modèle complet, il a été nécessaire de formaliser l'AMDEC et les arbres de défaillances.

5.1. Formalisation de l'AMDEC

La formalisation UML de l'AMDEC que nous avons réalisée est visible Figure III.13. Brièvement, on retrouve le fait qu'une cause implique un (ou plusieurs) mode de défaillance, qui lui-même peut provoquer un effet.

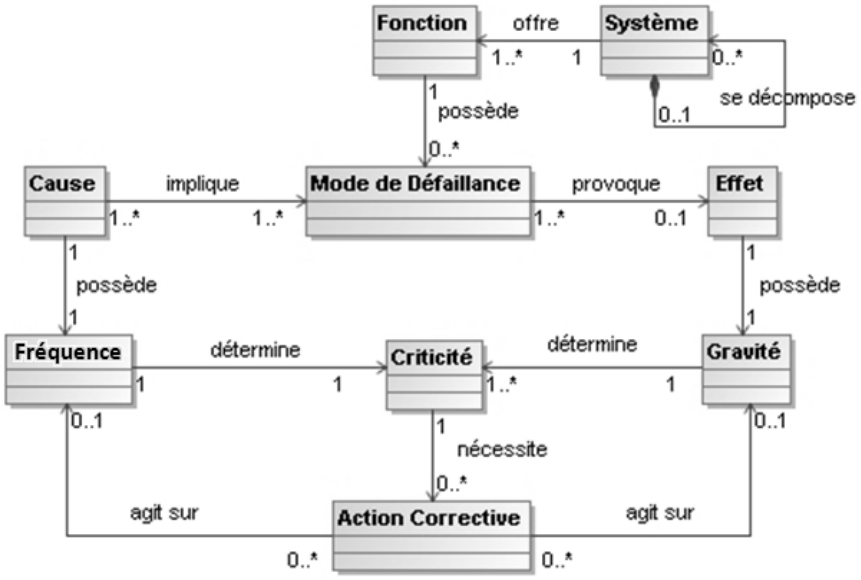


Figure III.13 : Formalisation de l'AMDEC en UML

Explication du modèle conceptuel :

Un système se décompose en sous-systèmes, qui eux-mêmes peuvent être considérés comme des systèmes. Chaque système offre un ensemble de fonctions. Une fonction possède des modes de défaillance. Des causes impliquent ces modes de défaillance. Les modes de défaillance provoquent des effets. Une cause possède une fréquence et un effet possède une gravité. La combinaison de la fréquence d'une cause avec la gravité d'un effet impliqué est source d'une criticité. Selon les cas, une criticité nécessite des actions correctives, qui vont agir sur la fréquence et/ou la gravité.

5.2. Formalisation de l'Arbre de Défaillances

La formalisation de l'arbre de défaillance en UML est donnée par la Figure III.14. Elle fait intervenir des événements. Ceux-ci peuvent être des causes ou des modes de défaillance, mais pas uniquement puisque des événements « intermédiaires » entre les

causes d'un système et les modes de défaillance des sous-systèmes peuvent être définis. L'autre concept qui apparaît est celui de « porte logique », qui peut être de type ET, OU, voire d'autres types (OU EXCLUSIF, OU PRIORITAIRE, ET PRIORITAIRE,…).

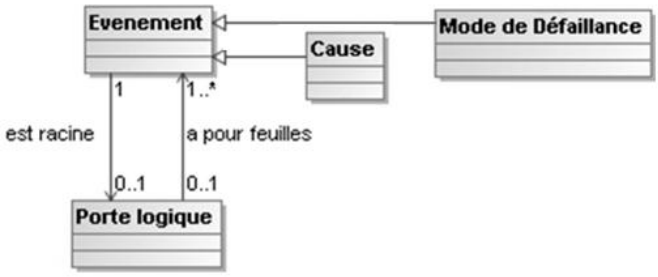


Figure III.14 : Formalisation de l'arbre de défaillance en UML

Un événement peut donc être « racine » (c'est-à-dire en sortie) pour une porte logique et/ou feuille (c'est-à-dire en entrée) pour une autre porte logique. Les termes « racine » et « feuille » sont utilisés ici pour rappeler que l'on construit un arbre.

5.3. Formalisation de la méthodologie complète

Pour la formalisation de la méthodologie complète, il s'agit d'intégrer les deux diagrammes précédents dans un seul schéma. A cela, il faut ajouter le fait que les actions correctives sont sources d'exigences de sûreté de fonctionnement. Ces exigences sont à prendre en compte pour la conception du système, elles sont donc allouées au système. Au final, le diagramme de la Figure III.15 formalise tous les concepts de la méthode dans un diagramme de classes UML.

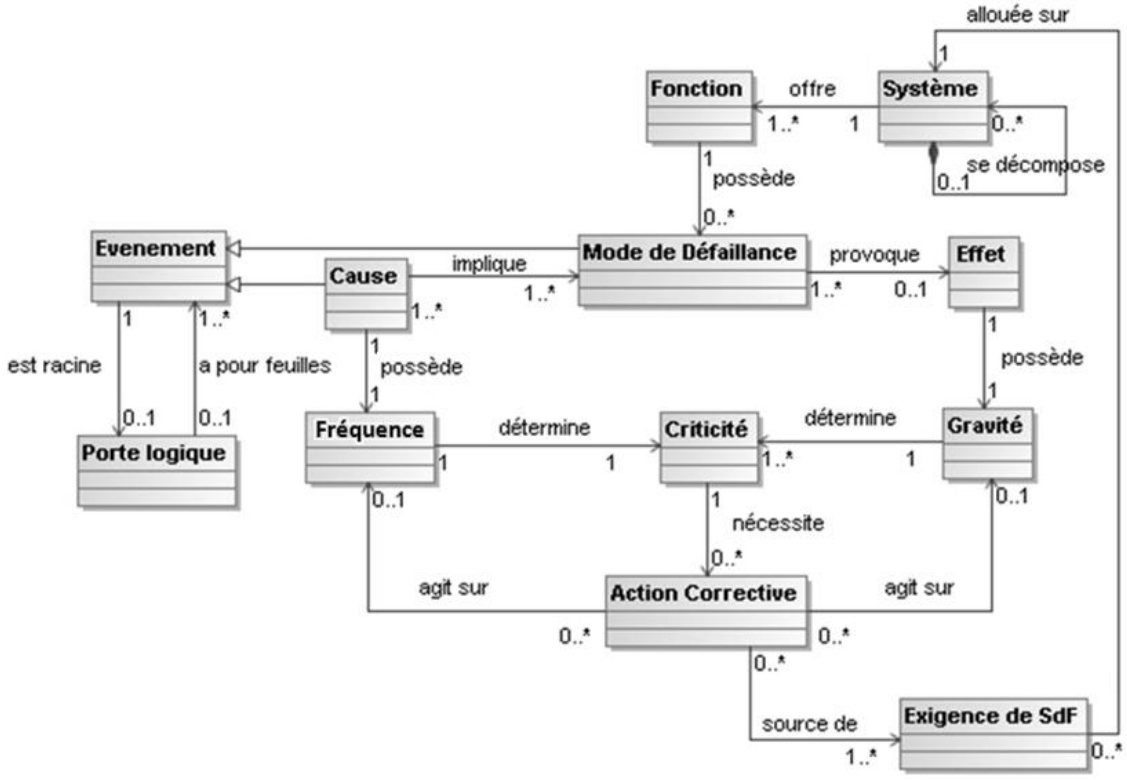


Figure III.15 : Formalisation UML des concepts de la méthode AMDEC+AdD

6. Cas d'étude

Afin d'illustrer la méthodologie de déclinaison d'exigences de sûreté de fonctionnement exposée dans ce chapitre, nous l'appliquons à un cas d'étude simple. Nous ne déclinerons qu'une seule exigence de sûreté système pour cet exemple. Un exemple plus complet sera présenté dans le chapitre 5.

6.1. Présentation de l'exemple

Le système étudié est celui d'un ascenseur, constitué d'une cabine mobile suspendue à un câble dans une cage. Un sous-système de contrôle du mouvement (incluant un treuil actionné par un moteur électrique) permet de mouvoir le câble rattaché à un contrepoids à l'autre extrémité. La cabine comporte une porte automatique qui actionne aussi l'ouverture de portes palières. Un ensemble de pupitres de commande, dont un est situé à l'intérieur de la cabine et les autres se trouvent près des portes palières, permet de gérer les demandes de mouvements de l'ascenseur. Un sous-système de ventilation garantie la qualité de l'air dans la cabine. Pour finir, quelques dispositifs de sécurité sont présents : un capteur de présence dans la zone de fermeture de la porte automatique, un interphone présent dans la cabine et des freins d'urgence sur le toit de la cabine.

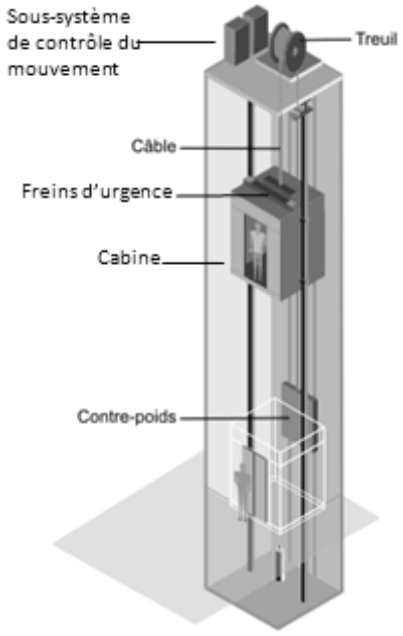


Figure III.16 : Système ascenseur

6.2. Application de la méthodologie

Nous allons maintenant appliquer la méthodologie de déclinaison d'exigences de sûreté de fonctionnement à ce système « ascenseur ».

6.2.1. Etape 1 : Analyse des défaillances du système

La première étape consiste à procéder à une AMDEC au niveau du système « ascenseur ». Dans la présentation de cet exemple, nous ne montrons que le mode de

défaillance « *descente non-maîtrisée* » de la fonction « *transporter les usagers* » de l'ascenseur. La Figure III.17 présente alors le tableau AMDEC représentatif de cette étude.

Système	Fonction	Mode de défaillance	Effet	Gravité	Cause	Actions Correctives *(Objectifs de SdF)*
Ascenseur	transporter les usagers	descente non-maîtrisée	Crash de la cabine, entrainant une mort possible de ces usagers	Catastrophique	**Chute de la cabine**	**Faire en sorte que la fréquence soit $<10^{-9}/h$**

Figure III.17 : AMDEC système « ascenseur » - fonction « transporter les usagers »

D'après ce tableau, nous pouvons identifier une exigence de sûreté au niveau système : « *La descente non-maîtrisée liée à la chute de la cabine doit avoir une fréquence* $< 10^{-9}/h$ ». Cette exigence est liée à la cause système : « *chute de la cabine* », qui va être étudiée dans la deuxième étape de la méthode.

6.2.2. Etape 2 : Analyses des causes des défaillances

Pour la deuxième étape, nous devons analyser toutes les causes système identifiées lors de la première étape. Dans la présentation de cet exemple, nous ne considérons que la cause système « *chute de la cabine* » visible dans le tableau AMDEC de la Figure III.17. Nous construisons alors un arbre de défaillance partant de cette cause système et faisant intervenir les modes de défaillances des sous-systèmes, donné en Figure III.18.

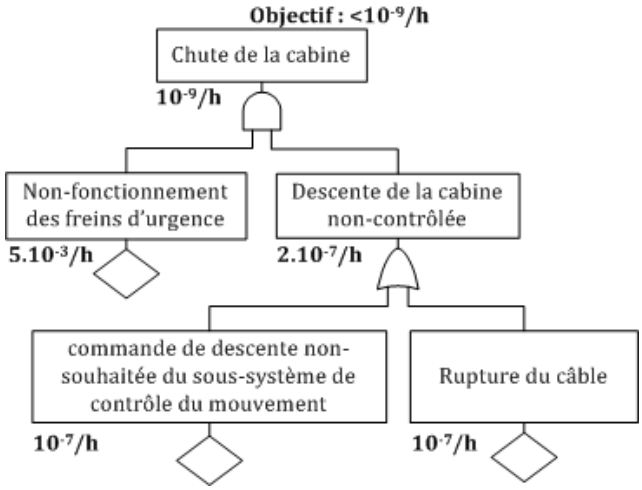


Figure III.18 : Arbre de défaillance de la cause système « chute de la cabine »

Ainsi, d'après la figure, la chute de la cabine fait intervenir la défaillance du sous-système de contrôle du mouvement (commande de descente non-souhaitée), la défaillance du câble (rupture) et la défaillance des freins d'urgence (non-fonctionnement).

6.2.3. Etape 3 : Analyses des défaillances des sous-systèmes

La troisième étape de la méthodologie de déclinaison des exigences de sûreté veut que nous procédions à des AMDEC pour tous les sous-systèmes qui interviennent dans les arbres de défaillances de la seconde étape.

Les sous-systèmes qui interviennent dans l'arbre de la Figure III.18 sont : le sous-système de contrôle du mouvement, le câble et les freins d'urgence. La Figure III.19 donne

les résultats de l'AMDEC pour le sous-système de contrôle du mouvement. C'est l'unique tableau AMDEC sous-système que nous présentons dans ce mémoire.

Système	Fonction	Mode de défaillance	Effet	Gravité	Cause	Actions Correctives *(Objectifs de SdF)*
sous-système de contrôle du mouvement	monter/ descendre la cabine	commande de descente non-souhaitée	Hors de contrôle, la cabine descend de manière non-maîtrisée, entrainant son crash possible	Hasardeux	**mode de commande bloqué dans l'état de descente**	**Faire en sorte que la fréquence soit $<10^{-7}$/h**

Figure III.19 : AMDEC « sous-système de contrôle du mouvement »

D'après ce tableau, nous pouvons identifier une exigence de sûreté au niveau du sous-système de contrôle du mouvement : « *La commande de descente non-souhaitée liée au mode de commande bloqué dans l'état de descente doit avoir une fréquence* $< 10^{-7}/h$ ». Cette exigence est liée au mode de défaillance du sous-système de contrôle du mouvement : « *commande de descente non-souhaitée* ».

Des AMDEC pour le câble et les freins d'urgence permettraient d'identifier respectivement les exigences :

- « *La rupture du câble doit avoir une fréquence* $< 10^{-7}/h$ », pour le câble.
- « *Le non-fonctionnement des freins d'urgence doit avoir une fréquence* $< 5.10^{-3}/h$ », pour les freins d'urgence.

6.2.4. Etape 4 : Synthèse

Dans la quatrième étape, la synthèse permet de montrer la déclinaison des exigences de sûreté au niveau système en exigences de sûreté sous-système. Nous donnons ici l'exemple de déclinaison qui correspond à l'arbre de défaillance de la Figure III.18 et qui est synthétisé en Figure III.20.

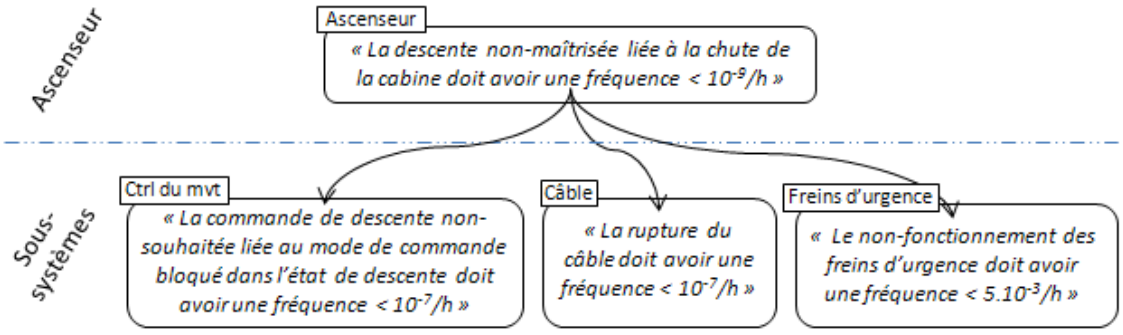


Figure III.20 : Synthèse de la déclinaison des exigences du système « ascenseur »

L'exigence de sûreté système « La descente non-maîtrisée liée à la chute de la cabine doit avoir une fréquence < 10^{-9}/h » est déclinée au niveau sous-systèmes en :

- Pour le sous-système de contrôle du mouvement :
  - « *La commande de descente non-souhaité lié au mode de commande bloqué dans l'état de descente doit avoir une fréquence* $< 10^{-7}/h$ »
- Pour le câble :
  - « *La rupture du câble doit avoir une fréquence* $< 10^{-7}/h$ »

- Pour les freins d'urgence :
  - « *Le non-fonctionnement des freins d'urgence doit avoir une fréquence* < $5.10^{-3}/h$ »

6.3. Utilisation de la formalisation UML

La formalisation UML, présentée au paragraphe 5.3, permet de poser les concepts manipulés dans la méthodologie de déclinaison, tout en présentant les relations entre ces concepts.

Nous donnons en Figure III.21, sur la base de l'exemple du système « ascenseur », un diagramme des objets qui instancient les classes du diagramme de la formalisation. Comme lors de l'étape 3 de l'application de la méthodologie de déclinaison d'exigences (présentée au paragraphe 6.2.3), nous nous sommes limités à un seul sous-système : celui du contrôle du mouvement.

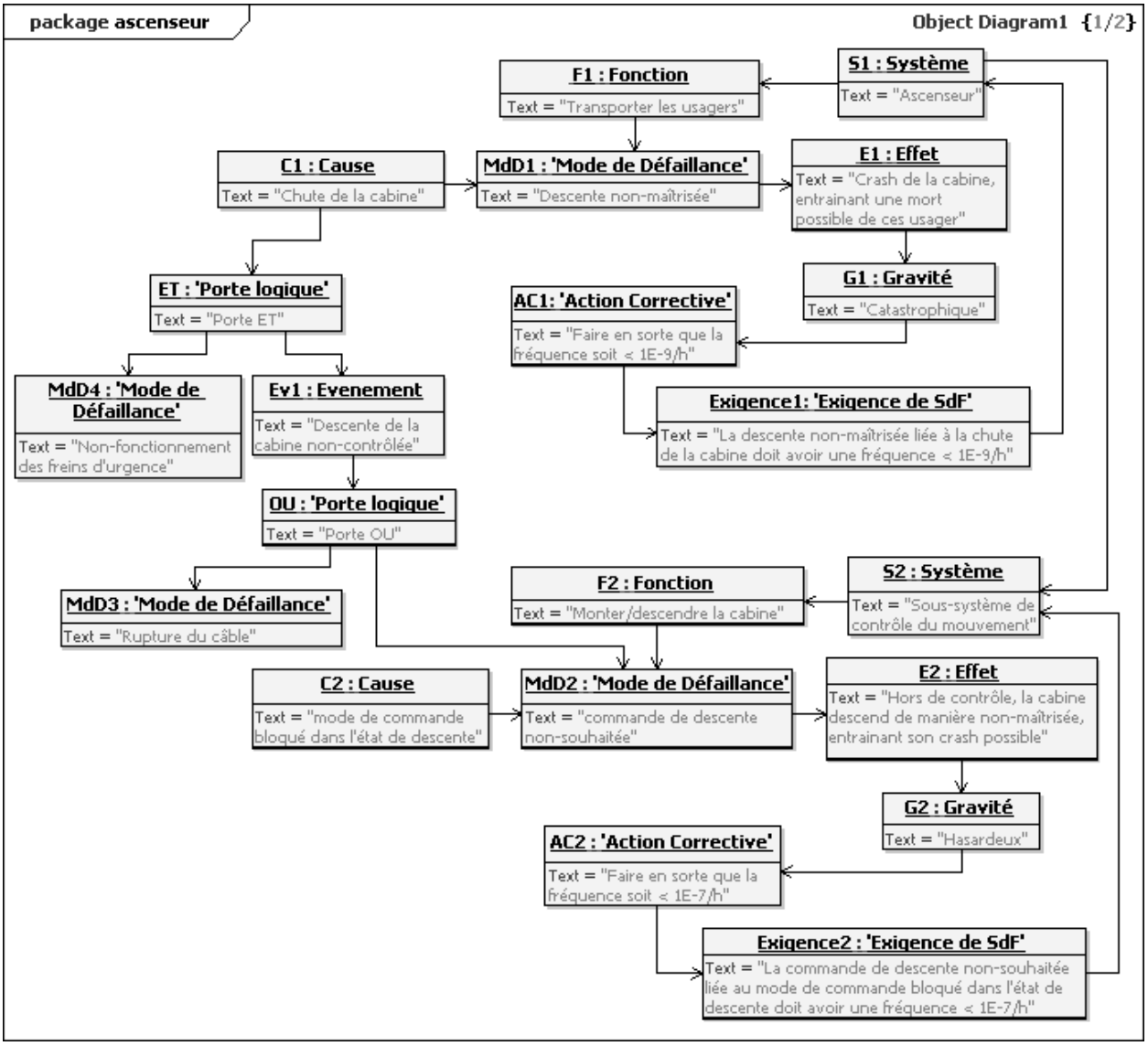


Figure III.21 : Diagramme d'objets de la formalisation UML pour l'exemple du système « ascenseur »

Partant du système **S1**, *l'ascenseur*, nous avons étudié la fonction **F1** : « *transporter les usagers* ». Nous nous sommes intéressés à un mode de défaillance de cette fonction : le mode **MdD1** de « *descente non-maîtrisée* ». L'identification de son effet **E1**, « *crash de la cabine entrainant une mort possible de ces usagers* », permet de définir une gravité **G1**, *catastrophique*. L'identification d'une cause **C1**, « *chute de la cabine* », n'est associée à aucune fréquence connue. Ainsi, une action corrective **AC1** qui agit sur la fréquence est définie : « *faire en sorte que la fréquence soit* $< 10^{-9}/h$ ». Cette action corrective est source de l'exigence de sûreté **Exigence1** : « *la descente non-maîtrisée liée à la chute de la cabine doit avoir une fréquence* $< 10^{-9}/h$ », qui doit être satisfaite par le système **S1**, *l'ascenseur*.

La cause **C1** est liée à une porte logique **ET** qui combine le mode de défaillance **MdD4**, « *non-fonctionnement des freins d'urgence* », avec l'événement **Ev1**, « *descente de la cabine non-contrôlée* ». Cet événement **Ev1** est lui-même lié à une autre porte logique **OU** qui combine les modes de défaillance **MdD3**, « *rupture du câble* », et **MdD2**, « *commande de descente non-souhaitée* ».

Ce dernier mode de défaillance **MdD2** est lié à une fonction **F2**, « *monter/descendre la cabine* », du système **S2** de « *contrôle du mouvement* », sous-système du système **S1**. L'analyse de ce mode de défaillance **MdD2** a permis d'identifier une cause **C2**, « *mode de commande bloqué dans l'état de descente* », et son effet **E2** : « *hors de contrôle, la cabine descend de manière non-maîtrisée, entrainant son crash possible* ». La cause **C2** n'est associée à aucune fréquence connue, quant à l'effet **E2**, sa gravité **G2**, *hasardeux*, est déterminée. Sur cette base, une action corrective **AC2** qui agit sur la fréquence est déterminée : « *faire en sorte que la fréquence soit* $< 10^{-7}/h$ ». Cette action corrective est source de l'exigence de sûreté **Exigence2** : « *la commande de descente non-souhaitée liée au mode de commande bloqué dans l'état de descente doit avoir une fréquence* $< 10^{-7}/h$ », qui doit être satisfaite par le système **S2**, *sous-système de contrôle du mouvement*.

7. Bilan de la méthodologie et complément

La méthodologie proposée, dans son état actuelle (Combinaison d'AMDEC et d'AdD), aide donc à la définition d'exigences de sûreté de fonctionnement niveau système, puis permet la décomposition et l'allocation de ces exigences sur les sous-systèmes. Mais cette décomposition et allocation d'exigences ne concernent en fait que les exigences qui agissent sur la fréquence, ceci à travers les liens causes-effets des arbres de défaillances.

Il s'agit de s'occuper aussi des exigences qui traitent de la gravité des effets. Le problème est le suivant : on considère qu'un mode de défaillance d'une fonction système apparaît, que peut-on faire pour en réduire les effets ? La réponse vient en ajoutant des mesures de protection ou des fonctions de sécurité qui vont empêcher la défaillance d'aboutir à un accident ou tout du moins réduire la gravité de l'accident.

Les nouvelles mesures/fonctions sont définies à travers des actions correctives traductibles en exigences de sûreté de fonctionnement, qui s'apparentent à des exigences fonctionnelles. Ainsi, les risques associés aux modes de défaillance sont traités par ces exigences de sûreté que l'on adjoint aux exigences qui définissent les mesures de protection déjà présentes. Toutes ces exigences sont au départ définies au niveau système, et vont être

traitées de la même manière que la « partie fonctionnelle » du système. Il n'y a donc pas d'autres méthodes spécifiques pour la déclinaison de ces exigences sur des sous-systèmes. Cette déclinaison va se faire classiquement avec les processus standard de l'ingénierie système.

En revanche, pour aider à l'identification des mesures protectives, une méthode possible consiste à utiliser des arbres d'événements (*event trees*) [Villemeur, 1988]. Elle complète l'AMDEC et montre de façon synthétique quelles mesures de protection permettent de contrer les effets d'un mode de défaillance. S'en suit alors la définition d'actions correctives, et donc d'exigences de sûreté de fonctionnement, qui agissent sur la gravité des modes de défaillances.

Dans une démarche de conception de systèmes complexes, SysML semble être un langage de modélisation de plus en plus utilisé. A ce propos, la méthodologie que nous venons de présenter peut s'appuyer sur les travaux de [David & al., 2010] qui s'intéressent à la fiabilité des systèmes complexes en utilisant SysML, en particulier pour la préparation de l'AMDEC. Ces travaux offrent la possibilité d'extraire des diagrammes SysML fonctionnels les informations nécessaires aux études de risques.

8. Conclusion

Dans le cas des systèmes complexes, un système est généralement défini comme la composition des sous-systèmes qui interagissent de manière non intuitive pour satisfaire un certain nombre de fonctions et répondre à un ensemble d'exigences. Dans ce chapitre, il a été question de définition d'exigences de sûreté et de la gestion de ces dernières à différents niveaux du système. Nous avons alors présenté une méthodologie de déclinaison qui permet d'obtenir une traçabilité et une déclinaison rigoureuse entre les exigences de sûreté définies au niveau « système » et celles définies au niveau « sous-systèmes », ceci en plus d'aider à cette définition d'exigences. Elle est basée sur des méthodes courantes de sûreté que sont l'AMDEC et l'analyse par arbre de défaillance. Cette méthodologie constitue un véritable atout vis-à-vis d'une conception sûre, en renforçant la prise en compte de la sûreté dans le processus de conception. Elle apporte une réponse aux problèmes de conception vus dans la problématique liés à la définition des exigences de sûreté, à leur gestion et à leur traçabilité. Le chapitre suivant continue dans ce sens, en fournissant un appui à la conception au travers d'un modèle d'information contenant des données d'ingénierie et des données de traçabilité.

Chapitre 4 : Vers une méthodologie ISBM

1. Introduction

Afin d'améliorer la gestion des exigences de sûreté et répondre à des faiblesses concernant la conception de systèmes complexes sûrs, nous proposons dans ce quatrième chapitre d'appuyer notre démarche par un modèle d'information. Ceci revient à orienter la démarche vers une méthodologie d'Ingénierie Système Basée sur les Modèles (ISBM).

Le contenu du chapitre commence alors par une section sur l'ISBM, la définition et l'intérêt d'un modèle d'information ainsi qu'un état de l'art du domaine. Suit la principale section du chapitre qui présente notre modèle à base du langage SysML. Le choix de ce langage est justifié et les différentes extensions sont présentées. Pour terminer, une section propose des compléments aux modèles sous forme d'analyses possibles à mettre en œuvre.

2. Modèle d'information et ISBM

Dans cette première section, nous présentons le concept d'Ingénierie Système Basée sur les Modèles (ISBM) et l'apport que peut avoir cette dernière dans la conception de systèmes complexes. Nous discutons ensuite de modèle d'information qui permet d'outiller cette ISBM.

Après avoir défini le modèle d'information, nous expliquons son intérêt et son apport vis-à-vis des activités de gestion des exigences, mais aussi pour communiquer entre les différentes équipes impliquées dans la conception. Suit un état de l'art relatif aux modèles d'information, exposant les travaux les plus significatifs du domaine. Puis, nous présentons un principe de bonne conception qui est fortement conseillé par les autorités de sûreté. Les recommandations de ces autorités sont prises en compte dans la définition de notre modèle d'information qui est présenté dans ce chapitre.

2.1. Ingénierie Système Basée sur les Modèles (ISBM)

L'Ingénierie Système Basée sur les Modèles (ISBM), en anglais *Model Based System Engineering*, propose de s'appuyer fortement sur les modèles, ceci pour toutes les étapes du cycle de développement (partant des spécifications et allant jusqu'aux tests d'intégration) [Estefan, 2008]. Le suivi d'une méthodologie d'ISBM permet de guider la conception avec l'utilisation de tels modèles pour telles étapes du cycle de conception.

L'INCOSE présente l'ISBM comme l'application formelle des techniques de modélisation pour réaliser les activités de spécification système, de conception, d'analyse, de vérification et de validation, en partant de la phase de conceptualisation, tout au long du développement, et sur l'ensemble des phases du cycle de vie [INCOSE, 2007]. Avec l'ISBM, il est question de remplacer les approches de conception essentiellement basée sur des documents par des approches complètes d'ingénierie système basées sur des modèles (voir Figure IV.1).

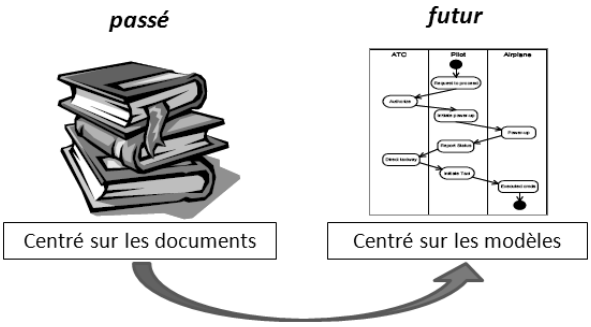


Figure IV.1 : Transition entre l'IS centrée sur les documents et l'IS centrée sur les modèles

En fait, les promesses du développement basé sur la modélisation sont nombreuses [Friedenthal & al., 2008] :

- Formaliser la pratique de l'ingénierie système à travers l'utilisation de modèle.
- Intégrer différents domaines de modélisation du système.
- Améliorer la communication :
  - faciliter la compréhension du système à tous les participants,
  - constituer des documents de référence pour les fournisseurs et sous-traitants.
- Améliorer la qualité des produits :
  - détecter les erreurs plus tôt,
  - assurer que toutes les exigences sont satisfaites,
  - apporter de la rigueur et de la précision.
- Maîtriser les coûts de développement et les plannings :
  - prendre en compte les changements durant la durée de vie du projet,
  - permettre une gestion fine et efficace des ressources,
  - accélérer les développements et la mise sur le marché des produits.
- Gérer et maîtriser la complexité.

En résumé, l'ISBM veut faire progresser l'Ingénierie Système dans de nombreux domaines tels que la communication, la précision des analyses, l'intégration des résultats, la réutilisation des connaissances produites mais surtout la maitrise de la complexité.

« *Nous avons réduit de 20% à 50% selon les projets, nos coûts de développement dès la première année d'utilisation de Rhapsody associé à notre outil de gestion des exigences Doors* » à déclarer un dirigeant d'une compagnie de télécommunication (Leader Telecom).

Comme le montre la Figure IV.2, l'ISBM implique que tous les participants à un projet de conception (Clients, sous-traitants, concepteurs…) travaillent sur des modèles, ceci pour toutes les étapes du cycle de développement le permettant (de « *Material Solution*

Analysis » à « *Operations & Support* », voir Figure IV.2). En fait, les modèles sont présents à tous les niveaux : aussi bien pour les conceptions détaillées des différents métiers (mécanique, électrique, logiciel…), que pour les analyses des différentes spécialités transverses (telle que la sûreté de fonctionnement), mais aussi et surtout au niveau du système. Pour ce dernier niveau, on parle de modèle système, qui est le point central de la conception du système. En effet, ce modèle se trouve à la jonction et fait le lien entre les exigences des parties prenantes, les analyses du système, la conception détaillée et les aspects de documentation (voir Figure IV.3 extrait de [David, 2009]).

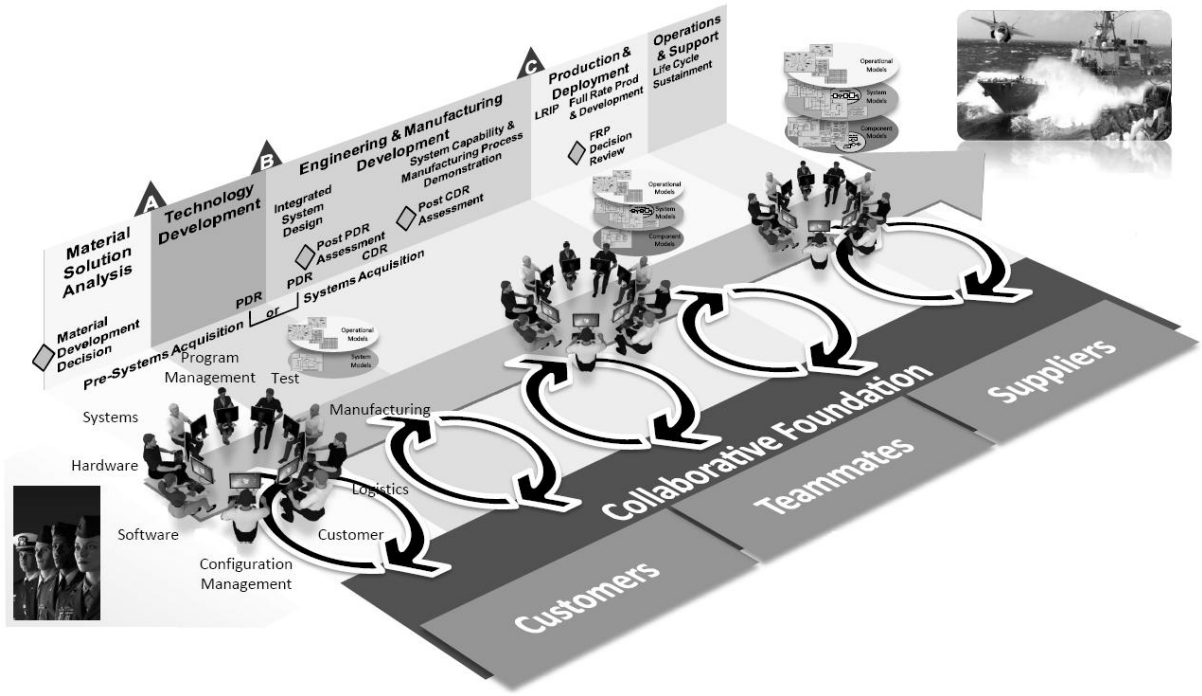


Figure IV.2 : ISBM tout au long du cycle de développement

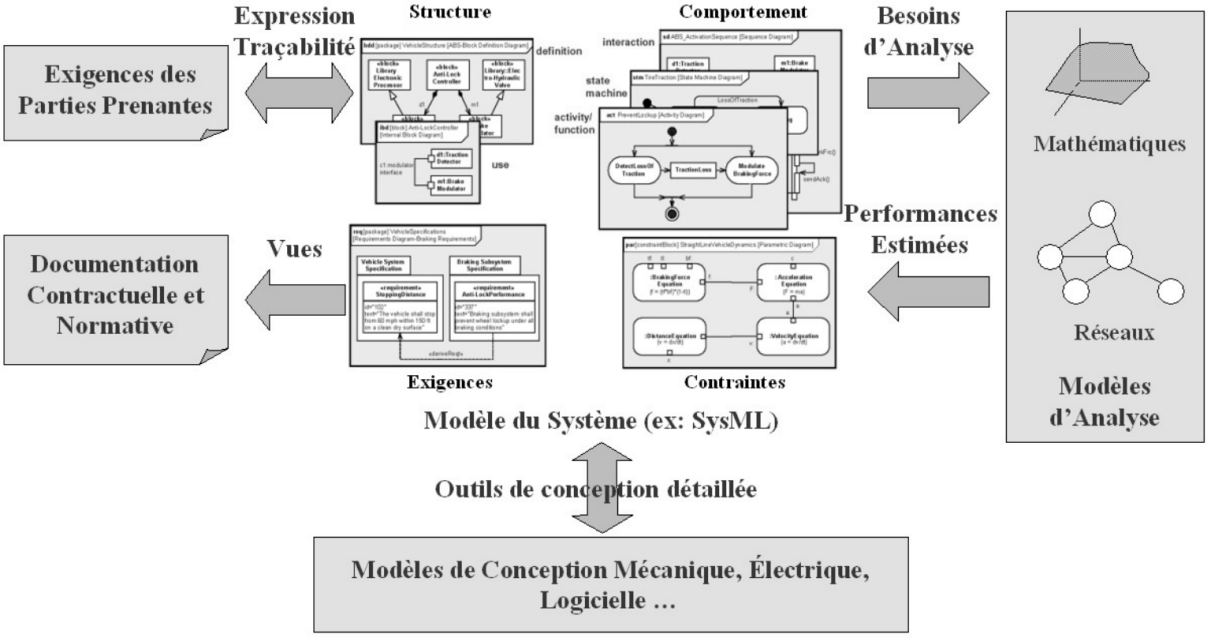


Figure IV.3 : Modèle du système au centre de l'I.S.

C'est donc pour se rapprocher d'une IS basée sur les modèles que nous proposons dans ce chapitre un modèle d'information qui rassemble l'ensemble des données d'ingénierie au sein d'un modèle commun. Ce modèle, qui permettra de remplacer de nombreux documents, se situe parfaitement au niveau du modèle du système de la Figure IV.3.

2.2. Définition d'un modèle d'information

Nous proposons la définition suivante pour la notion de modèle d'information, plus connu dans le passé sous le nom de modèle de données :

> ***Définition :*** *Un modèle d'information correspond à une base de données dans laquelle sont stockées toutes les connaissances produites par l'activité liée à ce modèle, ceci avec une sémantique précise et des relations logiques définies entre ces connaissances.*

Dans notre cas, nous utilisons ce concept de modèle d'information dans la conception système. Les connaissances produites peuvent alors correspondre à toutes sortes de données d'ingénierie. Pour n'en citer que quelques-unes, nous pouvons retrouver les exigences, les composants logiques, les composants physiques, les interfaces, les fonctions, etc. Des exemples de relations entre ces données, que nous détaillerons plus loin, peuvent être une relation d'allocation entre une fonction et un composant, ou encore une relation de satisfaction entre une exigence et une fonction.

De plus, chacune des données peut avoir une structure particulière constituée de différents champs prédéfinis de type déterminé. Par exemple, pour une exigence il est possible de la structurer en définissant des champs « partie prenante », « description » et « justification » de type chaîne de caractères, ou encore un champ « priorité » qui prendrait des valeurs entières dans l'intervalle [0, 5] par exemple.

2.3. Intérêt d'un modèle d'information

2.3.1. Appuyer la gestion des exigences

Dans le premier chapitre de cette thèse, nous avons vu l'importance des activités de gestion des exigences pour le bon déroulement d'un projet [Juristo & al., 2002], vu le nombre important de documents qui sont produits lors de la conception d'un système. Sans cette gestion des exigences, il serait extrêmement difficile de garantir la cohérence et la qualité nécessaire au succès du projet de conception d'un système complexe.

Appuyer la gestion des exigences par un modèle d'information apporte un soutien considérable au déroulement du projet. En effet, le modèle peut contenir toutes les données d'ingénierie système, dont notamment toutes sortes d'exigences. Il facilite l'expression des données en leur donnant une structure particulière et permet des liens logiques entre ces données (ou par les données elles-mêmes) pour justifier leur existence. Mais encore, un modèle d'information participe à la collecte des données, facilite leur partage, permet de détecter au plus tôt les incohérences, permet de gérer la validation des données et aide à la gestion de changement dans les données (par exemple un changement d'exigence). Il permet aussi, à travers des indicateurs dans la structure des données, d'évaluer la progression du projet, par exemple en suivant l'évolution des états de maturité des

exigences qui doivent être définies, déclinées, allouées, satisfaites, vérifiées ou encore justifiées.

En fait, l'avantage principal à utiliser un modèle d'information pour la gestion des exigences provient de la gestion rigoureuse de la traçabilité possible grâce à un tel modèle. Il permet de formaliser cette traçabilité qui peut d'ailleurs être réalisée de bout en bout, partant des besoins initiaux et allant jusqu'aux éléments les plus fins et détaillés de la conception, sans oublier les procédures ou cas de tests.

Concernant le modèle d'information proposé plus tard dans ce chapitre, celui-ci va s'appuyer en partie sur la vision des exigences dans la norme EIA-632 déjà présentée dans le chapitre 2, paragraphe 3.4.5 (voir Figure IV.4). Les exigences techniques, provenant des exigences des parties prenantes, conduisent aux représentations des solutions logique et physique, qui elles-mêmes sont sources de nouvelles exigences appelées exigences techniques dérivées. Puis la solution de conception répondant à l'ensemble des exigences techniques est figée par des exigences spécifiées.

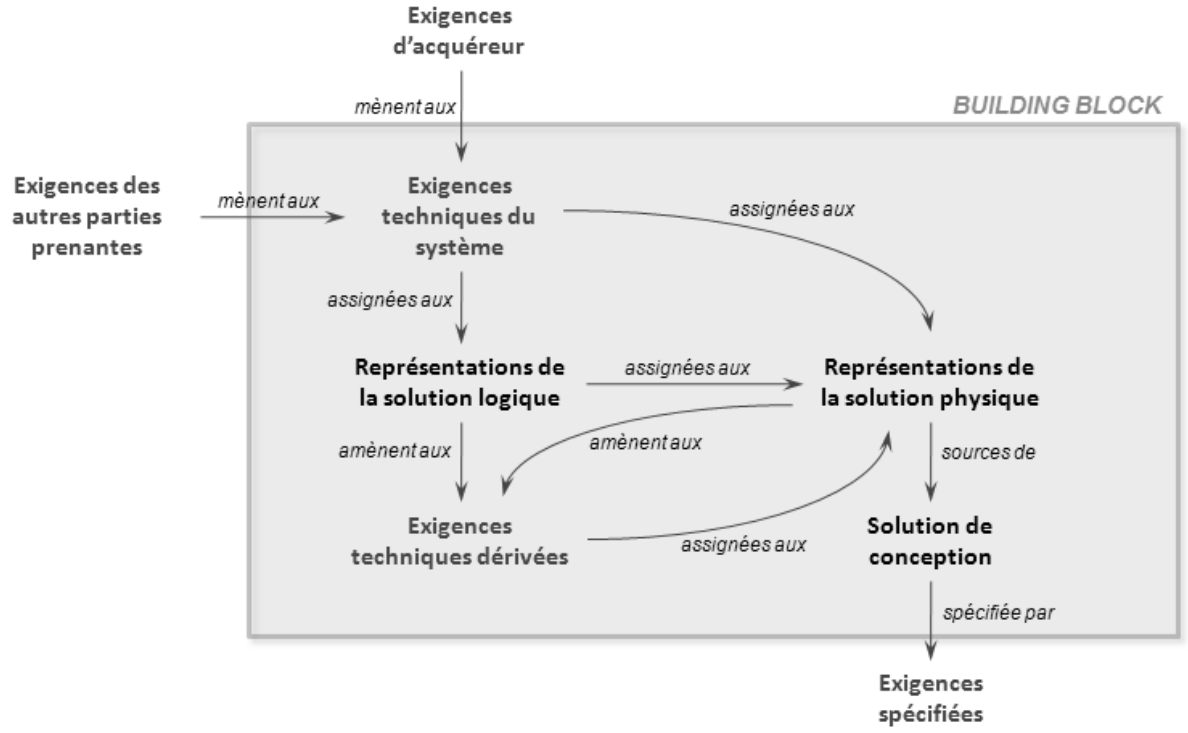


Figure IV.4 : Relation entre les exigences dans l'EIA-632

2.3.2. Partager les connaissances

Un modèle d'information est la base des connaissances « système » du projet de conception, permettant le partage des données entre tous les domaines de compétence (mécanique, hydraulique, thermique, électrique, …). Il permet entre autres le partage des connaissances entre ingénieurs de conception et ingénieurs de sûreté. En effet, la conception de système donne lieu à une accumulation de documentations qui doivent toutes être croisées et mises à jour pour maintenir la cohérence et respecter les spécifications du système. Un modèle d'information est un moyen de regrouper dans un modèle commun à tous les corps de métiers, les spécifications, les contraintes et les paramètres de l'ensemble

du système. Il est donc destiné à modéliser le niveau « système », en exhibant ses interactions avec l'environnement ainsi que les connexions entre les différents sous-systèmes.

En fait, les avantages apportés par un modèle d'information sont nombreux, dont notamment :

- Un partage des spécifications d'un système complexe entre tous les corps de métiers.
- L'identification des risques et la création d'une base d'analyse commune à tous les participants d'un projet.
- Facilite la gestion de projets complexes, l'évolutivité et la maintenabilité des systèmes complexes.
- Documente et capitalise le savoir de tous les corps de métiers dans un projet.

Le modèle d'information doit être vu comme un moyen de mise en commun des connaissances, incluant les trois volets : exigences, solution de conception et V&V (Vérification et Validation). Il est à considérer comme un véritable niveau d'interconnexion entre les différents métiers, comme le représente la Figure IV.5.

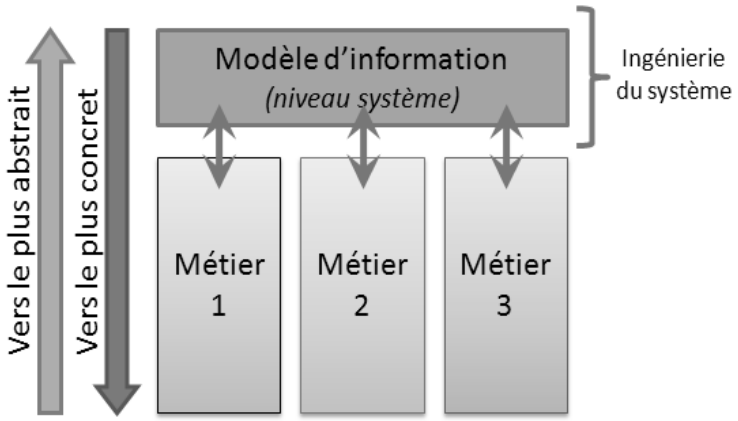


Figure IV.5 : Le modèle d'information : un niveau d'interconnexion

2.3.3. Supporter la conception

Les deux points précédents participent déjà au fait qu'un modèle d'information fournit un vrai support à la conception du système. Mais plus encore, le fait même de modéliser les connaissances à l'aide du modèle permet la transformation plus ou moins progressive des besoins en la définition du système à concevoir. Durant cette transformation, un passage graduel se fait depuis des concepts abstraits vers une définition rigoureuse du système. Il convient de distinguer deux domaines distincts dans la modélisation : l'espace du problème et l'espace des solutions possibles. Vraisemblablement, au début du projet l'espace de représentation du problème est plus important que celui de représentation des solutions possibles. Au fur et à mesure de l'avancement de la conception, l'ensemble de représentation des solutions possibles s'enrichit peu à peu, pour aboutir à la définition rigoureuse du système. En parallèle, l'ensemble de représentation du problème s'enrichit afin de mieux définir les attentes du système (besoins/exigences) jusqu'à se stabiliser. (cf. Figure IV.6)

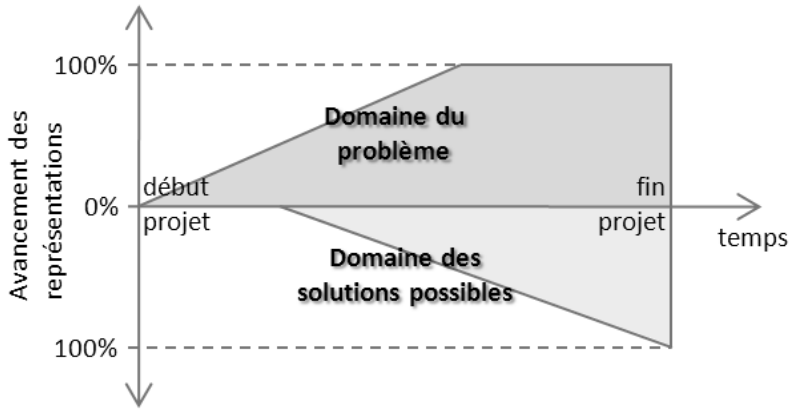


Figure IV.6 : Répartition des domaines au cours de la modélisation

D'ailleurs, la transition entre le domaine du problème et celui de la solution est un point très délicat de l'ingénierie système. Elle doit s'exprimer par des allocations d'exigences, de propriétés ou de contraintes sur des éléments de solution possible. Ces allocations engendrent des liens de traçabilité qui sont cruciaux pour les étapes de vérification et validation du système. En conséquence, l'utilisation d'un modèle d'information permet aussi de réduire le fossé entre le domaine du problème et celui de la solution, ceci à l'aide de liens de traçabilité.

2.3.4. Synthèse

Pour résumer, une manière de rendre efficace la gestion des exigences est de s'appuyer sur un modèle d'information qui serait la base et le cœur de connaissance du projet de conception, et sur lequel on va s'appuyer pour :

- guider la conception,
- définir tôt et bien les exigences,
- gérer rigoureusement la traçabilité des exigences,
- gérer les modifications/changements d'exigences,
- réaliser les analyses d'impacts,
- évaluer l'avancement du projet,
- comprendre le système, ceci sur la base d'un langage commun compréhensible par tous et d'un modèle partagé,
- passer progressivement du domaine du problème à celui de la solution,
- réutilisation des modèles existants pour supporter l'évolution technologique,

En fait, vis-à-vis de notre problématique de conception de systèmes sûrs, il faut garder à l'esprit que tout ce qui contribue à un système de développement meilleur (ou *système pour faire*) – il peut s'agir du projet, de son organisation et de son déroulement – constitue un facteur positif pour l'élaboration du bon système à développer (ou *système à faire*). Ici, l'utilisation d'un modèle d'information est clairement un facteur positif pour une bonne conception.

2.4. Travaux relatifs aux modèles d'information

2.4.1. MeMVaTEx

Le projet MeMVaTEx [Albinet & al., 2007] [Albinet & al., 2008], signifiant **Mé**thode de **M**odélisation pour la **Va**lidation et la **T**raçabilité des **Ex**igences, regroupe un ensemble de partenaires académiques (CEA, INRIA/UNSA, CNRS, UTC/HEUDIASYC) et de partenaires industriels (Siemens VDO, Embelec). Dans le domaine des systèmes temps réel embarqués, l'objectif du projet était de définir une méthodologie pour la modélisation des systèmes en prenant en compte l'expression des exigences et leur traçabilité tout au long du processus de conception.

Finalement plutôt destinée au monde logiciel, la méthodologie se base sur le langage EAST-ADL et sur deux profils UML 2.0 : le profil MARTE pour le temps réel et le profil SysML pour la modélisation des exigences. MeMVaTEx propose également quelques extensions à ces langages. Par exemple, la définition d'un nouveau stéréotype pour les exigences par un mécanisme d'héritage permet d'associer de nouveaux attributs pour les exigences (voir Figure IV.7). Dans ce projet, il était aussi question de modéliser les trois aspects : exigences, solution et V&V, de manière séparée mais au sein d'un même formalisme [Albinet & al., 2008].

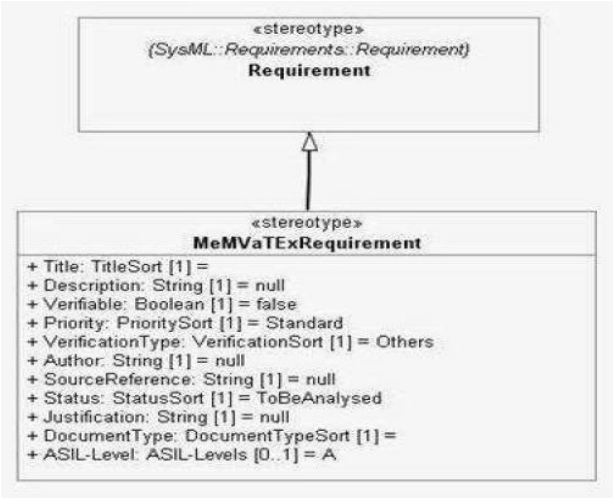


Figure IV.7 : Exigence MeMVaTEx

2.4.2. Modèle de données de l'AFIS

L'AFIS (Association Française d'Ingénierie Système) propose son propre modèle de données (ou modèle d'information) pour la conception système en ingénierie système. Il présente de nombreux aspects, notamment en se plaçant dans plusieurs vues différentes :

- La vue des concepts généraux,
- La vue affaire,
- La vue d'ingénierie : définition d'architecture (voir Figure IV.8),
- La vue d'ingénierie : exigences,
- La vue d'ingénierie : gestion de configuration,
- La vue d'ingénierie IVV.

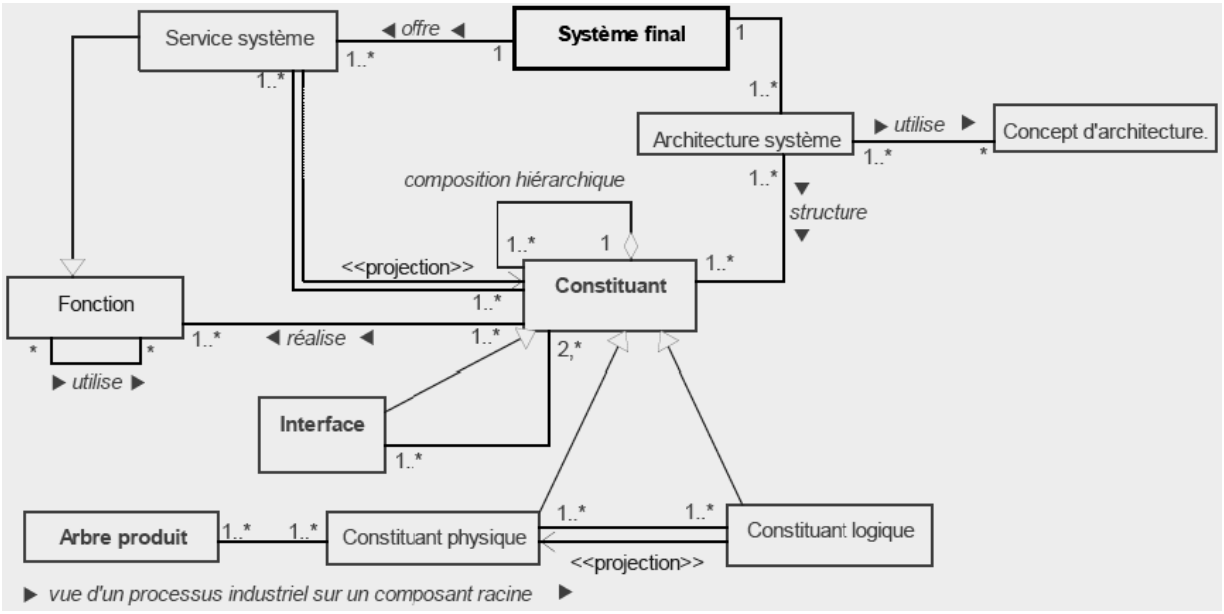


Figure IV.8 : Vue d'ingénierie de définition d'architecture système (AFIS)

Ce modèle est présenté dans un document de l'AFIS du groupe de travail MO (Méthodes et Outils) [AFIS, 2005b].

En complément, il existe des fiches qui détaillent de manière plus fine la description des données d'ingénierie. Par exemple, la fiche n°3 du groupe de travail IE (Ingénierie des Exigences), intitulée « un modèle de données pour la description d'exigences », décrit les propriétés caractéristiques des exigences. Elle fournit aussi un ensemble de liens entre les exigences et d'autres objets de l'ingénierie système (justification, source, condition, risque, compromis, produit, vérification) (voir Figure IV.9).

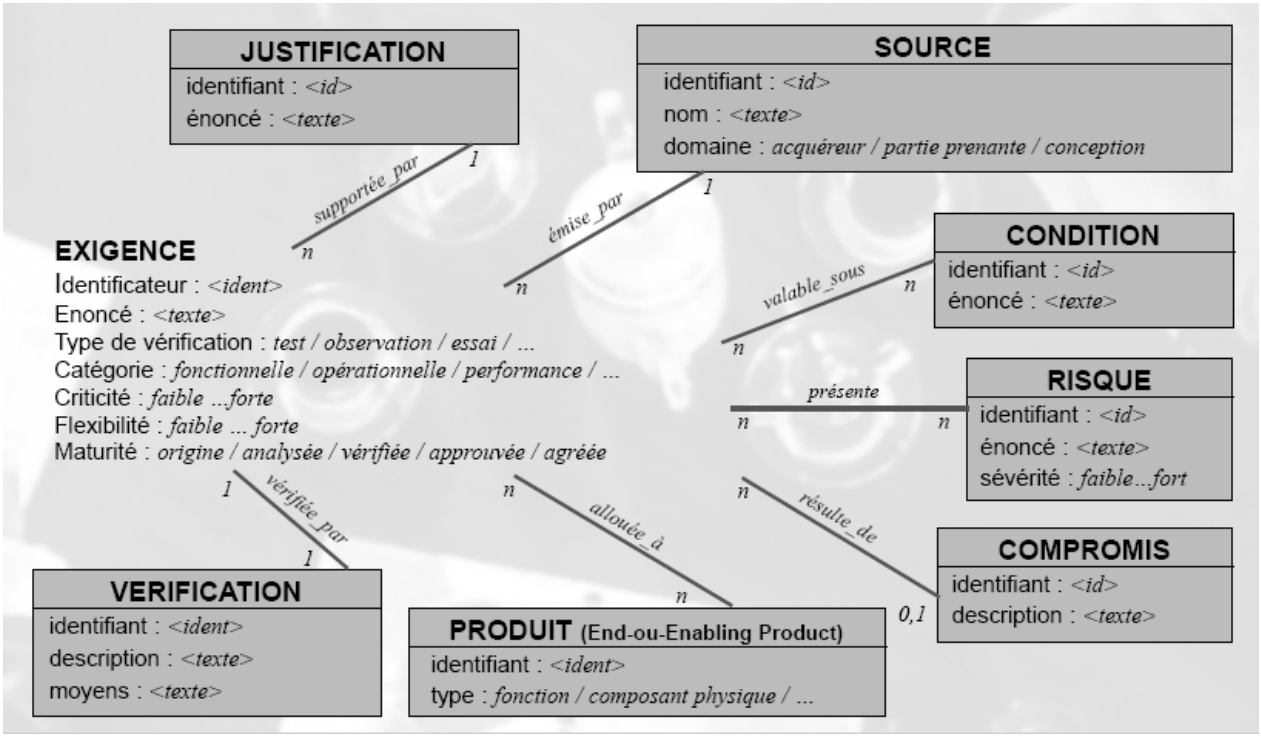


Figure IV.9 : Extrait fiche n°3 AFIS : un modèle de données pour la description d'exigences

2.4.3. Modèle de MAP Système

[MAP Système] est une société indépendante de conseil en Ingénierie Système, fondée par Alain Faisandier et Thérèse Renard. Ses consultants ont une longue expérience professionnelle qui leur permet d'avoir de solides connaissances et un recul important dans le développement, la méthodologie, le management, l'assurance qualité, la maîtrise d'ouvrage et la maîtrise d'œuvre des systèmes complexes civils et militaires des secteurs des télécommunications, du spatial, du nucléaire, de l'aéronautique, de l'automobile, du biomédical et de l'informatique.

Ils se sont également intéressés à la question de conception de systèmes sûrs de fonctionnement, mais en s'appuyant sur les processus d'ingénierie système de la norme ISO-15288 (voir Figure IV.10). Ils se sont appuyés en partie sur le guide de la sûreté de fonctionnement de Jean-Claude Laprie, avec les processus de développement à sûreté de fonctionnement explicite et la liste des points clés identifiés du guide.

Pour appuyer la démarche, ils ont aussi défini un modèle de données (voir Figure IV.11) [Faisandier, 2008], qui distingue 3 volets : le domaine du problème (besoins, exigences, …), le domaine fonctionnel et le domaine organique.

Ils ajoutent ensuite un quatrième volet qui est celui de la sûreté de fonctionnement, incluant notamment les notions de « risque » et de « danger ».

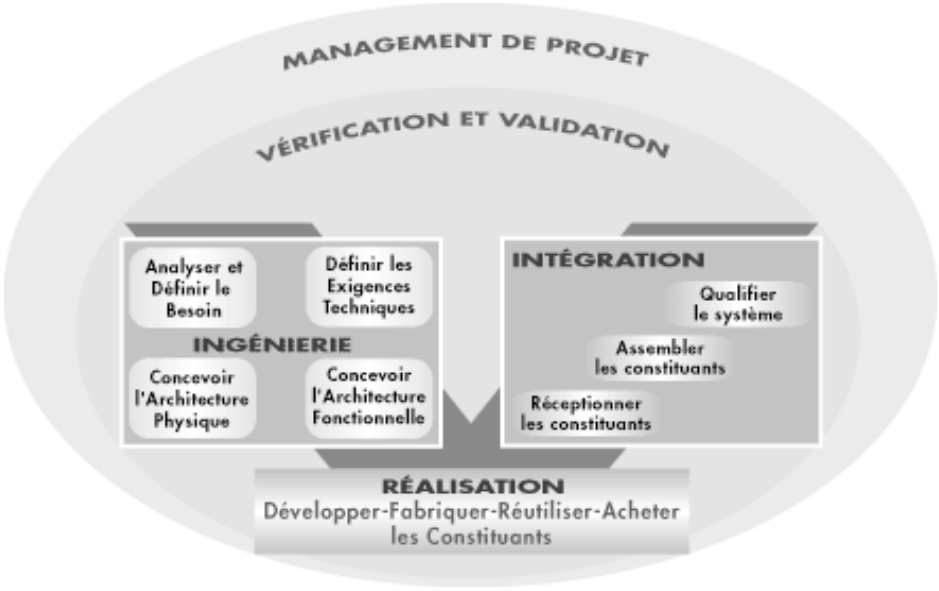


Figure IV.10 : Vue générale des processus IS - MAP Système

En plus de la vision processus et du modèle de données, MAP Système a développé un outil, qui s'appuie sur le modèle de données. Il intègre tous les aspects, des exigences aux constituants, en passant par les fonctions. L'outil propose aussi un certain nombre de « pattern » de sûreté de fonctionnement, qui sont disponibles dans une librairie et utilisables par le concepteur. Par exemple, il existe le pattern de la redondance double

active, ou encore celui de la redondance double passive. Bien entendu, le concepteur a la possibilité de construire son propre pattern.

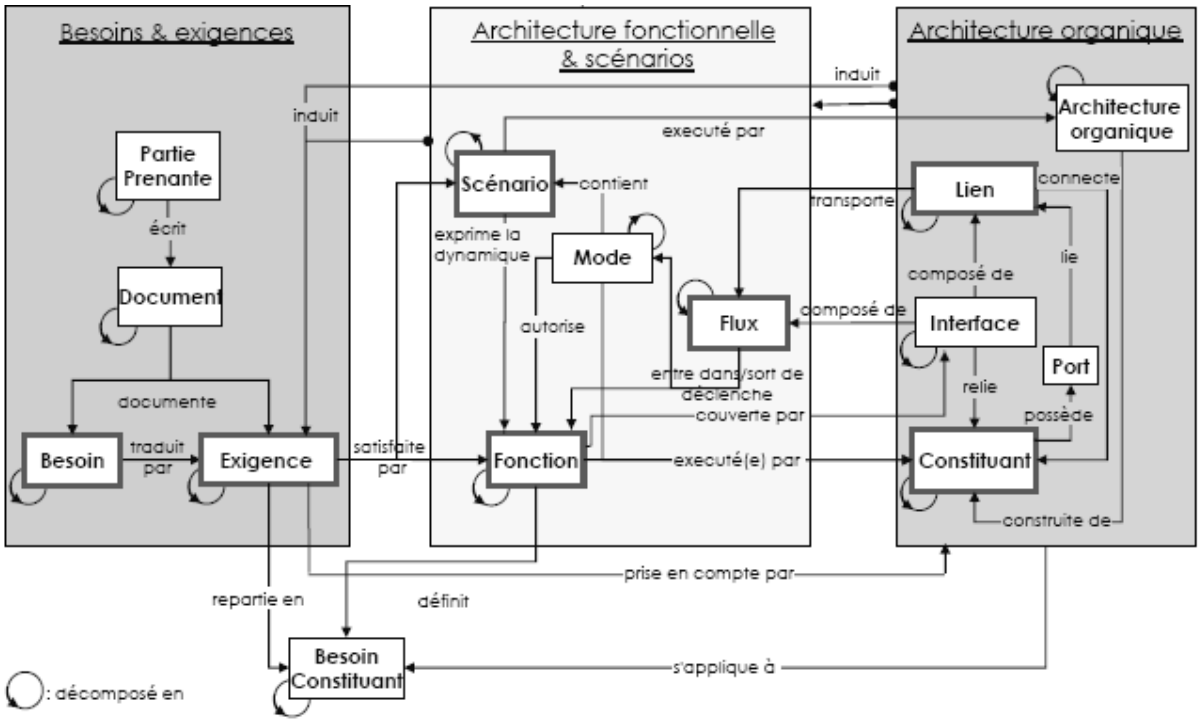


Figure IV.11 : Méta-modèle générique de l'Ingénierie Système de MAP Système

2.4.4. DoDAF et MoDAF

Le DoDAF (Department of Defense Architecture Framework) a été conçu par le département de la défense américaine. Il a pour objectif d'assister la conception, le développement et l'acquisition de systèmes ou systèmes de systèmes structurés hiérarchiquement. En fait, le DoDAF fournit un cadre d'architecture destiné au développement d'une architecture de systèmes ou d'une architecture d'entreprise.

Ce cadre d'architecture comporte deux couches :

- celle des données, pour les éléments entités, relations et attributs,
- celle de présentation, pour les vues et les produits.

Le référentiel est défini par le Core Architecture Data Model 2.0 (CADM, essentiellement un schéma de base de données commun) (voir Figure IV.12) et le système de référentiel de l'architecture du DoD (DARS).

Les vues permettent de visualiser l'architecture de données en organisant les données logiquement dans une perspective spécifique. Quant aux produits, ils sont un moyen de visualiser l'architecture de données par des graphiques, des tables ou des textes.

Somme toute, un des intérêts principaux de DoDAF est de rendre possible l'échange d'information entres les différents groupes collaboratifs d'utilisateurs pour atteindre leurs

objectifs partagés. Il s'agit donc d'apporter un vocabulaire commun aux différentes équipes afin de faciliter les échanges d'informations.

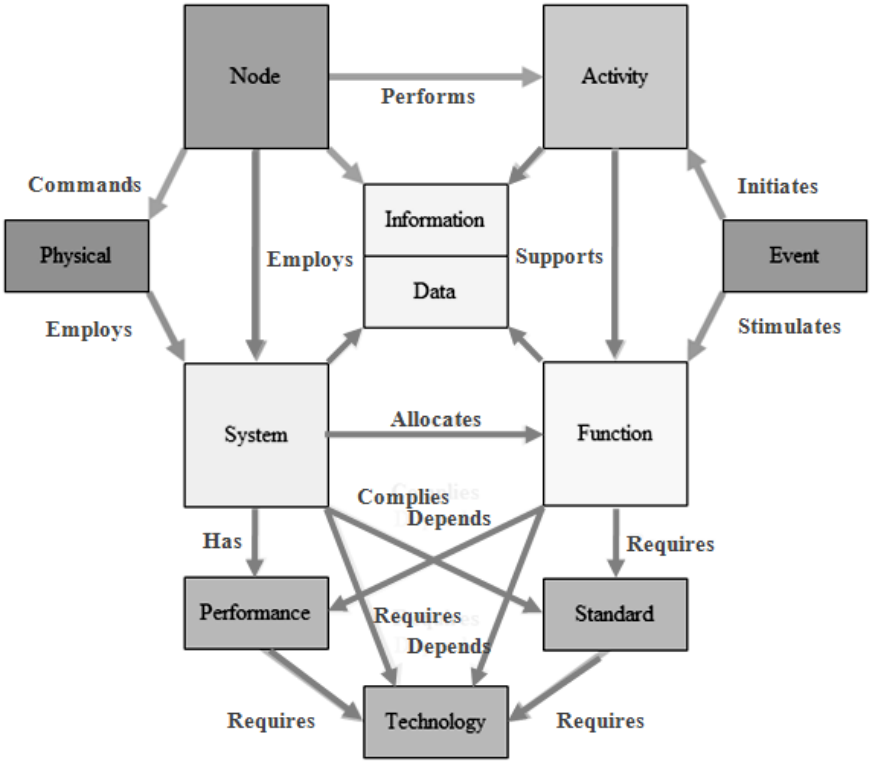


Figure IV.12 : Vue conceptuelle principale du Core Architecture Data Model 2.0

Le MoDAF (Ministry of Defense Architecture Framework) est, quant à lui, basé sur le DoDAF, mais réalisé par le ministère de la défense du Royaume-Uni. Il contient en fait d'autres vues, spécifique aux besoins du MoD.

2.4.5. Snow Card Volere

Volere est une marque déposée appartenant à Atlantic Systems Guild. Cette organisation est un centre de connaissance, de conseil et d'organismes de formation répartis entre Londres, Aachen et New York. Son objectif est de rester à l'avant-garde du développement et de l'ingénierie des systèmes. Volere est en fait le nom donné à un ensemble de ressources d'exigences, qui couvrent des cours, des modèles, des livres ou encore des processus. Pour notre travail, nous avons retenu de cet ensemble la proposition de représentation d'une exigence, appelée SnowCard Volere ou Atomic Requirement [Volere], visible en Figure IV.13.

L'exigence Volere est alors composée de :

- Un identifiant unique,
- Un type, qui peut être choisi à l'aide du Template Volere [Roberston, 2010] qui propose la classification suivante :
    - Exigences fonctionnelles
    - Exigences non-fonctionnelles :
        - d'apparence,

  - de convivialité et d'ergonomie,
  - de performance,
  - opérationnelles et d'environnement,
  - de maintenabilité et de support,
  - de sécurité, culturelles et politiques,
  - légales ou juridiques.
- La description de l'exigence,
- La justification de l'exigence,
- L'initiateur de l'exigence, qui correspond à une des parties prenantes,
- Un critère de satisfaction qui fournit une aide pour savoir si la solution remplit l'exigence,
- Le degré de satisfaction de la partie prenante dans le cas où l'exigence est implémentée (de 1 pour indifférent à 5 pour très satisfait),
- Le degré d'insatisfaction de la partie prenante dans le cas où l'exigence n'est pas implémentée (de 1 pour importe peu à 5 pour très mécontent),
- La priorité qui indique l'importance relative de l'exigence,
- Les conflits qui informent des exigences qui ne peuvent pas être implémentées si celle-ci l'est,
- Une liste des documents à l'appui qui illustrent et expliquent d'avantage l'exigence,
- Un historique.

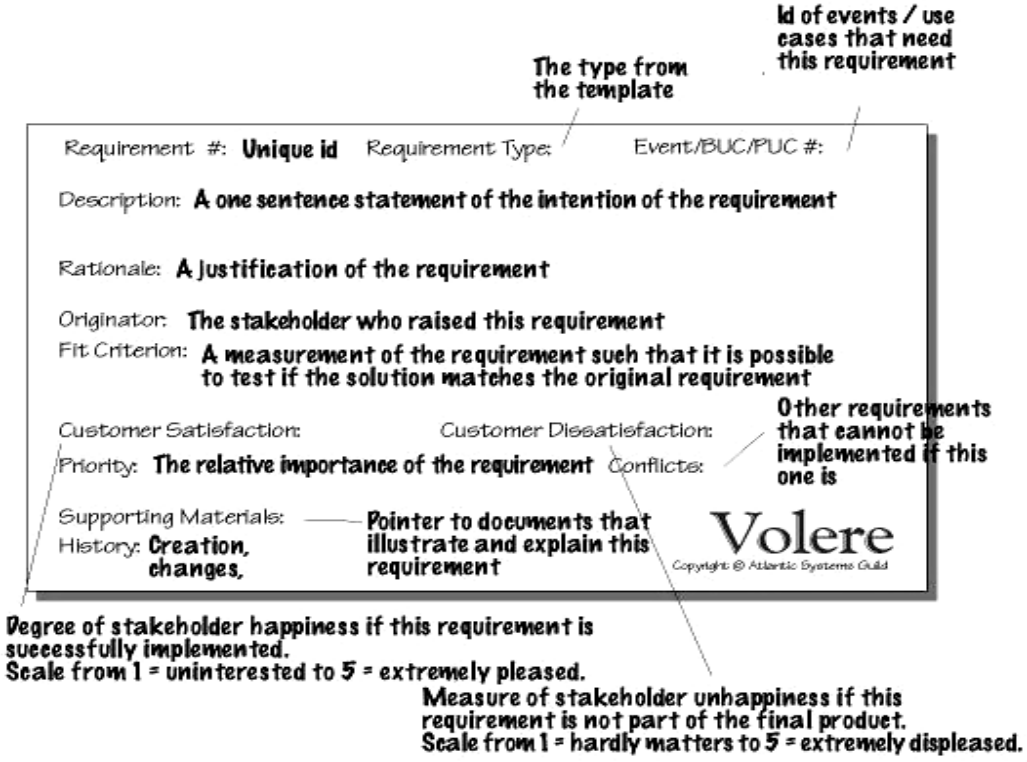


Figure IV.13 : SnowCard Volere

2.4.6. Travaux du CRAN

Dominique EVROT [Evrot, 2008] propose un modèle d'information succinct, repris en Figure IV.14, à partir d'une vision inspirée du modèle de données de l'AFIS [AFIS, 2005b].

Celui-ci s'inscrit dans le cadre du processus de définition du système, où il est nécessaire d'établir des relations entre les exigences, les fonctions et/ou les composants du système.

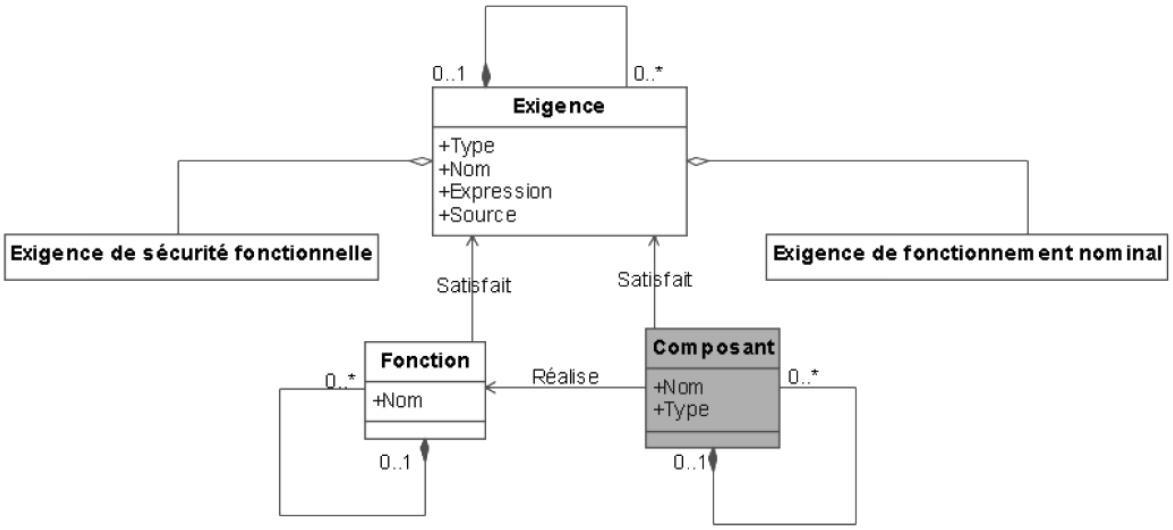


Figure IV.14 : Modèle du système à faire de D. EVROT

Ce modèle de traçabilité liant les exigences aux composants du système permet de réaliser des analyses d'impacts dans le cas d'évolutions d'exigences. On est capable d'évaluer les conséquences de la modification d'une exigence sur la sécurité du système à partir du réseau tissé entre exigences, fonctions et composants.

2.4.7. Travaux du LAAS-CNRS

Dans le cadre de sa méthodologie de conception système proposée dans ses travaux de thèse, Jean VERRIES [Verries & al., 2008] [Verries, 2010] a développé un modèle d'information très détaillé. Il fait bien la distinction entre la définition des données d'ingénierie, regroupées en différents ensembles, la définition des liens existants entre ces données, et la présence de points de vue qui forment les modèles de conception et d'analyse.

2.4.8. Synthèse

Nous constatons que de nombreux travaux se sont intéressés à la question d'un modèle d'information.

Plutôt défini pour le logiciel, MeMVaTEx n'intègre pas de concept spécifique à la sûreté de fonctionnement. Cependant, il propose de bonnes extensions pour les exigences et utilise le langage SysML. Le modèle de l'AFIS quant à lui s'intéresse beaucoup au concept de vue. Il intègre de nombreux concepts, dont une partie sera exploitée pour notre travail. Il en va de même avec le modèle proposé par MAP Système. Concernant les projets DoDAF et MoDAF, ils mettent en relation des concepts plus que des données d'ingénierie, mais l'idée de partage d'information entre les différentes équipes est bien présente. Dans les travaux de Volere, les champs définis par la snowcard et la classification en type des exigences pourront être exploités. D. Evrot propose un premier modèle d'information très succinct, défini en UML et celle de J. Verries propose un modèle détaillé concernant les exigences.

Notre proposition vient compléter ces travaux, en définissant un modèle centré sur les données d'ingénierie, redéfinissant des exigences, intégrant des aspects de sûreté de fonctionnement et utilisant le langage SysML.

2.5. Triptyque exigences-solutions-V&V, principe de bonne conception

Au sujet de notre modèle d'information, dont l'utilité a été justifiée ci-dessus, nous souhaitons qu'il respecte un principe de bonne conception provenant des autorités de sûreté (voir [Albinet & al., 2008]). En effet, celles-ci imposent une séparation des concepts manipulés lors de la conception du système. Les exigences, la solution de conception et les données concernant la V&V doivent être développées indépendamment. On fait d'ailleurs référence au triptyque exigences-solutions-V&V. Il faut donc pouvoir distinguer très clairement ces différents concepts. Cette distinction peut être retrouvée dans des normes, par exemple dans la norme ISO-26262 [ISO-26262, 2008], adaptation prochaine de l'CEI-61508 [CEI-61508, 2010] au domaine de l'automobile.

Afin d'illustrer ce discours, la Figure IV.15 présente ce qui ne doit pas être fait (à gauche) et ce qu'il convient de faire (à droite).

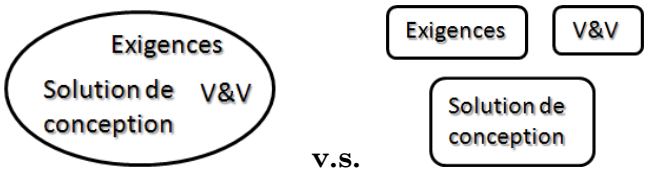


Figure IV.15 : Séparation des concepts manipulés

Or, le premier cas (celui de gauche) présente tout de même un avantage non-présent dans le second. Il permet d'attacher directement les exigences aux éléments de solutions, par exemple en restant dans le même formalisme.

L'avantage du second cas, quant à lui, est clair. Il permet une gestion efficace de chaque concept, appuyée par des outils spécifiques. Ceci étant, le problème de cette approche est que les exigences sont gérées de façon externe au processus de modélisation de la solution. Il ne facilite donc pas la compréhension et la revue des exigences par rapport aux équipes de développement de la solution.

Nous verrons que notre approche va, tout en respectant la séparation des concepts prévue par les autorités de sûreté, s'inspirer des deux cas précédents pour intégrer les deux avantages.

3. Modèle proposé

Dans cette section, nous présentons tout d'abord le langage SysML (*System Modeling Language*). Ensuite, nous détaillons quelques points particuliers de ce langage, notamment concernant les relations liées aux exigences. Puis nous justifions le choix de SysML pour le modèle d'information que nous proposons [Guillerm & al., 2010c] et nous exposons nos extensions du langage. La présentation de notre modèle d'information clôt la section.

3.1. Le langage choisi : SysML

SysML [Bock, 2006] est à l'ingénierie des systèmes complexes et/ou hétérogènes ce qu'UML est à l'informatique. Il doit permettre à des acteurs de corps de métiers différents de collaborer autour d'un modèle commun pour définir un système. La conception de système donne souvent lieu à une accumulation de documentations qui doivent toutes être croisées et mises à jour pour maintenir la cohérence et respecter les spécifications du système. SysML est un moyen de regrouper dans un modèle commun à tous les corps de métiers, les spécifications, les contraintes, et les paramètres de l'ensemble du système. SysML n'aborde plus la conception avec la notion de classes mais avec la notion de blocs qui deviendront des parties mécaniques, électroniques, informatiques ou autres.

SysML (System Modeling Language) est donc :

- Un langage de modélisation visuel pour l'Ingénierie Système.
- Compatible avec UML pour l'Ingénierie Logicielle. Ce qui est parfaitement logique, puisque SysML est un DSL (Domain Specific Language) d'UML pour le « Système », c'est-à-dire une adaptation d'UML.
- Un support à la spécification, l'analyse, le design, la vérification et la validation d'un large panel de systèmes et de « systèmes de systèmes ». Ces systèmes pouvant inclure du matériel, du logiciel, de l'information, des processus, du personnel et des services.

3.1.1. Historique

Voici un bref historique des méthodes, techniques ou langages de conception qui ont permis d'arriver jusqu'à SysML, en passant évidemment par le langage UML (cf. Figure IV.16) :

- Dans les années 1970, les premières méthodes d'analyse pour la conception système apparaissent.
- Les années 1980 annonce l'approche systémique, avec la modélisation des données et modélisation des traitements : Merise [Redouin, 1989], Axial, IE...
- Entre 1990 et 1995, il y a une véritable émergence d'une multitude de méthodes orientée-objets : Booch [Booch, 1993], Classe-Relation, Fusion, HOOD, OMT [Rumbaugh & al., 1990], OOA, OOD, OOM, OOSE [Jacobson, 1992] ...
- L'année 1995 marquera les premiers consensus, tels qu'OMT (James Rumbaugh), OOD (Grady Booch) ou OOSE (Ivar Jacobson).
- Les années 1995 à 1997 voit les premiers pas pour l'unification et la normalisation des méthodes avec la naissance d'UML.

Ainsi, UML (Unified Modeling Language) [Booch & al., 2005] est né de la fusion des 3 méthodes qui ont le plus influencé la modélisation orientée objets au milieu des années 90 :

- OMT (Object Modeling Technique). Méthode d'analyse et de conception orientée objets. Vues statiques, dynamiques et fonctionnelles d'un système.
- OOD (Object Oriented Design). Vues logiques et physiques du système.

- OOSE (Object Oriented Software Engineering). Couvre tout le cycle de développement. Une des plus anciennes méthodes objet focalisée sur le modèle statique.

Fin 1997, UML devient un standard OMG (Object Management Group). L'OMG est un organisme à but non lucratif, créé en 1989 à l'initiative de grandes sociétés (HP, Sun, Unisys, American Airlines, Philips...). Son rôle est de promouvoir des standards qui garantissent l'interopérabilité entre applications orientées objet, développées sur des réseaux hétérogènes.

Puis UML s'améliore et s'enrichit progressivement pour atteindre la version UML 2.0 en 2004. Finalement, suite aux premières prémisses d'un langage spécifique à l'ingénierie système dès 2001, c'est en 2005 qu'une première version de SysML voit le jour. Aujourd'hui, la version actuelle du langage est SysML 1.2.

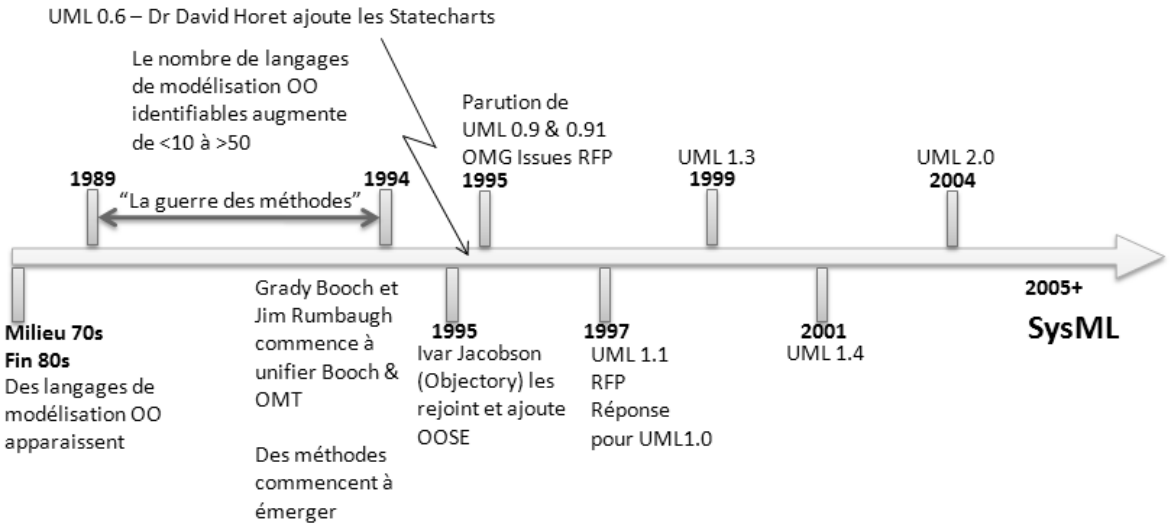


Figure IV.16 : Historique de SysML

3.1.2. Présentation

SysML est une adaptation d'UML 2 pour l'ingénierie système. Plus précisément, SysML réutilise une partie d'UML, appelée UML4SysML (interpréter « *UML for SysML* »), à laquelle est ajoutée une extension (voir Figure IV.17).

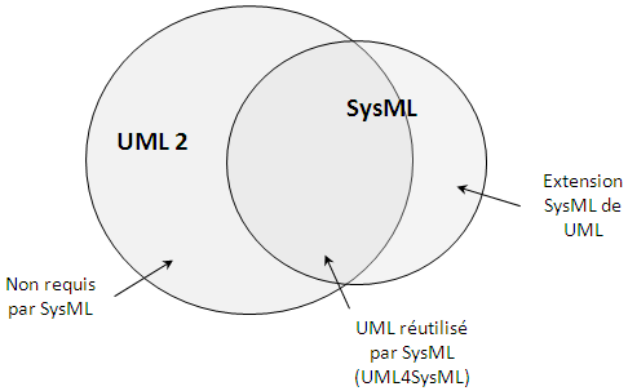


Figure IV.17 : Construction de SysML à partir d'UML 2

Pour définir SysML [Bock, 2005], 6 diagrammes ont été retirés des 13 diagrammes d'UML. Il s'agit des diagrammes de composants, d'objets, de déploiement, de vue d'ensemble des interactions et de temps, qui sont trop orientés pour le monde logiciel ou qui n'apporteraient rien de significatif à l'ingénierie système. Par ailleurs, 2 diagrammes ont été renommés : le diagramme de classes est remplacé par le diagramme de définition de blocks et le diagramme de structure composite devient le diagramme de blocks internes. La notion de classe, sur laquelle on pouvait s'appuyer pour créer des instances (c'est-à-dire des objets), n'a pas été reprise pour le domaine de l'ingénierie système, où celle-ci est substituée par la notion de block. Ainsi, un système ou un composant sont définis par des blocks en SysML. Le diagramme d'activité a subi quelques modifications et le diagramme paramétrique a été ajouté pour définir les relations entre les grandeurs physiques du système. Le dernier point, certainement l'innovation majeure de SysML, est le diagramme des exigences qui voit le jour. Il permet d'introduire les exigences systèmes dans le modèle, avec un certain nombre de liens de traçabilité rendu possible entre des exigences elles-mêmes ou entre des exigences et d'autres éléments de modélisation. La Figure IV.18 résume ce paragraphe traitant de l'ensemble des diagrammes offerts par SysML.

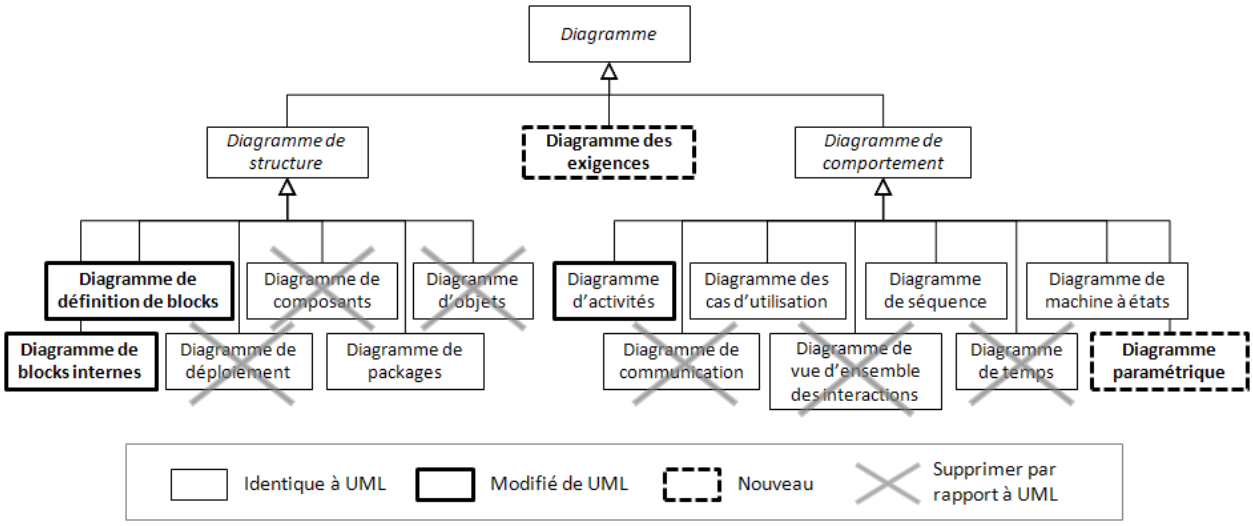


Figure IV.18 : Les diagrammes de SysML

Finalement, à travers un environnement unique intégrant les exigences et permettant la modélisation de la conception, SysML supporte différentes vues qui sont :

- les exigences, avec le diagramme des exigences,
- le comportement, avec les diagrammes des cas d'utilisation, de séquence, d'activités et de machine à états,
- la structure, avec les diagrammes de définition de blocks et de blocks internes,
- les contraintes, avec le diagramme paramétrique.

3.1.3. Stéréotype et block

Cette section présente deux notions importantes de SysML, essentielles à la présentation de nos travaux.

Dans SysML, on voit apparaitre la notion de stéréotype. Les stéréotypes sont définis par des extensions de « méta-classes » qui permettent de définir de nouveaux éléments de modélisation. En effet, la définition d'un nouveau stéréotype fait partie de la méta-modélisation : c'est-à-dire de la modélisation de moyens de modélisation. Un stéréotype peut être associé à de nouvelles propriétés (attributs) ou de nouvelles contraintes. Il permet en fait une extension sémantique du langage. Nous verrons plus loin des exemples de stéréotypes définis dans SysML ou défini par nous-mêmes.

La notion de classe disparait dans une modélisation SysML. Cet élément de base de la modélisation UML est remplacé par la notion de « block ». Le block est ainsi l'élément qui va permettre de décrire les architectures des systèmes (composants, sous-systèmes, système). (En fait, dans le méta-modèle SysML (profil de UML2.0), « block » est défini comme un stéréotype.)

3.1.4. Traçabilité avec SysML

Nous avons vu à maintes reprises l'importance de la traçabilité lors de la conception d'un système complexe. Dans ce paragraphe, il s'agit justement de présenter les aspects de traçabilité offerts par le langage SysML.

3.1.4.1. Relations entre exigences

SysML permet 3 types de liens entre les exigences. Ceux-ci sont la composition, la dérivation et la copie. La Figure IV.19 représente ces différents liens sur des exemples génériques.

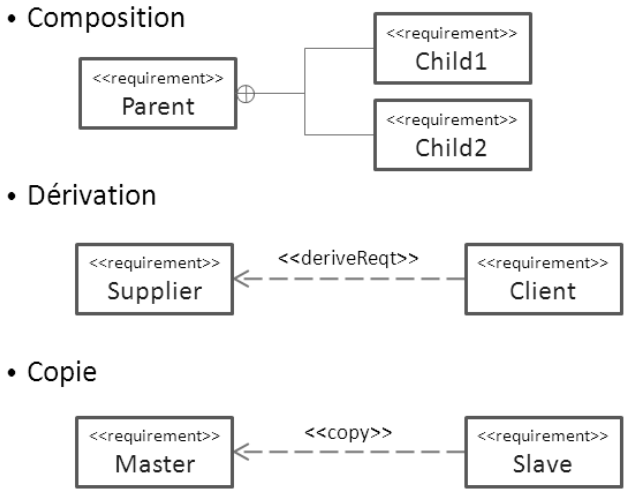


Figure IV.19 : Liens de traçabilité entre exigences et exigences

Le lien de composition indique qu'une exigence est décomposée en plusieurs autres exigences. Avec ce type de lien, si les exigences filles sont satisfaites, alors l'exigence mère l'est aussi.

Le lien de dérivation (stéréotypé par *deriveReqt*) exprime une dépendance entre deux exigences.

Le lien de copie (stéréotypé par *copy*) représente la simple copie d'une exigence. Cela permet de créer une exigence en « lecture seule ».

3.1.4.2. Relations entre exigences et autres éléments de modèle

SysML définit également des connexions entre les exigences et d'autres éléments de modèle. D'après le concept de traçabilité utilisé pour la conception de systèmes, il est effectivement tout à fait d'à-propos de proposer ce type de liens qui contribue à rendre l'ensemble du modèle SysML cohérent. Ces liens de connexion entre les exigences et d'autres éléments de modèle sont au nombre de 3 (voir Figure IV.20).

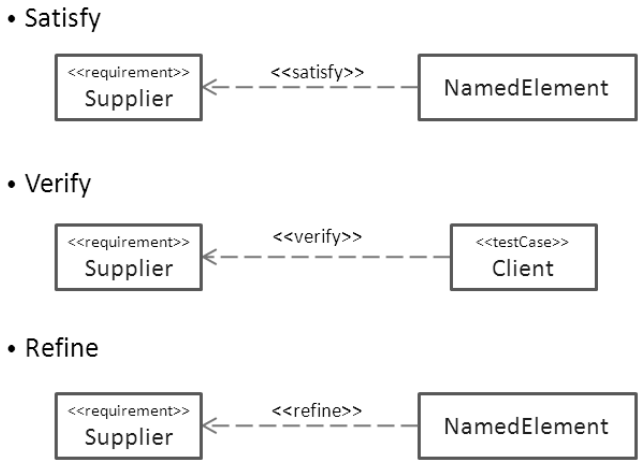


Figure IV.20 : Liens de traçabilité entre exigence et autres éléments de modèle

Le lien de satisfaction (stéréotypé par *satisfy*) signifie que l'élément du modèle satisfait l'exigence.

Le lien de vérification (stéréotypé par *verify*) est utilisé pour lier un *TestCase* à une exigence. Ce lien signifie que l'exigence peut être vérifiée par le *TestCase* en question, qui explique la méthode ou la procédure permettant la vérification (traduction française : cas de tests).

Le lien de raffinement (stéréotypé par *refine*) permet d'expliquer plus en détails une exigence, plus que le simple champ de description textuelle. Par exemple, un diagramme de machine à états peut expliciter plus clairement et synthétiquement l'idée du paragraphe textuel de l'exigence.

3.1.4.3. Relation d'allocation Activités/Blocks

Le dernier type de lien que nous présentons ici est celui d'allocation (stéréotypé par *allocate*). Son usage le plus courant correspond à l'allocation d'une fonction sur un composant, autrement dit : l'allocation d'une activité sur un block. Cette allocation peut se représenter sous de nombreuses formes dans les diagrammes SysML. Elle peut par exemple apparaître à travers les attributs *allocatedFrom* et *allocatedTo* des blocks, ou plus simplement en utilisant un lien d'association tagué du stéréotype *allocate*. Mais la façon le plus couramment utilisée reste l'utilisation des partitions dans les diagrammes d'activités (aussi appelées : *swimlanes*), comme le montre la Figure IV.21. Cela permet d'allouer les

fonctions représentatives des activités présentes dans la partition (*ActionName* dans la Figure IV.21) au block associé à cette partition (*ElementName* dans la Figure IV.21).

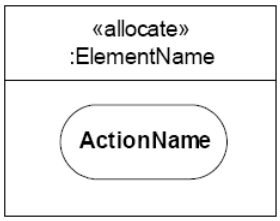


Figure IV.21 : Liens de traçabilité entre exigence et autres éléments de modèle

3.1.5. Justification du choix de SysML

Après avoir présenté très rapidement le langage SysML (la spécification faisant plus de 250 pages), nous justifions le choix de ce langage pour notre problématique.

Parmi les avantages que peut offrir SysML pour la conception système, nous pouvons citer :

- Tout d'abord, SysML est un excellent candidat au besoin d'un langage commun bien défini et compréhensible par tous les acteurs impliqués dans la conception, de disciplines ou de spécialités différentes.
- SysML permet la modélisation d'une large gamme de systèmes, incluant tant du matériel, que du logiciel, de l'information, des processus, du personnel ou des équipements.
- Il permet d'avoir un modèle complet du système, couvrant l'ensemble des étapes, partant des exigences jusqu'à la solution physique (aspects structurel et comportemental), ce qui est rendu possible par la variété des diagrammes de SysML.
- SysML permet une expression soignée des exigences, incluant toutes les informations nécessaires, structurée avec différents attributs.
- Il offre la possibilité d'avoir une traçabilité rigoureuse. Entre autres, celle-ci facilite les analyses d'impacts, par exemple dans le cadre d'un changement d'exigence.
- Avec SysML, il est possible de décrire les allocations des exigences sur les différents éléments du modèle.
- En plus de cela, il est possible d'intégrer et d'associer des cas de tests directement à la modélisation.
- Le dernier point important est le fait que SysML est un langage extensible. Nous proposons d'ailleurs une extension du langage, présenté dans la section ci-après.

3.2. Extension proposée de SysML

Les différentes relations présentées dans les paragraphes précédents (*composition*, *derivation*, *copy*, *satisfy*, *verify*, *refine* et *allocate*) sont essentielles pour la définition de notre modèle d'information et également pour satisfaire la gestion des exigences de l'EIA-632. Cependant, nous souhaitons ajouter au langage quelques autres éléments.

Nous définissons de nouvelles exigences, qui contiennent plus d'information que les exigences de base de SysML. Ces informations sont utiles pour une meilleure conception, en

fournissant plus de détails et en rendant possible un meilleur suivi du travail. A cela nous ajoutons aussi un nouveau lien, ainsi que de nouveau « type » d'exigences (plus précisément, il s'agit de stéréotype). Nous définissons entre autres un « sous-type » d'exigence technique du système qui sont des exigences de sûreté de fonctionnement. De plus, nous ajoutons un nouveau type de block à SysML : il s'agit d'un block « *risk* » qui représente les risques qui menacent le système.

3.2.1. Exigences enrichies

Le concept d'exigence est fondamental en conception système. C'est également un des intérêts et une des nouveautés qu'apporte SysML, de considérer les exigences et les lier à des éléments du modèle. L'objectif est de rapprocher et lier les besoins des parties prenantes à la solution de conception, pour guider le développement, aider la vérification et la validation, ou encore faciliter le suivi des changements d'exigences ou les évolutions.

SysML définit une exigence comme le montre l'élément *requirement* à gauche dans la Figure IV.22, avec 3 attributs :

- *Heading* : le nom de l'exigence, qui correspond au nom donné à l'élément *requirement.*
- *Text* : sa description sous forme textuelle.
- *Id* : son identifiant.

Nous enrichissons cette définition en préconisant de nouveaux attributs, dont une partie est inspirée du profil RPM (*Requirement Profil for MeMVaTEx*), des recommandations de l'AFIS ou de la *Snow Card Volere*. Ces attributs, visibles à droite dans la Figure IV.22, sont les suivants :

- *Catégorie* : indique si l'exigence est fonctionnelle, de fiabilité, de performance, de sécurité, de coût, de disponibilité, de maintenabilité, de sûreté, … (Une liste possible est donnée par Volere [Roberston, 2010]).
- *Justification* : justifie l'existence de l'exigence.
- *Source* : enregistre la provenance de l'exigence (acquéreur, partie prenante, conception, lois, règlementation, base technologique, standards, spécifications, capacités et tendances des produits concurrents, interfaces avec d'autres systèmes ou plateformes, analyse de risques, …).
- *Cible* : indique la cible concernée par l'exigence qui peut être le système ou des produits contributeurs (produits de développement, de production, de test, de déploiement/installation, d'entrainement, d'opération, de support/maintenance ou de retrait de service).
- *Maturité (origine, définie, validée, vérifiée)* : renseigne sur la vie de l'exigence qui passe à travers différents stades.
- *Criticité (faible…forte)* : caractérise l'importance de l'exigence en termes de risque.
- *Flexibilité (faible…forte)* : caractérise une certaine tolérance envers la satisfaction de l'exigence.
- *Priorité (faible…forte)* : ordonne les exigences pour leur prise en compte dans la conception.

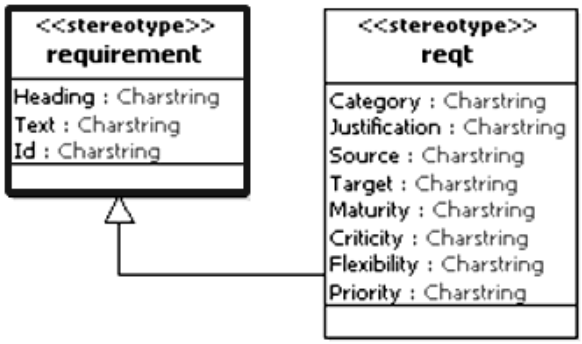


Figure IV.22 : Exigences SysML enrichies

Concernant le champ maturité qui renseigne sur la vie de l'exigence, la Figure IV.23 illustre les différents stades en les détaillant. En fait, l'exigence doit atteindre l'état « *validée* » pour pouvoir être alloué à un élément de modèle. Quant à l'état « *vérifiée* », il est obtenu après avoir suivi le *TestCase* associé à l'exigence, qui procède à une vérification sur le modèle ou le système réalisé.

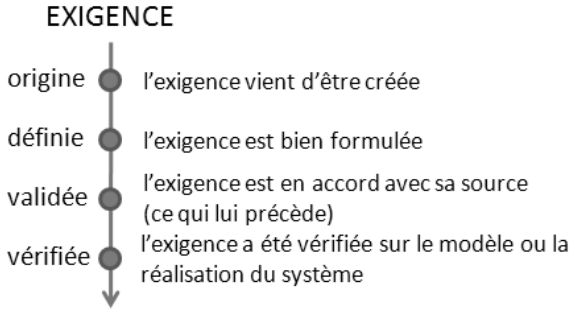


Figure IV.23 : Champ maturité d'une exigence

3.2.2. Nouveaux stéréotypes d'exigences

Les stéréotypes permettent d'étendre la sémantique des éléments de modélisation : il s'agit du mécanisme d'extension du méta-modèle d'UML (base de SysML). Ils permettent de définir de nouveaux types de block, en plus du noyau prédéfini par UML (ou SysML par extension). Comme exemples de stéréotypes, nous avons déjà vu <<*requirement*>> ou <<*testcase*>>.

Pour catégoriser les exigences en respectant la classification de la norme EIA-632, nous avons défini de nouveaux stéréotypes pour les exigences, comme le montre la Figure IV.24.

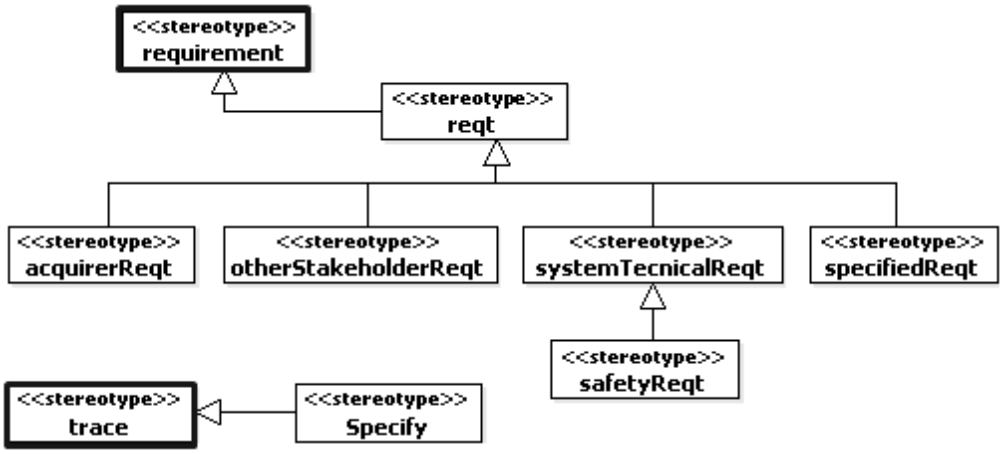


Figure IV.24 : Nouveaux stéréotypes pour les exigences

Ainsi, pour mieux distinguer les exigences entre-elles, il sera possible de créer des exigences de différents stéréotypes :

- <<acquirerReqt>> pour les exigences d'acquéreurs,
- <<otherStakeholderReqt>> pour les exigences des autres parties prenantes,
- <<systemTechnicalReqt>> pour les exigences techniques du système,
- <<safetyReqt>> pour les exigences techniques du système de sûreté de fonctionnement,
- <<specifiedReqt>> pour les exigences spécifiées.

Sur la même Figure IV.24, apparait aussi la définition d'un nouveau lien de traçabilité : le lien « *specify* ». Il permet de lier les exigences spécifiées et la solution de conception. Pour rappel, les exigences spécifiées, telles que définies dans l'EIA-632, servent à décrire et détailler la solution de conception finale.

Note : sur la Figure IV.24, les éléments de SysML sont les rectangles à contour épais. Le reste correspond à l'extension du langage que nous proposons.

3.2.3. Stéréotype « risk »

Les analyses de risques sont, comme nous l'avons déjà vu, à la base de la conception de système sûr de fonctionnement. C'est souvent le point de départ de réflexions concernant la fiabilité du système ou la sécurité du système et de ses utilisateurs.

Les risques, suite à leurs analyses par le processus de traitement des risques, peuvent être à la source d'exigences de sûreté de fonctionnement, voire de modification d'exigences déjà existantes. Les nouvelles exigences ainsi définies vont agir sur le risque en essayant soit de réduire l'impact de l'occurrence du risque (sa gravité), soit de réduire la fréquence d'occurrence du risque, ou les deux. Le but final étant de rendre le risque acceptable selon sa criticité, qui est une combinaison de sa gravité et de sa fréquence (voir Figure IV.25).

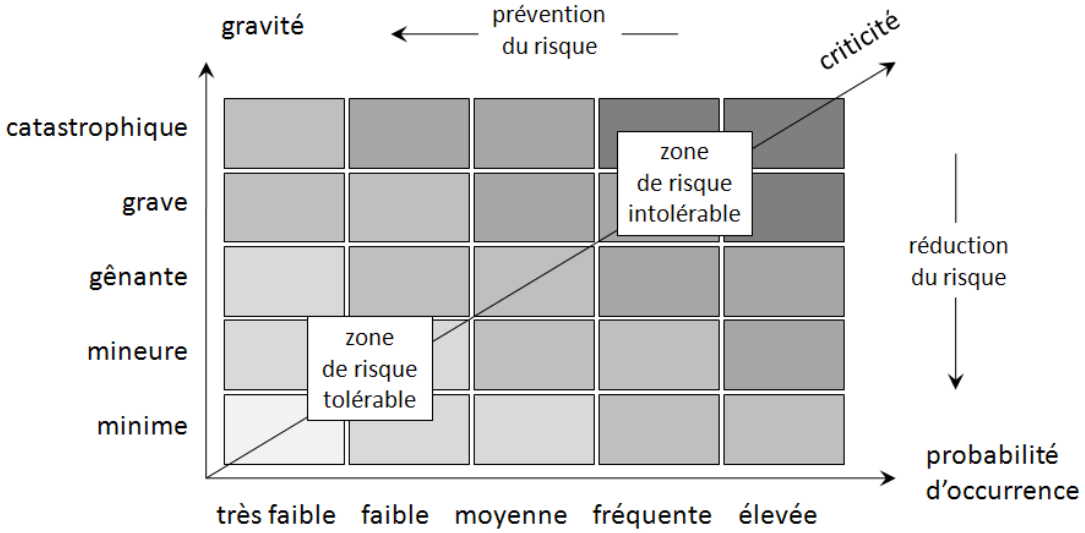


Figure IV.25 : Matrice de criticité

De par l'existence de cette relation entre les risques et les exigences de sûreté et de par l'importance que cela représente au niveau de la traçabilité et de la compréhension du modèle, nous introduisons un nouveau stéréotype de bloc « *risk* », visible en Figure IV.26. Les attributs que nous définissons pour ce bloc « *risk* » sont les suivants :

- *ID* : identifiant,
- *Statment* : description du risque,
- *Assumptions* : éventuelles hypothèses faites pour la définition de ce risque,
- *Severity* : sévérité (ou gravité) du risque (*sans gravité*, *mineur*, *majeur*, *hasardeux*, *catastrophique*).

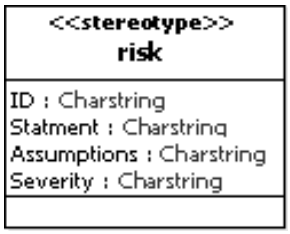


Figure IV.26 : Le bloc « risk »

Ce bloc apporte en fait une justification à la présence des exigences, tout en aidant à la gestion des risques. Les analyses d'impacts tirent aussi un bénéfice de cette traçabilité entre risques et exigences. Il est possible, par exemple, d'avoir directement accès aux risques remis en cause par la modification d'une exigence.

La Figure IV.27 présente la définition du bloc « *risk* » comme extension de SysML. Sur cette même figure apparaît la définition d'un nouveau lien de traçabilité « *treat* », qui permet de lier les exigences de sûreté de fonctionnement aux risques.

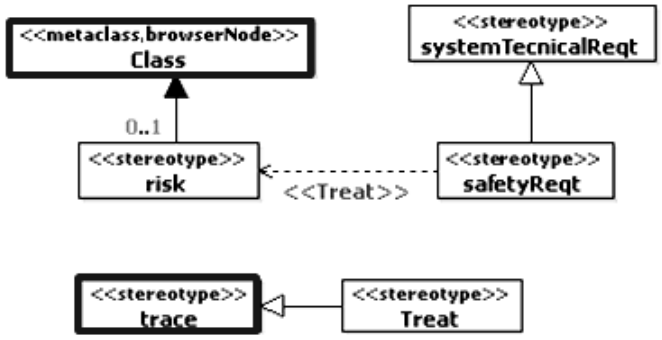


Figure IV.27 : Nouveau bloc « risk » lié aux exigences de sûreté de fonctionnement

Note : dans la Figure IV.27, de même que pour la Figure IV.24, les éléments de SysML sont les rectangles à contour épais. Le reste correspond à l'extension du langage que nous proposons.

3.3. Présentation du modèle d'information

Après avoir étendu le langage SysML, nous pouvons définir notre propre modèle d'information [Guillerm & al., 2010d], donné en Figure IV.28.

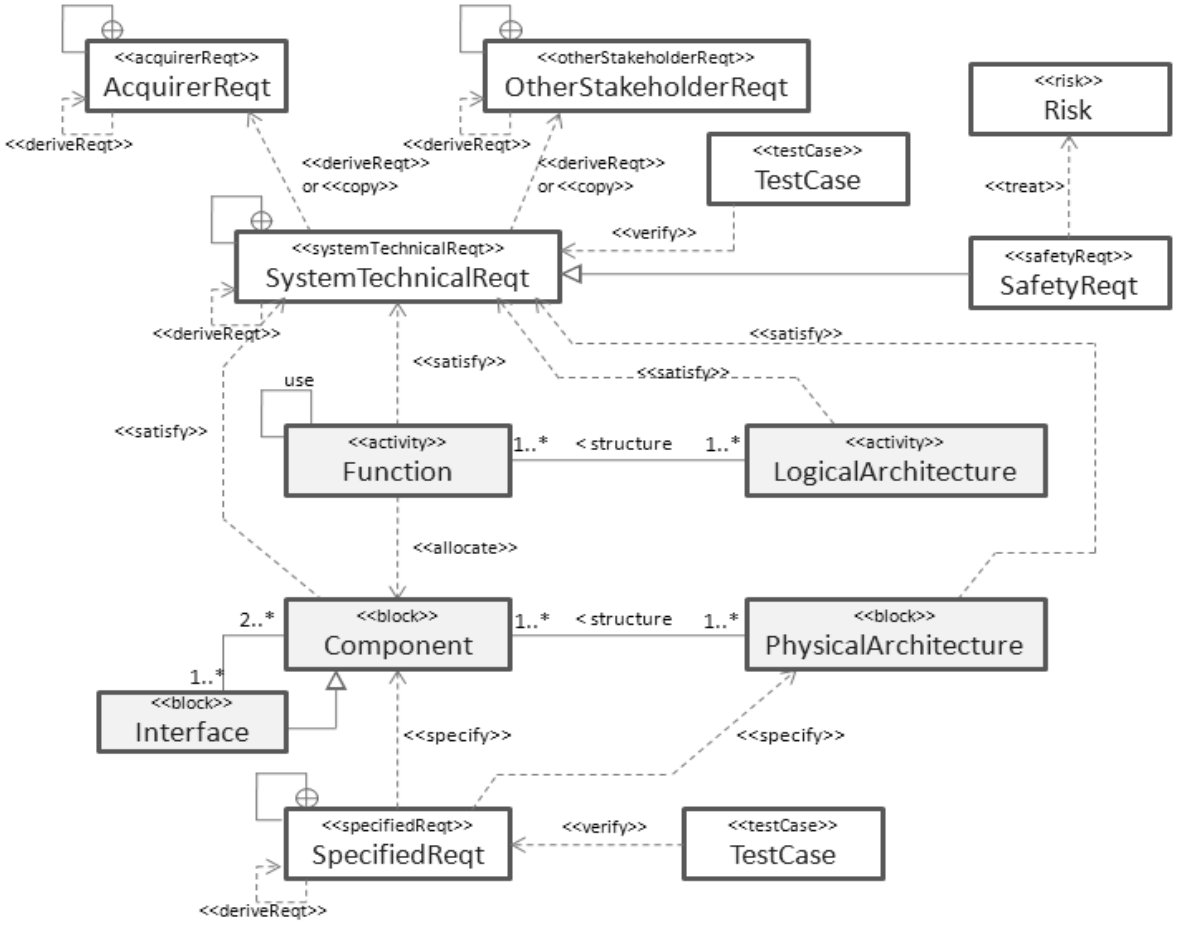


Figure IV.28 : Modèle d'information SysML proposé

Ce modèle peut être vu comme un méta-modèle pour la conception des systèmes complexes sûrs de fonctionnement. Utilisable pour la conception d'un *building block*, il

s'inspire en partie d'éléments de la norme EIA-632, notamment en faisant apparaître clairement la distinction entre les différentes classes d'exigences (acquéreur, autres parties prenantes, technique du système, et spécifiée).

A partir d'exigences d'acquéreur (*AcquirerReqt*) et d'exigences d'autres parties prenantes (*OtherStakeholderReqt*), des exigences techniques du système (*SystemTechnicalReqt*) sont définies. Ces exigences sont liées à des cas de tests (*TestCase*) qui prévoient d'ores et déjà leur vérification. Parmi ces exigences, certaines sont des exigences de sûreté de fonctionnement (*SafetyReqt*). Celles-ci sont liées également aux risques (*Risk*) qu'elles traitent.

Puis les architectures logique (*LogicalArchitecture*) et physique (*PhysicalArchitecture*) sont définies. Elles permettent de structurer respectivement l'ensemble des fonctions (*Function*) et des composants (*Component*) du système. Tous ces éléments satisfont l'ensemble des exigences techniques du système.

Dans le modèle, nous avons souhaité mettre en évidence la notion d'interface (*Interface*). En effet, comme nous l'avons vu dans le 1er chapitre, cette notion est importante. L'interface est ici définie comme un composant particulier, qui est lui-même lié à d'autres composants.

Pour finir, les exigences spécifiées décrivent la solution de conception finale. Elles sont elles-aussi liées à des cas de tests pour les vérifier.

4. Compléments au modèle

Pour accompagner le modèle d'information, un certain nombre de compléments peuvent être ajoutés et utilisés. Entre autre, nous avons pensé à la possibilité d'utiliser un langage formel pour l'expression de certaines exigences (lorsque cela est possible) comme OCL. Il est également intéressant de montrer la possibilité de retirer un certain nombre d'information du modèle utile au management du projet. Par exemple, il est possible (si les outils de modélisation implémentent correctement ces fonctionnalités…) d'obtenir toutes sortes de matrices de traçabilité, ou encore un inventaire des états des exigences.

4.1. Utilisation d'OCL

OCL (*Object Constraint Language*) a été défini pour formaliser des contraintes et être utilisé par UML. Il faisait partie intégrante de la norme UML depuis la version 1.3. Mais aujourd'hui, dans le cadre d'UML 2.0, les spécifications d'OCL figurent dans un document séparé, décrivant en détails la syntaxe formelle du langage.

OCL peut être utilisé dans de nombreux diagrammes d'UML afin de spécifier des contraintes sur l'état d'un objet ou d'un ensemble d'objets. Il pourrait parfaitement être utilisé pour formaliser certaines des exigences (celles qui peuvent être écrites avec ce langage). La suite logique à cette formalisation serait la vérification automatique de ces exigences sur le modèle du système.

4.2. Analyses et résultats issus du modèle de données

4.2.1. Matrices de traçabilité

Nous avons mainte fois répété l'importance de la traçabilité lors de la conception d'un système complexe. Si les outils le permettent, il serait possible avec l'utilisation de notre modèle d'information de récupérer toutes sortes de matrices de traçabilité. Celles-ci pourraient directement être utilisées comme éléments de synthèse pour la traçabilité qui permettraient de voir l'avancé de la conception. Elles pourraient également servir et aider les analyses d'impacts suite à un changement (exigences, composants, fonctions…).

Par exemple, l'outil Tau G2 d'IBM (appartenant auparavant à Telelogic) propose, avec le plugin pour SysML, la génération de matrices de traçabilité, appelée matrices de dépendance. Ces matrices affichent les éléments source et cible de dépendances (c'est-à-dire de lien de traçabilité). Il est possible d'obtenir une matrice qui prend en compte tous les liens (*all*), les liens d'allocations (*allocate*), les liens de dérivation (*derive*), les liens de satisfaction (*satisfy*) ou encore les liens de vérification (*verify*). Les éléments sources sont alors listés verticalement et les éléments cibles listés horizontalement. (Il est aussi possible de demander la transposée de la matrice (*reversed*), dans ce cas les sources sont affichées horizontalement et les cibles verticalement.)

Comme le montre la Figure IV.29, un ensemble d'exigences (REQ1 UC 1.1, REQ1 UC 1.2, etc.) est tracé vers un ensemble de cas de tests (1.1.1, 1.1.2, etc.) pour leur vérification.

Requirement Identifiers	Reqs Tested	REQ1 UC 1.1	REQ1 UC 1.2	REQ1 UC 1.3	REQ1 UC 2.1	REQ1 UC 2.2	REQ1 UC 2.3.1	REQ1 UC 2.3.2
Test Cases	321	3	2	3	1	1	1	1
Tested Implicitly	77							
1.1.1	1	x						
1.1.2	2		x	x				
1.1.3	2	x						
1.1.4	1			x				
1.1.5	2	x						
1.1.6	1		x					
1.1.7	1			x				
1.2.1	2				x		x	
1.2.2	2					x		x

Figure IV.29 : Exemple de matrice de traçabilité pour la vérification des exigences

4.2.2. Etats des exigences

D'autres informations importantes peuvent être déduites du modèle d'information et servir la gestion du projet de conception en fournissant des indications quant à l'état d'avancement de la conception. Il s'agit des états des exigences, contenus dans les champs « maturité » de nos extensions de l'exigence SysML. Il serait intéressant, toujours au

travers d'outils, d'obtenir le pourcentage d'exigences à l'état « origine », « définie », « validée » ou « vérifiée ».

5. Conclusion

Nous venons de présenter dans ce quatrième chapitre un modèle d'information qui vient appuyer la démarche processus d'ingénierie système définie dans le deuxième chapitre. Il permet de se rapprocher d'une méthodologie d'ingénierie système basée sur les modèles. Son rôle est de rendre plus efficace la gestion des exigences et de fournir une base commune entre tous les participants à la conception. Ce modèle d'information est basé sur le langage SysML et respecte les concepts de la norme EIA-632. Pour ce faire, nous avons défini des extensions de SysML, également présentées dans ce chapitre, en particulier pour prendre en compte des exigences de sûreté de fonctionnement ou encore le concept de risque dans le modèle. Le chapitre suivant présentera un exemple d'utilisation de ce modèle issu du domaine aéronautique.

Chapitre 5 : Exemple illustratif de l'avion S18

1. Introduction

Afin d'illustrer les travaux présentés dans ce mémoire, nous présentons un cas d'étude dans ce cinquième et dernier chapitre. Il s'agit d'un avion de ligne dont les informations ont été extraites de l'exemple traité par la norme ARP-4761 : l'avion fictif S18. Ce cas d'étude permet d'illustrer l'utilité de la méthodologie de déclinaison d'exigences de sûreté de fonctionnement sur un système suffisamment complexe. L'utilisation de notre modèle d'information SysML, réalisé à l'aide de l'outil Tau G2 d'IBM, sera elle aussi présentée. Cet exemple sera traité de manière structurée en suivant une décomposition hiérarchique en couche. Pour ce faire, nous nous baserons sur la notion de *building block* recommandée par la norme EIA-632, norme sur laquelle repose notre approche d'intégration de la sûreté de fonctionnement dans les processus d'ingénierie système présentée au chapitre 2.

Ainsi, ce chapitre est composé de trois grandes sections. La première section décrit l'exemple de manière détaillée, en présentant les fonctions et les sous-systèmes, ceci pour trois systèmes de niveaux différents. La deuxième section s'attache à appliquer la méthodologie de déclinaison d'exigences de sûreté de fonctionnement aux différents niveaux hiérarchiques. Quant à la troisième section, celle-ci propose la modélisation SysML de cet exemple utilisant notre modèle d'information.

2. Présentation de l'exemple

L'exemple illustratif choisi et traité est celui de l'avion fictif S18 issu de la norme ARP-4761. Le niveau système complet est alors décrit dans un premier temps, avec les différentes fonctions et la décomposition en sous-systèmes.

2.1. Le système « avion S18 »

L'avion S18 est un quadriréacteur servant au transport de passagers. Il peut transporter de 300 à 500 passagers sur 5000 miles nautiques à mach 0.86. Le temps de vol moyen de cet appareil est de 5 heures.

2.1.1. Les fonctions

Les différentes fonctions réalisées par ce système sont :

- « contrôler la poussée » pour commander la force exercée par les moteurs,

- « contrôler le plan de vol » pour commander les différentes gouvernes de l'appareil, permettant ainsi d'agir sur l'orientation spatiale de l'avion (roulis, lacet, tangage),
- « déterminer l'orientation » pour avoir un retour sur l'orientation spatial relative de l'appareil concernant les angles roulis, lacet et tangage,
- « déterminer la position et la direction » pour avoir un retour sur l'orientation spatial absolue de l'appareil par rapport au sol,
- « contrôler l'avion au sol » pour commander l'appareil au sol, avec comme sous-fonctions :
    - « déterminer la transition air/sol »
    - « décélérer l'avion au sol »
    - « contrôler la direction de l'avion au sol »
    - …
- « contrôler l'environnement cabine » pour gérer l'environnement de la cabine des passagers (température, air, lumière…),
- …

Une étude et une modélisation complète nécessiteraient un trop grand nombre de détails pour être inclus dans ce mémoire de thèse, c'est pourquoi nous nous limitons à cette liste non-exhaustive qui nous permet de modéliser une partie du système en guise d'exemple. Par la suite, nous nous intéressons plus en détail à la sous-fonction « décélérer l'avion au sol ».

2.1.2. Les sous-systèmes

Le système avion se décompose en plusieurs sous-systèmes. C'est la mise en relation et l'interaction de ces sous-systèmes qui permet de réaliser les fonctions listées plus haut. Une liste, encore une fois non-exhaustive, des sous-systèmes est :

- La structure
- Le système de poussée
- Le système des gouvernes
- **Le système des aérofreins**
- **Le système des inverseurs de poussée**
- **Le système des freins des roues**
- …

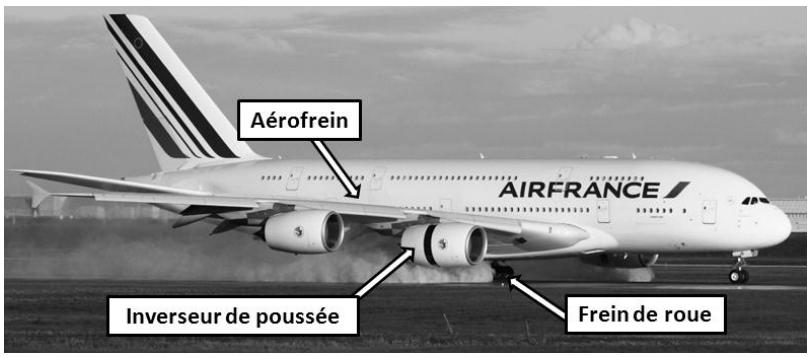


Figure V.1 : Sous-systèmes impliqués dans le freinage

Note : On note ici l'utilisation du mot « système », et non « sous-système ». Ceci est justifié par le fait qu'un sous-système est lui-même un système, mais aussi par le langage employé par les industriels de construction aéronautique qui emploient plutôt le mot « système avion » pour l'avion complet et le mot « système » pour ce niveau (« sous-système » ou « équipement » est utilisé pour le niveau suivant).

Dans la suite de notre étude, nous nous intéressons en particulier à la fonction de décélération de l'avion au sol. Les sous-systèmes qui interviennent pour cette fonction de décélération sont les « inverseurs de poussée », les « aérofreins » et les « freins des roues ».

Les **inverseurs de poussée** sont associés aux réacteurs et peuvent inverser la direction de la poussée. Ils ne peuvent être utilisés qu'au-delà d'une certaine vitesse (autrement les réacteurs réinjectent du gaz chaud et se détériorent).

Les **aérofreins** sont des surfaces mobiles sur les ailes qui servent à réduire la portance et à augmenter la traînée. En conséquence, ils ralentissent la vitesse de l'avion, en l'empêchant de redécoller et en transférant d'avantage de poids sur les roues. Ils sont efficaces seulement au-dessus d'une certaine vitesse.

Enfin, les **freins des roues** sont utilisés à toutes les vitesses et sont situés sur les trains d'atterrissages principaux (pas sur le train avant). Ils peuvent être utilisés de façon dissymétrique, pour lutter contre un vent de travers ou pour effectuer des virages serrés.

Sachant que nous souhaitons parcourir plusieurs niveaux de décomposition hiérarchique, nous continuons la présentation de l'exemple en détaillant le sous-système de freins des roues. Viendra ensuite celle d'un calculateur présent dans ce sous-système de freins des roues : le calculateur BSCU.

2.2. Le système « freins des roues »

Le système « freins des roues » permet de ralentir la rotation des roues de l'avion afin de réduire la vitesse de l'appareil. D'autres systèmes permettent de diminuer la vitesse de l'avion, comme les reverses ou les aérofreins, mais ceux-ci n'ont qu'un rôle secondaire comparé à l'efficacité des freins des roues. Pour l'exemple, nous ne traitons en détails que ce système de freins des roues.

2.2.1. Les fonctions

Le premier rôle du système de freins des roues est de décélérer l'avion au sol sans faire déraper les pneus. Le système de freins des roues accomplit cette fonction automatiquement à l'atterrissage ou manuellement quand le pilote l'active.

En plus de décélérer l'avion, le système de freins des roues est utilisé pour :

- contrôler la direction au sol avec un freinage différentiel (pour les virages serrés),
- stopper la rotation des roues lors de la rétraction des trains,
- empêcher l'avion de bouger quand il est parqué,
- informer du freinage et de l'état du système.

2.2.2. Les sous-systèmes

Le système « freins des roues » n'inclut pas uniquement les freins multi-disques en carbone, mais bien d'autres sous-systèmes nécessaire à leurs contrôles et leur fonctionnement.

Deux ensembles indépendants (Figure V.2) de pistons hydrauliques permettent d'actionner ces freins en carbone. Le premier ensemble est opéré à partir d'une ressource hydraulique VERTE et est utilisé par le mode de freinage NORMAL. Un mode ALTERNATIF est en veille et est sélectionné automatiquement lorsque le système NORMAL défaille. Ce mode est opéré indépendamment à partir d'une ressource hydraulique BLEUE (appartenant au deuxième ensemble) et est soutenu par un accumulateur qui est également utilisé pour activer le frein de parking. L'accumulateur approvisionne le système ALTERNATIF dans le mode de freinage d'URGENCE lorsque la ressource BLEUE est perdue et que le mode NORMAL n'est pas disponible. La commutation entre ces modes est automatique, dépendant des conditions de défaillances, mais elle peut aussi être sélectionnée manuellement.

Un calculateur, appelé calculateur BSCU pour *Braking System Control Unit*, permet d'élaborer les ordres de commandes de freinage en fonction du mode sélectionné (automatique ou manuel) et/ou des ordres des pilotes provenant des pédales. Alimenté par une ressource électrique, ce calculateur envoi aussi des informations à un système d'annonciation de freinage. Ce système est chargé de renseigner les pilotes sur l'état général de tout le système de freinage.

Ce calculateur permet de fournir les commandes de freinage à la chaine NORMALE, tandis que concernant la chaine ALTERNATIVE, celle-ci est commandée directement par une liaison mécanique liée aux pédales.

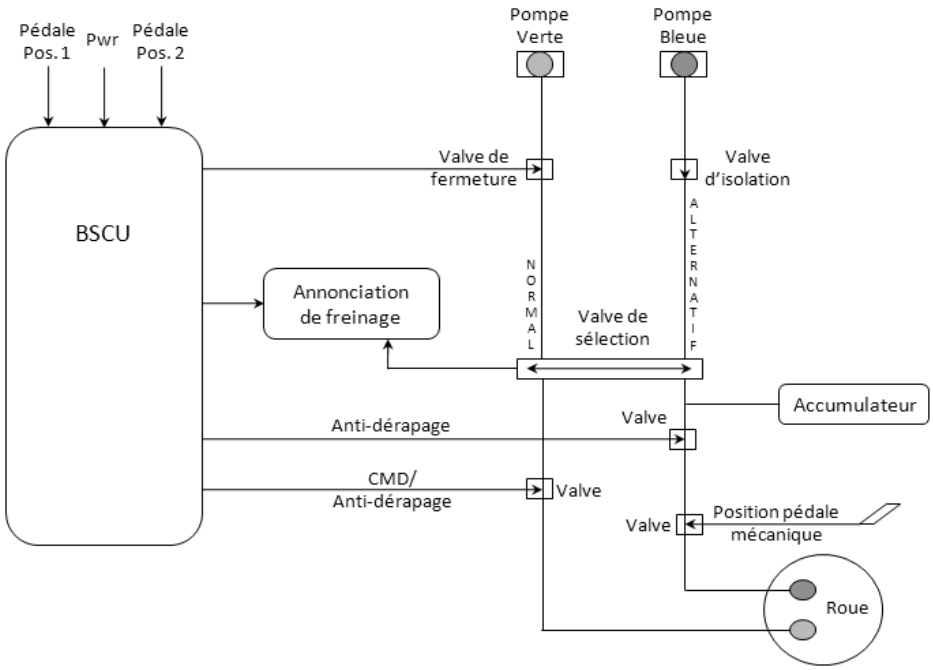


Figure V.2 : Système de freins des roues

Ainsi, les sous-systèmes considérés dans notre étude sont les suivants (cf. Figure V.2) :

- Le calculateur BSCU,
- La ressource électrique,
- Le système d'annonciation,
- Les composants hydrauliques du mode NORMAL,
- Les composants hydrauliques du mode ALTERNATIF,
- L'accumulateur (mode d'URGENCE et frein de parking),
- La ressource hydraulique VERTE,
- La ressource hydraulique BLEUE.

2.3. Le système « calculateur BSCU »

Nous venons de voir le système de freins des roues de l'avion S18. Ce système contient un calculateur : le BSCU, que nous détaillons pour le troisième et dernier niveau de notre étude. Cet élément correspond en fait à la partie « intelligente » du système de freins des roues, ce que nous allons retrouver dans le détail de ses fonctions.

2.3.1. Les fonctions

Le rôle du calculateur BSCU, *Braking System Control Unit*, est, comme son nom le laisse deviner, d'élaborer et de fournir des commandes pour le freinage. En fait, ses activités peuvent être définies par deux fonctions principales qui sont :

- Générer la commande de freinage à partir de l'information des pédales, en assurant l'anti-dérapage des roues,
- Sélectionner, grâce à son moniteur général, le mode de freinage alternatif si le BSCU est défaillant (ne peut plus générer la commande de freinage).

2.3.2. Les sous-systèmes

Pour des raisons de sûreté (que nous verrons plus loin), l'architecture du BSCU est composée de 2 unités de calcul en parallèle : le **BSCU1** et le **BSCU2**. Chaque unité de calcul comporte :

- Une **ressource électrique**, pour l'alimentation des sous-systèmes en énergie électrique,
- Un **CPU**, pour le calcul des fonctions de commande à réaliser,
- Un **moniteur de puissance**, pour la surveillance de la disponibilité de la ressource de puissance,
- Un **moniteur système**, pour le contrôle du fonctionnement de l'unité de calcul.

En plus de ces deux unités de calcul, le BSCU inclut également un **moniteur général** et un **commutateur**. Le moniteur général contrôle l'état de fonctionnement des deux unités de calcul (BSCU1 et BSCU2) et permet de fournir les résultats des opérations du BSCU1 ou ceux du BSCU2 en sortie du BSCU à l'aide du commutateur. Il permet également de signaler l'invalidité générale du BSCU lorsque celui-ci est défaillant. De manière plus détaillée, le fonctionnement en fonction des défaillances est le suivant :

- En opération normal, le BSCU1 fournit les commandes de freinage et d'anti-dérapage aux freins des roues.
- Si le moniteur système du BSCU1 reporte une défaillance, alors le BSCU2, si valide, fournit la commande.
- Si les BSCU1 et BSCU2 sont défaillants, alors le moniteur général reporte l'état invalide. Dans ce cas, le circuit normal du système de freins des roues ne peut plus être utilisé.

La Figure V.3 synthétise sur un schéma l'ensemble des relations entre les différents composants du système BSCU.

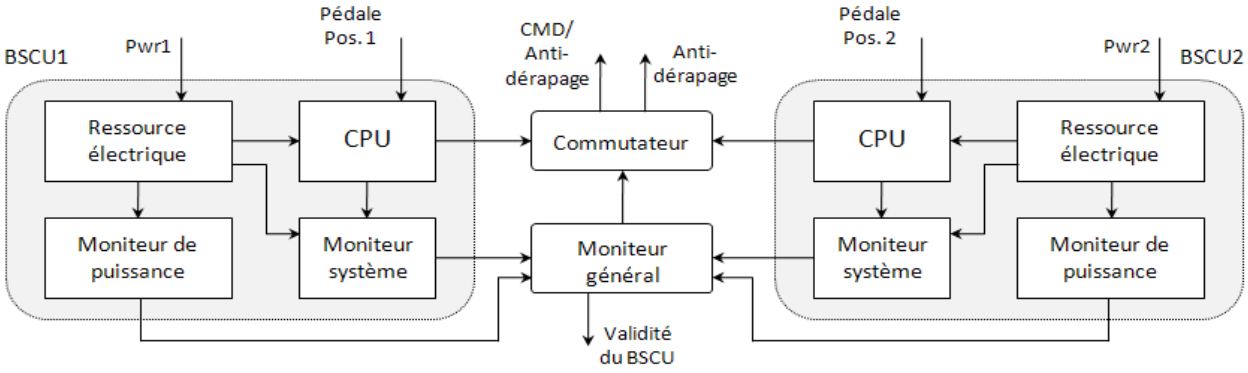


Figure V.3 : Système BSCU

3. Application de la méthodologie de déclinaison d'exigences de Sûreté de Fonctionnement

Suite à la présentation de l'exemple, nous allons maintenant appliquer la méthodologie de déclinaison d'exigences de sûreté de fonctionnement définie dans le chapitre 3. Pour bien illustrer l'intérêt de la méthodologie, nous allons l'appliquer sur trois niveaux : celui de « l'avion S18 » (en ne s'intéressant qu'à la fonction de décélération au sol), celui du système « freins des roues » et celui de l'équipement « calculateur BSCU ».

3.1. Niveau « avion S18 »

Pour le cas d'étude, nous nous intéressons à la fonction de décélération au sol de l'avion S18. Les phases considérées sont les phases d'atterrissage, de décollage dans le cas d'interruption avant la vitesse V1 (l'avion commence son décollage, l'équipage découvre une anomalie avant la limite de vitesse V1, il décide alors d'interrompre le décollage : freinage d'urgence de l'avion) et la circulation sur les voies de roulage (mode taxi). Nous rappelons que les sous-systèmes qui interviennent pour la fonction de décélération sont les « inverseurs de poussée », les « aérofreins » et les « freins des roues » (revoir Figure V.1).

Nous allons appliquer la méthode sur les aspects « décélération au sol » du système avion, pour identifier les exigences de sûreté et déterminer leur déclinaison au niveau des sous-systèmes.

3.1.1. Etape 1 : Analyse des défaillances du système

Donnée par la Figure V.4, l'application de l'AMDEC au niveau du système « Avion S18 » pour la fonction de décélération fait apparaître les modes de défaillance suivants :

- « perte de la capacité de décélération »,
- « décélération involontaire après V1 »,
- « perte partielle de la capacité de décélération »,
- « perte de la capacité de décélération automatique »,
- « décélération asymétrique ».

Il faut noter que le tableau AMDEC de la Figure V.4 n'est pas exhaustif. D'autres modes de défaillance peuvent être définis.

Dans ce tableau, nous avons examiné les différentes phases d'utilisation du système car les contraintes changent en fonction de ces phases. Cependant, la fréquence n'apparait pas, car elle n'est pas disponible au début de l'étude. Ainsi, les actions correctives agissent sur les causes et permettent de définir les fréquences associées qui conduisent à une criticité acceptable (criticité = fréquence x gravité) pour une gravité donnée, qui elle est connue.

Pour donner un exemple, nous pouvons identifier une exigence de sûreté au niveau système avion S18 : « *la perte non-annoncée de la capacité de décélération doit avoir une fréquence* $< 5.10^{-9}/flt$ ». Cette exigence est liée à la cause système : « *perte non-annoncée de la capacité de décélération* », qui va d'ailleurs être étudiée dans la deuxième étape de la méthode.

Remarque : concernant les données quantitatives visibles dans les arbres qui vont suivre, une distinction d'unité sera à faire entre celle des objectifs : par heure de vol (/fh), et celle des probabilités : par vol (/flt). Sachant que l'on considère une durée moyenne des vols de 5h, il y aura équivalence entre, par exemple, un objectif de $10^{-9}/fh$ *et une probabilité de* $5.10^{-9}/flt$.

3.1.2. Etape 2 : Analyses des causes des défaillances

L'arbre de défaillances de la Figure V.5 s'intéresse à la cause système : « **perte non-annoncée de la capacité de décélération** ».

Il est construit progressivement dans le but d'obtenir des feuilles qui représentent les modes de défaillance des sous-systèmes. Ainsi, d'après la figure, la perte non-annoncée de la capacité de décélération fait intervenir : la perte non-annoncée des inverseurs de poussée, la perte non-annoncée de tous les freins sur piste contaminée (par de l'eau, de la neige, de la glace…) et la perte non-annoncée de tous les freins.

Dans son arbre de défaillance, nous constatons que la cause système « *perte non-annoncée de la capacité de décélération* » fait intervenir une autre cause système : « *perte non-annoncée des freins des roues* ». Il est effectivement possible qu'un arbre implique une cause associée à un autre mode de défaillance du système. Mais cela implique que, pour les calculs de probabilité, nous devons respecter l'objectif de la cause, mais aussi ceux des autres causes survenant dans l'arbre.

Syst.	Fonction	Mode de défaillance	Phase	Effet	Gravité	Cause	Actions Correctives *(Objectifs de SdF)*
Avion	Décélérer l'avion au sol	Perte de la capacité de décélération					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage ne peut pas décélérer l'avion, il en résulte une sortie de piste à haute vitesse.	Catastro-phique	**Perte non-annoncée de la capacité de décélération**	**Faire en sorte que la fréquence soit $<10^{-9}$/fh (fh: flight hour)**
		b) annoncée	Atterrissage	L'équipage choisit un aéroport approprié et prépare les passagers à une sortie de piste.	Hasardeux	**Perte annoncée de la capacité de décélération**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		c) non-annoncée	Taxi	L'équipage ne peut pas stopper l'avion, il en résulte un contact avec obstacles à faible vitesse.	Majeur	**Perte non-annoncée des freins**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		d) annoncée	Taxi	L'équipage dirige l'appareil en évitant les obstacles et demande un remorquage.	Sans gravité	**Perte annoncée des freins**	**rien**
		Décélération involontaire après V1	Décollage	L'équipage ne peut pas décoller dû à l'application des freins en même temps qu'une poussée importante, il en résulte une sortie de piste à haute vitesse.	Catastro-phique	**Décélération involontaire après V1**	**Faire en sorte que la fréquence soit <10-9/fh**
		Perte partielle de la capacité de décélération					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage ne peut pas décélérer complètement l'avion, il en résulte une sortie de piste potentielle.	Hasardeux	**Perte partielle non-annoncée de la capacité de décélération**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		b) annoncée	Atterrissage	L'équipage choisit un aéroport approprié et prépare les passagers à une sortie de piste.	Majeur	**Perte partielle annoncée de la capacité de décélération**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		c) non-annoncée	Taxi	L'équipage peut ne pas être capable de stopper de manière adéquate l'avion avant un obstacle, il en résulte une collision à faible vitesse.	Mineur	**Perte partielle non-annoncée des freins**	**Faire en sorte que la fréquence soit $<10^{-3}$/fh**
		d) annoncée	Taxi	L'équipage dirige l'appareil en évitant les obstacles et demande un remorquage.	Sans gravité	**Perte partielle annoncée des freins**	**rien**
		Perte de la capacité de décélération automatique					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage arme le freinage automatique. A l'atterrissage ou pendant le décollage annulé, il défaille. L'équipage reconnait la situation et active manuellement la capacité de décélération. Suivant le temps de réaction de l'équipage , il en résulte une sortie de piste potentielle.	Majeur	**Perte non-annoncée de la capacité de décélération automatique**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		b) annoncée	Atterrissage / Décollage annulé	L'équipage active manuellement la capacité de décélération pendant l'atterrissage ou le décollage annulé.	Sans gravité	**Perte annoncée de la capacité de décélération automatique**	**rien**
		Décélération asymétrique					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage n'est pas préparé à une décélération asymétrique et réagit trop tard pour maintenir le contrôle en direction, il en résulte une sortie de piste sur le coté.	Majeur	**Décélération asymétrique non-annoncée**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		b) annoncée	Atterrissage	L'équipage est préparé à une décélération asymétrique et contre avec la gouverne de direction et le train directionnel avant.	Mineur	**Décélération asymétrique annoncée**	**Faire en sorte que la fréquence soit $<10^{-3}$/fh**
		c) Décélération asymétrique	Taxi	L'avion dévie légèrement de sa trajectoire.	Sans gravité	**Décélération asymétrique**	**rien**

Figure V.4 : AMDEC système « Avion S18 » pour la fonction de décélération au sol

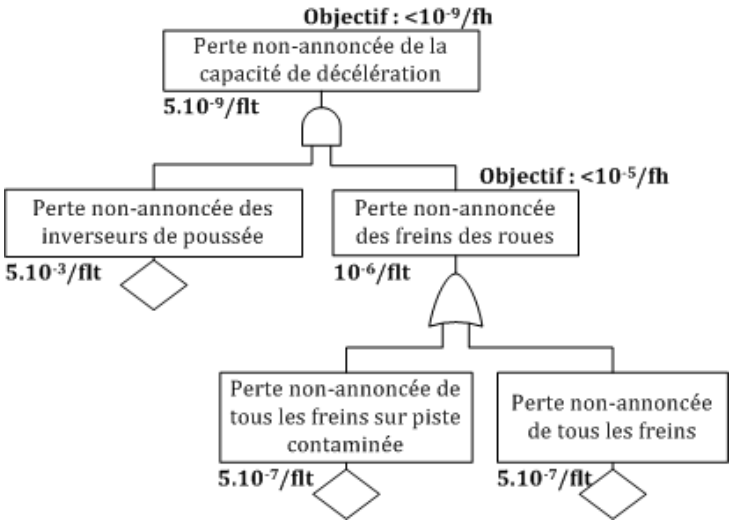


Figure V.5 : Arbre de défaillance de la « perte non-annoncée de la capacité de décélération »

Remarque : l'arbre de la Figure V.5 ne fait pas intervenir la défaillance des aérofreins, car l'efficacité de ces derniers est bien plus faible que celle des freins ou des inverseurs de poussée (surtout à faible vitesse). Ils ne sont donc pas pris en compte dans la capacité de décélération. En revanche, ils sont à considérer pour des décélérations involontaires.

L'arbre de défaillance de la Figure V.6 s'intéresse à la cause système : « **décélération involontaire après V1** ».

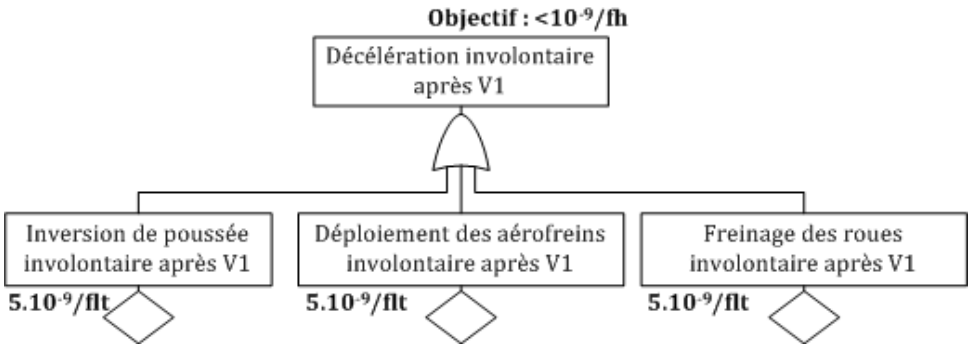


Figure V.6 : Arbre de défaillance de la « décélération involontaire après V1 »

D'après cet arbre, la décélération involontaire après V1 fait intervenir l'inversion de poussée, le déploiement des aérofreins et le freinage des roues, toutes ces actions de manière involontaire et après la vitesse V1.

3.1.3. Etape 3 : Analyses des défaillances des sous-systèmes

Pour cet exemple, nous ne présentons, dans la Figure V.7, que l'AMDEC du sous-système « freins des roues », avec l'étude de la fonction « *freiner les roues sur le sol sans dérapage* ». Dans cette étude, nous retrouvons quelques modes de défaillance du sous-système « freins des roues » présents dans les arbres de défaillance des Figure V.5 et Figure V.6, comme :

- *« perte totale du freinage des roues »*
- *« freinage des roues involontaire après V1 »*

Syst.	Fonction	Mode de défaillance	Phase	Effet	Gravité	Cause	Actions Correctives *(Objectifs de SdF)*
Freins des roues	Freiner les roues sur le sol sans dérapage	Perte totale du freinage des roues					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage détecte la défaillance quand les freins sont opérés. L'équipage utilise au maximum les aérofreins et les inverseurs de poussée. Il peut en résulter une sortie de piste.	Hasardeux	**Perte non-annoncée de tous les freins des roues**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		b) annoncée	Atterrissage	L'équipage choisit un aéroport approprié et prépare les passagers à une sortie de piste. L'équipage utilise au maximum les aérofreins et les inverseurs de poussée.	Hasardeux	**Perte annoncée de tous les freins des roues**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		Perte symétrique partielle du freinage des roues					
		a) non-annoncée	Atterrissage / Décollage annulé	L'équipage détecte la défaillance quand les freins sont opérés. L'équipage utilise au maximum les freins disponibles, les aérofreins et les inverseurs de poussée. La température des roues freinées augmente et peut conduire à la défaillance des roues ou pneus. Suivant le nombre de freins perdus, il peut en résulter une sortie de piste.	Majeur à hasardeux	**Perte symétrique partielle non-annoncée du freinage des roues**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		b) annoncée	Atterrissage	L'équipage est au courant de la perte symétrique partielle du freinage des roues avant l'atterrissage. L'équipage utilise au maximum les freins disponibles, les aérofreins et les inverseurs de poussée. Suivant le nombre de freins perdus, il peut en résulter une sortie de piste.	Majeur	**Perte symétrique partielle annoncée du freinage des roues**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		Perte asymétrique du freinage des roues					
		a) sans perte de la gouverne de direction ou du contrôle directionnel du train avant	Atterrissage / Décollage annulé	Perte de performance de freinage. Tendance à virer sur le coté de la piste. Pour les performances de freinages et la température, les effets sont les mêmes que la perte partielle de freinage ci-dessus. L'équipage garde l'avion sur la piste en compensant avec la gouverne de direction à haute vitesse et avec le train avant directionnel à faible vitesse. Suivant le nombre de freins perdus, il peut en résulter une sortie de piste sur le coté.	Majeur	**Défaillance du système de freins sans perte de la gouverne de direction ou du contrôle directionnel du train avant**	**Faire en sorte que la fréquence soit $<10^{-5}$/fh**
		b) avec perte de la gouverne de direction ou du contrôle directionnel du train avant	Atterrissage	Perte de performance de freinage. Tendance à virer sur le coté de la piste. Pour les performances de freinages et la température, les effets sont les mêmes que la perte partielle de freinage ci-dessus. L'équipage ne peut pas garder l'avion sur la piste, il en résulte une sortie de piste sur le coté.	Hasardeux	**Défaillance du système de freins combinée avec la perte de la gouverne de direction ou du contrôle directionnel du train avant**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		Feinage involontaire des roues					
		a) sans blocage des roues	Décollage, avant V1	L'équipage stoppe l'avion sur la piste.	Mineur	**Freinage involontaire sans blocage des roues**	**Faire en sorte que la fréquence soit $<10^{-3}$/fh**
		b) avec blocage de toutes les roues	Décollage, avant V1	Eclatement potentiel de tous les pneus et perte de l'efficacité du freinage.	Hasardeux	**Freinage involontaire avec blocage de toutes les roues**	**Faire en sorte que la fréquence soit $<10^{-7}$/fh**
		c) avec ou sans blocage de toutes les roues	Décollage, après V1	l'équipage ne peut pas décoller ou procéder à une interruption de décollage en toute sécurité, il en résulte une sortie de piste à haute vitesse.	Catastro-phique	**Freinage involontaire avec ou sans blocage de toutes les roues**	**Faire en sorte que la fréquence soit $<10^{-9}$/fh**
		d) freinage involontaire non-détecté d'une roue sans blocage	Décollage	L'équipage ne peut pas détecter la défaillance par l'asymétrie qui est très faible. La température de la roue peut atteindre une valeur très importante. L'équipage rétracte les trains, il en résulte un feu des pneus possible.	Catastro-phique	**Freinage involontaire non-détecté d'une roue sans blocage**	**Faire en sorte que la fréquence soit $<10^{-9}$/fh**
		e) freinage involontaire d'une roue sans blocage couplé avec la détection de la température élevée	Décollage	L'équipage ne peut pas détecter la défaillance par l'asymétrie qui est très faible. La température de la roue peut atteindre une valeur très importante. L'équipage détecte la température élevée et garde les trains sortis pour refroidir les freins.	Mineur	**Freinage involontaire d'une roue sans blocage couplé avec la détection de la température élevée**	**Faire en sorte que la fréquence soit $<10^{-3}$/fh**

Figure V.7 : AMDEC du sous-système « freins des roues »

Un exemple d'exigence de sûreté identifiée au niveau de ce sous-système de freins des roues est : « *la perte non-annoncée de tous les freins doit avoir une fréquence < 5.10^{-7}/flt* », liée au mode de défaillance « *perte totale du freinage des roues* ».

3.1.4. Etape 4 : Synthèse

La synthèse permet de montrer la déclinaison des exigences de sûreté au niveau système en exigences de sûreté sous-système. Nous donnons ici deux exemples de déclinaison qui correspondent respectivement aux deux arbres de défaillance créés (voir Figure V.5 et Figure V.6).

L'exigence de sûreté système « *la perte non-annoncée de la capacité de décélération doit avoir une fréquence <5.10^{-9}/flt* » est déclinée au niveau sous-systèmes en :

- Pour les **freins des roues** :
  - « *la perte non-annoncée de tous les freins sur une piste contaminée doit avoir une fréquence < 5.10^{-7}/flt* »
  - « *la perte non-annoncée de tous les freins doit avoir une fréquence < 5.10^{-7}/flt* »
- Pour les **inverseurs de poussée** :
  - « *la perte non-annoncée des inverseurs de poussée doit avoir une fréquence < 5.10^{-3}/flt* »

L'exigence de sûreté système « *la décélération involontaire après V1 doit avoir une fréquence < 5.10^{-9}/flt* » est déclinée au niveau sous-systèmes en :

- Pour les **freins des roues** :
  - « *le freinage involontaire des roues après V1 doit avoir une fréquence < 5.10^{-9}/flt* »
- Pour les **inverseurs de poussée** :
  - « *l'inversion involontaire de la poussée après V1 doit avoir une fréquence < 5.10^{-9}/flt »*
- *Pour les **aérofreins** :*
  - *« le déploiement involontaire des aérofreins après V1 doit avoir une fréquence < 5.10^{-9}/flt* »

3.2. Niveau « freins des roues »

Nous poursuivons l'étude de cet exemple par l'application de la méthodologie de déclinaison d'exigences de sûreté de fonctionnement au niveau suivant : celui du système « freins des roues ».

3.2.1. Etape 1 : Analyse des défaillances du système

L'analyse des défaillances au niveau du système « freins des roues » a déjà été réalisée dans l'étape 3 du niveau précédent. Pour cette étape, il suffit donc de reprendre l'AMDEC réalisée en Figure V.7 au paragraphe 3.1.3.

Les modes de défaillances identifiés sont les suivants (liste non-exhaustive) :

- « *Perte totale du freinage des roues* »,
- « *Perte symétrique partielle du freinage des roues* »,
- « *Perte asymétrique du freinage des roues* »,
- « *Freinage involontaire des roues* ».

Un exemple d'exigence de sûreté identifiée à l'aide de cette AMDEC est : « *la perte non-annoncée de tous les freins doit avoir une fréquence* < $5.10^{-7}/flt$ », liée à la cause système « *perte non-annoncée de tous les freins des roues* ».

3.2.2. Etape 2 : Analyses des causes des défaillances

L'arbre de défaillance de la Figure V.8 s'intéresse à la cause système : « **perte non-annoncée de tous les freins des roues** ».

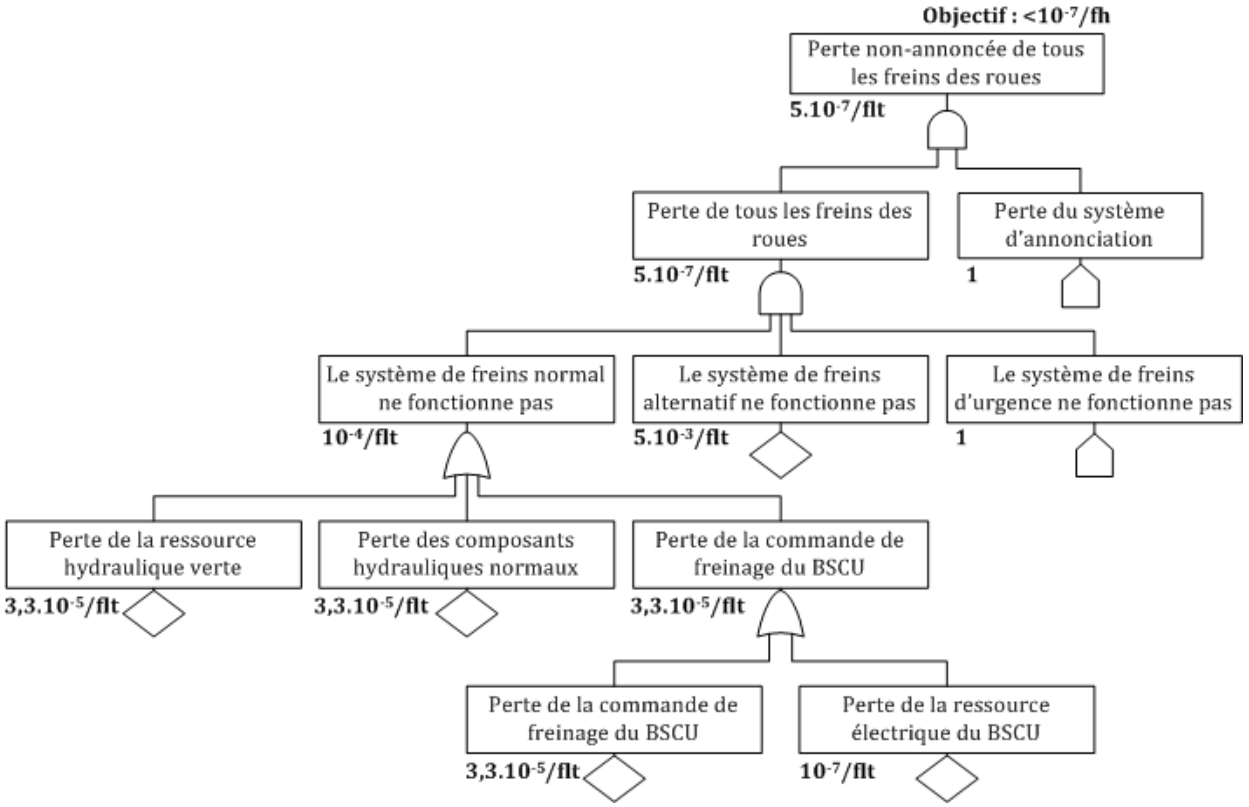


Figure V.8 : Arbre de défaillance de la « perte non-annoncée de tous les freins des roues »

Cette cause fait intervenir des pertes (défaillance) au niveau de la ressource et des composants hydrauliques de la chaine normale, du calculateur BSCU, ainsi qu'au niveau des systèmes de freins alternatif et d'urgence et du système d'annonciation. Mais d'après la figure, nous constatons que nous n'étudions pas plus en détails la défaillance du système de freins alternatif. Egalement, nous ne considérons pas de contrainte de fiabilité pour les défaillances du système de freins d'urgence et du système d'annonciation.

3.2.3. Etape 3 : Analyses des défaillances des sous-systèmes

Pour cet exemple, nous ne présentons, dans la Figure V.9, que l'AMDEC sous-système du « calculateur BSCU », avec l'étude de la fonction « *fournir la commande de freinage* ».

Dans cette AMDEC, nous reconnaissons le mode de défaillance « *perte de la commande de freinage du BSCU* » présent dans l'arbre de défaillance de la Figure V.8.

Syst.	Fonction	Mode de défaillance	Phase	Effet	Gravité	Cause	Actions Correctives *(Objectifs de SdF)*
BSCU	Fournir la commande de freinage	perte de la fonction	Atterrissage/ Décollage annulé	Perte du freinage	Hasardeux	**Une défaillance du BSCU provoque la perte de la commande de freinage**	**Faire en sorte que la fréquence soit <$3{,}3.10^{-5}$/fh**
		déclanchement intempestif de la fonction	Décollage	Freinage involontaire	Mineur à catastro-phique	**Le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire**	**Faire en sorte que la fréquence soit <$2{,}9.10^{-9}$/fh**

Figure V.9 : AMDEC du sous-système « calculateur BSCU »

Ce mode de défaillance est d'ailleurs lié à une exigence de sûreté que nous identifions : « *la perte du calculateur BSCU entrainant la perte de la commande de freinage doit avoir une fréquence* < $3{,}3.10^{-5}/flt$ ».

3.2.4. Etape 4 : Synthèse

Suite à l'application de la méthodologie de déclinaison d'exigences de sûreté de fonctionnement au niveau du système de freins des roues, nous pouvons alors identifier une déclinaison. Elle concerne la perte non-annoncée de tous les freins.

L'exigence de sûreté système « *la perte non-annoncée de tous les freins doit avoir une fréquence* < $5.10^{-7}/flt$ » est déclinée au niveau sous-systèmes en :

- Pour le **système d'annonciation** :
  - « *Il n'y a pas de contrainte concernant la fiabilité du système d'annonciation* »
- Pour le **système de freinage alternatif** :
  - « *La perte du système de freinage alternatif doit avoir une fréquence* < $5.10^{-3}/flt$ »
- Pour le **système de freinage d'urgence** :
  - « *Il n'y a pas de contrainte concernant la fiabilité du système de freinage d'urgence* »
- Pour la **ressource hydraulique normale** :
  - « *La perte de la ressource hydraulique du mode normal doit avoir une fréquence* < $3{,}3.10^{-5}/flt$ »
- Pour les **composants hydrauliques normaux** :
  - « *La perte des composants hydraulique du système de freinage du mode normal doit avoir une fréquence* < $3{,}3.10^{-5}/flt$ »
- Pour la **ressource électrique** :
  - « *La perte de la ressource électrique doit avoir une fréquence* < $10^{-7}/flt$ »
- Pour le **calculateur BSCU** :
  - « *La perte du calculateur BSCU entrainant la perte de la commande de freinage doit avoir une fréquence* < $3{,}3.10^{-5}/flt$ »

3.3. Niveau « calculateur BSCU »

Nous poursuivons l'étude de cet exemple par l'application de la méthodologie de déclinaison d'exigences de sûreté de fonctionnement au niveau suivant : celui du système « calculateur BSCU ».

3.3.1. Etape 1 : Analyse des défaillances du système

L'analyse des défaillances au niveau du système « calculateur BSCU » a déjà été réalisée dans l'étape 3 du niveau précédent. Pour cette étape, il suffit donc de reprendre l'AMDEC réalisée en Figure V.9 au paragraphe 3.2.3.

Les modes de défaillances identifiés sont les suivants :

- Perte de la commande de freinage,
- Déclanchement intempestif de la commande de freinage.

Un exemple d'exigence de sûreté identifiée à l'aide de cette AMDEC est : « *la perte du calculateur BSCU entrainant la perte de la commande de freinage doit avoir une fréquence* < *3,3.10-5/flt* », liée à la cause système « *une défaillance du BSCU provoque la perte de la commande de freinage* ».

3.3.2. Etape 2 : Analyses des causes des défaillances

L'arbre de défaillance présent dans les Figure V.10 et Figure V.11 s'intéresse à la cause système : « **une défaillance du BSCU provoque la perte de la commande de freinage** ».

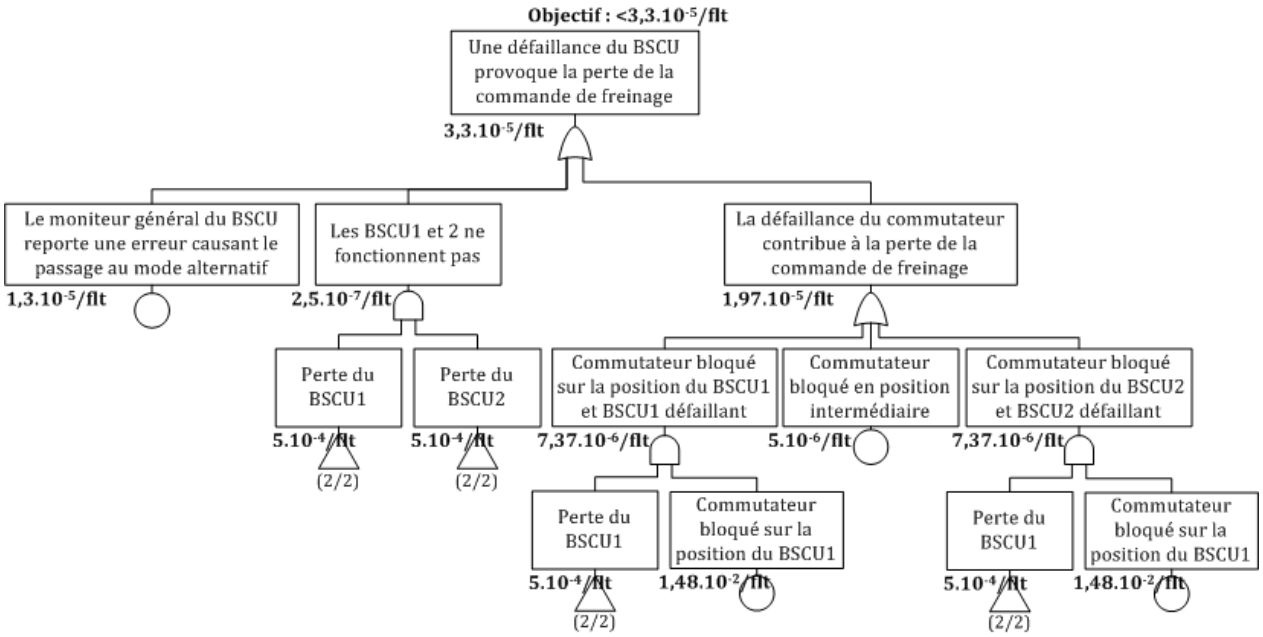


Figure V.10 : Arbre de défaillance de « une défaillance du BSCU provoque la perte de la commande de freinage » (1/2)

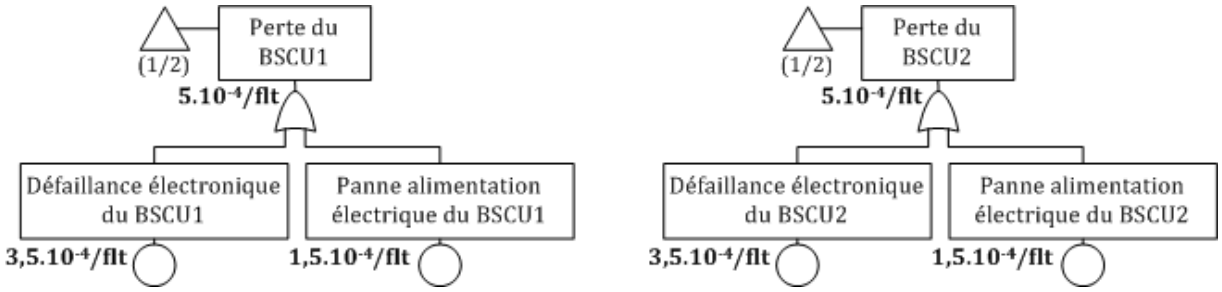


Figure V.11 : Arbre de défaillance de « une défaillance du BSCU provoque la perte de la commande de freinage » (2/2)

La cause système « *une défaillance du BSCU provoque la perte de la commande de freinage* » provient donc : d'une défaillance du moniteur général, de la perte des deux unités de calcul, ou alors d'une défaillance du commutateur liée ou non à une défaillance d'une unité de calcul (suivant la position dans laquelle le commutateur est bloqué). Quant aux défaillances mêmes des unités de calcul, celles-ci proviennent soit d'une défaillance électronique, soit d'une panne d'alimentation électrique.

L'arbre de défaillance présent dans les Figure V.12, Figure V.13 et Figure V.14 s'intéresse à la cause système : « **le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire** ».

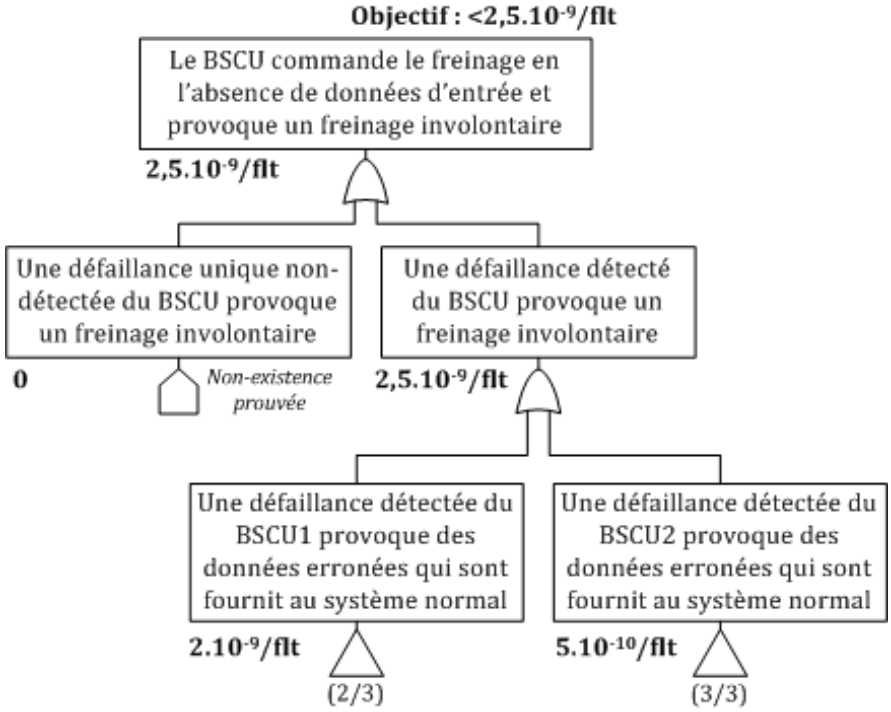


Figure V.12 : Arbre de défaillance de « le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire » (1/3)

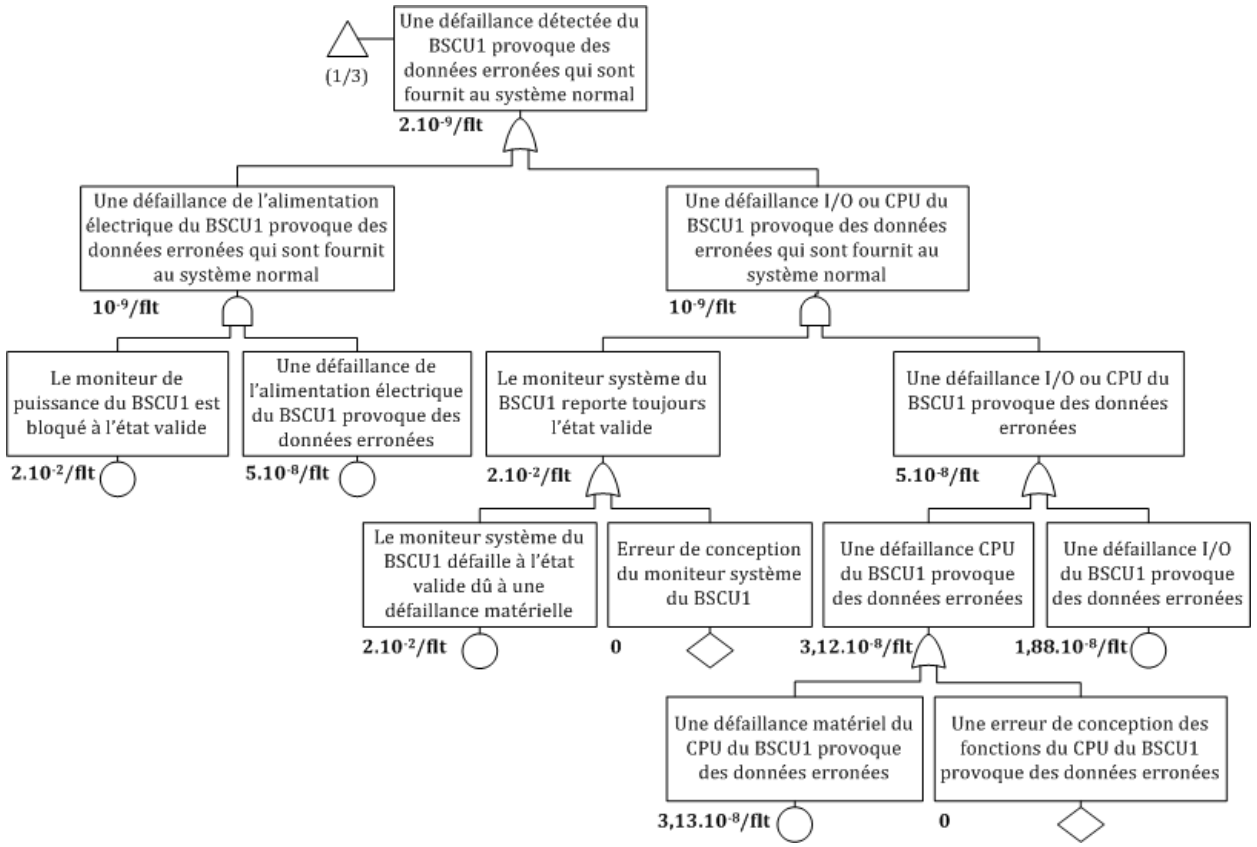


Figure V.13 : Arbre de défaillance de « le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire » (2/3)

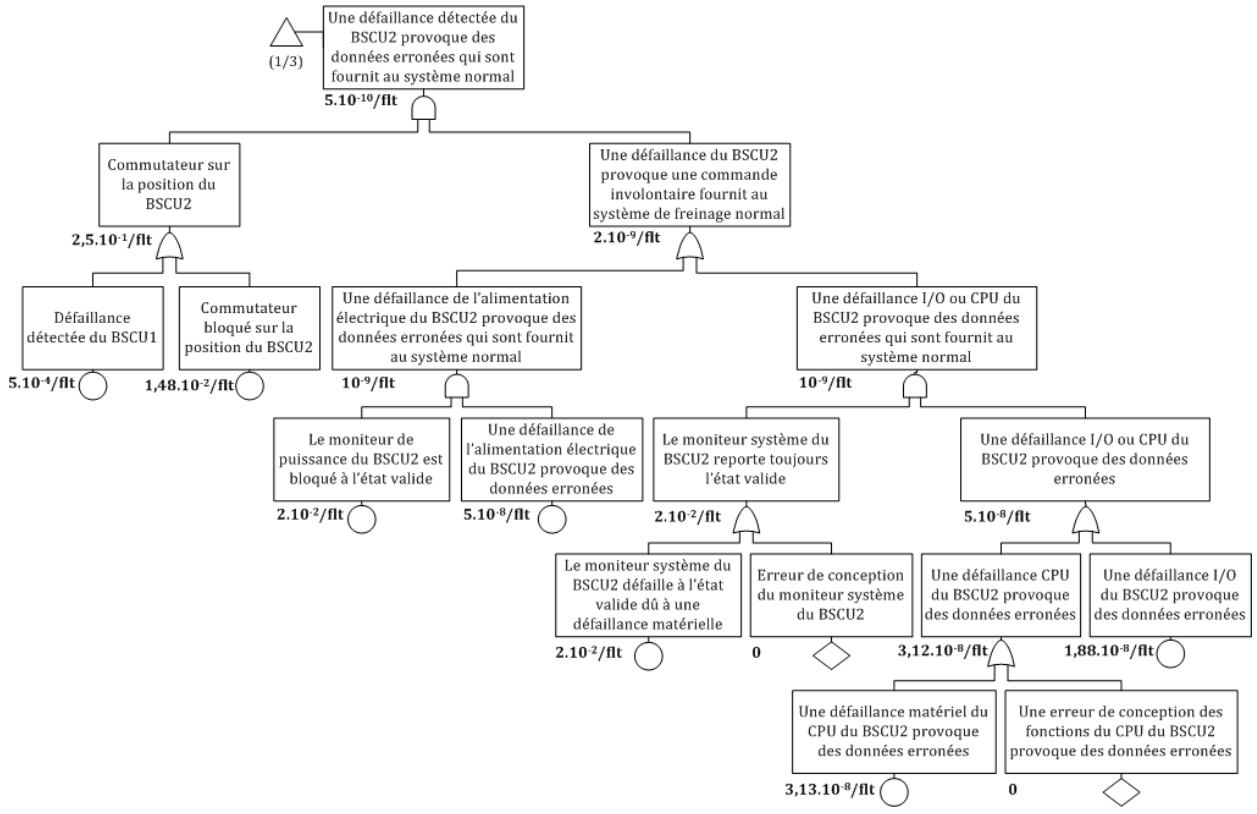


Figure V.14 : Arbre de défaillance de « le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire » (3/3)

Lors de l'analyse de la cause système « *le BSCU commande le freinage en l'absence de données d'entrée et provoque un freinage involontaire* », nous voyons sur l'arbre que nous ne considérons que les défaillances détectées du BSCU. Nous supposons que la non-existence d'une défaillance unique non-détectée à été prouvé (d'où la probabilité de « 0 » dans l'arbre).

Cette cause système provient donc soit d'une défaillance détectée du BSCU1, soit d'une défaillance détectée du BSCU2 dans le cas où le BSCU2 est utilisé. Cette utilisation du BSCU2 se produit lorsque le BSCU1 est défaillant ou lorsque le commutateur est bloqué sur la position du BSCU2. Concernant la défaillance détectée d'une unité de calcul (BSCU1 ou BSCU2), celle-ci se produit lors d'une défaillance de son alimentation électrique combinée avec une défaillance de son moniteur de puissance, ou bien lors d'une défaillance I/O ou CPU combinée avec une défaillance de son moniteur système (qui reporte toujours l'état valide).

3.3.3. Etape 3 : Analyses des défaillances des sous-systèmes

Nous choisissons de ne pas présenter d'AMDEC relatif aux sous-systèmes du calculateur BSCU dans ce mémoire. Le niveau atteint présente en effet des composants dont les fonctions et leurs modes de défaillance sont compréhensible aisément. L'intérêt de la présentation de ces AMDEC n'est donc pas essentiel.

3.3.4. Etape 4 : Synthèse

Suite à l'application de la méthodologie de déclinaison d'exigences de sûreté de fonctionnement au niveau du calculateur BSCU, nous pouvons alors identifier deux déclinaisons. La première concerne la perte du calculateur BSCU et la deuxième le freinage involontaire.

L'exigence de sûreté système « la perte du calculateur BSCU entrainant la perte de la commande de freinage doit avoir une fréquence < $3{,}3.10^{-5}$/flt » est déclinée au niveau sous-systèmes en :

- Pour le **moniteur général** :
  - *« L'erreur du système de surveillance du BSCU entraînant un passage en mode alternatif doit avoir une fréquence < $1{,}3.10^{-5}$/flt »*
- Pour le **commutateur** :
  - *« Le blocage du commutateur sur la position de l'unité de calcul BSCU1 doit avoir une fréquence < $1{,}48.10^{-2}$/flt »*
  - *« Le blocage du commutateur dans une position intermédiaire doit avoir une fréquence < 5.10^{-6}/flt »*
  - *« Le blocage du commutateur sur la position de l'unité de calcul BSCU2 doit avoir une fréquence < $1{,}48.10^{-2}$/flt »*
- Pour le **CPU du BSCU1** :
  - *« La défaillance électronique du BSCU1 doit avoir une fréquence < $3{,}5.10^{-4}$/flt »*
- Pour le **CPU du BSCU2** :
  - *« La défaillance électronique du BSCU2 doit avoir une fréquence < $3{,}5.10^{-4}$/flt »*

- Pour la **ressource électrique du BSCU1** :
  - *« La panne d'alimentation électrique du BSCU1 doit avoir une fréquence < 1,5.10^{-4}/flt »*
- Pour la **ressource électrique du BSCU2** :
  - *« La panne d'alimentation électrique du BSCU2 doit avoir une fréquence < 1,5.10^{-4}/flt »*

L'exigence de sûreté système « le freinage involontaire en raison d'une panne du calculateur BSCU doit avoir une fréquence < 2,5.10^{-9}/flt » est déclinée au niveau sous-systèmes en :

- Pour le **commutateur** :
  - *« Le blocage du commutateur sur la position de l'unité de calcul BSCU2 doit avoir une fréquence < 1,48.10^{-2}/flt »*
- Pour le **moniteur de puissance du BSCU1** :
  - *« Le blocage dans l'état valide du système de surveillance de l'alimentation électrique du BSCU1 doit avoir une fréquence < 2.10^{-2}/flt »*
- Pour le **moniteur de puissance du BSCU2** :
  - *« Le blocage dans l'état valide du système de surveillance de l'alimentation électrique du BSCU2 doit avoir une fréquence < 2.10^{-2}/flt »*
- Pour le **moniteur système du BSCU1** :
  - *« Le système de surveillance de validité du BSCU1 qui est défaillant dans l'état valide en raison d'une erreur matérielle doit avoir une fréquence < 2.10^{-2}/flt »*
  - *« L'existence d'erreur de conception du moniteur du BSCU1 n'a pas de contrainte concernant la fiabilité »*
- Pour le **moniteur système du BSCU2** :
  - *« Le système de surveillance de validité du BSCU2 qui est défaillant dans l'état valide en raison d'une erreur matérielle doit avoir une fréquence < 2.10^{-2}/flt »*
  - *« L'existence d'erreur de conception du moniteur du BSCU1 n'a pas de contrainte concernant la fiabilité »*
- Pour la **ressource électrique du BSCU1** :
  - *« La panne d'alimentation électrique du BSCU1 entraînant des données erronées doit avoir une fréquence < 5.10^{-8}/flt »*
- Pour la **ressource électrique du BSCU2** :
  - *« La panne d'alimentation électrique du BSCU2 entraînant des données erronées doit avoir une fréquence < 5.10^{-8}/flt »*
- Pour le **CPU du BSCU1** :
  - *« La défaillance matériel du CPU du BSCU1 entraînant des données erronées doit avoir une fréquence < 3,13.10^{-8}/flt »*
  - *« L'erreur de conception des fonctions du CPU du BSCU1 entraînant des données erronées n'a pas de contrainte de fiabilité »*
  - *« La défaillance I/O du BSCU1 entraînant des données erronées doit avoir une fréquence < 1,88.10^{-8}/flt »*
- Pour le **CPU du BSCU2** :

- *« La défaillance matériel du CPU du BSCU2 entraînant des données erronées doit avoir une fréquence < 3,13.10^{-8}/flt »*
- *« L'erreur de conception des fonctions du CPU du BSCU2 entraînant des données erronées n'a pas de contrainte de fiabilité »*
- *« La défaillance I/O du BSCU2 entraînant des données erronées doit avoir une fréquence < 1,88.10^{-8}/flt »*

Remarque : cette deuxième décomposition refait intervenir une exigence au niveau du commutateur, concernant son blocage sur la position du BSCU2, déjà présente dans la première décomposition. Ce genre de configuration est tout à fait acceptable, du moment que les exigences ne sont pas incohérentes. Il s'agit en fait d'accorder les contraintes de sûreté ou de prendre en compte la contrainte la plus sévère. La détection de l'incohérence potentielle et l'accord fait entre les exigences se gèrent lors des phases d'analyse et de validation des exigences.

4. Modélisation SysML

La section qui suit présente la modélisation SysML de l'exemple de l'avion S18 pour les 3 niveaux de *building block* étudiés précédemment (avion S18, système de freinage, calculateur BSCU) en s'intéressant plus particulièrement à la fonction de freinage.

Nous rappelons que nous souhaitons modéliser, en respectant le modèle d'information défini dans le chapitre 4, afin de partager les données d'ingénierie sur une base commune lors de l'ingénierie du système, ceci entre les différents participants à la conception (particulièrement entre les ingénieurs système et les ingénieurs de sûreté de fonctionnement pour ce qui concerne notre problématique). Nous verrons que l'essentiel de la modélisation présenté ici s'attache à la définition des exigences et aux liens de traçabilité.

4.1. Préparation du projet sous Tau-G2

L'outil que nous avons choisi d'utiliser pour modéliser en SysML est l'outil Tau G2, qui est disponible au LAAS-CNRS. Autrefois développer par la société Telelogic, Tau G2 appartient maintenant à IBM. L'objectif principal de Tau G2 est d'offrir un environnement de modélisation UML, avec des possibilités de simulation et de génération de codes. Toutefois, le méta-modèle de SysML a été implanté dans l'outil pour permettre également de modéliser en langage SysML.

Tout d'abord, avant de commencer la modélisation, il s'agit de préparer le projet sous TauG2. Plus que la création même d'un nouveau projet, ce que nous présentons ici correspond à l'extension nécessaire de SysML à définir et à l'organisation en packages proposée pour notre projet.

4.1.1. Extension de SysML

Concernant l'extension de SysML, nous avons créé un nouveau package appelé « *ExtensionSysML* ». Celui-ci contient alors tous les nouveaux stéréotypes à ajouter à SysML, tels que proposés dans la section 3.2 du chapitre 4. La visualisation de ces stéréotypes et de leurs relations est rendue visible à travers un diagramme présenté en Figure V.15.

Dans cette figure, les nouveaux stéréotypes d'exigences apparaissent : « *acquirerReqt* », « *otherStakeholderReqt* », « *systemTechnicalReqt* », « *specifiedReqt* » et « *safetyReqt* », ainsi que la définition du bloc « *risk* ». Les définitions des deux nouveaux liens de traçabilité sont également visibles : lien « *treat* », pour une exigence de sûreté qui traite un risque, et lien « *specify* », pour une exigence spécifiée qui spécifie un élément de la solution.

4.1.2. Organisation en packages

Concernant l'organisation du projet, un package est défini pour chacun des trois niveaux de *building block* étudié (voir Figure V.16) :

- **« AvionS18 »**,
- **« Système FreinsDesRoues »**,
- **« BSCU »**.

Pour rappel, la notion de building block provient de la norme EIA-632 que nous utilisons, présentée au chapitre 2.

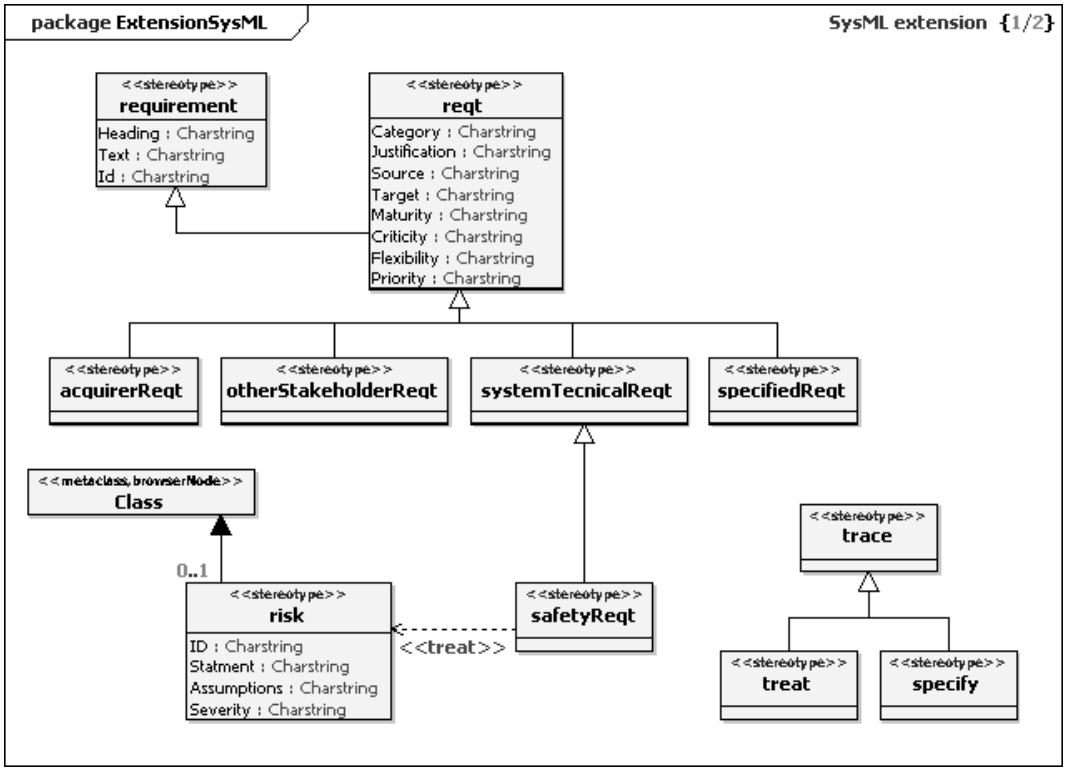


Figure V.15 : Extension de SysML

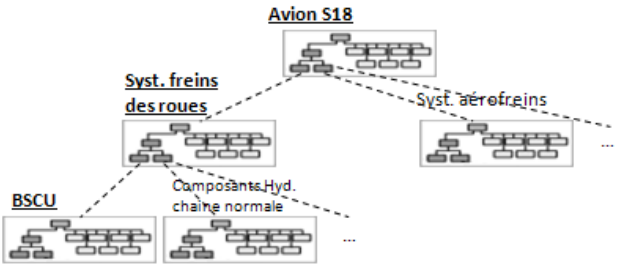


Figure V.16 : Building blocks

Puis, pour chacun de ces niveaux, une structure en trois (autres) packages permet d'organiser la modélisation. Il s'agit de suivre le principe de bonne conception présenté au paragraphe 2.5 du chapitre 4, qui stipulait de séparer les concepts d'exigences, de solution de conception et de vérification et validation dans la modélisation. Ainsi, les trois packages sont (voir Figure V.17) :

- « **Requirements** »,
- « **Design Solution** »,
- « **V&V** ».

Le package « *Requirements* » accède aux deux autres packages : « *Design Solution* » et « V&V », et importe l'extension de SysML contenu dans le package « *ExtensionSysML* » pour l'utiliser.

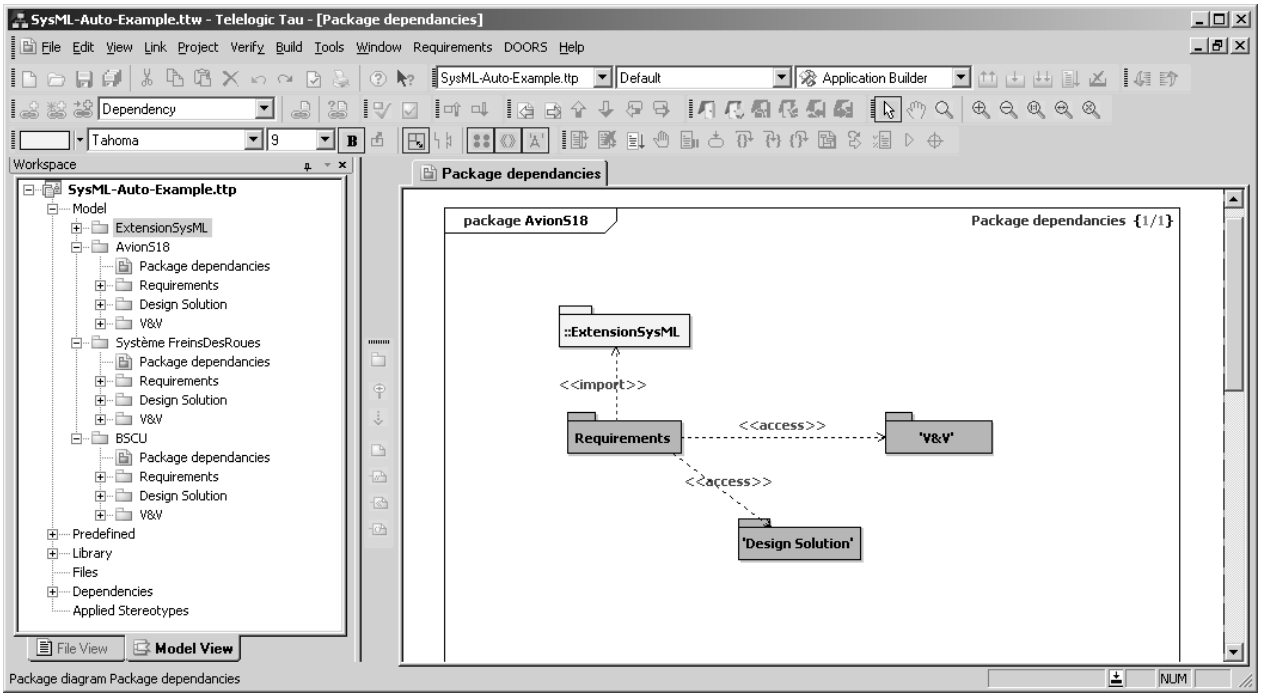


Figure V.17 : Organisation en packages

Note : afin de faciliter la compréhension de l'exemple, nous présentons par la suite les packages dans l'ordre suivant : « Design Solution », « Requirements » et « V&V ». Ceci se justifie par le fait que le package « Requirements » fait référence à beaucoup d'éléments contenus dans le package « Design Solution ». Pour autant, cet ordre de présentation n'est absolument pas représentatif de la temporalité de création des éléments du modèle.

4.2. Niveau « avion S18 »

Suite à la préparation du projet, nous pouvons commencer la modélisation du système. Nous commençons par le *building block* « avion S18 », en présentant successivement les différents diagrammes présents dans les différents packages annoncés ci-dessus.

4.2.1. Package de la solution de conception

Le premier package présenté est celui de la solution de conception (« Design Solution »). Celui-ci contient deux diagrammes :

1. **Un diagramme de cas d'utilisation**, présentant les fonctions du système,
2. **Un diagramme de block**, pour la structure du système.

4.2.1.1. Fonctions du système

La Figure V.18 présente les fonctions du système à travers un diagramme de cas d'utilisation. En plus de montrer les acteurs du système dans ce diagramme, ce sont bien entendu les cas d'utilisation qui sont représentatifs des fonctions. Une structure hiérarchique entre ces fonctions est réalisée à l'aide de relations « *include* ».

Par exemple, la fonction « *décélérer l'avion au sol* » est une sous-fonction de la fonction « *contrôler l'avion au sol* », qui elle-même est une sous-fonction de la fonction « *piloter l'avion* ».

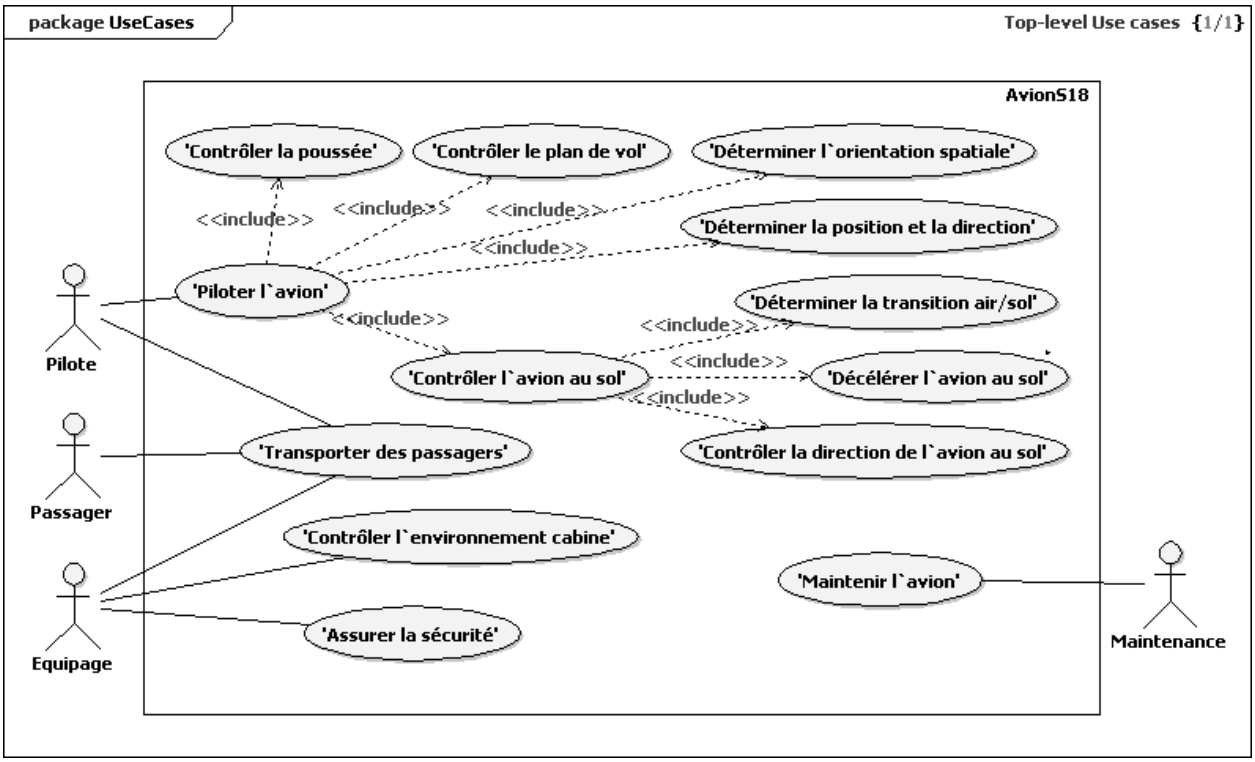


Figure V.18 : Avion S18 – Fonctions du système

4.2.1.2. Structure du système

Une partie de la structure du système « avion S18 » est représentée en Figure V.19 dans un diagramme de blocks. Nous retrouvons le fait que « l'avion S18 » est composé de :

- Une structure,
- Un système de poussée,
- Un système de gouvernes,
- Un système d'aérofreins,
- Un système d'inverseurs de poussée,
- Un système de freins des roues.

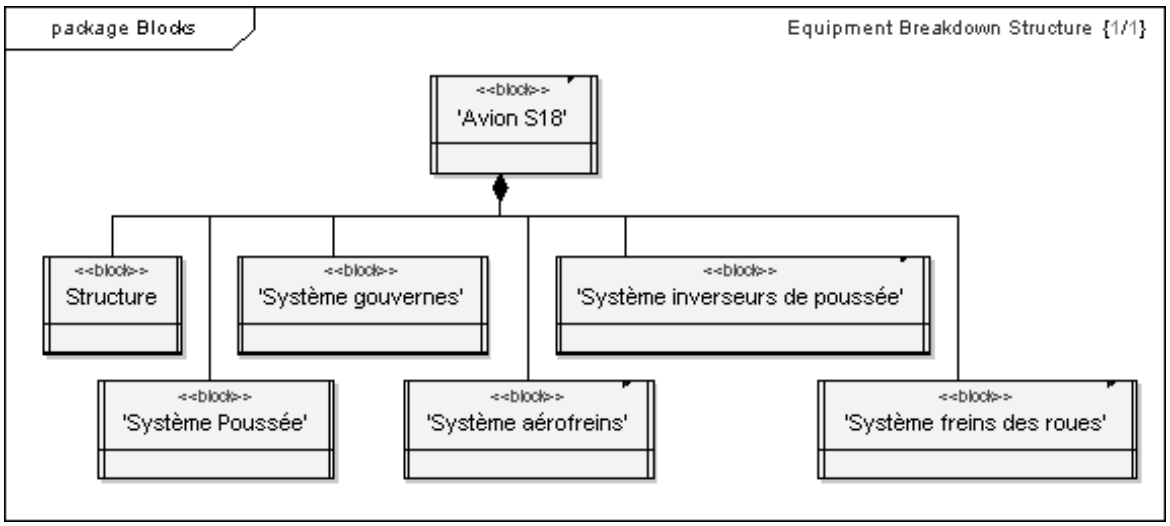


Figure V.19 : Avion S18 – Structure du système

4.2.2. Package des exigences

Le second package présenté est celui des exigences (« Requirements »). Nous avons choisi de créer quatre diagrammes d'exigences pour :

1. **Les exigences haut-niveau**, pour présenter les exigences au niveau du système, « l'avion S18 » pour ce premier *building block*, et lier ces exigences aux risques qu'elles traitent par la relation « *treat* »,
2. **La déclinaison des exigences**, pour montrer la déclinaison des exigences système en exigences sous-système par la relation de composition,
3. **La satisfaction des exigences**, pour indiquer les éléments du système (système ou sous-systèmes) qui satisfont les exigences par la relation « *satisfy* »,
4. **La vérification des exigences**, pour désigner les éléments de V&V qui vérifient les exigences par la relation « *verify* ».

4.2.2.1. Diagramme des exigences haut-niveau

La Figure V.20 montre les exigences haut-niveau de l'avion S18. Seulement deux risques apparaissent : « sortie de piste haute vitesse » et « sortie piste sur coté ». Leurs sévérités sont comprises entre deux valeurs, ceci est dû à une simplification de modélisation

que nous faisons pour ne pas alourdir le modèle : par exemple pour ne pas multiplier le nombre de risques en fonction des phases atterrissage/taxi/décollage annulé.

Dans ce diagramme, nous retrouvons par exemple l'exigence de sûreté système « *la perte non-annoncée de la capacité de décélération doit avoir une fréquence <5.10⁻⁹/flt* » identifiée lors de l'AMDEC du système avion S18 (voir paragraphe 3.1.1). Cette exigence, dont l'identifiant est « *R2* », permet de traiter le risque « *Risk_1* » : « *l'avion sort de la piste à haute vitesse et peut rentrer en contact avec des obstacles* ».

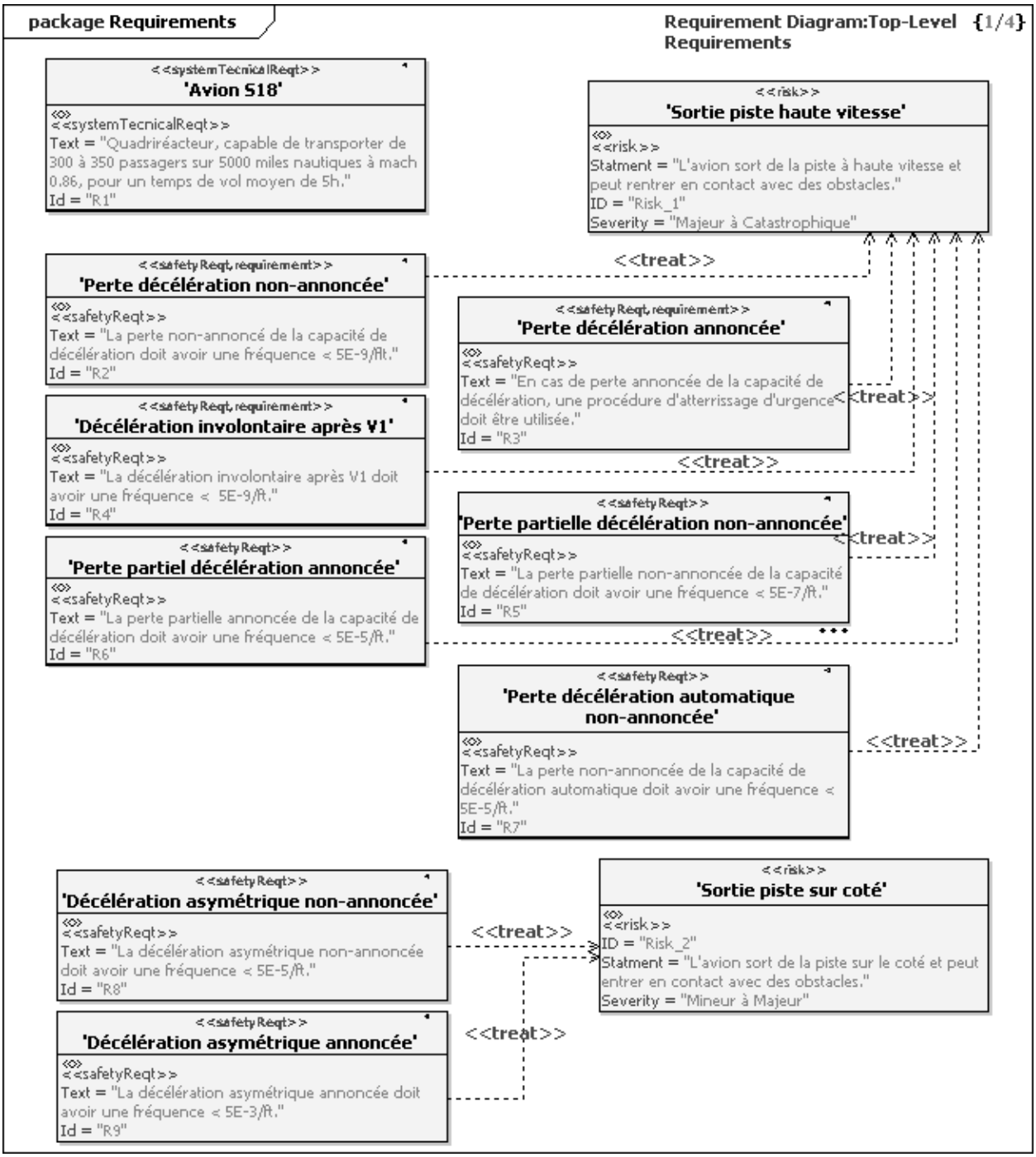


Figure V.20 : Avion S18 – Diagramme des exigences haut-niveau

4.2.2.2. Diagramme de déclinaison des exigences

La Figure V.21 montre la déclinaison des exigences haut-niveau (système avion S18) en exigences sous-systèmes.

Nous retrouvons ici les deux déclinaisons synthétisées au paragraphe 0 concernant les exigences système « *perte décélération non-annoncée* » et « *décélération involontaire après V1* », qui se déclinent au niveau des sous-systèmes : inverseurs de poussée, frein des roues et aérofreins.

Note : Concernant les identifiants des exigences (champ Id), nous avons choisis de réutiliser l'identifiant de l'exigence mère pour les exigences déclinées. Par exemple, si l'identifiant de l'exigence mère est « R2 », alors la première exigence déclinée aura comme identifiant « R2.1 ».

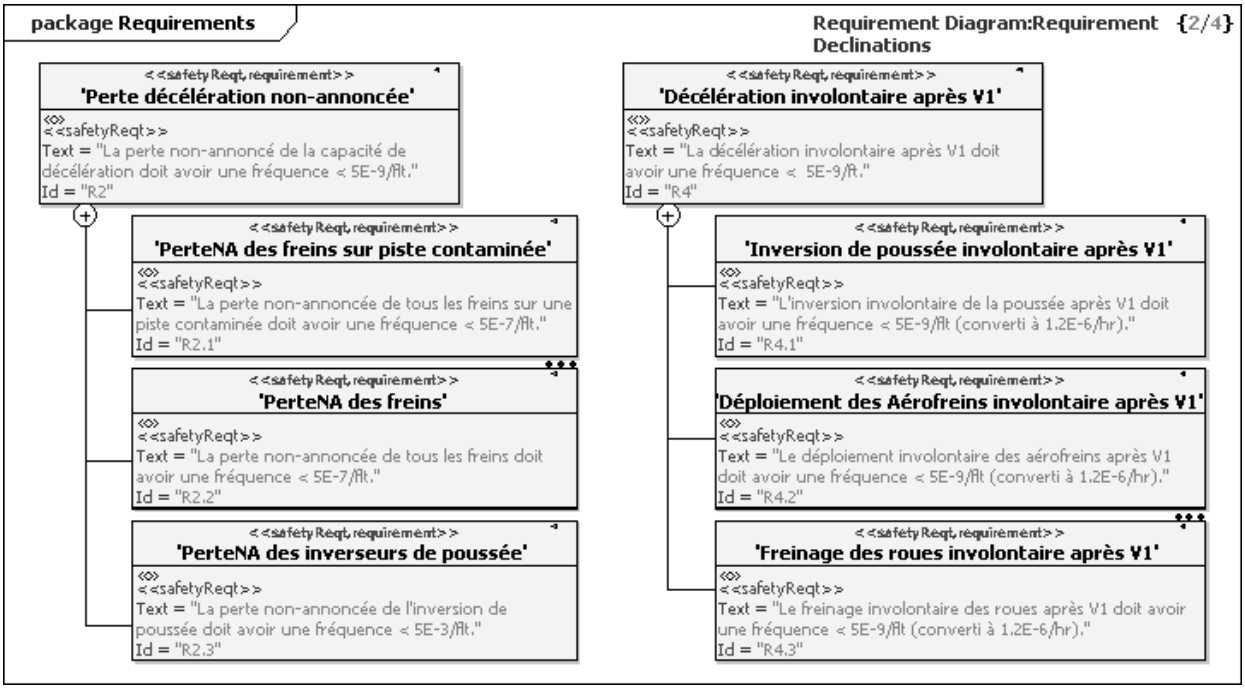


Figure V.21 : Avion S18 – Diagramme de déclinaison des exigences

4.2.2.3. Diagramme de satisfaction des exigences

La Figure V.22 montre les affectations des exigences aux éléments de la solution de conception, ceci à travers le lien « *satisy* » de SysML. Ces éléments peuvent être tous des éléments du langage SysML autres que des exigences, mais dans notre modélisation, ceux-ci correspondent à des blocks (représentatifs de systèmes) ou des usecases (représentatifs de fonctions).

Concernant les deux déclinaisons d'exigences précédentes, les exigences système sont très logiquement liées à des éléments système, qui ici correspondent aux fonctions du système « avion S18 » (modéliser à l'aide de *usecase*). Quant aux exigences sous-système, celles-ci sont associées à des *blocks* représentant des sous-systèmes (freins des roues, inverseurs de poussée, aérofreins).

En bas du diagramme, une série d'exigences a été simplement liée au block « avion S18 » représentant le système. Pour une modélisation complète, ces exigences devraient également être déclinées. Mais pour limiter la taille de l'exemple, nous n'étudions pas en profondeur ces exigences.

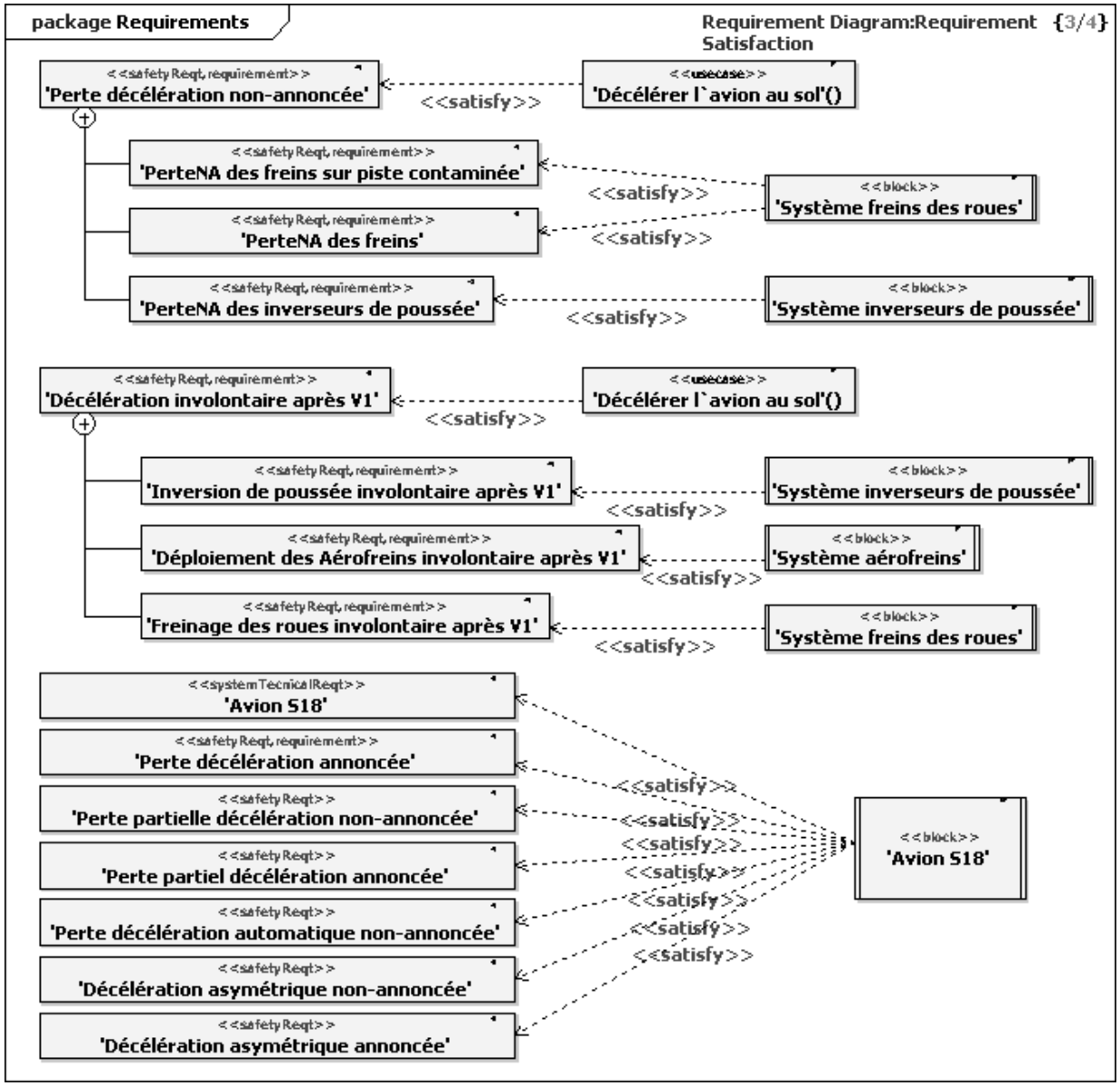


Figure V.22 : Avion S18 – Diagramme de satisfaction des exigences

4.2.2.4. Diagramme de vérification des exigences

La Figure V.23 montre les relations de vérification entre les opérations de V&V (qui sont stéréotypés par *TestCase*) et les exigences système. Pour l'exemple, seul les deux exigences système à la base des déclinaisons ont été considérées dans ce diagramme. Les opérations de V&V correspondent alors à des analyses de SdF. Elles sont définies dans le package « *V&V* » présenté ci-après.

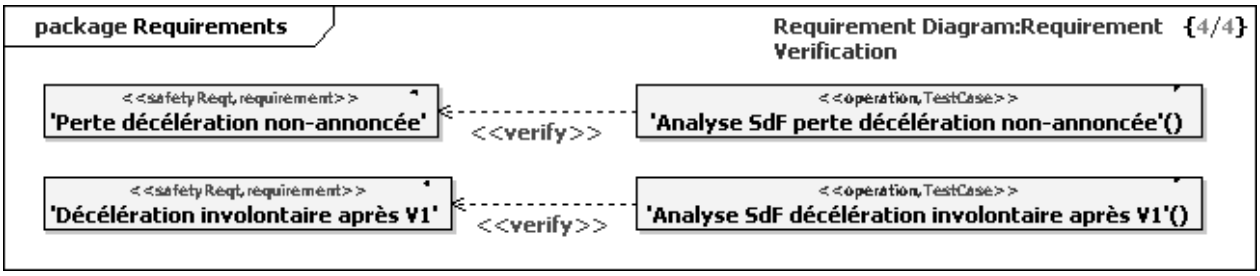


Figure V.23 : Avion S18 – Diagramme de vérification des exigences

Ainsi, l'exigence nommée « *perte décélération non-annoncée* » sera vérifiée, d'après le diagramme, par une opération nommée « *analyse SdF perte décélération non-annoncée* ».

4.2.3. Package de V&V : les cas de tests

Le dernier package à présenter est celui de V&V, qui renferme les cas de tests. Celui-ci contient le diagramme de la Figure V.24, qui définit les deux opérations de V&V entrevues au paragraphe 4.2.2.4. A ces opérations, nous associons des commentaires dans le diagramme des *TestCases* afin de les spécifier. Comme nous pouvons le voir sur la Figure V.24, ces commentaires sont les suivants :

- Pour l'opération « **analyse SdF perte décélération non-annoncée** » : analyser avec l'arbre de défaillance de l'événement racine 'perte décélération non-annoncée' (cf. Figure V.5),
- Pour l'opération « **analyse SdF décélération involontaire après V1** » : analyser avec l'arbre de défaillance de l'événement racine 'décélération involontaire après V1' (cf. Figure V.6).

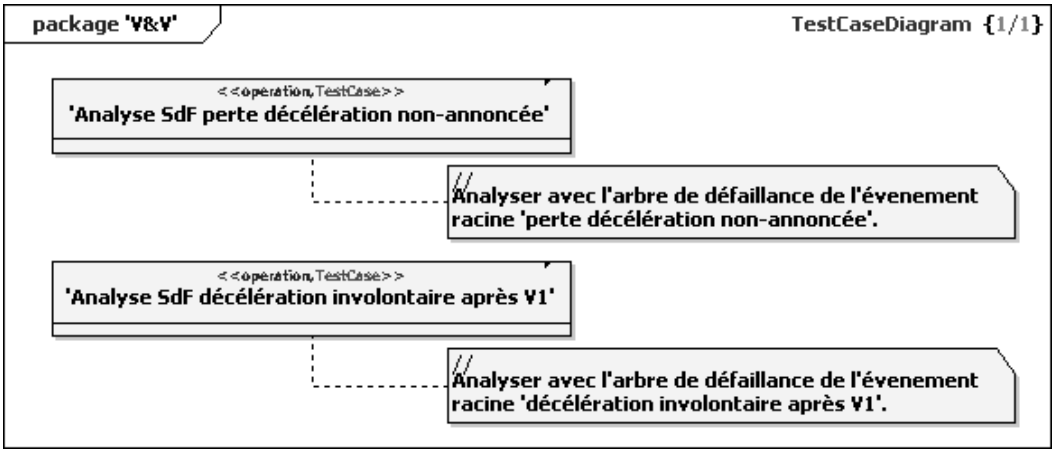


Figure V.24 : Avion S18 – Cas de tests

4.3. Niveau « freins des roues »

Le second *building block* étudié est celui des « freins des roues ». Nous présentons également les différents diagrammes présents dans les trois packages (solution de conception, exigences, V&V) de ce *building block*.

4.3.1. Package de la solution de conception

Le premier package présenté est celui de la solution de conception (« Design Solution »), présentant les fonctions et la structure du système.

4.3.1.1. Fonctions du système

La Figure V.25 présente les fonctions du système « freins des roues » à travers un diagramme de cas d'utilisation. Les différentes fonctions retrouvées ici correspondent à :

- Le freinage de l'avion au sol, qui peut être manuel ou automatique et qui inclut une fonction d'anti-dérapage,
- Le contrôle de la direction au sol avec un freinage différentiel,
- Le maintien immobile de l'avion quand celui-ci est parqué,
- L'arrêt de la rotation des roues lors de la rétractation des trains après un décollage,
- La transmission d'informations concernant le freinage et l'état général du système de freinage des roues.

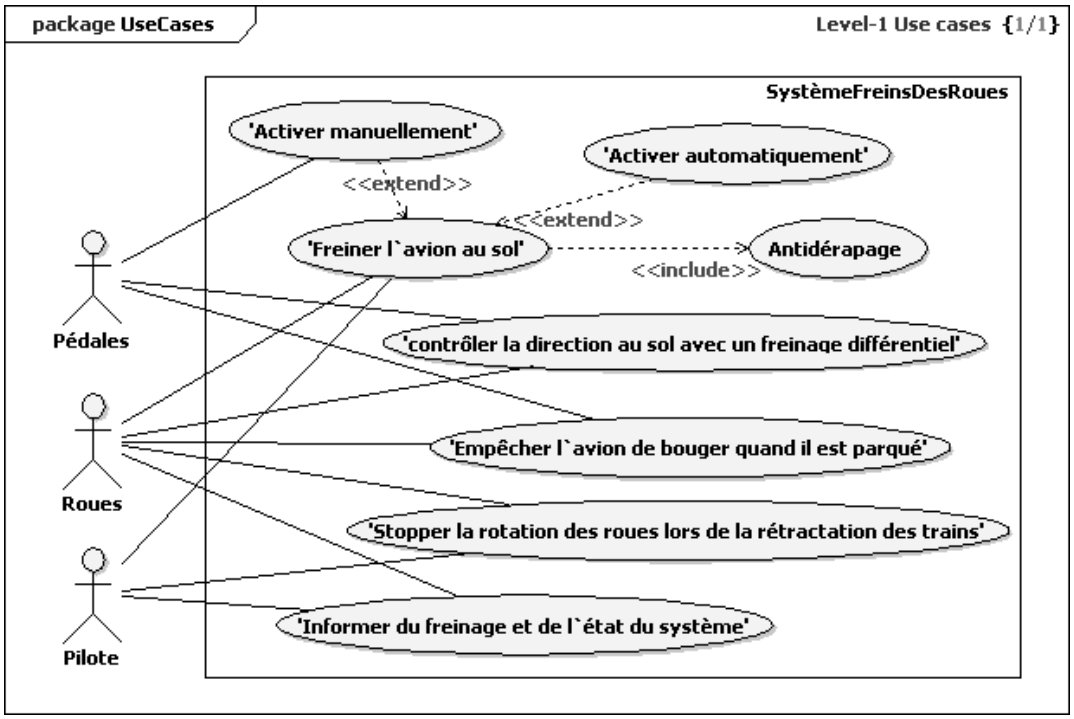


Figure V.25 : Freins des roues – Fonctions du système

4.3.1.2. Structure du système

La structure du système « freins des roues » est représentée en Figure V.26 dans un diagramme de blocks. Nous retrouvons le fait que le système « freins des roues » est composé de :

- Une ressource électrique,
- Un calculateur BSCU
- Un système d'annonciation,
- Une chaine normale, incluant : une ressource et des composants hydrauliques,

- Une chaine alternative, incluant : une ressource et des composants hydrauliques, ainsi qu'un accumulateur.

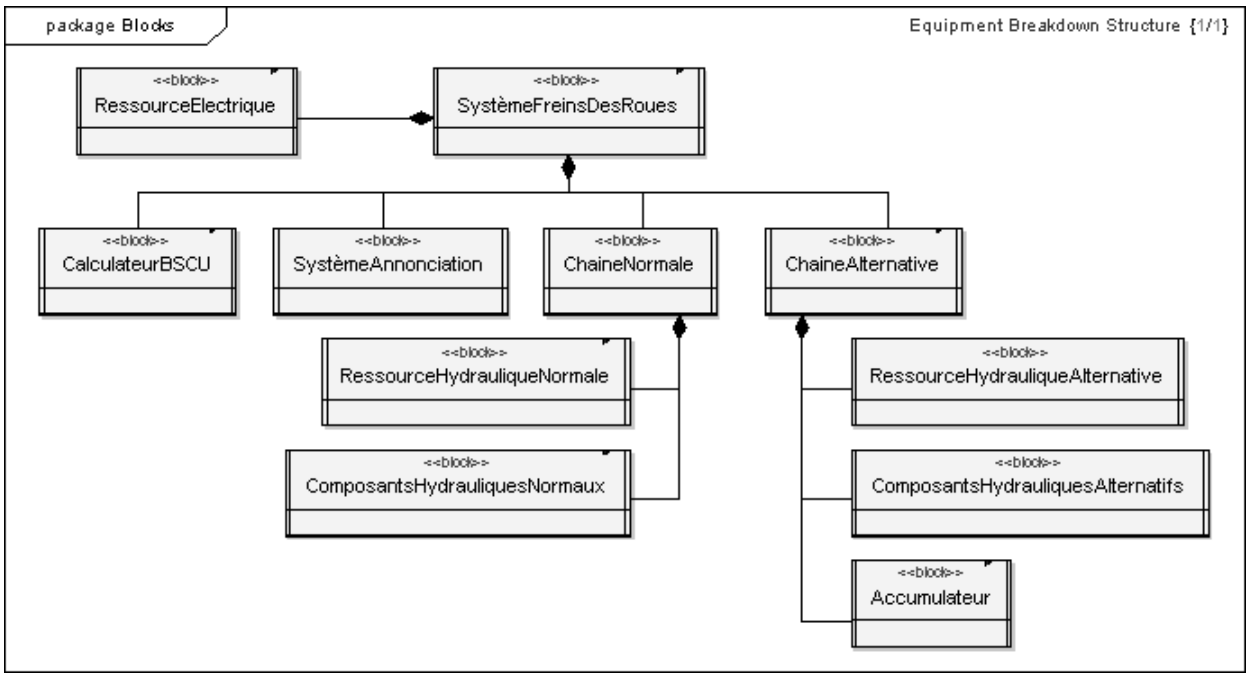


Figure V.26 : Freins des roues – Structure du système

La Figure V.27 propose un diagramme de structure composite pour le système de freins des roues, dans lequel les relations entre les différents composants sont visibles.

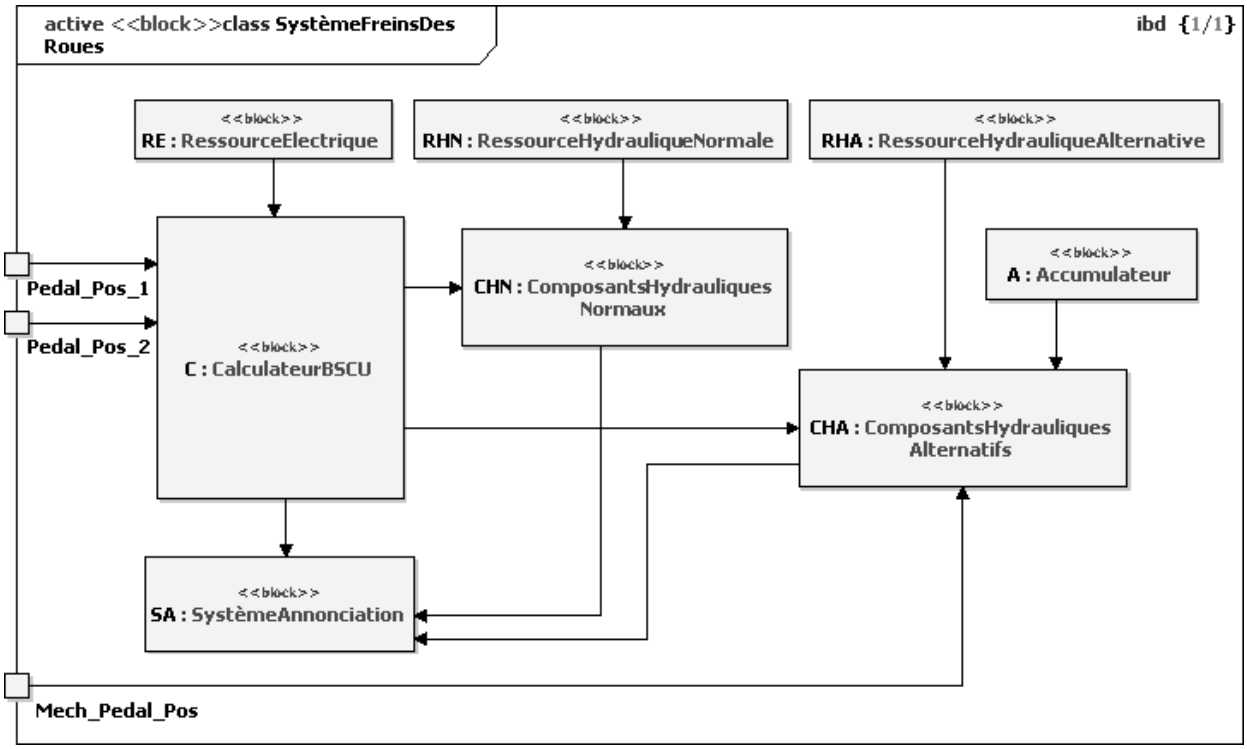


Figure V.27 : Freins des roues – Structure du système (2)

4.3.2. Package des exigences

Le second package présenté est celui des exigences (« *Requirements* »).

4.3.2.1. Diagramme des exigences haut-niveau

La Figure V.28 montre les exigences haut-niveau du système de freins des roues. Dans ce diagramme, les risques traités par les exigences de sûreté haut-niveau apparaissent aussi. Ceux-ci sont :

- « **Sortie piste haute vitesse** » : L'avion sort de la piste à haute vitesse et peut rentrer en contact avec des obstacles.
- « **Haute température** » : La température sur les roues atteint le point où la défaillance du pneu/roue se produit. Feu des pneus possible.
- « **Sortie piste sur coté** » : L'avion sort de la piste sur le coté et peut entrer en contact avec des obstacles.
- « **Avion stoppé sur piste** » : L'avion est stoppé sur la piste.
- « **Eclatement pneus** » : Eclatement potentiel de tous les pneus et perte de l'efficacité du freinage.

Dans ce diagramme, nous retrouvons par exemple l'exigence de sûreté système « *la perte non-annoncée de tous les freins doit avoir une fréquence* $< 5.10^{-7}/flt$ » identifiée lors de l'AMDEC du sous-système « freins des roues » (voir paragraphe 3.2.1). Cette exigence, dont l'identifiant est « *R2.1* », permet de traiter le risque « *Risk_1* » : « *l'avion sort de la piste à haute vitesse et peut rentrer en contact avec des obstacles* ».

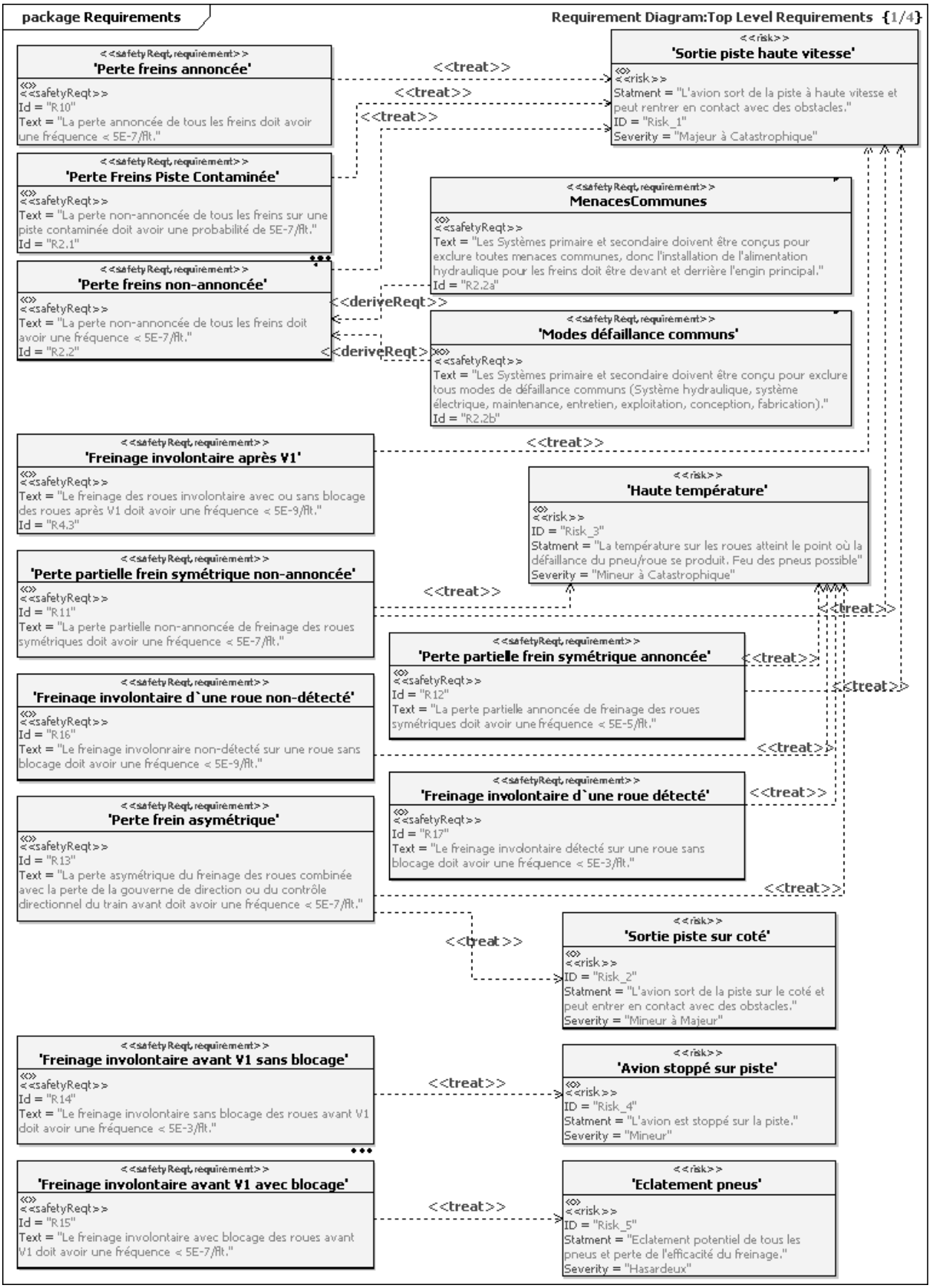


Figure V.28 : Freins des roues – Diagramme des exigences haut-niveau

4.3.2.2. Diagramme de déclinaison des exigences

La Figure V.29 montre la déclinaison des exigences haut-niveau (système freins des roues) en exigences sous-systèmes.

Nous retrouvons ici la déclinaison synthétisée au paragraphe 3.2.4 concernant l'exigence sous-système « *perte freins non-annoncée* ».

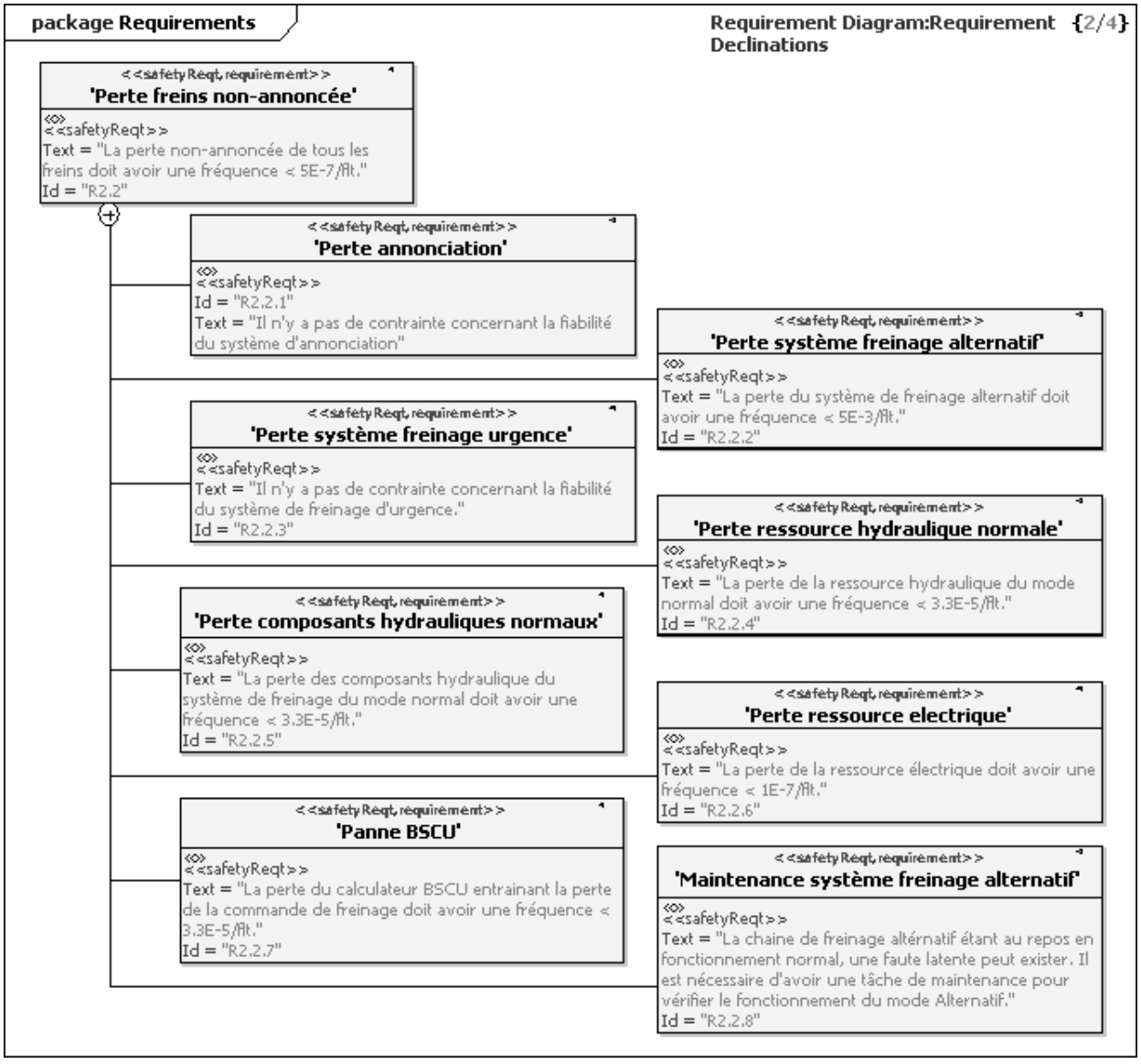


Figure V.29 : Freins des roues – Diagramme de déclinaison des exigences

4.3.2.3. Diagramme de satisfaction des exigences

La Figure V.30 montre les affectations des exigences aux éléments de la solution de conception (*block* dans notre exemple), ceci à travers le lien « *satisy* » de SysML.

Dans ce diagramme, nous avons choisi de ne représenter que les affectations des exigences concernées par la déclinaison présentée au paragraphe précédent. Ainsi, l'exigence haut-niveau est liée au système « freins des roues ». Quant aux autres exigences,

elles sont connectées aux sous-systèmes qui les concernent, comme illustré par la Figure V.30.

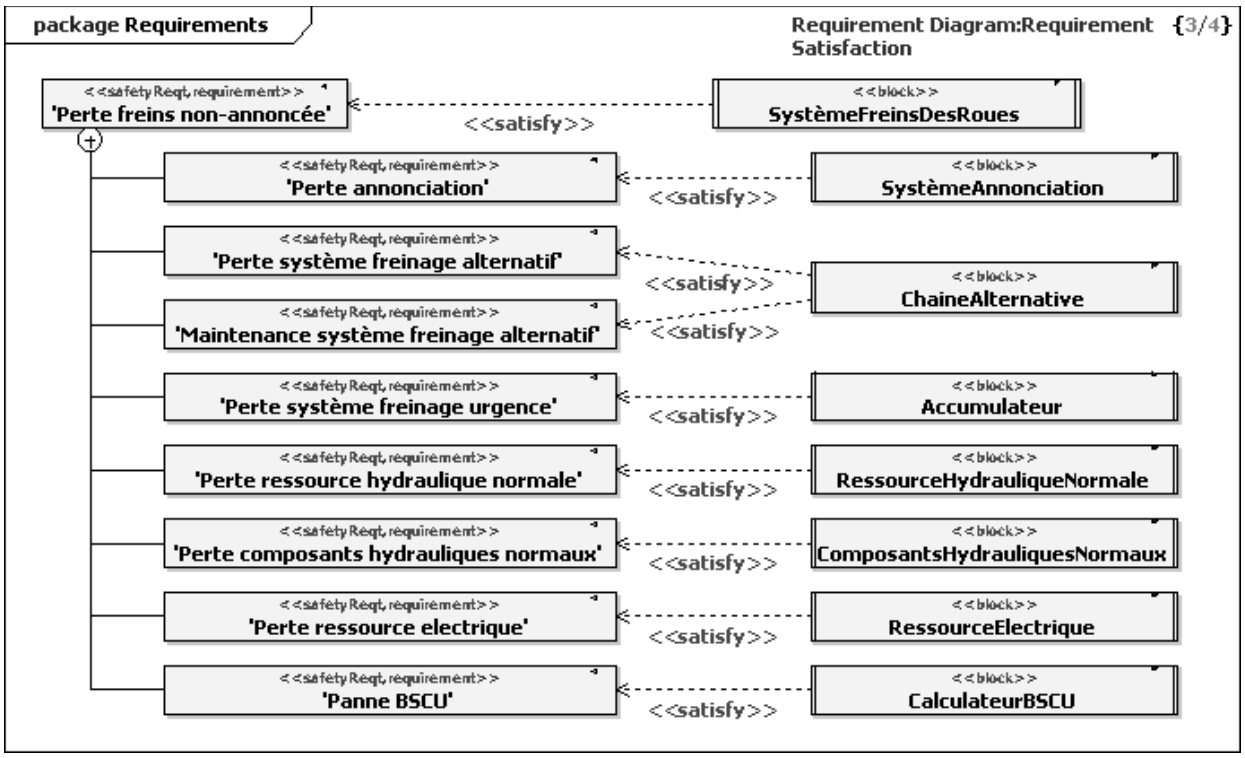


Figure V.30 : Freins des roues – Diagramme de satisfaction des exigences

4.3.2.4. Diagramme de vérification des exigences

La Figure V.31 montre les relations de vérification entre les opérations de V&V (qui sont stéréotypés par *TestCase*) et les exigences système. Dans cette figure, seule l'exigence système à la base de la déclinaison a été considérée. L'opération de V&V correspond alors à une analyse de SdF. Elle est définie dans le package « *V&V* » présenté ci-après.

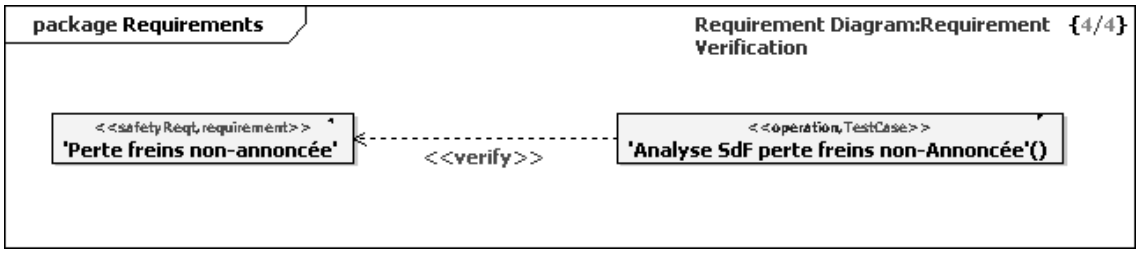


Figure V.31 : Freins des roues – Diagramme de vérification des exigences

Ainsi, l'exigence nommée « *perte freins non-annoncée* » sera vérifiée, d'après le diagramme, par une opération nommée « *analyse SdF perte freins non-annoncée* ».

4.3.3. Package de V&V : les cas de tests

Le dernier package à présenter est celui de V&V, qui renferme les cas de tests. Celui-ci contient le diagramme de la Figure V.32, qui définit l'opération de V&V entrevue au paragraphe 4.3.2.4. A cette opération, nous associons un commentaire dans le diagramme

des *TestCases* afin de la spécifier. Comme nous pouvons le voir sur la Figure V.32, ce commentaire est le suivant :

- Pour l'opération « **analyse SdF perte freins non-annoncée** » : analyser avec l'arbre de défaillance de l'événement racine 'perte des freins non-annoncée' (cf. Figure V.8).

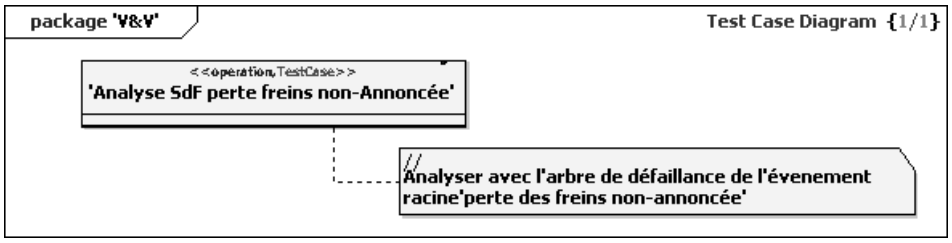


Figure V.32 : Freins des roues – Cas de tests

4.4. Niveau « calculateur BSCU »

Le troisième et dernier *building block* que nous avons étudié est celui du « calculateur BSCU ». La même organisation que pour les deux *building blocks* précédents a été faite. Etant donné que le travail réalisé est similaire, nous ne présentons pas ce niveau dans le chapitre. Néanmoins, tous les diagrammes sont fournis en annexe B de ce mémoire.

5. Synthèse de l'étude de cas

Pour résumer et faire une courte synthèse sur le cas d'étude présenté dans ce chapitre 5, nous avons donc réussi à appliquer successivement, trois fois la méthodologie de déclinaison des exigences de sûreté de fonctionnement définie au chapitre 3, ceci pour trois *building blocks* successifs : l'avion S18, le sous-système de freins des roues et le calculateur BSCU. Nous avons alors identifié et défini :

- Deux exigences de sûreté au niveau « avion S18 », déclinées au niveau des sous-systèmes : freins des roues, inverseurs de poussée et aérofreins.
- Une exigence de sûreté au niveau « freins des roues », déclinée au niveau du système d'annonciation, du système de freinage alternatif (non déclinée d'avantage), du système de freinage d'urgence (idem), de la ressource hydraulique normale, des composants hydrauliques normaux, de la ressource électrique et du calculateur BSCU.
- Deux exigences de sûreté au niveau « calculateur BSCU », déclinées au niveau du moniteur général, du commutateur, des CPUs, des ressources électriques, des moniteurs de puissance et des moniteurs système.

Nous avons également modélisé l'exemple à l'aide du langage SysML augmenté de notre extension afin de respecter le modèle d'information défini au chapitre 4. Nous avons ainsi illustré notre proposition pour intégrer des aspects de sûreté dans SysML, tel que les exigences de sûreté ou les risques. Un aspect important qui ressort de ce modèle d'information, qui pourra être utilisé autant par les ingénieurs système que ceux de sûreté, est la traçabilité. Pour donner un exemple, la Figure V.33 synthétise une partie des

déclinaisons d'exigences de sûreté, dont toutes les informations (traçabilité incluse) sont accessibles dans le modèle d'information. Sur cette figure, nous avons replacé les identifiants des exigences du modèle SysML au niveau des systèmes qui les concernent. En considérant le système général « avion S18 », les systèmes ont été catégorisés en : « avion », « systèmes », « équipements » ou « composants ». D'après la figure, nous pouvons par exemple retrouver le fait que l'exigence R2.2.7.5 associée au CPU1 provient de l'exigence R2.2.7 du BSCU, qui venait de l'exigence R2.2 du système de freins des roues, qui elle-même était résultat de la déclinaison de l'exigence R2 associée au système avion S18.

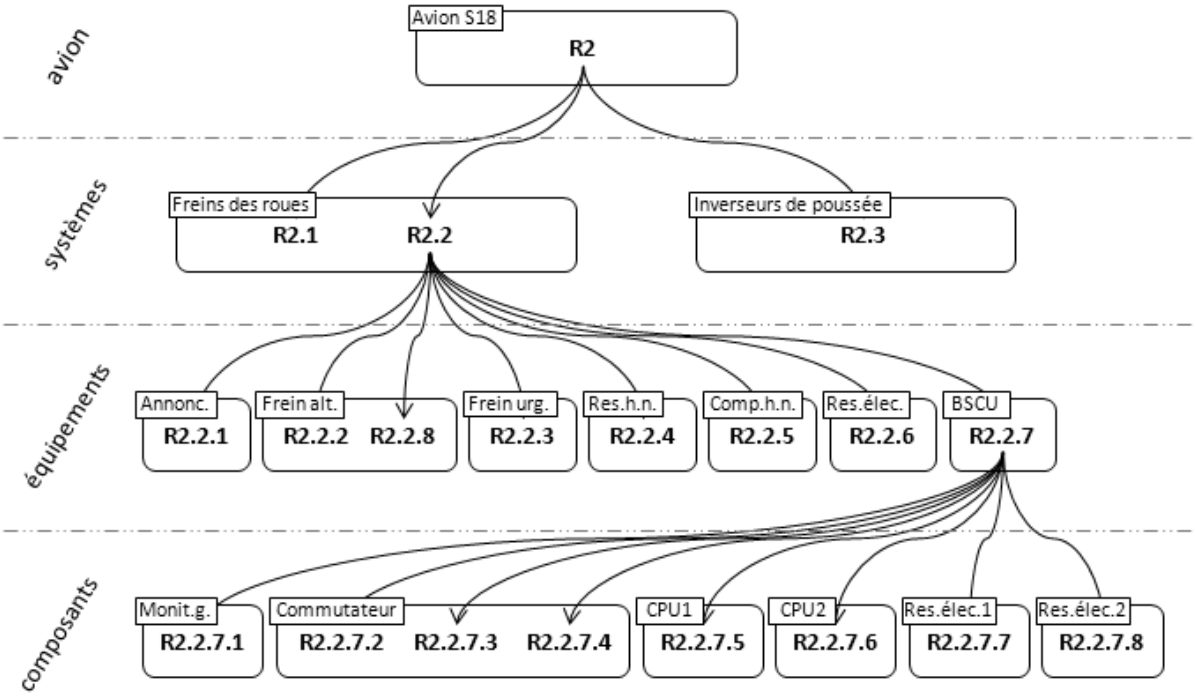


Figure V.33 : Synthèse d'une partie des déclinaisons

6. Conclusion

Ce chapitre 5, dernier chapitre du mémoire, a donc permis d'illustrer les propositions du chapitre 3, concernant la méthodologie de déclinaison d'exigences de sûreté de fonctionnement, et du chapitre 4, concernant l'utilisation d'un modèle d'information en SysML, ceci dans le contexte proposé au chapitre 2.

Nous avons montré la faisabilité de notre méthodologie de déclinaison d'exigence de sûreté sur un exemple complexe, ceci au travers de différents niveaux de *building blocks*. Nous avons bien vu qu'en combinant des AMDECs et des analyses par arbres de défaillances, nous avons réussi à définir des exigences de sûreté et à identifier les déclinaisons des exigences de sûreté associées au système en exigences de sûreté associées aux sous-systèmes. Comme nous l'avons vu en conclusion du chapitre 1, cette traçabilité de la sûreté de fonctionnement est extrêmement importante pour une vérification et une validation correcte du système, ainsi que pour des analyses d'impacts suite à un changement d'exigences.

De surcroit, la section sur la modélisation SysML (incluant notre extension au langage) a montré les possibilités de ce langage pour une modélisation respectant notre modèle d'information. La gestion de la traçabilité est parfaitement visible et les concepts d'exigences, de solution de conception et de V&V sont séparés en respectant une organisation claire du projet. La visualisation directement dans le modèle des risques et des exigences de sûreté renforce la modélisation.

Conclusion Générale

Pour ce travail de thèse, nous nous sommes intéressés à la sûreté de fonctionnement des systèmes complexes et à sa prise en compte dans la conception de ces systèmes. Une analyse des origines des problèmes de sûreté de fonctionnement des systèmes complexes a été réalisée. De nombreux points sont à l'origine de ces problèmes, tel que de mauvaises définitions ou formulations des exigences, une mauvaise ou absence de traçabilité, une absence d'un langage commun entre les différents corps de métiers, etc. Il nous a alors paru essentiel de définir une approche globale de prise en compte de la sûreté de fonctionnement au niveau processus d'ingénierie système pour apporter des réponses aux faiblesses identifiées. Le cadre de processus d'ingénierie système parait évident pour traiter de la conception des systèmes complexes. Par nature, les propriétés de sûreté de fonctionnement des systèmes complexes nécessitent un point de vue général sur le système. Intégrer leur gestion dans un tel cadre est la solution aux problèmes de sûreté.

Ainsi, le premier objectif a été de proposer une approche de prise en compte de la sûreté de fonctionnement dans les processus d'ingénierie système. Cette approche a été construite en se basant sur la norme EIA-632 et elle définit les activités spécifiques à mettre en œuvre concernant la sûreté de fonctionnement. A l'occasion, nous avons identifié un processus particulièrement critique qui est celui d'ingénierie des exigences. Notamment une attention toute particulière doit être prêtée concernant les exigences de sûreté. Cette approche structure et impose une formulation claire des exigences (notamment de sûreté) et recommande leurs formalisations. Elle revendique une traçabilité, très importante pour la conception de systèmes complexes, particulièrement en cas d'évolution des exigences et de gestion de changement.

Ensuite, nous nous sommes concentrés sur une méthodologie qui permet d'aider à la définition d'exigences de sûreté au niveau du système, puis à leurs déclinaisons au niveau des sous-systèmes. L'intérêt concernant la traçabilité est immédiat concernant ces exigences de sûreté. Cette méthodologie, construite à base d'AMDEC et d'analyses d'arbres de défaillance, améliore ainsi la définition des exigences de sûreté, leur gestion et leur traçabilité.

Nous avons également voulu se rapprocher d'une approche d'Ingénierie Système Basée sur les Modèles (ISBM). Un modèle d'information a été défini et présenté afin de rendre plus efficace la gestion des exigences et de fournir une base commune entre tous les participants à la conception, notamment entre les différents métiers. Ce modèle est basé sur le langage SysML et respecte les concepts définis dans la norme EIA-632. Il inclut explicitement des concepts important pour notre problématique que sont les risques et les exigences de sûreté. Il permet une très bonne vision des aspects de traçabilité entre les

différentes données d'ingénierie, tout en séparant les trois concepts : exigences, solution de conception et V&V.

Pour finir, nous avons illustré les différentes propositions sur l'exemple de l'avion fictif S18. Dans le cadre d'une conception en suivant l'approche définie au deuxième chapitre et les concepts de la norme EIA-632, le cas d'étude a permis d'appliquer la méthodologie de déclinaison d'exigences de sûreté du troisième chapitre, ceci sur trois niveaux. L'exemple a également été modélisé en SysML en suivant le modèle d'information présenté au quatrième chapitre. Les liens de traçabilité étaient alors directement visibles, ainsi que les risques et les exigences de sûreté qui font partie intégrante du modèle.

Problème d'ingénierie ou problème de gestion ?

Pour faire une synthèse des problèmes de sûreté de fonctionnement, deux grandes familles de problèmes de sûreté se distinguent : les problèmes d'ingénierie et les problèmes de gestion. Il ne faut pas confondre les deux. Un problème d'ingénierie ne peut être résolu qu'en s'occupant de l'ingénierie, alors qu'un problème de gestion qu'en améliorant la gestion. Typiquement, un souci dans l'architecture du système est un problème d'ingénierie, alors qu'une mauvaise traçabilité est un problème de gestion.

Dans notre travail, nous nous sommes finalement intéressés aux deux fronts en même temps, avec entre autres :

- L'approche processus intégrant la sûreté de fonctionnement qui donne à la fois des recommandations concernant la gestion des données d'ingénierie et l'ingénierie du système,
- La méthodologie de déclinaison d'exigence qui concerne plus l'ingénierie du système,
- Le modèle d'information qui se retrouve plus à un niveau gestion des données d'ingénierie.

Perspectives

De nombreuses perspectives sont envisageables pour améliorer et continuer le travail réalisé au cours de cette thèse et présenté dans ce mémoire. Celles-ci se retrouvent au niveau des trois axes de recherches présents dans le travail : la démarche processus proposée, la méthodologie de déclinaison et le modèle d'information.

La démarche d'Ingénierie Système intégrant la sûreté de fonctionnement

D'autres processus de la norme EIA-632 pourrait être explorés et détaillés en termes de sûreté de fonctionnement, pour améliorer la démarche processus proposée. En effet, nous nous sommes essentiellement concentrés sur les processus de conception système et sur quelques processus d'évaluation technique (analyse des risques, validation des exigences, vérification du système), car ces processus sont au cœur de l'ingénierie du système et sont primordiaux. Ce sont effectivement eux qui permettent les définitions des exigences et des solutions logique et physique, ainsi que des validations et vérifications de la conception. Mais nous pourrions explorer d'autres processus, comme les processus de fourniture et d'acquisition, les processus de planification, d'évaluation et de pilotage, les processus de

réalisation et de transfert pour l'utilisateur. L'étude de ces processus d'un point de vue sûreté de fonctionnement permettrait d'améliorer encore la démarche de conception proposée et ainsi la sûreté des systèmes conçus en suivant cette démarche.

La méthodologie de déclinaison d'exigences de sûreté de fonctionnement :

Concernant la méthodologie de déclinaison d'exigences de sûreté de fonctionnement, nous nous sommes concentrés sur la partie « cause » des modes de défaillances, ceci en utilisant les arbres de défaillances. Les actions correctives s'attachaient à réduire la fréquence des causes. Il s'agirait, pour compléter la méthodologie, de s'intéresser a la partie « effet » des modes de défaillances. Les actions correctives auraient alors comme objectif de réduire la gravite des effets. Une possibilité serait d'utiliser des arbres d'événements pour analyser cette partie « effet ».

Une autre continuité possible concernant la méthodologie de déclinaison serait d'envisager des exigences émanant d'actions correctives associées aux modes de défaillances des sous-systèmes qui n'interviennent pas dans les arbres de défaillances. La question serait de savoir comment considérer ces exigences au niveau du système. Devraient-elles provenir d'une déclinaison ? Sont-elles indicatrices d'oublis au niveau des analyses des défaillances systèmes ou au niveau des arbres de défaillances ?

Le modèle d'information en SysML

Concernant le modèle d'information, nous avons évoqué, à la fin du quatrième chapitre, l'utilisation potentielle du langage OCL pour la formalisation des exigences. Il s'agirait donc d'intégrer cette possibilité dans le modèle d'information. Egalement, nous avons envisagé des analyses à définir sur le modèle d'information, tel que des générations de matrices de traçabilité ou encore des évaluations d'indicateurs ou de métriques. Les besoins de ces analyses devraient être identifiés et leurs formalisations pourraient être définies.

Une extension plus importante du modèle d'information, que nous avons définit à l'aide de SysML, serait de le lier à un langage plus spécifique pour la solution physique. Un bon candidat pour ce langage pourrait être AADL.

Bibliographie

[Aer, 2004] SAE Aerospace. SAE AS5506: Architecture Analysis and Design Language (AADL). SAE International, 2004.

[AFIS] AFIS, Association Française d'Ingénierie Système, www.afis.fr.

[AFIS, 2005a] Document de vulgarisation de l'IS, 2005, www.afis.fr.

[AFIS, 2005b] AFIS, Modèle de données AFIS, version 2.0, groupe de travail Méthodes et Outils, 2005.

[Akerlund & al., 2006] Akerlund O., P. Bieber, E. Boede, M. Bozzano, M. Bretschneider, C. Castel, A. Cavallo, M. Cifaldi, J. Gauthier, A. Griffault, O. Lisagor, A. Lüdtke, S. Metge, C. Papadopoulos, T. Peikenkamp, L. Sagaspe, C. Seguin, H. Trivedi, L. Valacca, ISAAC, a framework for integrated safety analysis of functional geometrical and human aspects, ERTS 2006, Toulouse, France, 25-27 january, 2006.

[Albinet & al., 2007] Albinet A., J-L. Boulanger, H. Dubois, M-A. Peraldi-Frati, Y. Sorel and Q-D. Van. Model-based methodology for requirements traceability in embedded systems, 3rd ECMDA workshop on traceability, Haifa, Israel, June 2007.

[Albinet & al., 2008] Albinet A., S. Begoc, J.-L. Boulanger, O. Casse, I. Dal, H. Dubois, F. Lakhal, D. Louar, M.-A. Peraldi-Frati, Y. Sorel and Q.-D. Van, The MeMVaTEx methodology: from requirements to models in automotive application design, in ERTS'08, Toulouse, France, January 2008.

[Aldemir & al., 2006] Aldemir T., D.W. Miller, M.P. Stovsky, J. Kirschenbaum, P. Bucci, A.W. Fentiman, et L.T. Mangan, Current state of reliability modeling methodologies for digital systems and their acceptance criteria for nuclear power plant assessments. NUREG/CR-6901, U.S. Nuclear Regulatory Commission, Washington, DC, 2006.

[Alexander & al., 2009] Alexander R., N. Herbert and T. Kelly, Deriving Safety Requirements for Autonomous Systems, 4th SEAS DTC Technical Conference, July 2009.

[Allenby & al., 2001] Allenby K. and T. Kelly, Deriving Safety Requirements using Scenarios, 5th IEEE International Symposium on Requirements Engineering (RE'01), IEEE Computer Society Press, 2001.

[Arlat & al., 1999] J. Arlat, Y. Crouzet, P. David, J.-L. Dega, Y. Deswarte, J.-C. Laprie, D. Powell, C. Rabéjac, H. Schindler, J.-F. Soucailles, "Fault Tolerant Computing," dans Encyclopedia of Electrical and Electronics Engineering, (Ed. J. G. Webster), Ch. 7, pp. 285-313, J. Wiley & Sons, 1999.

[ARP-4754, 1996] ARP-4754: Certification considerations for highly-integrated or complex aircraft systems, Society of Automotive Engineers (SAE) standard, novembre 1996.

[ARP-4761, 1996] ARP-4761: Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment, Society of Automotive Engineers (SAE) standard, décembre 1996.

[Avizienis & al., 2001] Avizienis A., J.-C. Laprie, and B. Randell, Fundamental Concepts of Dependability, LAAS-CNRS, Technical Report N01145, Apr. 2001.

[Avizienis & al., 2004] Avizienis A., J.-C. Laprie, B. Randell, et C. Landwehr, Basic Concepts and Taxonomy of Dependable and Secure Computing. IEEE Transactions on Dependable and Secure Computing, vol. 1, pp. 11-33, 2004.

[Bar-Yam, 2005] Bar-Yam Y.. About engineering complex systems: Multiscale analysis and evolutionary engineering. Engineering self-organizing systems: methodologies and applications, vol. 3464, pp. 16-3, 2005.

[Batut, 1986] Batut. J Fiabilité prévisionnelle du réseau à très haute tension d'edf. In 5eme Colloque international de fiabilité et de maintenabilité, Biarritz, France, October 1986.

[Beizer, 1990] Beizer. B, Software Testing Techniques, Van Nostrand Reinhold, New York, 2nd Edition, 1990.

[Bernoulli, 1738] Bernoulli D., Specimen theoriae novae de mensura sortis, Commentarii Academiae Scientiarum Imperialis Petropolitanae 5, p. 175-192, 1738.

[Black & al., 2009] Black J., P. Koopman, System Safety as an Emergent Property in Composite Systems. IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 369-378, Estoril, Lisbon, June 29-July 2, 2009.

[Bock, 2005] Bock C., Systems engineering in the product lifecycle, Int J Product Development 2(1–2), 123–137, 2005.

[Bock, 2006] Bock C., "SysML and UML 2 Support for Activity Modeling," Systems Engineering, 9, No. 2, pp. 160-186, 2006.

[Boehm, 1986] Boehm B., A spiral model of software development and enhancement, ACM SIGSOFT, vol 11, N° 4, August 1986.

[Boehm, 1988] Boehm B., A Spiral Model of Software Development and Enhancement, Computer, pp. 61-72, May 1988.

[Boehm & al., 1994] Boehm B., P. Bose, A Collaborative Spiral Software Process Model Based on Theory W, October 11, 1994.

[Booch, 1993] Booch G., *Object-Oriented Analysis and Design with Applications (2nd Edition)*, Addison-Wesley Professional, 10 October, 1993.

[Booch & al., 2005] Booch G., J. Rumbaugh et I. Jacobson, The Unified Modeling Language User Guide, 2nd Edition, Addison Wesley, 2005.

[Bozzano & al., 2003] Bozzano M, Cavallo. A, Cifaldi. M, Valacca. L, Villafiorita. A. Improving Safety Assessment of Complex Systems: An Industrial case study. FM 2003. Pisa, 8-14 September 2003.

[Buren & al., 1998] Buren, JV & Cook, DA 1998, Experiences in the Adoption of Requirements Engineering Technologies, Software Technology Support Center.

[Buzzatto, 1999] Buzzatto J.L., Failure mode, effects and criticality analysis (FMECA) use in the Federal Aviation Administration (FAA) reusable launch vehicle (RLV) licensing process. Digital Avionics Systems Conference, 1999. Proceedings 18th. vol.2 10/24-29/1999. Location: St Louis, MO, USA.

[CEI-60812, 2006] CEI-60812: Techniques d'analyse de la fiabilité du système – procédure d'analyse des modes de défaillance et de leurs effets, International Electrotechnical Commission standard, 2006.

[CEI-61508, 2010] CEI-61508: Functional safety of electrical/electronic/ programmable electronic safety-related systems, International Electrotechnical Commission standard, 2010.

[Chatelet, 2000] Chatelet E., Sûreté de fonctionnement : méthodes et outils de base. Université de technologie de Troyes edition, 2000.

[Cohn, 1989] Cohn. A, "The notion of proof in hardware verification", Journal of Automated Reasoning, vol. 5, pp. 127-139, 1989.

[Common Criteria, 1998] Common Criteria for Information Technology Security Evaluation, Common Criteria Implementation Board, Version 2.0, CCIB-98-026, CCIB-98-027, CCIB-98-027A, CCIB-98-028, May 1998.

[Conquet & al., 2005] E. Conquet and P. David, "Preparing the System and Software engineering of the 21st century for critical systems with the ASSERT project", in Fifth European Dependable Computing Conference, Supplementary. Volume, (Budapest, Hungary), pp.27-32, 2005.

[Conquet, 2008] Conquet E., The ASSERT project: a step towards reliable and scientific system and software engineering, ERTS2008, Toulouse, France, January 2008.

[Coulin, 2007] Coulin C. R.. A Situational Approach and Intelligent Tool for Collaborative Requirements Elicitation. Thèse de doctorat, University of technology, Sydney and Université Paul Sabatier, Toulouse, 2007.

[David, 2009] David P., Contribution à l'analyse de sûreté de fonctionnement des systèmes complexes en phase de conception : application à l'évaluation des missions d'un réseau de capteurs de présence humaine, Thèse doctorat, Université d'Orléans, 2009.

[David & al., 2010] David P., Idasiak V., Kratz F., Reliability study of complex physical systems using SysML, Reliability Engineering and System Safety, Vol. 95, p.431-450, 2010.

[Desroches & al., 2009] Desroches A., Baudrin D., Dadoun M. L'analyse préliminaire des risques. Edition Hermès – Lavoisier, 2009.

[Deswarte, 1998] Deswarte Y. « SQUALE: Dependability Assessment Criteria », SQUALE Open Workshop, Toulouse, 24-25 novembre 1998.

[Deswarte & al., 1998] Deswarte Y., Kaâniche M., Corneillie P., Benoit P., «SQUALE : critères d'évaluation de la sûreté de fonctionnement », Actes du 11ème Colloque National de Fiabilité & Maintenabilité (/-11), Arcachon (France), 29 septembre - 1er octobre 1998, pp. 367-376

[Deswarte & al., 1999] Deswarte Y., Kaaniche M., Corneillie P., Goodson J., "SQUALE Dependability Assessment Criteria", 18th International Conference on Computer Safety, Reliability and Security (SAFECOMP'99), (Toulouse, France), Lecture Notes on Computer Science 1698, Springer-Verlag, pp.27-38, 1999.

[DO-178B, 1992] DO-178B: Software considerations in airborne systems and equipment certification, RTCA et EUROCAE, 1er décembre 1992.

[DO-254, 2000] DO-254: Design assurance guidance for airborne electronic hardware, RTCA et EUROCAE, 19 avril 2000.

[Dubi, 2000] Dubi A., Monte Carlo Applications in Systems Engineering. John Wiley & Sons, Ltd. England, 2000.

[EIA-632, 1999] EIA-632: Processes for engineering systems, Electronic Industries Alliance standard, 7 janvier 1999.

[ESACS, 2001] Requirements for improving the safety process on Complex Systems and definition of the tools integration concepts, WP1, ESACS, 17 mai 2001.

[ESACS] Consortium, Enhanced Safety Assessment for Complex Systems (Project Portal), http://www.cert.fr/esacs/.

[Esteban et al., 2009] Esteban P., J.-C. Pascal et D. Esteve, Une méthodologie de Conception Produit basée sur la norme EIA-632, 8ème Congrès International de Génie Industriel (CIGI 2009), Bagnères de Bigorre (France), 8p, 10-12 Juin 2009.

[Estefan, 2008] Estefan J.A., Survey of Model-Based Systems Engineering (MBSE) Methodologies. Part B. Survey of Candidate Model-Based Engineering (MBSE) Methodologie. 47p. May, 2007. INCOSE MBSE Focus Group.

[Evrot, 2008] Evrot D., Contribution à la vérification d'exigences de sécurité : application au domaine de la machine industrielle, Thèse de doctorat, Université Henri Poincaré, Nancy I, 2008.

[Faisandier, 2008] Faisandier A., Sensibilisation à l'Ingénierie des Systèmes, Toulouse, MAP Système, 2008.

[Faucher, 2004] Faucher J., Pratique de l'AMDEC, Edition DUNOD, Paris 2004.

[Faucher, 2009] Faucher J., Pratique de l'AMDEC (2ème édition), Editeur Dunod, 17 juin 2009.

[Forsberg & al., 1991a] Forsberg K., Mooz H., Visualizing project management: models and frameworks for mastering complex systems. Proceeding of the 1991 INCOSE Symposium.

[Forsberg & al., 1991b] Forsberg K., Mooz H., The relationship of system engineering to the project cycle. Proceedings of the 1991 INCOSE Symposium.

[Forsberg & al., 1995] Forsberg K., Mooz H., "Application of the "Vee" to Incremental and Evolutionary Development," Proceedings of the Fifth Annual International Symposium of the National Council on Systems Engineering, St. Louis, MO, July 1995.

[Friedenthal & al., 2008] Friedenthal S., A. Moore and R. Steiner, A Practical Guide to SysML: The Systems Modeling Language. The MK/OMG press, Elsevier, 2008.

[Goguen & al., 1993] Goguen J. and C. Linde, Techniques for requirements elicitation. In 1st IEEE International Symposium on Requirements Engineering, pages 152-164, San Diego, 4-6th January 1993.

[Gotel & al., 1994] Gotel O. C. Z. and C. W. Finkelstein, An analysis of the requirements traceability problem, in International Conference on Requirements Engineering, 1994, pp. 94– 101.

[Guillerm & al., 2009a] Guillerm. R, Sadou. N, Demmou H. et Alloula K., Hybrid Approach for Deriving Feared Scenarios in Industrial Systems, SAFE PROCESS 2009 - 7th IFAC Symposium on Fauult Detection Supervision and Safety of Technical Processes, Barcelone (Espagne), 30 Juin - 3 Juillet 2009.

[Guillerm & al., 2009b] Guillerm R., H. Demmou et N. Sadou, System engineering approach for safety management of complex systems, European Simulation and Modelling Conference (ESM'2009), Leicester (Royaume-Unis), 26-28 Octobre 2009.

[Guillerm & al., 2010a] Guillerm. R, Sadou. N, Demmou. H.. ESA Petri net: Dynamic reliability analysis Tool. International Journal of Adaptive and Innovative Systems 1, 3/4 (2010) 201-216.

[Guillerm & al., 2010b] Guillerm R., H. Demmou et N. Sadou, Safety Evaluation of complex system – Integration in system engineering process, IEEE International Systems Conference 2010, San Diego (California, USA), 5-8 Avril 2010.

[Guillerm & al., 2010c] Guillerm R., H. Demmou et N. Sadou, Engineering dependability requirements for complex systems - A new information model definition, IEEE International Systems Conference 2010, San Diego (California, USA), 5-8 Avril 2010.

[Guillerm & al., 2010d] Guillerm R., H. Demmou et N. Sadou, Information model for model driven design of complex system based on system engineering approach, Complex Systems Design and Management Conference (CSDM'2010), Paris, 27-29 Octobre 2010.

[Guillerm & al., 2011] Guillerm R., N. Sadou et H. Demmou, Combining FMECA and Fault Trees for declining safety requirements of complex systems, European Safety and Reliability Conference (ESREL), 2011.

[IEEE-1220, 2005] IEEE-1220: Standard for application and management of the systems engineering process, IEEE standard, 20 mars 2005.

[INCOSE, 2004]. Systems engineering handbook. International Council on Systems Engineering (INCOSE) Working Group, 2004.

[INCOSE, 2007] INCOSE, Systems Engineering Vision 2020, INCOSE-TP-2004-004-02, version 2.03, septembre 2007.

[ISAAC] ISAAC Consortium, Improvement of Safety Activities on Aeronautical Complex systems (Project Portal), http://www.cert.fr/isaac/.

[ISO-15288, 2008] ISO-15288: Systems and software engineering – system life cycle processes, International Organization of Standardization standard, 2008.

[ISO-26262, 2008] ISO-26262, Véhicules routiers – sécurité fonctionnelle, version projet de comité, International Organization of Standardization standard, 2008.

[Jacobson, 1992] Jacobson I., *Object Oriented Software Engineering: A Use Case Driven Approach*, Addison Wesley, 1 juillet 1992.

[Jardine, 1987] Jardine. A. Maintenance, replacement and reliability, Pitman, Boston, MA, 1987.

[Juristo & al., 2002] Juristo. N, A. M. Moreno, and A. Silva, Is the European Industry Moving Toward Solving Requirements: Engineering Problems?, IEEE Software, vol. 19, no. 6, pp. 70–77, 2002.

[Khalfaoui, 2003] Khalfaoui. S . Méthode de recherche des scénarios redoutés pour l'évaluation de la sûreté de fonctionnement des systèmes mécatroniques du monde automobile. Thèse de Doctorat, No 03574, Institut National Polytechnique, Toulouse, 26 septembre 2003.

[Komi-Sirvio & al., 2003] Komi-Sirvio S. and M. Tihinen, Great Challenges and Opportunities of Distributed Software Development – An Industrial Survey. In Proceedings of the Fifteenth International Conference on Software Engineering & Knowledge Engineering (SEKE'2003), 2003, pp. 489–496.

[Konate, 2009] Konate J., Approche système pour la conception d'une méthodologie pour l'élicitation collaborative des exigences, Thèse de doctorat, Université de Toulouse, 2009.

[Kotovsky & al., 1985] Kotovsky K., J.R. Hayes et H.A. Simon, Why are some problems hard? Evidence from Tower of Hanoi, Cognitive Psychology, vol. 17, 1985.

[Landy, 2007] Landy G., AMDEC : Guide pratique, Edition AFNOR 2007.

[Lannoy, 2008] A. Lannoy, Maîtrise des risques et sûreté de fonctionnement : repères historiques et méthodologiques, Collection Sciences du risque et du danger, série Notes de synthèse et de recherche, 128p. janvier 2008.

[Laprie & al., 1995] Laprie J.-C., J. Arlat, J.-P. Blanquart, A. Costes, Y. Crouzet, Y. Deswarte, J.-C. Fabre, H. Guillermain, M. Kaâniche, K. Kanoun, C. Mazet, D. Powell, C. Rabéjac, P. Thévenod, Guide de la sûreté de fonctionnement, Editions Cépaduès, Toulouse, mai 1995, 369 p.

[Laprie, 2004] Laprie. J-C, "Sûreté de fonctionnement des systèmes : concepts de base et terminologie", Revue de l'Électricité et de l'Électronique (REE), (11), pp.95-105, novembre 2004.

[Ledoux & al., 2007] Ledoux J., Gaudoin O., Modélisation aléatoire en fiabilité des logiciels. Ed Hermes Science Publications, 2007.

[Lee & al., 1985] Lee W.S, Grosh D.L, Tillman F.A, Lie C.H. "Fault tree analysis, methods, and applications - A review", IEEE Transactions on Reliability, August 1, 1985; ISBN 0018-9529; r-34, page 194-203.

[Leveson, 2003] Leveson N., White Paper on Approaches to Safety Engineering, 23 avril 2003.

[Leveson & al., 2003] Leveson N., M. Daouk, N. Dulac et K. Marais, A Systems Theoritic Approach to Safety Engineering, 30 October 2003.

[Leveson, 2004] Leveson N., Model-based analysis of socio-technical risk. Technical Report ESD-WP-2004-08, MIT, Cambridge, MA, December 2004.

[Leveson & al., 2005] Leveson N. et N. Dulac, Safety and Risk-Driven Design in Complex Systems-of-Systems, 1st NASA/AIAA Space Exploration Conference, Orlando (Floride), 30 janvier – 1er février 2005.

[Leveson & al., 2007] Leveson N., M. S. Herring, B. D. Owens, M. Ingham et K. A. Weiss, Safety-Driven Model-Based System Engineering

Methodology Part I: Methodology Description, 16 décembre 2007.

[Lindsay & al., 1999] Lindsay P. A., J. A. McDermid et D. J. Tombs, A Process for Derivation and Quantification of Safety Requirements for Components of Complex Systems, Technical report, No. 99-46, décembre 1999.

[Lindsay & al., 2000] Lindsay P. A., J. A. McDermid et D. J. Tombs, Deriving quantified safety requirements in complex systems, Computer safety, reliability and security, SAFECOMP 2000, Rotterdam, Pays-Bas, 24-27 Octobre 2000, vol. 1943, pp. 117-130.

[Lindsay & al., 2002] Lindsay P. A. et J. A. McDermid, Derivation of safety requirements for an embedded control system, Systems Engineering, Test and Evaluation Conference, Sydney, 29-30 Octobre 2002.

[M2OS, 2010] Groupe de travail Management, Méthodes, Outils, Standards (M2OS), Fiches méthodes, Institut pour la Maîtrise des Risques (IMdR), septembre 2010.

[Magee & al., 2004] Magee C. and De Weck O. L., Complex System Classification, Fourteenth Annual International Symposium of the International Council on Systems Engineering (INCOSE), Toulouse, France, June 20-24, 2004.

[MAP Système] MAP Système, http://sinergie.mapsysteme.pagesperso-orange.fr

[Martin, 2001] Martin. N. J., Chapter 24: Processes for Engineering a System. Citation Information. Digital Avionics Handbook, Second Edition - 2 Volume Set Edited by Cary R . Spitzer CRC Press 2001

[Meinadier, 1998] Meinadier J.-P., Ingénierie et intégration des systèmes, Editions Hermès, 1998.

[Meinadier, 2002] Meinadier J.-P., Le métier d'intégration de systèmes, Edition Hermès-Lavoisier (540p), 2002.

[Messaadia, 2008] Messaadia M., Ingénierie système et système de production manufacturière : Intégration de l'évolution des exigences dans le PLM, Thèse de Doctorat, Université de Toulouse, avril 2008.

[Mortureux, 2002] Mortureux Y., Analyse préliminaire des risques, Technique de l'ingénieur, Octobre 2002.

[Parviainen & al., 2004] Parviainen P., Tihinen M., Lormans M., and Van Solingen R., Requirements Engineering: Dealing with the Complexity of Sociotechnical Systems Development, Requirements

Engineering for Sociotechnical Systems, J. L. Mat´e and A. Silva, Eds. IdeaGroup Inc, ch.1, pp.1–20, 2004.

[Ramesh, 1998] Ramesh B., Factors influencing requirements traceability practice, communication of the ACM, vol. 41, issue 12, pp.37-44, New York, USA, 1998.

[Ramesh & al., 2001] Ramesh B. and Jarke M., Towards reference models for requirements traceability, IEEE Transactions on Software Engineering, vol. 27, issue 1, pp.58-93, 2001.

[Rasmussen, 1975] Rasmussen N., Reactor Safety Study – An Assessment of Accident Risks in U.S., Commercial Nuclear Power Plants, WASH-1400, October 1975.

[Rasmussen, 1997] Rasmussen J., Risk Management in a Dynamic Society: A Modelling Problem, Safety Science, vol. 27, No. 2/3, Elsevier Science Ltd., pp.183-213, 1997.

[Redouin, 1989] Redouin P., *Merise, comprendre et pratiquer*, EDITEST, 1989.

[Roberston, 2010] Robertson J. and S., Volere Requirements Specification Template, edition 15, Atlantic Systems Guild, 2010.

[Rochet, 2007] Rochet S., Formalisation des processus de l'Ingénierie Système : proposition d'une méthode d'adaptation des processus génériques à différents contexte d'application, Thèse de Doctorat, INSA de Toulouse, 26 novembre 2007.

[Rogovin, 1979] Rogovin M., Three Mile Island: a report to the commissioners and to the public, Technical Report, Nuclear Regulatory Commission, Washington, DC (USA), January 1979.

[Rumbaugh & al., 1990] Rumbaugh J. R., Blaha M. R., Lorensen W., Eddy F., Premerlani W., *Object-Oriented Modeling and Design*, Prentice-Hall, 1st October, 1990.

[Sadou, 2007] Sadou N., Aide à la conception des systèmes embarqués sûrs de fonctionnement, Thèse de Doctorat, Université de Toulouse, 6 novembre 2007.

[Sadou & al., 2009] Sadou N. and Demmou H., Reliability analysis of discrete event dynamic systems with Petri nets, Reliability Engineering & System Safety, pp.1862-1868, november 2009.

[Sahraoui, 2005] Sahraoui A.-E.-K., Requirements Traceability Issues: Generic Model, Methodology And Formal Basis, International Journal of Information Technology and Decision Making, vol. 4, no. 1, pp. 59–80, 2005.

[Sheard, 2006] Sheard S., Complex Systems Science and its Effects on Systems Engineering, European Systems Engineering Conference, Edinburgh, UK, 18-20 September 2006.

[Sinnamon & al., 1996] Sinnamon M. and Andrews J. D., Fault trees and binary decision diagrams, Proceedings of the Annual Reliability and Maintainability Symposium, pp.215-222, 1996.

[Sommerville & al., 1997] Sommerville I. and Sawyer P., Requirements Engineering: A Good Practice Guide, 1st edition, John Wiley & Sons, 1997.

[Sommerville, 2006] Sommerville I., Software Engineering (Update) (8th Edition), International Computer Science, Addison-Wesley Longman Publishing Co., Boston, MA, USA, 2006.

[SPEM, 2008] Software & Systems Process Engineering Meta-Model Specification (SPEM), version 2.0, Object Management Group (OMG), avril 2008.

[SQUALE, 1999a] SQUALE (Security, Safety and Quality Evaluation for Dependable Systems): Dependability Assessment Criteria, 4th draft, janvier 1999.

[SQUALE, 1999b] SQUALE (Security, Safety and Quality Evaluation for Dependable Systems): End of Project Report, 26 février 1999.

[SysML, 2008] Systems Modeling Language (SysML), version 1.1, Object Management Group (OMG), novembre 2008.

[TCSEC, 1985] TCSEC: Department of Defense Trusted Computer System Evaluation Criteria, U.S. Department of Defense, 1985.

[UML, 2010] Unified Modeling Language (UML), Superstructure, version 2.3, Object Management Group (OMG), mai 2010.

[Verries & al., 2008] Verries J., Paludetto M. and Saharaoui A.-E.-K., From design with SysML to VHDL-AMS simulation, Proceeding of ESM'08, pp.115-120, Le Havre, Oct. 2008.

[Verries, 2010] Verries J., Approche pour la conception de systèmes aéronautiques innovants en vue d'optimiser l'architecture - Application au système portes passagers, Thèse de doctorat, Université de Toulouse, 2010.

[Vesely & al., 1987] Vesely W. E. and Roberts N. H., Fault Tree Handbook, U.S. Nuclear Regulatory Commission, Government Printing Office, 1987.

[Vhd, 1999] 1076.1-1999 IEEE Standard VHDL Analog and Mixed-Signal Extensions Language Reference Manual, IEEE Press, ISBN 0-7381-1640-8.

[Villemeur, 1988] Villemeur A., Sûreté de Fonctionnement des Systèmes Industriels, Paris : Edition Eyrolles, 785p, 1988.

[Volere] Volere, Atomic Requirement, www.volere.co.uk.

[Watson, 1961] Watson H.A., Launch Control Safety Study, section VII Vol. 1, Bell Labs, Murray Hill, NJ, 1961.

[Wu & al., 2006] Wu W. and Kelly T., Deriving Safety Requirements as Part of System Architecture Definition, 24th International System Safety Conference, System Safety Society, Albuquerque (USA), August 2006.

[Zeigler & al., 2000] Zeigler B.P, Praehofer H. and Kim T.G., Theory of modelling and simulation, Academic Press, San Diego, California, USA, 2000.

Annexe A : Mots-clés associés au modèle de développement à SdF explicite

1. Expression des exigences

1.1. Processus de création

- **Spécifications fonctionnelles**
  - Définitions des fonctions (services attendus en valeur et en temps)
  - Description et enchaînement des phases (systèmes phasés)
  - Répartition préliminaire des tâches entre l'homme et le système
- **Description de l'environnement d'utilisation**
  - Frontières du système (environnement sociotechnique)
  - Caractéristiques de l'environnement et profils des utilisateurs
  - Modes d'exploitation
- **Contraintes de développement, de validation et d'exploitation**
  - Contraintes physiques (poids, technologie, etc.)
  - Evolutions prévisibles
  - Portabilité et interopérabilité souhaitées
  - Réutilisation
  - Testabilité
  - Contraintes d'exploitation et de maintenance

1.2. Processus de prévention de fautes

- **Formalismes et langages**
  - Exigences de certification, réglementation, normes et standards
  - Choix des formalismes, outils et environnements pour le développement
- **Organisation projet**
  - Attribution des tâches et organisation des équipes
  - Gestion de la sous-traitance
  - Gestion des ressources
- **Planification projet et évaluation des risques liés au développement**
  - Identification des risques (contraintes de développement et d'exploitation, évolution des technologies)
  - Moyens pour la réduction des risques
  - Choix d'une stratégie de développement
  - Planification de chaque étape du projet (objectifs, moyens, coûts, délais)
  - Définition des critères de transition entre les différentes étapes

- Planification des revues de projet et des audits
- Planification de la gestion des configurations

1.3. Processus de tolérance aux fautes

- **Comportement en présence de défaillances**
  - Identification et dosage des attributs de la sûreté de fonctionnement mis en jeu
  - Hypothèses de défaillance et modes dégradés possibles
    - Définition des défaillances
    - Modes de défaillance (domaine, perception, conséquences)
    - Durée maximale d'interruption de service admissible
    - Nombre de fautes successives à tolérer dans chaque mode

1.4. Processus d'élimination des fautes

- **Préparation du plan de vérification**
  - Définition des stratégies de vérification statique (revues, inspection, …)
  - Définition des stratégies de test (critères de test, méthodes de génération)
  - Spécification des simulateurs d'environnement éventuels
- **Hypothèses pour la vérification**
- **Vérification des exigences**
  - Analyse comportementale (vérification de propriétés)
  - Revues et inspection de la spécification
  - Agrément de la spécification
    - Maquettage
    - Mise en situation d'opérateurs
    - Revue par experts
- **Conception de scénarios de vérification fonctionnelle**

1.5. Processus de prévision des fautes

- **Expression des objectifs de sûreté de fonctionnement**
  - Choix des mesures et assignation d'objectifs quantifiés
- **Analyse des modes de défaillance, de leurs effets et de leur criticité**
  - Identification des modes de défaillance
  - Classification des défaillances/sévérité
  - Classes de défaillance à prendre en compte
- **Hypothèses pour la prévision**
  - Hypothèses pour la modélisation et paramètres
  - Caractérisation de l'environnement d'utilisation et de maintenance
    - Paramètres de l'environnement (température, pression, etc.)
    - Profil opérationnel (fréquence d'utilisation/fonction)
- **Allocation sûreté de fonctionnement par fonction**
  - Classement des fonctions par niveau de criticité
- **Préparation du plan de prévision des fautes**
  - Choix des méthodes et outils d'analyse et dévaluation ordinale et probabiliste
  - Définition d'une procédure de collecte de données
- **Collecte et analyse**
  - Produits existants : retour d'expérience
    - Données de fiabilité
    - Données sur l'environnement de développement et d'utilisation

- Produits en cours : suivi du produit (nombre de fautes, etc.)

2. Conception

2.1. Processus de création

- **Définition de l'architecture**
  - Structure
    - Décomposition en couches et/ou en composants
    - Répartition des tâches entre l'homme et le système
  - Comportement
    - Requêtes/réponses entre couches et les interactions entre composants
  - Données
    - Nature et flots des données entre composants
- **Choix des technologies**
  - Identification des composants matériels et logiciels
- **Identification des composants réutilisables**
  - Définition des adaptations nécessaires
- **Préparation des procédures d'exploitation et de maintenance**
- **Préparation du plan d'intégration**
  - Stratégie d'intégration de l'architecture
  - Plan d'intégration du système dans son environnement

2.2. Processus de prévention de fautes

- **Formalismes et langages**
  - Choix des formalismes, outils et environnements pour le développement
  - Normes et standards
- **Gestion de projet**
- **Planification projet et évaluation des risques liés au développement**

2.3. Processus de tolérance aux fautes

- **Comportement en présence de fautes**
  - Hypothèses de fautes (fautes prises en compte et fautes exclues)
    - Nature des fautes (accidentelles, intentionnelles)
    - Cause phénoménologique (physiques, humaines)
    - Frontières du système (internes, externes)
    - Persistance temporelle (permanentes, temporaires)
    - Modes de défaillance (domaine, perception, conséquences)
  - Modèles d'erreurs
- **Partitionnement**
  - Niveau(x) de détail de l'application de la tolérance
    - Zones d'indépendance des fautes
    - Zones de confinement des erreurs
  - Couches d'application de la tolérance
  - Supports de tolérance et répartition des tâches entre l'homme et le système
- **Choix des techniques de tolérance aux fautes**
  - Diversification fonctionnelle
  - Programmation défensive

  - Techniques de protection
- **Mécanismes de traitement d'erreurs**
  - Détection
  - Diagnostic
  - Recouvrement
- **Mécanismes de traitement des fautes**
  - Diagnostic
  - Passivation
  - Reconfiguration
- **Identification des points durs**

2.4. Processus d'élimination des fautes

- **Hypothèses pour la vérification**
- **Vérification de la conception**
  - Analyse comportementale (vérification de propriétés)
  - Revues et inspections de la conception
  - Prototypage et mise en situation des opérateurs
  - Vérification de la conception par rapport aux exigences
- **Vérification des mécanismes de tolérance aux fautes**
  - Vérification (formelle) des algorithmes de traitements d'erreurs et de fautes
  - Injection de fautes par simulation (modèle de fautes)
- **Préparation des tests unitaires et des tests d'intégration**
  - Définition des stratégies de test (critères de test, méthodes de génération)
  - Définition des stratégies d'injection de fautes
- **Conception de scénarios de vérification fonctionnelle et structurelle**
- **Vérification de l'évaluation**

2.5. Processus de prévision des fautes

- **Hypothèses pour la prévision**
- **Analyse des modes de défaillance, de leurs effets et de leur criticité**
- **Allocation de sûreté de fonctionnement par composant**
- **Evaluation préliminaire de la sûreté de fonctionnement**
  - Méthodes
    - Modèles analytiques
    - Simulation
  - Dimensionnement
    - Redondances (hommes et système)
    - Couvertures
- **Collecte et analyse**
  - Données sur les défaillances des composants (à partir des bases de données existantes)
  - Suivi et analyse des fautes identifiées

3. Réalisation

3.1. Processus de création

- **Réalisation des composants**
  - Fabrication des composants matériels
  - Codage des composants logiciels
- **Adaptation des composants réutilisés**
- **Paramétrisation**

3.2. Processus de prévention de fautes

- **Formalismes et langages**
  - Choix des formalismes, outils et environnements pour le développement
  - Normes et standards pour la réalisation
- **Gestion de projet**
- **Planification projet et évaluation des risques liés au développement**

3.3. Processus de tolérance aux fautes

- **Mise en œuvre des mécanismes de tolérance aux fautes**

3.4. Processus d'élimination des fautes

- **Hypothèses pour la vérification**
- **Vérification de la réalisation**
  - Vérification de chaque composant vis-à-vis de sa spécification (test, revues)
  - Revues et inspection de la réalisation
  - Tests de charge et de robustesse
- **Vérification de l'évaluation**

3.5. Processus de prévision des fautes

- **Hypothèses pour la prévision**
- **Evaluation de la couverture des tests**
- **Evaluation de la fiabilité des composants**
- **Evaluation préliminaire de la sûreté de fonctionnement du système**
- **Collecte et analyse**
  - Relevés de défaillance et de correction des composants
  - Catégorisation et analyse des fautes, erreurs, défaillances

4. Intégration

4.1. Processus de création

- **Intégration des composants**
- **Intégration du système dans son environnement**

4.2. Processus de prévention de fautes

- **Formalismes et langages**
  - Choix des formalismes, outils et langages pour l'intégration
  - Normes et standards pour l'intégration
- **Gestion de projet**
- **Planification projet et évaluation des risques liés au développement**

4.3. Processus de tolérance aux fautes

- **Intégration des composants de tolérance aux fautes avec les autres composants**

4.4. Processus d'élimination des fautes

- **Hypothèses pour la vérification**
- **Vérification de l'intégration par rapport à l'architecture**
- **Vérification des mécanismes de tolérance aux fautes**
- **Tests d'interopérabilité**
- **Vérification du produit par rapport à sa spécification**
- **Test du système dans son environnement**
- **Tests de qualification**
- **Vérification de l'évaluation**

4.5. Processus de prévision des fautes

- **Hypothèses pour la prévision**
- **Evaluation de la couverture des tests**
- **Evaluation de la tolérance aux fautes**
- **Evaluation des mesures de sûreté de fonctionnement**
- **Collecte et analyse**
  - Relevés de défaillance, d'erreur et de correction (système)
  - Catégorisation et analyse des fautes, erreurs, défaillances

Annexe B : Modélisation SysML du niveau « calculateur BSCU »

Le troisième et dernier *building block* modélisé dans le cas d'étude de l'avion S18, présenté au chapitre 5, est celui du « calculateur BSCU ». Nous présentons ici les différents diagrammes présents dans les trois packages (solution de conception, exigences, V&V) de ce *building block*.

1. Package de la solution de conception

Le premier package présenté est celui de la solution de conception (« Design Solution »), présentant les fonctions et la structure du système.

1.1. Fonctions du système

La Figure B.1 présente les fonctions du système « calculateur BSCU » à travers un diagramme de cas d'utilisation. Les différentes fonctions retrouvées ici correspondent à :

- Le réglage du freinage afin d'éviter le dérapage des roues,
- La sélection du mode de freinage (manuel ou automatique),
- La transmission d'informations concernant le statut du calculateur BSCU,
- La génération de commande de freinage en fonction des commandes reçues et de l'état du système.

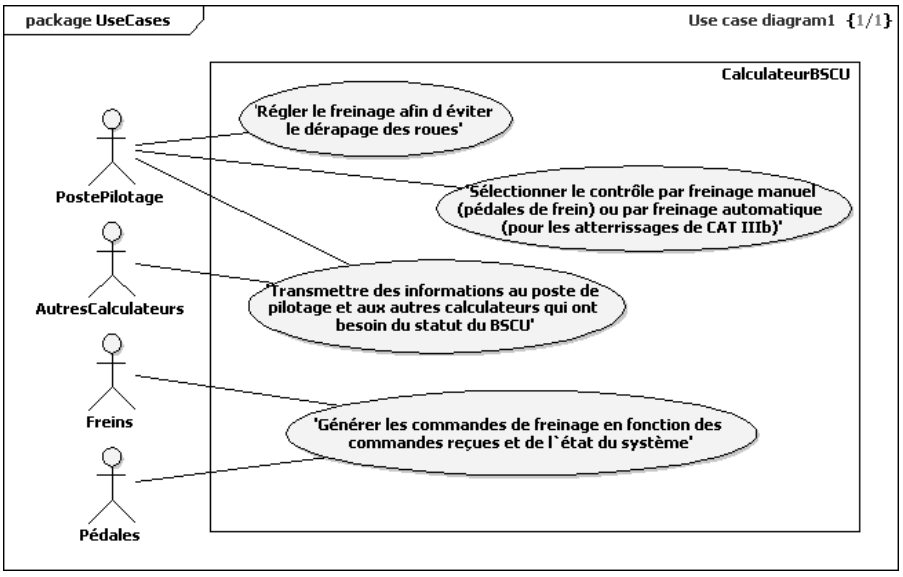


Figure B.1 : Calculateur BSCU – Fonctions du système

1.2. Structure du système

La structure du système « calculateur BSCU » est représentée en Figure B.2 dans un diagramme de blocks. Nous retrouvons le fait que le système « calculateur BSCU » est composé de :

- Deux ressources électriques,
- Deux moniteurs de puissance,
- Deux CPU,
- Deux moniteurs système,
- Un moniteur général,
- Un commutateur.

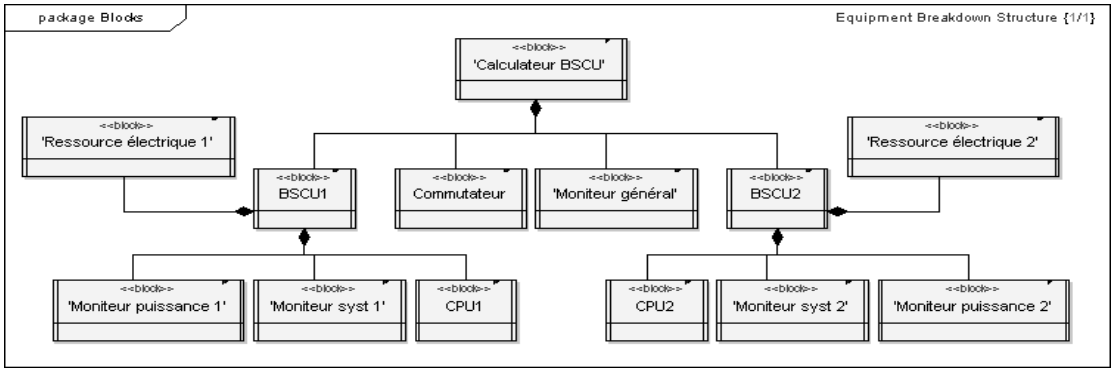


Figure B.2 : Calculateur BSCU – Structure du système

2. Package des exigences

Le second package présenté est celui des exigences (« *Requirements* »).

2.1. Diagramme des exigences haut-niveau

La Figure B.3 montre les exigences haut-niveau du système « calculateur BSCU ». Dans ce diagramme, nous retrouvons par exemple l'exigence de sûreté système « *la perte du calculateur BSCU entrainant la perte de la commande de freinage doit avoir une fréquence < 3,3.10^{-5}/flt* » identifiée lors de l'AMDEC du calculateur BSCU (voir paragraphe 3.3.1 du chapitre 5).

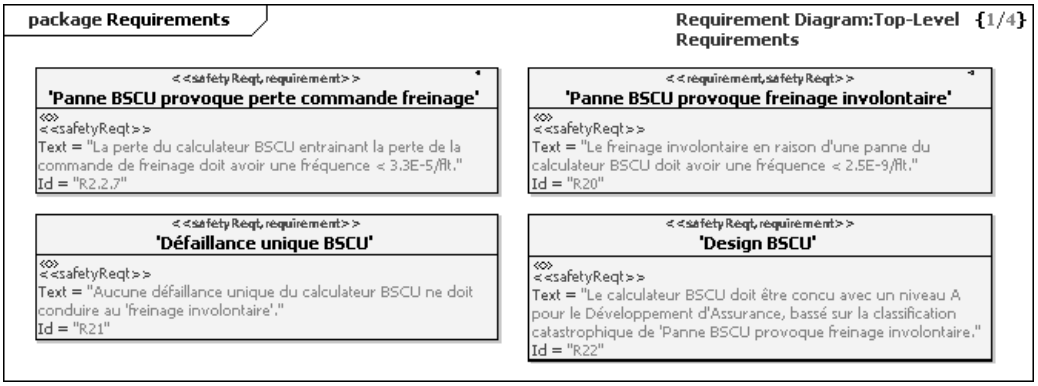


Figure B.3 : Calculateur BSCU – Diagramme des exigences haut-niveau

2.2. Diagramme de déclinaison des exigences

Le diagramme répartit entre la Figure B.4 et la Figure B.5 montrent les déclinaisons des exigences haut-niveau (système calculateur BSCU) en exigences sous-systèmes.

Nous retrouvons les déclinaisons synthétisées au paragraphe 3.3.4 du chapitre 5 concernant les exigences sous-système :

- « **panne BSCU provoque perte commande freinage** », pour la Figure B.4,
- « **panne BSCU provoque freinage involontaire** », pour la Figure B.5.

Remarque : dans ce diagramme, nous constatons que l'exigence sous-système R2.2.7.4 appartient aux deux déclinaisons (celle de l'exigence R2.2.7 et celle de l'exigence R20) Une fois l'exigence R2.2.7.4 liée à l'exigence R2.2.7 pour la première déclinaison, il était impossible d'utiliser la relation de composition « containement » (prévue pour les exigences) pour la deuxième déclinaison (celle de l'exigence R20). Cela signifie que, tel que SysML est défini, une exigence sous-système ne peut pas appartenir à deux exigences systèmes. Dans TauG2, nous avons opté pour une solution rapide et simple qui est l'utilisation du lien de composition « classique ».

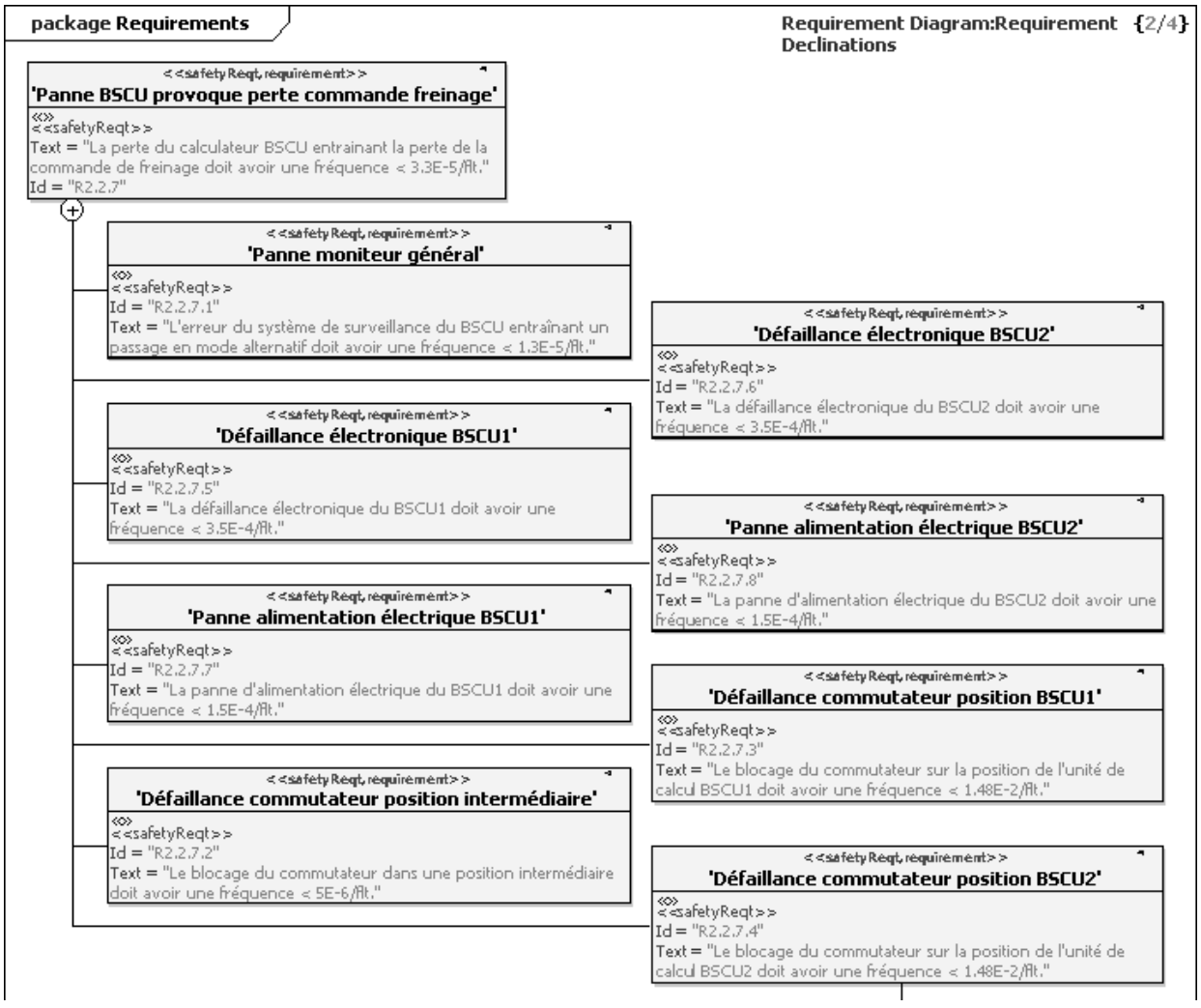


Figure B.4 : Calculateur BSCU – Diagramme de déclinaison des exigences (partie 1/2)

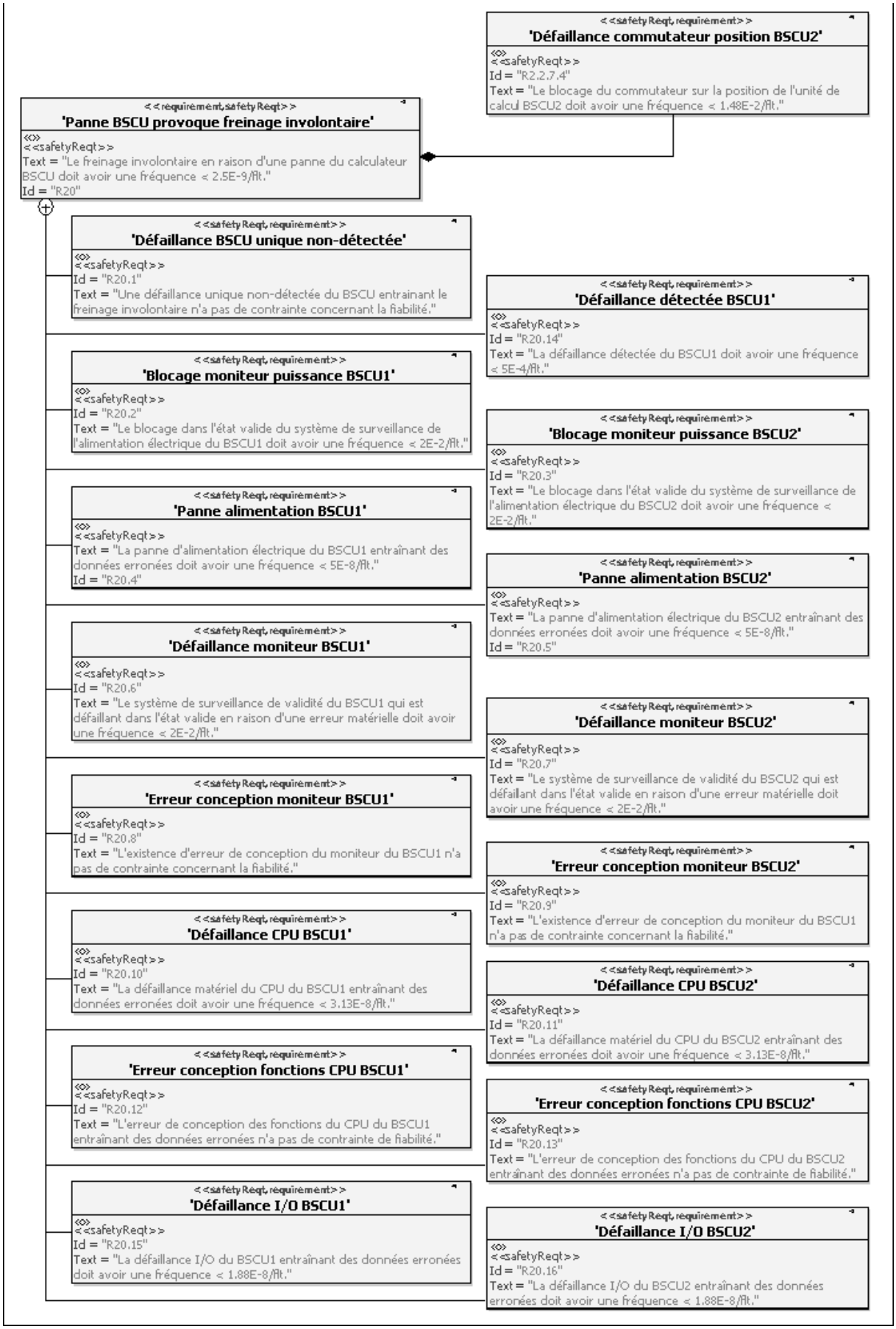


Figure B.5 : Calculateur BSCU – Diagramme de déclinaison des exigences (partie 2/2)

2.3. Diagramme de satisfaction des exigences

La Figure B.6 montre les affectations des exigences aux éléments de la solution de conception (*block* dans notre exemple), ceci à travers le lien « *satisy* » de SysML.

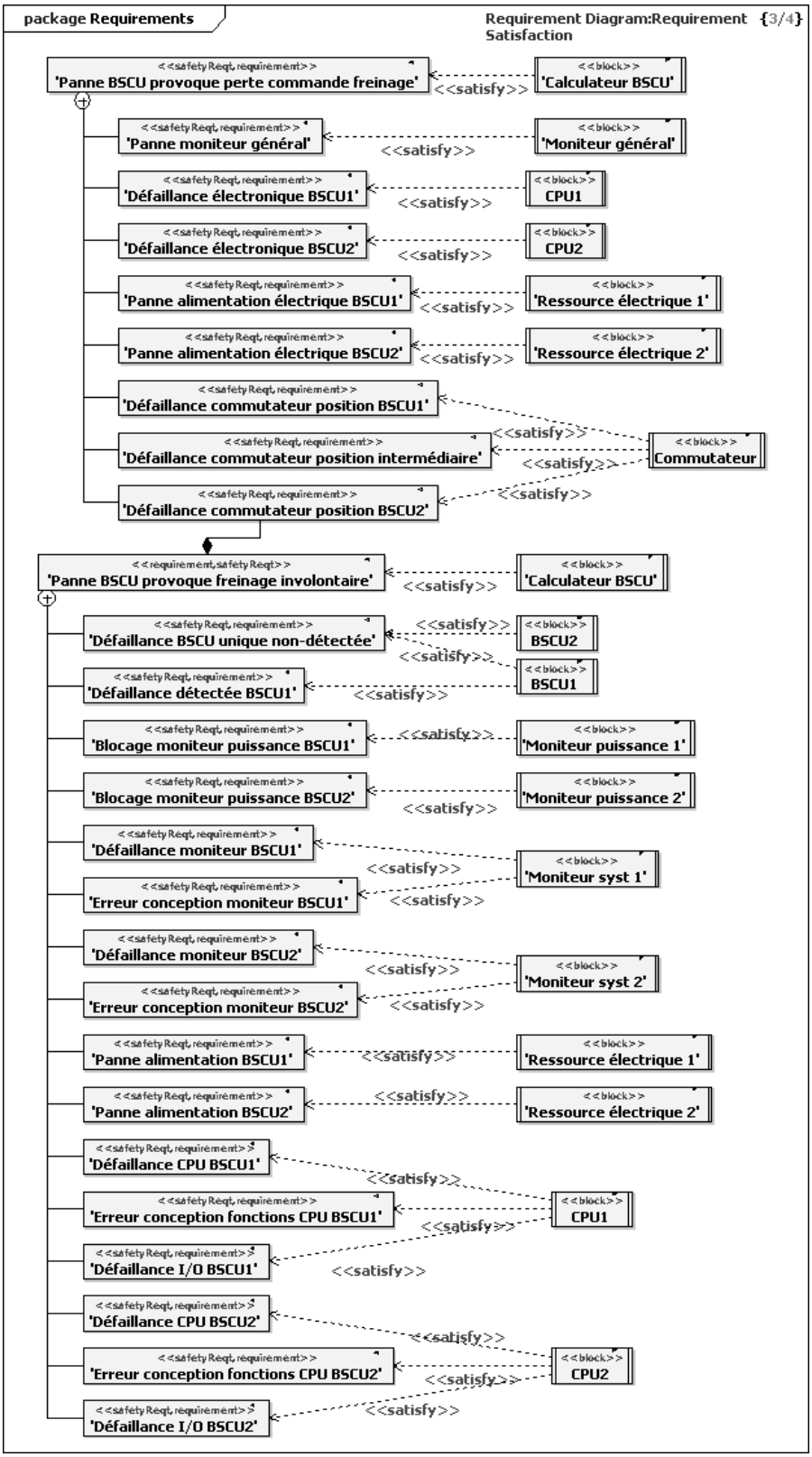


Figure B.6 : Calculateur BSCU – Diagramme de satisfaction des exigences

Dans ce diagramme, nous avons choisi de ne représenter que les affectations des exigences concernées par les déclinaisons présentées au paragraphe précédent. Ainsi, les exigences haut-niveaux sont liées au système « calculateur BSCU ». Quant aux autres exigences, elles sont connectées aux sous-systèmes qui les concernent (moniteur général, CPU1, CPU2, ressource électrique 1, ressource électrique 2, commutateur, etc.).

2.4. Diagramme de vérification des exigences

La Figure B.7 montre les relations de vérification entre les opérations de V&V (qui sont stéréotypés par *TestCase*) et les exigences système. Pour l'exemple, seul les deux exigences système à la base des déclinaisons ont été considérées dans ce diagramme. Les opérations de V&V correspondent alors à des analyses de SdF. Elles sont définies dans le package « *V&V* » présenté ci-après.

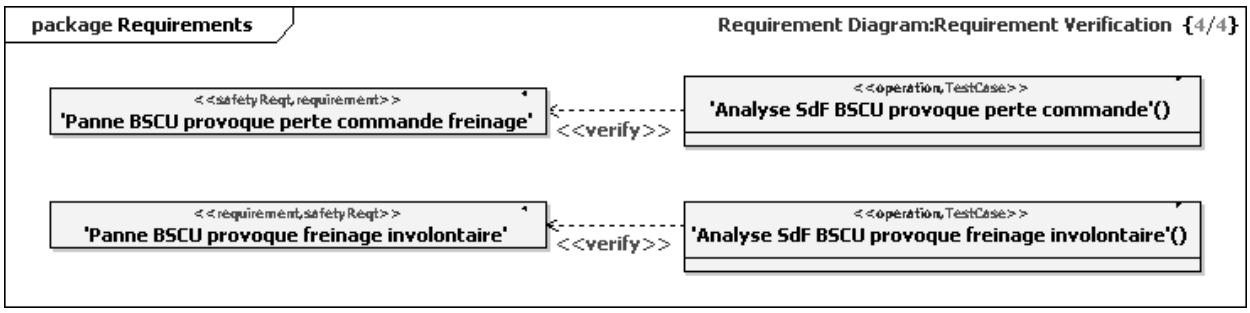


Figure B.7 : Calculateur BSCU – Diagramme de vérification des exigences

Ainsi, l'exigence nommée « *panne BSCU provoque perte commande freinage* » sera vérifiée, d'après le diagramme, par une opération nommée « *analyse SdF BSCU provoque perte commande* ».

3. Package de V&V : les cas de tests

Le dernier package à présenter est celui de V&V, qui renferme les cas de tests. Celui-ci contient le diagramme de la Figure B.8, qui définit les deux opérations de V&V entrevues au paragraphe 2.4 de ce chapitre. A ces opérations, nous associons des commentaires dans le diagramme des *TestCases* afin de les spécifier. Comme nous pouvons le voir sur la Figure B.8, ces commentaires sont les suivants :

- Pour l'opération « **analyse SdF BSCU provoque perte commande** » : analyser avec l'arbre de défaillance de l'événement racine 'panne du calculateur BSCU provoque perte de commande de freinage' (cf. Figure V.10 et Figure V.11),
- Pour l'opération « **analyse SdF BSCU provoque freinage involontaire** » : analyser avec l'arbre de défaillance de l'événement racine 'panne du calculateur BSCU provoque freinage involontaire' (cf. Figure V.12, Figure V.13 et Figure V.14).

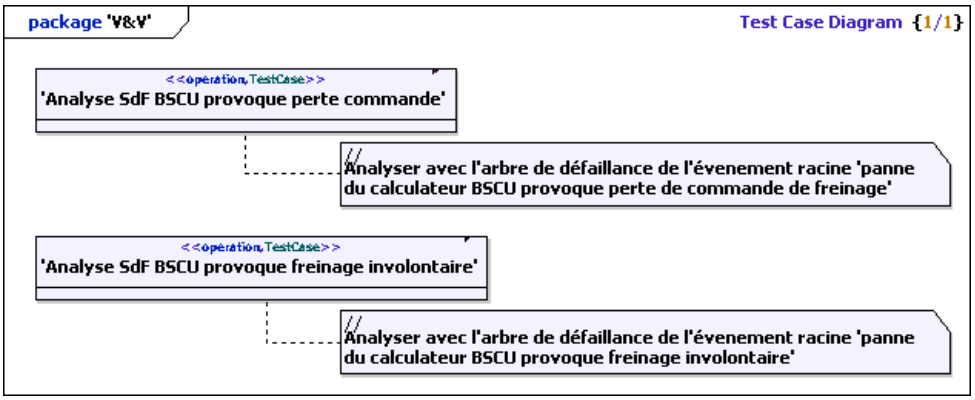


Figure B.8 : Calculateur BSCU – Cas de tests

Integration of Dependability in System Engineering Processes

Abstract:

The integration of various technologies, including computer and electronics, makes the nowadays designed systems increasingly complex. They have behaviors which are more elaborate and difficult to predict, they have a greater number of components in interaction and/or perform highest level functions. Parallel to this increasing complexity of these systems, the competitive of the global market imposes strong constraints of cost and time to the system developers. Other strong constraints deal with the quality of these systems, especially when they involve human risks or significant financial risks.

Thus, developers are forced to adopt a rigorous design approach to meet the desired system requirements and satisfy the various constraints (cost, time, quality, dependability...). Several methodological approaches to guide the system design are defined through system engineering standards. Our work is based on the EIA-632 standard, which is widely used, especially in the aeronautical and military fields. It is to improve the systems engineering process described by the EIA-632, in order to incorporate a global and explicit consideration of dependability. Indeed, till now the dependability was achieved by reusing generic models after having studied and developed independently each function. So there was no specific consideration of the risks associated with the integration of several technologies. For this reason, we propose to concern ourselves with the dependability requirements at the global level and as early as possible in the development phase. Then, these requirements will be decline to lower levels. We based our approach on the processes of the EIA-632 standard that we expand. We also propose an original method for the declination of the dependability requirements based on fault trees and FMEAC, and an information model based on SysML in order to support our approach. An example from the aeronautical field illustrates our proposals.

Keywords: System engineering, Processes, EIA-632, Dependability, FMECA, Fault trees, Requirements, Information model, SysML.

AUTEUR : Romaric GUILLERM

TITRE : Intégration de la Sûreté de Fonctionnement dans les Processus d'Ingénierie Système

DIRECTEURS DE THESE : Hamid DEMMOU et Nabil SADOU

LIEU ET DATE DE SOUTENANCE : LAAS-CNRS, 15 juin 2011

RESUME

L'intégration de diverses technologies, notamment celles de l'informatique et l'électronique, fait que les systèmes conçus de nos jours sont de plus en plus complexes. Ils ont des comportements plus élaborés et plus difficiles à prévoir, ont un nombre de constituants en interaction plus important et/ou réalisent des fonctions de plus haut niveau. Parallèlement à cette complexification des systèmes, la compétitivité du marché mondial impose aux développeurs de systèmes des contraintes de coût et de délais de plus en plus strictes. La même course s'opère concernant la qualité des systèmes, notamment lorsque ceux-ci mettent en jeu un risque en vies humaines ou un risque financier important.

Ainsi, les développeurs sont contraints d'adopter une approche de conception rigoureuse pour répondre aux exigences du système souhaité et satisfaire les diverses contraintes (coût, délais, qualité, sûreté de fonctionnement,…). Plusieurs démarches méthodologiques visant à guider la conception de système sont définies par l'intermédiaire de normes d'Ingénierie Système. Notre travail s'appuie sur la norme EIA-632, qui est largement employée, en particulier dans les domaines aéronautique et militaire. Il consiste à améliorer les processus d'ingénierie système décrits par l'EIA-632, afin d'intégrer une prise en compte globale et explicite de la sûreté de fonctionnement. En effet, jusqu'à présent la sûreté de fonctionnement était obtenue par la réutilisation de modèles génériques après avoir étudié et développé chaque fonction indépendamment. Il n'y avait donc pas de prise en compte spécifique des risques liés à l'intégration de plusieurs technologies. Pour cette raison, nous proposons de nous intéresser aux exigences de Sûreté de Fonctionnement au niveau global et le plus tôt possible dans la phase de développement, pour ensuite les décliner aux niveaux inférieurs, ceci en s'appuyant sur les processus de la norme EIA-632 que nous étoffons. Nous proposons également une méthode originale de déclinaison d'exigences de sûreté de fonctionnement à base d'arbres de défaillances et d'AMDEC, ainsi qu'un modèle d'information basé sur SysML pour appuyer notre approche. Un exemple issu du monde aéronautique permet d'illustrer nos propositions.

MOTS-CLES

Ingénierie système, Processus, EIA-632, Sûreté de fonctionnement, AMDEC, Arbres de défaillances, Exigences, Modèle d'information, SysML.

DISCIPLINE : Systèmes Industriels

LAAS-CNRS

7, avenue du Colonel Roche - 31077 TOULOUSE Cedex 4 – France
Tél. : +33 (0)5 61 33 32 00 - Fax. : +33 (0)5 61 55 35 77 - Mail : laas-contact@laas.fr - http://www.laas.fr