Modélisation des dépendances fonctionnelles pour l'analyse des risques de niveau avion.

S. Maitrehenry

École Nationale Supérieure de Mécanique et d'Aérotechnique

Laboratoire d'Informatique et d'Automatique pour les Systèmes

THESE

pour l'obtention du Grade de

DOCTEUR DE L'ÉCOLE NATIONALE SUPÉRIEURE DE MÉCANIQUE ET D'AEROTECHNIQUE

(Faculté des Sciences Fondamentales et Appliquées)

(Diplôme National — Arrêté du 7 août 2006)

École Doctorale : Sciences et Ingénierie pour l'Information, Mathématiques

Secteur de Recherche : INFORMATIQUE ET APPLICATIONS

Présentée par :

Sébastien MAITREHENRY

****

Modélisation des dépendances fonctionnelles pour l'analyse des risques de niveau avion

****

Directeur de Thèse : **Yamine AIT-AMEUR**

****

Soutenue le 04/10/2013

devant la Commission d'Examen

****

JURY

Rapporteurs :	**Nicole Levy**	Professeur au CEDRIC-CNAM
	Philippe Palanque	Professeur à l'université Paul Sabatier - IRIT
Examinateurs :	**Yamine Ait-Ameur**	Professeur à l'ENSEEIHT
	Bieber Pierre	Chargé de recherche ONERA
	Sylvain Metge	Ingénieur AIRBUS
	Jérémy Guiochet	Maitre de conférence à l'université Paul Sabatier - LAAS
	Patrick Girard	Professeur à l'université de Poitiers/ENSMA - LIAS

Remerciements

Je tiens tout d'abord à remercier les trois personnes qui m'ont assisté et soutenu au quotidien dans mes travaux de thèse :

- Merci à Sylvain Metge qui m'a permis de réaliser ma thèse dans les meilleures conditions possibles. Sa rigueur, son esprit critique, sa vision industrielle et pragmatique ainsi que ses encouragements m'ont énormément apporté. Je n'oublie pas non plus son attachement pendant la thèse à un juste équilibre entre la recherche et les applications industrielles. Ce fut pour moi une chance de l'avoir à mes cotés chez Airbus pendant les trois années de ma thèse CIFRE.
- Merci à Pierre Bieber qui m'a beaucoup appris sur la sûreté de fonctionnement et la modélisation en AltaRica. Sa gentillesse, sa pédagogie et sa patience m'ont permis d'appréhender sereinement les nombreuses embûches que j'ai rencontrées au cours de cette thèse. Il s'est montré très disponible et a su s'accommoder de mes longs moments passés chez Airbus.
- Merci à Yamine Ait-Ameur dont l'expertise en ingénierie des modèles a été une aide très précieuse ! En s'appropriant rapidement les spécificités du domaine de la sûreté de fonctionnement, il a su me guider au mieux pour définir une approche de travail rigoureuse. Je retiens sa rigueur et sa patience, mais aussi son talent à me stimuler quand il le fallait.

Je remercie ensuite Frederic Marani, Charles Zamora et Luis Compte qui m'ont accueilli chez Airbus dans le service en charge de la définition du processus de sûreté de fonctionnement et de la réalisation des activités de sécurité de niveau avion. Grâce à eux j'ai pu faire mes premiers pas dans l'industrie. J'ai été parfaitement intégré dans cette grande équipe où se côtoient Français, Allemands, Anglais, Italien et bien sûr Écossais. J'ai une pensée particulière pour François Pouzolz qui a quitté Airbus peu après mon arrivée, mais dont les travaux ont permis l'existence de ma thèse. Un grand merci également à Chris, Alexandro et Michel avec qui j'ai beaucoup partagé.

Je remercie l'équipe ALFA qui m'a intégré aux travaux de définition des architectures fonctionnelles des systèmes. Merci notamment au chef du projet, Elio Vecchione et à son collaborateur de la première heure Thierry qui m'a appuyé sur le cas d'étude du décollage.

Un grand merci aussi à Christel Seguin qui a suivit avec intérêt mes travaux et m'a aidé par ses remarques pertinentes.

Je n'oublie pas non plus les nombreuses personnes qui m'ont soutenues et m'ont changé les idées pendant ces trois années de thèse.

- Merci aux joueurs du club de go d'Airbus qui m'ont offert des moments de détente et de réflexion autour d'un goban. Je remercie particulièrement notre président de club, Daniel, qui m'a tout appris sur ce jeu.
- Merci aux doctorants de l'Onera avec qui je n'ai malheureusement pas passé autant de temps que je l'aurais voulu. Néanmoins nos soirées wok et jeux de sociétés resteront dans ma mémoire. J'ai une pensée particulière pour Antoine, son talent en pâtisserie, ses coups bas au badminton et sa bonne humeur. Je pense également à Pierre et à sa fameuse phrase « je suis complètement foutu » prononcé vingt fois au cours de chaque partie pour finir bien souvent par une victoire écrasante.
- Merci aux doctorants du laboratoire LIAS de l'ENSMA que j'ai appris à connaître lors des séminaires de l'école doctorale. Je remercie particulièrement Youness pour son aide administrative, notamment le jour de la soutenance.
- Merci à mes amis qui m'ont changé les idées grâce à des soirées, week-ends et vacances toujours géniales. Merci notamment aux ex-Night Riders qui se reconnaîtront et à ma bestfriend Fanny.

Enfin je remercie ma famille et notamment mes parents. Vraiment, merci pour votre amour et vos encouragements. Je vous dédies ce manuscrit de thèse.

Table des matières

Définition et synthèse de modèles pour l'analyse des risques

Introduction

Le 4 novembre 2010, un A380-842 d'Airbus de la compagnie australienne Qantas, effectuant un vol commercial de Singapour à Sydney avec 440 passagers à bord et 29 membres d'équipage, a du retourner à l'aéroport de Singapour à la suite d'une panne moteur. Quatre minutes après le décollage, une explosion non contenue du disque de la turbine à pression intermédiaire du deuxième réacteur a provoqué de nombreux dégâts, dont principalement la destruction du réacteur, la perforation de l'aile gauche et l'endommagement du système de distribution du kérosène. L'avion étant resté contrôlable, l'équipage a pu faire atterrir l'appareil. L'incident n'a fait aucune victime parmi les occupants de l'appareil. Seules deux personnes au sol ont été blessées par des chutes de débris lors de l'atterrissage. L'agence australienne de la sécurité des transports a établi que la cause de cet incident est un « défaut de fabrication des turbines haute pression et à pression intermédiaire, là où le circuit distribue l'huile pour les roulements des arbres » [ATSB, 2010].

Cet événement – comme beaucoup d'autres, parfois plus dramatiques, et issus de différents domaines industriels (l'industrie chimique, le nucléaire, le transport ferroviaire ou automobile etc.) – met en lumière l'importance d'étudier les répercussions de pannes, au cours de la conception des systèmes, et en particulier de l'avion, pour déterminer les moyens de réduction des risques d'accidents et d'incidents. Cette problématique est l'affaire de tous les ingénieurs travaillant dans de nombreux domaines techniques différents liés aux systèmes étudiés. Par exemple, pour un avion : les moteurs, les commandes de vol, le cockpit, la structure etc. Ces ingénieurs sont assistés par des méthodes et des spécialistes de la **sûreté de fonctionnement**, qui est la discipline qui permet l'analyse et la maîtrise des pannes.

Il existe de nombreux moyens pour améliorer la sûreté de fonctionnement d'un produit comme les avions : renforcement de la robustesse du produit aux pannes, amélioration des procédures d'utilisation (définir en particulier des procédures spécifiques en cas de pannes), formation des pilotes et des opérateurs de maintenance etc. Le choix des moyens qui seront mis en œuvre découle d'exigences formulées lors des phases préliminaires de la conception. Parmi ces exigences, certaines sont émises par des spécialistes de la sûreté de fonctionnement grâce aux **analyses des risques**.

Cette thèse, initiée par Airbus en collaboration avec l'ONERA (Toulouse) et l'ISAE ENSMA (Poitiers), a pour objectif de proposer des méthodes et outils pour renforcer les analyses des risques. Partant du constat que ces analyses nécessitent une parfaite compréhension du fonctionnement du produit, notre approche consiste à représenter formellement son comportement à l'aide de **modèles**, dans le cadre de l'Ingénierie Dirigée par les

Modèles. Le produit, considéré comme un **système socio-technique**, est décrit par les tâches réalisées par ses utilisateurs et ses composants matériels et logiciels. De tels modèles sont déjà développés et exploités par les concepteurs de l'avion pour faire entre autre des analyses des besoins et des performances. Cependant, ces modèles ne permettent pas de réaliser des analyses outillées des conséquences des pannes sur les systèmes. Notre objectif est double.

- Ajouter une description formelle de la propagation des pannes dans le système afin d'assister l'analyse des risques. Le modèle formel permet de mieux évaluer les conséquences d'une panne fonctionnelle et de rechercher les combinaisons de pannes les plus critiques.
- Formaliser les liens entre les modèles de la conception et les modèles de sécurité en se basant sur la notion de transformation de modèles issue de l'Ingénierie Dirigée par les Modèles. Ce lien permet de renforcer la traçabilité des exigences et de faciliter le développement des modèles de sécurité.

Ce manuscrit de thèse est composé de huit chapitres. Les trois premiers chapitres présentent le cadre de la thèse et explicitent les objectifs. Le premier chapitre définit précisément les notions propres au domaine de la sûreté de fonctionnement puis donne les détails de l'analyse des risques telle qu'elle est réalisée dans l'industrie aéronautique. Dans ce chapitre, nous présentons aussi les axes d'optimisation que nous avons identifiés pour cette analyse. Comme notre proposition d'amélioration de l'analyse des risques s'appuie sur les modèles, le deuxième chapitre introduit les outils de l'ingénierie des modèles permettant de construire et de manipuler des modèles. Puis, dans le deuxième chapitre, nous présentons des modèles développés pour assister les phases préliminaires de la conception des systèmes et des modèles utilisés actuellement pour renforcer les analyses de sûreté de fonctionnement. Le chapitre 3 définit les objectifs de la thèse et introduit notre approche, à savoir, développer et générer de nouveaux modèles pour assister l'analyse des risques à partir des modèles de la conception.

Les cinq chapitres suivants développent les contributions théoriques et pratiques. Dans le chapitre 4, nous formalisons les concepts des modèles de sûreté de fonctionnement que nous proposons pour assister l'analyse des risques. Puis, le chapitre 5 montre comment ces concepts formalisés peuvent être modélisés à l'aide du langage AltaRica. Nous présentons également des méthodes d'analyse de ces modèles. Comme la génération des modèles AltaRica découle des modèles de la conception des systèmes, il est particulièrement intéressant de pouvoir générer les modèles AltaRica à partir des modèles de la conception. Le chapitre 6 présente la théorie et la mise en pratique de cette passerelle entre les deux activités de modélisation. Dans le chapitre 7, nous réalisons un déploiement de la nouvelle activité de modélisation en AltaRica et de la transformation de modèles sur un cas d'étude industriel : l'interruption du décollage pour éviter une situation dangereuse[1], telle qu'une sortie de piste. Cette mise en application permet de valider l'intérêt des modèles proposés pour les analystes de la sûreté de fonctionnement. Puis, nous présentons au chapitre 8 une possible extension des modèles de sécurité à la prise en compte des Facteurs Humains.

1 « *Si vous êtes en danger, abandonnez quelque chose* », sixième précepte du jeu de go par Wang Chi-Shin

Première partie

Vers des analyses de sécurité assistées par les modèles

Chapitre 1

La sûreté de fonctionnement dans l'aéronautique

Sommaire

Résumé. Le contexte applicatif de nos travaux est celui de l'étude de la sécurité des architectures de la conception aéronautique. Ce chapitre présente les fondements de la sûreté de fonctionnement et les analyses de sécurité qui sont réalisées lors de la conception d'un nouvel avion. Nos travaux se concentrent en particulier sur l'analyse des risques dont l'objectif est d'identifier les risques potentiels que peut subir l'avion au cours de son exploitation.

1 Les objectifs de la sûreté de fonctionnement

Avant de présenter, à partir de la section 2, les analyses de sécurité réalisées au cours de la conception d'un avion, nous définissons la sûreté de fonctionnement et les notions sur lesquelles elle s'appuie.

1.1 Les principes de la sûreté de fonctionnement

Il est impossible de définir avec précision la sûreté de fonctionnement sans définir au préalable ce sur quoi elle s'applique, c'est à dire la notion de système. De nombreuses définitions existent dans la littérature. Nous retenons celles de [Villemeur, 1988], de [Laprie et al., 1995], de [Mo, 1997] et de [Kehren, 2005] pour proposer la définition suivante.

Définition : Système et système socio-technique

Un système est un ensemble déterminé d'éléments en interaction. Il peut être constitué de sous-systèmes et lui-même être en interaction avec d'autres systèmes. L'environnement d'un système désigne tous ces éléments extérieurs.

Dans la littérature, en particulier dans le domaine des Facteurs Humains, l'ensemble Homme-Machine est également appelé système ou plus précisément système socio-technique.

Dans l'industrie aéronautique, le terme « système » désigne plus spécifiquement des ensembles d'éléments à un niveau de détail bien défini, appelé niveau système (nous reviendrons sur la notion de système aéronautique dans la section 2.1 de ce chapitre). Les notions présentées dans la section 1 de ce chapitre sont applicables à tous systèmes socio-techniques et pas seulement aux systèmes aéronautiques.

Exemple. A partir de la définition précédente, l'avion et ses occupants est un système socio-technique, composé de nombreux systèmes aéronautiques et lui-même pouvant faire partie de systèmes plus vastes tels que l'ensemble des avions composant la flotte d'une compagnie aérienne ou un aéroport avec tout ce qui le compose (avions, infrastructure aéroportuaire, passagers etc.).

1.1.1 Définition de la sûreté de fonctionnement

Dans [Laprie et al., 1995], J.-C. Laprie a formulé une définition de la sûreté de fonctionnement d'un système informatique. Cette définition s'adapte à tous les types de système, quelque soit leur périmètre et leurs utilisateurs (humains ou machines) :

Définition : Sûreté de Fonctionnement (SdF)

La sûreté de fonctionnement est le domaine d'étude qui permet aux utilisateurs d'un système de placer une confiance justifiée dans le comportement de ce système.

Le pilier des études de sûreté de fonctionnement est la **maîtrise des risques**. Un risque est défini dans [Villemeur, 1988] comme une *mesure d'un danger associant une mesure de l'occurrence d'un événement indésirable et une mesure de ses effets ou conséquences*. Les risques étudiés sont très diversifiés car ils dépendent des conséquences observées : coût humain, économique, écologique etc.

En fonction des risques que nous souhaitons maîtriser sur un système donné, nous pouvons être amenés à nous intéresser à différentes facettes, non disjointes, de la sûreté de fonctionnement, dont principalement :

- la sécurité : Les études de sécurité visent à maîtriser les conséquences néfastes pour les utilisateurs et d'autres éventuels acteurs extérieurs, de dysfonctionnements internes ou d'événements extérieurs au système étudié. Comme nous l'avons illustré au début de cette section, certains dysfonctionnements peuvent avoir des conséquences dramatiques sur le plan humain, en particulier dans les domaines du transport ou de l'énergie par exemple.
- la disponibilité / la fiabilité : Les études de disponibilité et de fiabilité opérationnelle s'intéressent au maintien de l'utilisation du système considéré. Il s'agit de garantir que le système est utilisable quand l'exploitant en a besoin. Des événements indésirables peuvent en effet perturber le bon fonctionnement d'un système et le rendre indisponible à l'utilisation temporairement ou définitivement. Les conséquences peuvent par exemple être des pertes financières pour l'exploitant.
- la sûreté : Les études de sûreté se concentrent sur les violations du système par des éléments extérieurs malveillants, tels que le piratage de données ou le crochetage d'une porte. Ces risques ont la particularité d'être intentionnels, car ils résultent d'attaques ciblées, à l'inverse des dysfonctionnements internes qui sont aléatoires.
- la maintenabilité : Les études de maintenabilité s'assurent que le système puisse être réparé. De nombreux événements indésirables rendent nécessaire une réparation du système. Il faut alors garantir que ces réparations ne deviennent pas irréalisables à cause des événements considérés.

Nos travaux s'inscrivent essentiellement dans le cadre des études de la sécurité des avions. Ces études, réalisées pendant la phase de conception, ont deux objectifs : permettre le développement d'avions robustes aux pannes et démontrer publiquement leur robustesse.

1.1.2 Taxonomie de la sûreté de fonctionnement

Les analyses de sécurité se focalisent sur différents concepts pour étudier avec précision les dysfonctionnements internes des systèmes. [Laprie et al., 1995] définit plusieurs termes largement employés aujourd'hui. Une **défaillance** d'un système est la conséquence immédiate sur le système de son dysfonctionnement. Il s'agit d'une déviation du système par rapport à son comportement prévu. Un système peut donc défaillir de différentes façons, selon son comportement attendu, son environnement et le point de vue des utilisateurs. La cause originelle d'une défaillance est la **faute**. Il existe de nombreuses familles de fautes potentielles : fautes internes ou externes au système, fautes accidentelles ou intentionnelles, fautes permanentes ou temporaires etc. Les fautes n'entraînent que rarement une défaillance immédiate. Elles créent une ou plusieurs perturbations dans le système. Ces dernières, appelées **erreurs**, sont susceptibles de

se propager dans le système jusqu'à provoquer éventuellement la défaillance du système. Certaines erreurs n'ont parfois aucun effet néfaste pour le système, par exemple dans le cas de données erronées écrasées et remplacées par de nouvelles données saines.

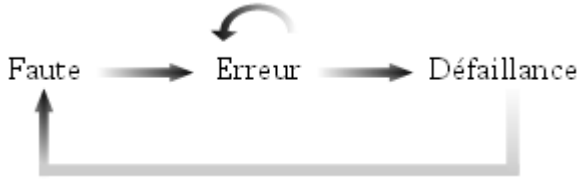


Fig. 1.1 - Pathologie des fautes

La chaîne de dégradation d'un système, de la faute à la défaillance, est récursive, puisque la défaillance d'un système peut impulser une faute dans un autre système. Les flèches de la figure 1.1, ci-dessus, représentent les liens de causalité entre les fautes, les erreurs et les défaillances. Il est également important de noter que plusieurs erreurs peuvent se succéder avant de provoquer éventuellement une défaillance.

Dans le domaine industriel, le vocabulaire scientifique présenté précédemment est rarement utilisé. Le terme « faute » en particulier peut-être ambiguë s'il est employé lors de l'expertise d'un incident ; il peut alors signifier une responsabilité morale voire pénale. Nous utilisons alors plus communément le terme « panne » (« *failure* » en anglais) pour regrouper l'ensemble des trois termes précédents. La totalité de ce manuscrit s'appuie sur la définition suivante des pannes (basée sur le document [CS-25 1309]) :

Définition : Panne

Une panne d'un système est l'ensemble composé d'un événement indésirable, causant un comportement du système différent de celui attendu, et de ses conséquences perçues par le système et ses utilisateurs.

Nous retrouvons dans ce terme les trois notions définies précédemment. Nous désignerons une faute par la cause d'une panne et une défaillance par la conséquence d'une panne. Nous utilisons ensuite la notion de propagation de pannes au lieu de propagation d'erreurs. D'ailleurs, si le terme « erreur » est souvent utilisé dans le domaine de la sécurité industrielle, il peut avoir une signification différente (voir le paragraphe 3.2.1 de ce chapitre pour la définition du mode de fonctionnement erroné). Ce terme est également utilisé dans l'expression « erreur humaine » pour désigner une faute commise par un acteur humain du système.

1.2 Importance de la sécurité dans l'aéronautique

Durant le dernier siècle, l'augmentation du trafic aérien s'est sensiblement accru pour faire face à une demande de transport de plus en plus forte. Non seulement il y a de plus en plus de vols, mais la capacité de transport et le confort n'a eu de cesse d'augmenter. Ainsi, ce moyen de transport déplace de plus en plus de passagers au fil des années. Cela montre la grande confiance que les passagers ont en ce moyen de déplacement, malgré le fait que les accidents les plus graves marquent fortement les esprits à cause du

nombre généralement important de victimes. L'avion évolue en effet dans un environnement à forts risques et très hostile et incompatible avec la physiologie de l'être humain : vitesse et altitude importantes, raréfaction de l'oxygène, pression atmosphérique faible, température excessivement basse, etc.

1.2.1 La sécurité des avions

La confiance des passagers se fonde sur les nombreuses études de sécurité réalisées et publiées par les avionneurs et des organismes indépendants. Les accidents et les incidents sont relevés et analysés pour renforcer la sécurité des avions en service et en développement. La figure 1.2, publiée dans le rapport annuel de 2011 de l'EASA ([EASA, 2011]), s'appuie sur les données capitalisées par l'OACI, l'Organisation de l'Aviation Civile Internationale, depuis 1945.

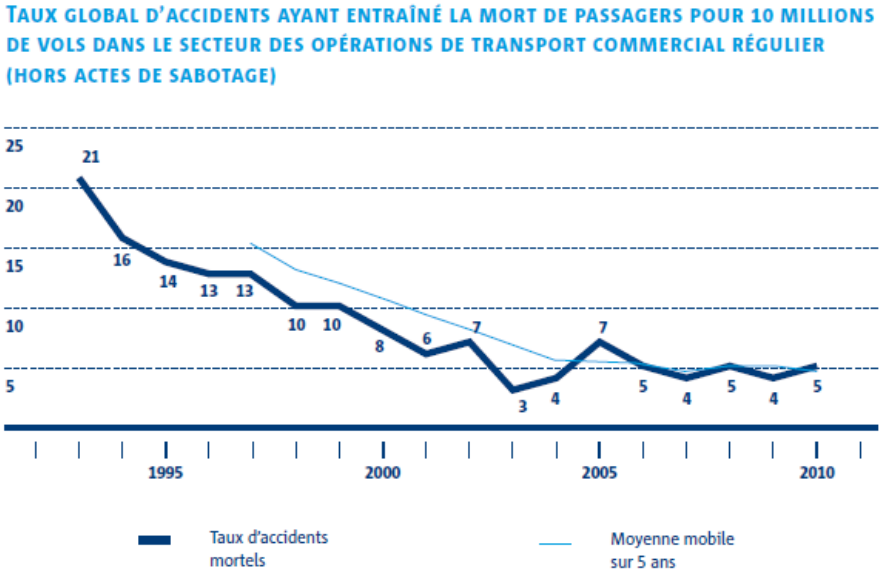


Fig. 1.2 - Évolution du taux d'accident mortel dans le transport aérien (Source : EASA)

Ces données montrent que le taux d'accidents mortels par nombre de vols a fortement diminué jusqu'en 2003 pour se stabiliser depuis autour de 5 accidents mortels tous les 10 millions de vols.

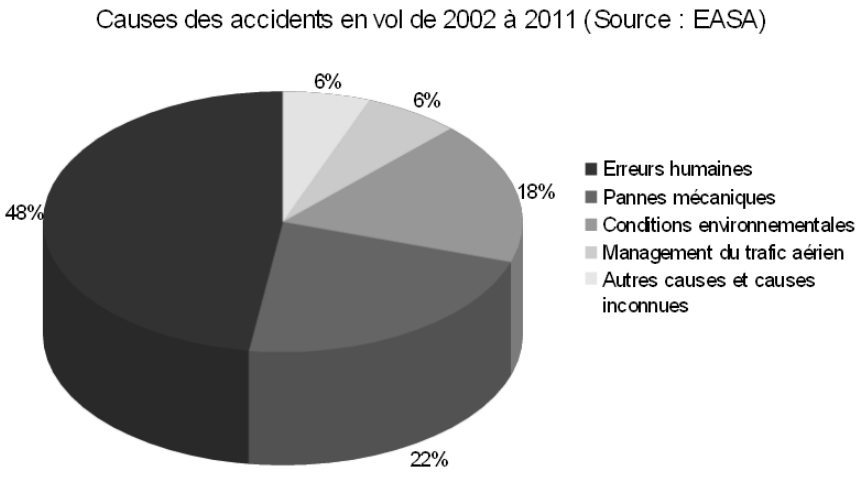


Fig. 1.3 - Ordre de grandeur des causes des accidents aériens

Les premières causes d'incidents et d'accidents sur des avions, pendant la dernière décennie, sont les erreurs humaines à bord de l'appareil (ordre de grandeur de 50%) et les pannes systèmes (ordre de grandeur de 25%), devant l'environnement, les problèmes de maintenance ou de contrôle aérien. Cela montre que les futurs efforts à réaliser, pour rendre le transport aérien encore plus sûr, est d'améliorer les interfaces entre les utilisateurs humains et les systèmes et de renforcer la robustesse des systèmes.

1.2.2 La certification

Afin de garantir un niveau de confiance dans le transport aérien, les autorités de certification définissent des exigences de sécurité qui doivent être respectées par l'ensemble du monde aérien, c'est à dire essentiellement, pour les appareils gros porteurs, les avionneurs, les compagnies aériennes et le contrôle aérien. La démonstration de la tenue des exigences de sécurité s'appelle la certification. Elle permet d'obtenir l'autorisation de voler ([EASA, 2011]).

Les autorités de certification

Les autorités de certification sont indépendantes des avionneurs et des compagnies aériennes. Elles réglementent l'exploitation des avions dans la zone géographique dont elles ont la charge. Il en existe de nombreuses à travers le monde, néanmoins, les deux principales agences sont :

- L'agence européenne EASA (« *European Aviation Safety Agency* »), créée en 2003 à la suite de la JAA (« *Joint Aviation Authorities* »).
- L'agence américaine FAA (« Federal Aviation agency »), créée en 1958.

L'organisation de l'Aviation Civile Internationale (OACI) a été créé en 1944 par plusieurs pays afin que les différentes autorités de certification du monde s'entendent sur des règles communes. Cette agence dépend des Nations Unies. Son rôle est la définition de normes réglementaires pour le transport des biens et des personnes. Elle définit entre autre les règles de l'air et les impératifs de sécurité. Ainsi, même si chaque autorité de certification publie ses propres documents de réglementation, de nombreuses exigences sont communes ou liées par correspondance. Les principales exigences pour la sûreté de fonctionnement des systèmes sont définies dans le paragraphe 1309 du document CS-25 de l'EASA ([CS-25 1309]) et du document FAR-25 de la FAA.

Les groupes de travail sur la sécurité

Les principaux avionneurs et équipementiers collaborent dans des groupes de travail pour étudier et définir ensemble des processus de développement, des méthodes et des outils afin de répondre au mieux aux exigences de sécurité des autorités de certification : le groupe WG-63 de l'EUROCAE (« *European Organisation of Civil Aviation Equipment* ») et le groupe S-18 de la SAE (« *Society of Automotive Engineers* »). Ces groupes élaborent deux documents, dits ARP (« *Aerospace Recommended Practice* »).

- [ARP4754A / ED-79A] : ce document fournit des recommandations en terme de processus pour garantir la sécurité des avions lors de la conception.

– [ARP4761 / ED-135] : ce document propose des méthodes acceptées par les autorités pour évaluer la sécurité des avions au sein du processus décrit dans le document précédent.

La suite du chapitre 1 s'appuie fortement sur le document [ARP4754A / ED-79A] pour présenter le processus d'analyse de la sécurité des avions en cours de développement, tel qu'appliqué par Airbus.

2 Analyses de sécurité réalisées pendant la conception aéronautique

La sécurité est analysée au cours de toute la vie de l'avion. Par exemple, des investigations sont réalisées sur chaque accident et incident afin de prévenir à l'avenir que de tels événements se répètent. Dans la suite, nous nous concentrons uniquement sur les analyses de sécurité réalisées pendant la phase de conception des avions. Cette section présente simultanément les processus de la conception et de la sécurité.

2.1 Le processus de la conception et de la sécurité

Le processus de sécurité, réalisé au cours de la conception, suit et supporte la conception lors de ses différentes étapes, des premiers concepts à l'implémentation.

2.1.1 Introduction du processus de la conception

La conception aéronautique suit le développement d'un cycle en V. La figure 1.4 en donne une vision globale, des premiers concepts à l'implémentation et la mise en production.

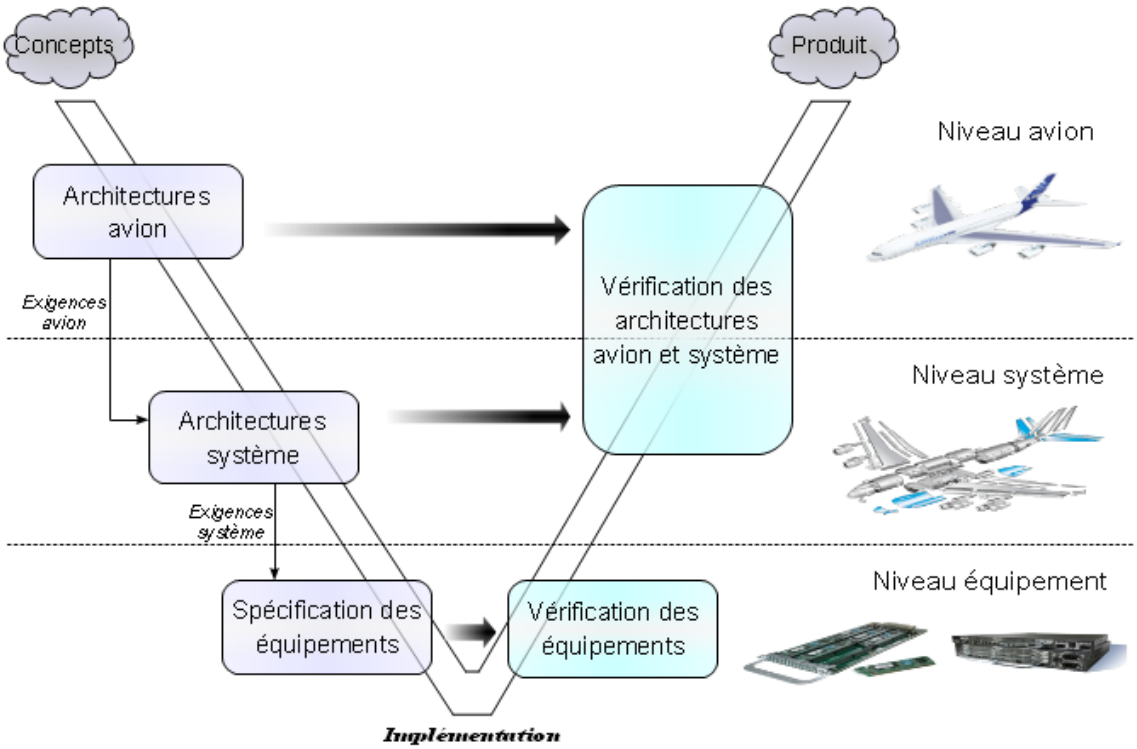


Fig. 1.4 - Processus de la conception aéronautique

Une fois les grands concepts identifiés au cours d'avant-projets, la conception d'un nouvel avion est guidée par différentes étapes de définition permettant de détailler et préciser de plus en plus le produit. Trois niveaux d'étude sont mis en évidence. Le niveau le plus bas est celui des **équipements**, tels que le matériel électronique, les logiciels embarqués ou encore les éléments de structure. Les équipements sont combinés pour former des **systèmes** de l'avion. On peut citer par exemple les moteurs, la roulette de direction ou encore la gouverne de direction. Les systèmes de l'avion interagissent entre eux pour gérer le comportement global de l'avion. Cette intégration des systèmes entre eux est étudiée au niveau **avion**.

Dans la phase descendante du cycle en V, la conception propose des **architectures** de définition de l'avion, puis de ses systèmes, jusqu'à une spécification précise de tous les équipements constituant tous les systèmes de l'avion. Aux niveaux avion et système, les architectures sont abstraites par rapport aux solutions matérielles réalisées par la suite. Elles s'appuient sur la notion de fonction. Ce terme est utilisé dans de nombreux domaines de l'ingénierie, sans qu'il en existe une définition commune et précise. Dans la suite, nous nous appuierons sur la définition suivante, tirée du document [ARP4754A / ED-79A].

Définition : Fonction

> *Une fonction est un comportement attendu d'un produit, basé sur ses spécifications. Elle est indépendante de toute solution d'implémentation.*

La vérification de la conception est la phase remontante du cycle en V. Dans cette étape de vérification, les différents domaines d'expertise de l'avion démontrent que les architectures conçues ont un comportement correct selon leur point de vue. Par exemple, les analystes de la sécurité sont en charge de la vérification de la robustesse aux pannes des architectures.

2.1.2 Conception par les exigences

Pour faire leur choix d'architecture et les vérifier, les concepteurs s'appuient sur la notion d'exigence. Elle permet de spécifier les fonctions et les composants de telle manière que le bon choix d'architecture s'imposera alors comme le meilleur compromis entre différentes options d'architectures envisagées, chacune devant toutefois respecter toutes les exigences. La conception basée sur les exigences est un principe datant des années 60. Depuis, de nombreuses méthodes et outils ont été développés pour assister la capitalisation et le traitement des exigences pendant toutes les phases du développement. [Robertson, 2006] présente une vision générale et des moyens d'aide à la gestion des exigences. Sa définition des exigences est conforme à celle des ARP. A partir de [ARP4754A / ED-79A], nous disposons de la définition suivante.

Définition : Exigence

> *Une exigence est liée à une fonction ou un composant matériel ou logiciel. Elle en représente une unique caractéristique précise, documentée, mesurable, pouvant être validée et tracée, et permettant de vérifier l'implémentation.*

Les exigences sur lesquelles s'appuient la conception ont diverses origines, telles que la performance, la sûreté, la maintenance et la sécurité. Chacun de ces domaines est un métier à part entière. Les concepteurs de l'avion prennent en compte les exigences des différentes spécialités puis déterminent les meilleures architectures par rapport à leurs concepts et ces exigences. Toutes les exigences, quelle que soit leur origine, servent à définir le niveau de détail suivant. Ainsi, les exigences avion, déterminées à partir des architectures et d'analyses de niveau avion, pilotent les choix d'architectures des différents systèmes composant l'avion.

Dans certains domaines les architectures proposées doivent ensuite être vérifiées par les spécialistes en vue de la certification. C'est le cas pour la sécurité. La gestion des exigences de sécurité par les spécialistes suit un processus clairement défini et validé par les autorités de certification.

2.1.3 Vue générale du processus de sécurité

La figure 1.5 décrit le processus de sécurité réalisé dans l'industrie aéronautique et en particulier chez Airbus. Ce processus suit un cycle en V, très proche de celui de la conception (voir la figure 1.4). Il est d'ailleurs réalisé en parallèle du cycle en V de la conception.

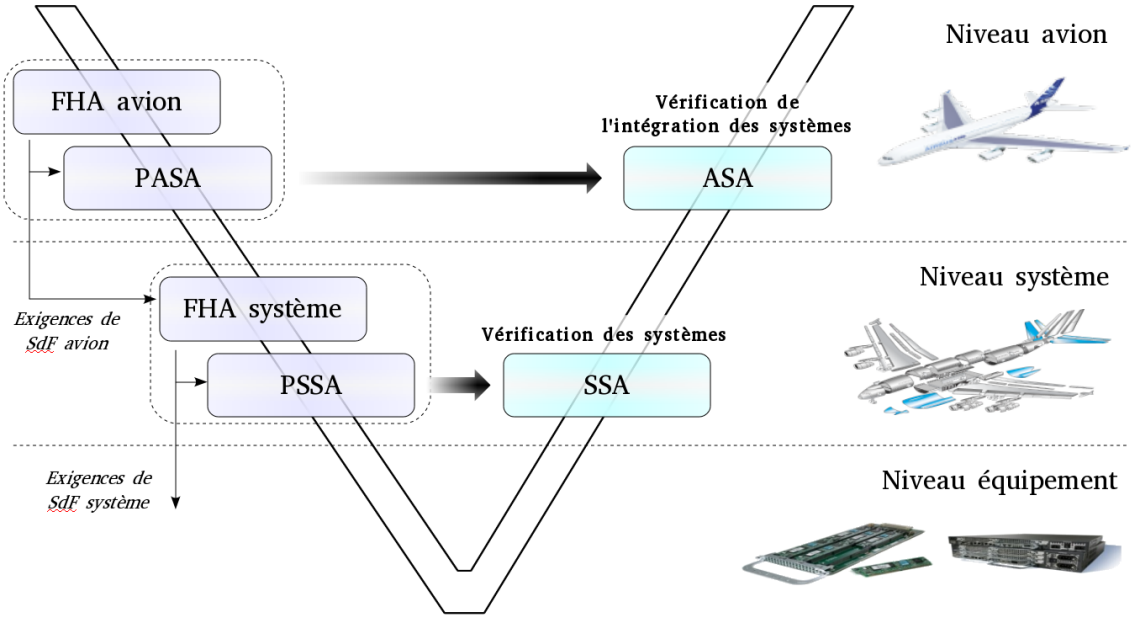


Fig. 1.5 - Processus de sécurité réalisé lors de la conception d'un avion

Le processus de sécurité est composé de plusieurs analyses, détaillées à la section 2.2. A chaque niveau d'étude, avion, système et équipement, il est possible d'identifier deux tâches majeures :

– L'identification et l'allocation des exigences de sécurité :

  L'identification des exigences de sécurité est réalisée dans la phase descendante du cycle en V. Elle est composée à chaque niveau de deux activités. La première identifie des risques potentiels, avec les FHA (Functional Hazard Assessment) et la seconde consiste en une analyse préliminaire des architectures proposées, avec les PASA/PSSA (Preliminary Aircraft/System Safety Assessment).

– La vérification que les architectures proposées respectent les exigences identifiées :

  La conception des équipements et des systèmes est réalisée en bas du cycle en V. La phase remontante du cycle est alors destinée à vérifier les choix de conception.

  Les équipements sont vérifiés par les équipementiers. L'avionneur est en charge de la vérification et de l'intégration de ces équipements au sein des systèmes, puis des interactions entre les systèmes à l'intérieur de l'avion tout entier. Les activités de vérification sont les ASA/SSA (Aircraft/System Safety Assessment), dans la continuité des travaux réalisés au cours des analyses préliminaires PASA/PSSA.

2.2 Activités du processus de sécurité

2.2.1 Activités de FHA

Les *Functional Hazard Assessment* (FHA), aussi appelées **analyses des risques**, sont des examens complets systématiques des fonctions afin d'identifier et de classifier les pannes fonctionnelles, selon leur sévérité. L'objectif de l'analyse FHA est d'exprimer les exigences de sécurité qui devront être tenues par lors de la conception et de la production. Dans les premières phases de la conception, une première analyse FHA de niveau avion est réalisée. Elle considère l'avion dans sa globalité, ce qui permet d'identifier des exigences de sécurité sur l'intégration des systèmes. Puis, différentes analyses FHA de niveau système sont réalisées pour traiter chaque système embarqué individuellement.

L'activité de FHA, étant au cœur de nos travaux, est présentée en détail dans la section 3 de ce chapitre.

2.2.2 Activités de PASA/PSSA

Les *Preliminary Aircraft/System Safety Assessment (PASA/PSSA)* sont les évaluations préliminaires de sécurité des systèmes et de l'avion. Ces analyses consistent à examiner les architectures proposées pour déterminer comment des pannes sur les éléments de ces architectures peuvent conduire aux scénarios de panne identifiés lors de l'analyse des risques (FHA). Il s'agit alors d'analyses itératives ayant pour but :

– de compléter et/ou préciser les exigences de sécurité identifiés au différents niveaux : les PASA/PSSA participent à la vérification du travail réalisé lors des activités de FHA.
– d'estimer que les architectures proposées peuvent satisfaire les exigences de sécurité : les PASA/PSSA permettent d'écarter des architectures jugées pas assez sûres, avant les étapes de développement et de validation finale des architectures.
– d'identifier d'éventuels besoins en protection alternative permettant d'atteindre les objectifs de sécurité : les PASA/PSSA identifient les points faibles des architectures proposées du point de vue de la sécurité et peuvent ainsi proposer des moyens de protection locale.

En pratique, l'identification des combinaisons de pannes, qui conduisent aux risques identifiées par les FHA, est réalisée à l'aide d'Arbres de Défaillances ou de Diagrammes de Dépendances. Ces techniques sont détaillées dans le paragraphe 4.1.1 du chapitre 2.

2.2.3 Activités de ASA/SSA

Les *Aircraft/System Safety Assessment* (ASA/SSA) sont les évaluations des architectures avion et système implémentées au regard des exigences de sécurité. Ces activités s'appuient sur les PASA/PSSA précédemment réalisées sur les mêmes architectures. Il s'agit d'activités similaires aux analyses préliminaires mais qui diffèrent par leurs intentions : alors que les PASA/PSSA assistent la conception, les ASA/SSA ont un objectif de vérification finale des architectures tant sur les aspects qualitatifs que quantitatifs des exigences.

A partir des Arbres de Défaillances ou des Diagrammes de Dépendances réalisés au cours des analyses de PASA/PSSA, les analystes de la sécurité peuvent réaliser des calculs de probabilité d'occurrence des scénarios de pannes redoutés, à partir des taux de défaillance des équipements transmis par les fournisseurs. L'étape de vérification quantitative s'assure alors que les probabilités obtenues sont satisfaisantes au regard des exigences de sécurité (les objectifs tant quantitatifs que qualitatifs sont présentés dans le tableau 1.2 du paragraphe 3.2).

3 L'analyse des risques

Toute analyse de sûreté de fonctionnement commence par une identification et une compréhension des risques pouvant affecter l'avion. Cette étape est cruciale et requise avec de plus en plus de rigueur par les autorités de certification pour tous les grands systèmes dits complexes, tels que les moyens de transport (avion, sous-marin, automobile) ou encore les centrales nucléaires. Historiquement, l'identification des risques était réalisée dès les premiers processus de sécurité, à partir de listes des pannes survenues sur les appareils en service. Depuis quelques décennies, l'identification des risques est devenue moins réactive mais davantage proactive afin de prédire les risques potentiels, pouvant affecter un nouveau produit, dès les premières phases de son développement. Cette activité est l'Analyse Préliminaire des Risques (APR), encore connue sous le nom de *Preliminary Hazard Analysis* (PHA).

Afin d'adapter les principes des APR à l'aéronautique ([ARP4754A / ED-79A]), les autorités de certification recommandent fortement l'application d'**analyses des risques**, identifiées dans le processus sous le nom de *Functional Hazard Assessment* (FHA).

3.1 Présentation générale de l'analyse des risques

L'analyse des risques vise à explorer toutes les pannes fonctionnelles potentielles pour les classifier et en déterminer des exigences de sécurité. La méthode suit des étapes clairement établies par les recommandations [ARP4754A / ED-79A] :

- identification des scénarios de pannes fonctionnelles
- classification des scénarios de pannes fonctionnelles
- détermination des exigences de sécurité à partir des scénarios de pannes
- allocation des exigences de sécurité au niveau d'étude inférieur

Les résultats de l'analyse des risques sont enregistrés dans des tables ayant la forme du tableau 1.1. Les différents champs de cette table sont explicités par la suite. Une application du processus d'analyse FHA est présentée dans le tableau 1.3 (Les champs de la table sont en anglais ; le texte en français et en italique dans la table est une annotation montrant les familles de données, qui remplissent certains champs).

Aircraft/System Function : *(nom et numéro identifiant la fonction étudiée)*				
Failure : *(numéro identifiant le scénario de pannes)*	*(nom du scénario de pannes)*	Classification	Justifications & Parameters	Requirements (FC)
Flight Phase : Repercussions : Crew detections : Crew actions :				**FC :**

Tab. 1.1 - Format d'enregistrement des résultats de l'analyse FHA

3.2 Étude des scénarios de pannes fonctionnelles

La première étape nécessaire pour réaliser une analyse des risques est d'identifier les fonctions à analyser ([ARP4754A / ED-79A]). La description fonctionnelle est construite sous la forme d'arbres de décomposition fonctionnel (voir le chapitre 2, section 3.3.1, pour plus de détails sur les arbres de décomposition fonctionnel). L'analyse des risques de niveau avion et chacune des analyses des risques de niveau système travaillent sur son propre arbre de décomposition fonctionnel.

Exemple. A un niveau fonctionnel avion, nous pouvons identifier une fonction de contrôle de la trajectoire de l'avion au sol : « Contrôle de la trajectoire ». Nous garderons cet exemple jusqu'à la fin de ce chapitre.

La connaissance des fonctions permet alors de décrire des scénarios de pannes fonctionnelles. Ces derniers sont définis comme des combinaisons de pannes affectant différentes fonctions.

3.2.1 Description de scénarios de pannes par les modes de panne

Il est cependant moins aisé d'exprimer des pannes fonctionnelles que des défaillances d'équipement. En effet, pour un tuyau, par exemple, nous pouvons identifier immédiatement une défaillance : la rupture du tuyau, menant à une fuite. Mais travailler sur des fonctions implique de s'abstraire de l'implémentation physique. Les pannes ne sont donc plus évidentes puisque les solutions techniques ne sont pas connues. Le document [ARP4754A / ED-79A] propose une solution sous la forme de modes de panne fonctionnelle (« *functional failure modes* »). Les quatre modes de panne fonctionnelle, définis ci dessous, assurent que tous les cas de pannes sont bien répertoriés.

– Perte totale (« *total loss* ») : la fonction est considérée comme perdue, c'est à dire qu'elle n'est plus en

mesure de réaliser la moindre fonctionnalité. Par exemple, la perte totale de la fonction « Contrôle de la trajectoire » signifie que l'avion ne dispose plus d'aucun moyen pour modifier la trajectoire de l'avion.

- Perte partielle (« *partial loss* ») : la fonction est considérée comme partiellement perdue, c'est à dire qu'elle est toujours fonctionnelle, mais qu'elle a perdu un certain nombre de ses capacités, ce qui diminue sa performance. Par exemple, la perte partielle de la fonction « Contrôle de la trajectoire » peut correspondre à la perte d'un des moyens (par exemple la roulette de direction) de contrôle de la trajectoire parmi l'ensemble de ses moyens (gouverne de direction, freinage différentiel etc.).
- Fonctionnement erroné (« *erroneous* ») : la fonction remplit toutes ses fonctionnalités mais avec une perte d'intégrité. Un des cas les plus dimensionnants est celui où la fonctionnalité devient contraire à celle qui est attendue. Par exemple, le fonctionnement erroné de la fonction « Contrôle de la trajectoire » peut provoquer un virage à gauche quand le pilote désirait virer à droite.
- Fonctionnement intempestif (« *inadvertant* ») : la fonction s'exécute spontanément sans aucune demande et à n'importe quel moment. Dans certains cas, cette activation n'a aucun impact, mais parfois elle perturbe les fonctionnalités rendues par d'autres fonctions. La sortie intempestive des trains d'atterrissage en plein vol est un exemple critique de fonctionnement intempestif.

La perte partielle, le fonctionnement erroné et le fonctionnement intempestif peuvent être raffinés, si nécessaire, lors de la description des scénarios de panne pour identifier de nouvelles exigences de sécurité.

Exemple. Prenons la fonction de production de la poussée motrice de l'avion : « Fournir la poussée ». La perte partielle de cette fonction peut se traduire concrètement de deux façons : une diminution globale de la poussée de l'avion (fonctionnement à 50% de performance de tous les moteurs de l'avion par exemple) ou alors une diminution asymétrique de la poussée (perte totale d'un des deux moteurs dans le cas d'un avion bimoteur), qui risque d'avoir un fort impact sur le contrôle de la trajectoire de l'avion en plus de l'impact évident sur la poussée motrice. Ces deux scénarios de pannes sont considérés comme différents et sont donc analysés séparément dans le cadre de l'analyse des risques.

Les autorités de certification imposent une analyse exhaustive de toutes les fonctions à travers chacun de ces quatre modes de panne. Du fait du nombre très important de scénarios de pannes que cela induit, la FHA est une activité très longue, qui requiert une grande rigueur.

3.2.2 Détermination des effets des scénarios de pannes

Chaque scénario de panne est étudié indépendamment des autres. Les analystes de la sécurité doivent alors déterminer les effets de chacun des scénarios dans le but de pouvoir en déduire leur sévérité. Pour cela, ils peuvent s'appuyer sur les documents de la conception, ainsi que sur des discussions avec des concepteurs avion, des pilotes et des spécialistes des performances avion. Pour les aider dans cette tâche, le processus de FHA exige d'étudier les quatre champs suivants.

1. **Phases de vol** (« *Flight Phase* ») : le contexte du scénario de pannes doit être connu car il est un paramètre important à prendre en compte pour comprendre l'effet des pannes. En effet, un même scénario de panne peut avoir des conséquences très différentes selon la vitesse de l'avion, son altitude, son environnement etc. Ce contexte est appelé contexte opérationnel. Le terme opérationnel est important puisque le contexte évolue justement tout au long des missions de l'avion. Les phases de vol sont définies comme un découpage de ces missions en périodes à l'intérieur desquelles les répercussions de pannes sont considérées invariantes. Dans le cas d'un vol simple, tel que représenté par la figure 1.6 par exemple, nous identifions plusieurs phases de vol du décollage à l'atterrissage. L'enchaînement de ces phases de vol est appelé scénario de vol. Il est possible de décrire différents scénarios de vol pour faire varier le contexte opérationnel. Toutefois, le vol simple est le cas d'étude recommandé par les autorités de certification.

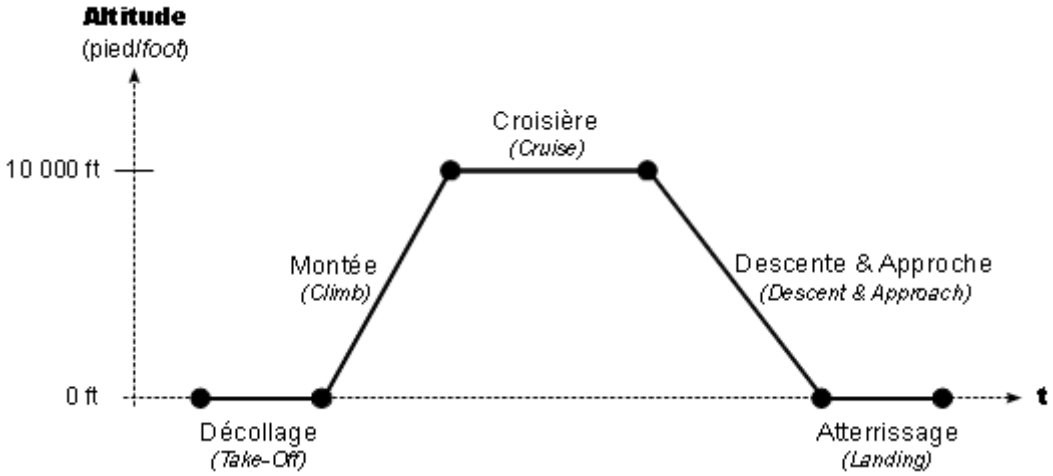


Fig. 1.6 - Scénario de vol simple du décollage à l'atterrissage

Les scénarios de pannes doivent être étudiés dans la FHA sur chaque phase de vol.

2. **Répercussions** (« *Repercussions* ») : ce champs permet aux analystes de la sécurité de décrire les effets directs du scénario de pannes considéré et ses conséquences si aucune action corrective n'est réalisée. Il s'agit d'une étape délicate de l'analyse des risques car il faut prendre en compte beaucoup de paramètres dont entre autre les phases de vol concernées et d'éventuelles conditions environnementales.
3. **Moyens de détection** (« *Crew detection* ») : ce champs renseigne les moyens de détection des pannes mis à disposition de l'équipage. Il peut s'agir aussi bien d'alarme auditive que visuelle ou de simples sensations physiques liées aux effets de la panne. Ce champs est difficile à remplir pour l'analyse des risques de niveau avion, car, en théorie, à ce stade de l'étude les systèmes de l'avion ne sont pas définis. Il n'est donc pas évident de prédire l'existence de moyens de détection. Toutefois, il est possible de faire des hypothèses, voire émettre des exigences.

4. **Moyens d'actions correctives** (« *Crew actions* ») : ce champs explique les moyens d'actions correctifs mis à disposition de l'équipage dans le cas où il serait en mesure de détecter la panne. Parfois les moyens correctifs permettent de revenir complètement dans le mode de fonctionnement nominal, sans panne, mais le plus souvent il s'agit de moyens permettant de diminuer et/ou d'éviter les effets et donc de réduire la sévérité de l'incident.

3.2.3 Classification des scénarios de pannes

Une fois les effets des scénarios de pannes connus, les analystes de la sécurité peuvent classifier ces scénarios de pannes afin d'en déduire les exigences de sécurité. La classification est définie par les autorités de certification dans le document de certification [CS-25 1309]. Il s'agit d'une discrétisation par rapport aux effets des pannes sur l'avion, ses occupants et sur la charge de travail de l'équipage (voir tableau 1.2).

Classification	**Effet de la défaillance**	**Objectif quantitatif** (probabilité de défaillance par heure de vol)	**Objectif qualitatif**
NSE *(No Safety Effect)*	Aucun effet sur la sécurité.	-	-
MIN *(MINor)*	Effets mineurs sur la sécurité : • Faible réduction des marges de sécurité • Faible augmentation de la charge de travail de l'équipage • Inconfort des passagers	10^{-3}	-
MAJ *(MAJor)*	Effets majeurs sur la sécurité : • Réduction significative des marges de sécurité et des fonctionnalités • Augmentation significative de la charge de travail de l'équipage • Inconfort des passagers et de l'équipage	10^{-5}	-
HAZ *(HAZardous)*	Effets périlleux sur la sécurité : • Forte réduction des marges de sécurité et des capacités fonctionnelles • Détresse de l'équipage • Quelques blessures sévères ou fatales parmi les occupants de l'avion (sauf pilotes)	10^{-7}	-
CAT *(CATastrophic)*	Effets catastrophiques sur la sécurité : • Perte de l'avion et de ses occupants	10^{-9}	Aucune panne unitaire ne doit conduire à un scénario catastrophique

Tab. 1.2 - Classification des scénarios de pannes (issue du document [CS-25 1309])

Si le choix de la classification se base essentiellement sur les répercussions des pannes, les champs décrivant les moyens de détection des pannes et d'actions correctives permettent éventuellement de diminuer la classification. Le champ « Justifications et Paramètres » entre également en compte, parfois, quand les effets décrits se basent sur des hypothèses. Il existe deux types d'hypothèses.

- Les justifications : il s'agit d'hypothèses actées par les autorités de certification. Elles s'appuient sur des études scientifiques auxquelles les avionneurs participent, mais ces derniers n'ont pas besoin de les prouver pour la certification. Généralement ces hypothèses sont relatives au contexte opérationnel.
- Les Paramètres : il s'agit d'hypothèses faites par les analystes de la sécurité par rapport aux systèmes qui seront implémentés. Ces hypothèses sont nécessaires bien souvent pour justifier des moyens de détection et de correction des pannes. Elles sont des exigences à part entière puisqu'il faut apporter la preuve a posteriori de leur véracité (existence de procédures, tests, analyses complémentaires etc.).

3.2.4 Exemple d'analyse d'un scénario de pannes

Le tableau 1.3 est un exemple d'application de ces principes avec comme scénario de pannes la perte totale de la fonction de contrôle de la trajectoire au sol. Ce scénario de pannes est analysé sur les phases de roulage de l'avion lors du décollage (« *Take-Off* ») et de l'atterrissage (« *Landing* »). Ses répercussions directes sont la perte de tous les moyens de contrôle de la trajectoire et donc le risque d'une sortie de piste à haute vitesse (supérieure à 60 nœuds). L'équipage dispose d'alarmes, informant de l'état de cette fonction critique, et peut se fier également aux sensations physiques liées à la perte de contrôle de la trajectoire. Enfin, le seul moyen de l'équipage pour réduire la sévérité du scénario de pannes est de freiner au maximum l'avion pour réduire la vitesse de sortie de piste.

Aircraft/System Function : Contrôle de la trajectoire				
Failure : A	Perte totale de contrôle de la trajectoire	Classification	Justifications & Parameters	Requirements (FC)
Flight Phase : Décollage, Atterrissage. Repercussions : Perte totale ou quasi totale des systèmes nécessaires au contrôle de la direction de l'avion pendant le décollage ou l'atterrissage. Les capacités restantes sont insuffisantes pour empêcher une possible sortie de piste à haute vitesse. Crew detection : • Alarmes des systèmes concernés. • Sensations physiques associées à la perte de contrôle de la trajectoire.		**CAT**	Classification criteria : Une sortie de piste avec une vitesse supérieure à 60 nœuds est classée CAT. STUDY : Étudier la pertinence d'alarmes pour informer l'équipage cockpit de la perte totale de guidage de l'avion.	**FC :** **CAT** – Perte totale du contrôle de la trajectoire pendant le décollage ou l'atterrissage.

Crew action : L'équipage cockpit applique une décélération maximale afin de minimiser les répercussions.			

Tab. 1.3 - Analyse de la perte totale du contrôle de la trajectoire de l'avion (exemple théorique)

Dans cet exemple d'analyse, deux hypothèses sont faites. La première indique qu'une sortie de piste d'un appareil avec une vitesse supérieure à 60 nœuds est un événement classifié « Catastrophique ». Il s'agit d'une justification puisque c'est une hypothèse de contexte, valable pour tous les avionneurs. La seconde hypothèse concerne les moyens de détection de la perte de contrôle de la direction. Là, il faut prouver a posteriori que l'équipage a des moyens de détection suffisants pour prendre connaissance de la panne. Pourtant, malgré ce paramètre, le scénario de pannes est classifié CAT selon la première hypothèse.

3.3 Identification des exigences de sécurité

Le but de la classification des scénarios de pannes est de bien estimer les risques pour en déduire des exigences adaptées. Les analystes de la sécurité doivent en effet respecter deux principes.

- Ne pas sous-estimer un risque : il faut toujours prendre en considération les pires cas possibles afin de ne pas négliger une situation potentiellement sévère. C'est le principe de base de la sûreté de fonctionnement pour la conception des systèmes complexes.
- Sur-estimer raisonnablement un risque : plus un risque est estimé important et plus la conception doit se protéger contre ce risque, que ce soit par prévention (diminuer la probabilité d'occurrence du scénario de pannes) ou par protection (diminuer les effets néfastes du scénario de pannes). La figure 1.7 illustre bien la relation entre la probabilité d'occurrence d'un scénario de pannes par heure de vol et la sévérité de ses effets.

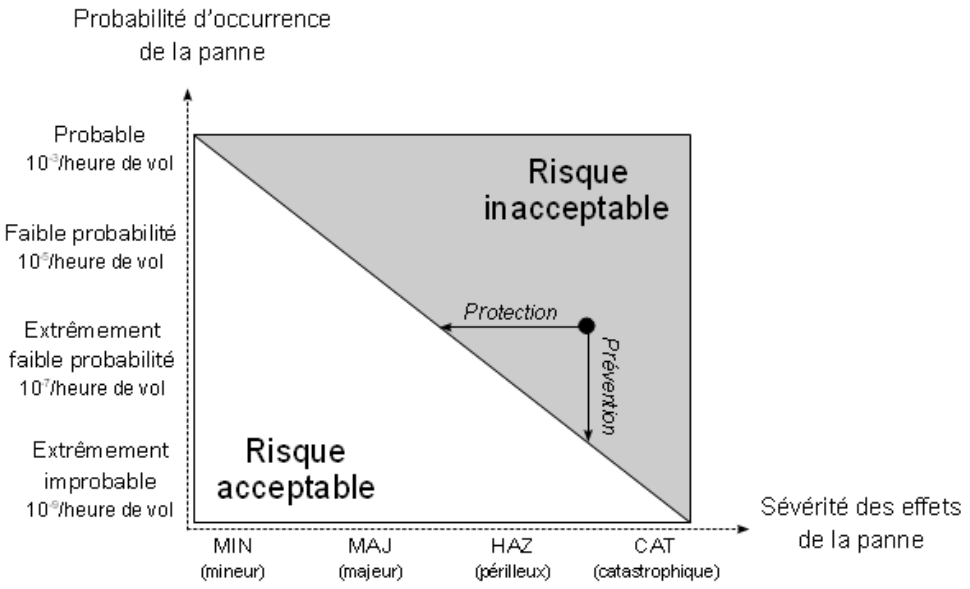


Fig. 1.7 - Relation entre probabilité d'occurrence et sévérité des pannes

Le problème est que, prévention comme protection induisent toutes deux une forte augmentation des coûts de développement et de production, ainsi qu'une possible baisse des performances générales de l'avion (consommation de carburant par exemple). Un risque trop sur-estimé conduit alors à une conception trop robuste, et donc trop coûteuse, par rapport au risque réel.

3.3.1 Les conditions de panne

Les exigences de sécurité découlent directement des **conditions de panne**, « *Failure Conditions* » (FC) en anglais, qui se définissent comme l'association de scénarios de pannes avec leur critère de risque associé. Par commodité, tous les scénarios de pannes basés sur le même mode de panne et la même fonction, mais avec des différences de contexte, peuvent être regroupés autour d'une seule FC si leur sévérité est identique.

La criticité est la clé de l'exigence, puisqu'elle lui est associée un objectif quantitatif de probabilité maximum d'occurrence par heure de vol et un objectif qualitatif (voir le tableau 1.2). Ces objectifs sont ceux qui sont vérifiés dans les dernières étapes de la conception par les activités de ASA/SSA pour chaque scénario de pannes.

Exemple. Pour le scénario de pannes concernant la perte totale du contrôle de la trajectoire au sol, nous en déduisons une FC capitalisant le fait que ce scénario est classifié Catastrophique. Cela signifie que ce scénario de pannes ne doit pas se produire plus d'une fois toutes les 10^9 heures de vol et qu'aucune panne unitaire ne doit mener à ce scénario de panne. Les systèmes doivent donc être conçus pour respecter cette exigence !

3.3.2 Introduction au DAL

Dans le cas d'un événement aléatoire dont la probabilité d'occurrence est connue, l'objectif quantitatif lié aux FC suffit en tant qu'exigences de sécurité mesurable. Cependant, les erreurs de conception ne rentrent pas dans l'ensemble des risques aléatoires ([CS-25 1309]).

Exemple. Une erreur de code logiciel par exemple se manifestera à chaque fois dans les mêmes situations. En outre, il est impossible de garantir qu'un logiciel ne contient aucune erreur de code. Ainsi, si nous devions estimer une probabilité pessimiste de la présence d'une erreur de code, elle serait de 1.

L'objectif quantitatif lié aux FC ne permet donc pas de définir un degré de rigueur de la conception. Pour palier ce manque, depuis plusieurs années, un nouvel outil d'aide à la conception s'impose de plus en plus comme un moyen supplémentaire d'adresser des exigences de sécurité : le « *Development Assurance Level* » (DAL). Chaque catégorie de DAL impose des règles de conception (ségrégation, redondance etc.) tant sur les architectures fonctionnelles (FDAL, « *Functional DAL* »), que sur les techniques d'implémentation matérielles et logicielles (IDAL, « *Item DAL* »). Le DAL est déterminé à partir des FC issues de la FHA avion. La valeur de FDAL d'une fonction avion correspond à la pire criticité d'une de ses FC, selon la table 1.4.

Classification des FC avion	CAT	HAZ	MAJ	MIN	NSE
FDAL associé à la fonction	A	B	C	D	E

Tab. 1.4 - Correspondance entre la classification des FC et le FDAL

Le DAL est ensuite déduit sur les différentes couches à partir du FDAL de niveau fonctionnel avion. Des règles d'allocation du DAL permettent des ajustements des niveaux de conception afin de minimiser les coûts de développement tout en assurant le respect des exigences de sécurité. Les règles d'allocation sont définies dans le document [ARP4754A / ED-79A].

Exemple. A supposer qu'il existe un scénario de panne, conduisant à une FC classée MAJ, à partir de notre fonction « Contrôle de la trajectoire », le FDAL de cette fonction serait tout de même A, car nous avons déjà identifié une FC classée CAT, attachée à cette fonction (voir table 1.3). Si ce contrôle de la trajectoire au sol est réalisé par une roulette de direction et une gouverne de direction alors ces deux éléments sont DAL A. Mais grâce aux règles d'allocation, l'une des deux manières de contrôler la trajectoire peut être dégradé DAL C, ce qui diminue fortement son coût de développement.

3.4 Axes d'amélioration de l'analyse des risques

L'analyse des risques est une étape nécessaire pour la certification, et elle joue aussi un rôle déterminant pour assister la conception des systèmes et de l'avion. Cependant, la majorité des exemples d'application de cette activité, et en particulier l'exemple des ARPs ([ARP4761 / ED-135]) sur la décélération de l'avion au sol, est réalisée sur des fonctions de niveau avion ou système qui ont des effets fonctionnels et dysfonctionnels évidents et peu liés à d'autres fonctionnalités. L'expérience a montré, notamment chez Airbus, que la complexité toujours grandissante des systèmes rend délicate la réalisation de l'analyse des risques. Cette difficulté se traduit au niveau du coût toujours plus important des analyses, à cause du temps nécessaire pour les réaliser.

Dans le cadre de cette thèse, nous avons identifié plusieurs points pouvant être optimisés :

1. mieux identifier les fonctions à analyser ;
2. mieux déterminer les effets des scénarios de pannes ;
3. mieux prendre en compte des données supplémentaires comme la charge de travail de l'équipage, les conditions environnementales ou des aspects facteurs humains ;
4. mieux traiter les scénarios de pannes multiples ;
5. renforcer la communication et la traçabilité entre la FHA et les documents de la conception.

D'autres industriels et chercheurs ont également présenté leurs points de vue sur les difficultés qu'ils pouvaient rencontrer lors d'une analyse des risques. Parmi eux on peut citer : [Wilkinson et al., 1998], [Allenby et al., 2001], [Flaus, 2008] et [Wassmuth et al., 2011]. Les différents axes d'amélioration sont détaillés dans les sections suivantes.

3.4.1 Mieux identifier les fonctions à analyser

En pratique, jusqu'à très récemment, les fonctions analysées dans le domaine de la sécurité étaient définies par des analystes de la sécurité. Les concepteurs de l'avion travaillaient également sur leurs propres descriptions fonctionnelles de l'avion. Les descriptions fonctionnelles de la sécurité et de la conception étaient construites toutes les deux sur la base d'un arbre de décomposition fonctionnel. Cependant, ces deux familles de descriptions fonctionnelles présentent de nombreuses différences entre elles car elles sont adaptées à leur domaine d'expertise. Nous identifions trois sources principales de différences.

- Différences de périmètre : comme chaque description fonctionnelle tend à décrire l'avion, il est possible d'y retrouver toutes les caractéristiques. Ainsi, il n'est pas rare d'y retrouver les mêmes fonctions. Cependant, les fonctions identifiées peuvent avoir des périmètres légèrement différents, principalement à cause de leurs niveaux de détails respectifs.
- Différences d'identificateur : une même fonction dans deux décompositions fonctionnelles peut avoir des identificateurs différents.
- Différences d'organisation : une même fonction peut être placée à des endroits différents dans les arbres de décompositions fonctionnelles.

Toutes ces différences ne favorisent pas la transmission des exigences de sécurité à la conception ([Wilkinson et al., 1998]). Certes, les concepteurs et les analystes de la sécurité communiquent pour trouver les correspondances entre leurs descriptions fonctionnelles. Mais cette situation était problématique pour les concepteurs puisque non seulement la sécurité mais aussi d'autres métiers, tels que les facteurs humains ou les analystes des performances avions, produisent leur propre description fonctionnelle.

Face à cette difficulté, les concepteurs ont proposé de fédérer toutes les activités autour d'une décomposition fonctionnelle unique. Cette approche est en cours d'évaluation chez Airbus dans le cadre du projet ALFA ([ALFA, 2012]), présenté dans la section 3.3.1 du chapitre 2. Pour les futurs programmes avion, l'analyse des risques devrait donc pouvoir se baser sur une description fonctionnelle de référence.

3.4.2 Mieux évaluer les conséquences des pannes fonctionnelles

Les analystes de la sécurité rencontrent des difficultés pour évaluer efficacement les conséquences des pannes fonctionnelles.

La première d'entre elles a pour origine le format de capitalisation des données de l'analyse des risques. En effet, ce format n'offre que très peu de support. En particulier, le champ « Répercussion » est un champ texte libre. Certes, comme le rappelle J.-M. Flaus dans [Flaus, 2008], cela laisse une grande liberté d'expression aux analystes, mais il constate également que le manque de formalisation pénalise fortement les analystes les

moins experts. Il est aisé de décrire le même scénario de pannes de deux façons différentes sans être capable de reconnaître leur lien de cause à effet. De plus, comme l'analyse des risques est un processus très long et difficile à maintenir, plusieurs analystes travaillent sur cette activité tout au long de la conception de l'avion. Cela peut conduire notamment à des incohérences ([Allenby et al., 2001]). Les études citées précédemment montrent les avantages de la mise en évidence et la traçabilité de données plus précises telles que, pour les répercussions des pannes, la distinction entre les répercussions sur l'avion et celles sur les occupants. En pratique, ces données sont manipulées par les analystes de la sécurité, mais sans être nécessairement tracées dans les documents d'analyse.

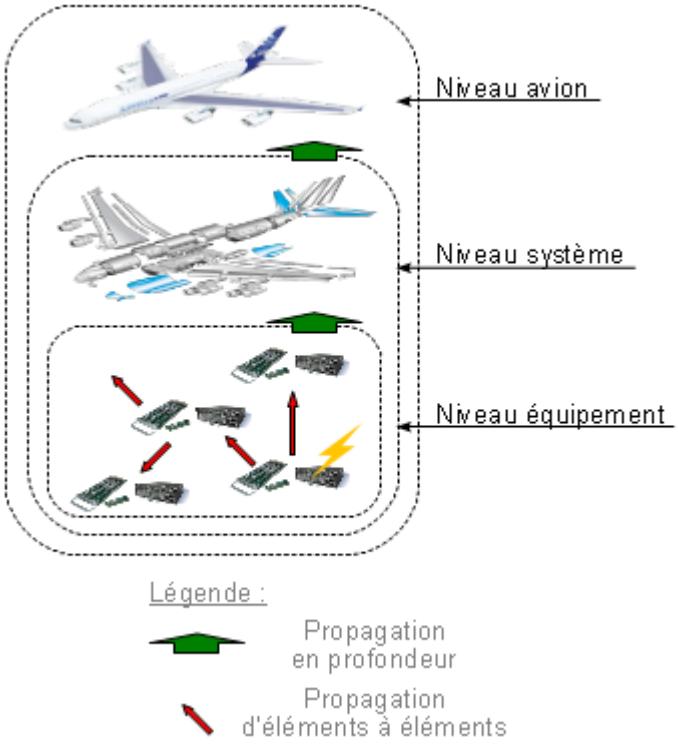


Fig. 1.8 - Propagation des pannes

Une seconde difficulté associée à l'évaluation des conséquences est l'effort de travail nécessaire pour analyser les scénarios de panne. La propagation des pannes, introduite dans la section 1.1.2 dans le cadre global de la sûreté de fonctionnement, est en effet de plus en plus difficile à identifier et ses effets difficiles à estimer. On distingue deux types de propagations des pannes, illustrés par la figure 1.8.

– Propagation d'éléments à éléments : ce chemin de propagation des pannes est le plus facile à appréhender. Il décrit le fait que pour fonctionner, certains composants et systèmes nécessitent le fonctionnement d'autres composants et systèmes ou d'un état particulier de l'avion, qui lui même dépend du fonctionnement de tous les composants et systèmes de l'avion. Par exemple, une panne peut se propager, par les réseaux, d'un capteur aux calculateurs.

- Propagation en profondeur : la propagation des pannes en profondeur ne pose de problèmes que depuis quelques décennies. Elle représente le fait qu'une défaillance d'un équipement contribue à la dégradation d'un système et que la défaillance d'un système se traduit par une panne fonctionnelle à l'échelle de l'avion. Or, un système peut intervenir dans le cadre de la réalisation de plusieurs fonctions avion et inversement une fonction avion peut s'appuyer sur l'action conjointe de plusieurs systèmes. Le phénomène est similaire entre les niveaux système et équipement. Au niveau avion ce phénomène est appelé intégration des systèmes. Cette intégration peut remettre en cause des indépendances entre fonctions et peut donc augmenter significativement la complexité des analyses des risques.

La complexité des chemins de propagation des pannes découle d'une caractéristique identifiée par de nombreux travaux, tels que [Wilkinson et al., 1998] et [Wassmuth et al., 2011], comme étant le principal facteur de difficulté rencontré lors des analyses de risques : la complexité des systèmes et de leur intégration.

La solution présentée dans [Wassmuth et al., 2011] consiste à mettre à jour le document d'analyse FHA au fur et à mesure que l'architecture de l'avion est définie afin d'intégrer les spécificités des systèmes. Cette approche est assistée par une base de données commune permettant d'établir des correspondances entre les fonctions et les équipements d'un produit. Cette méthode est déjà appliquée chez Airbus depuis de nombreuses années. La proposition de [Wilkinson et al., 1998] est plus théorique et porte sur la séparation entre la phase d'identification des exigences d'un produit et la phase de définition de l'architecture de ce produit. Elle est constituée de quatre étapes.

- La suppression de tout terme technologique dans le nom et la description des fonctions, afin d'éviter d'influer la conception ou d'être influencé par la conception lors de l'analyse. Par exemple, la fonction « diriger/orienter la poussée » serait préférable à « contrôler les inverseurs de poussée », même si les analystes de la sécurité savent que l'avion qu'ils analysent aura des inverseurs de poussée comme seul moyen de contrôle de la direction de la poussée.
- Le groupement des fonctions par famille, tels que les moyens de contrôle, les moyens de communication etc.
- L'identification de fonctions non-critiques, qui ne seront pas analysées. Ce point pose toutefois problème : comment juger la criticité d'une fonction sans en faire justement une analyse des risques ?
- S'appuyer sur des méthodes de modélisation des interactions entre les fonctions, pour faciliter l'analyse des effets des pannes. Cependant, les utilisations de modèles ne couvrent pas la phase d'identification des exigences de sécurité. En revanche, elles le font avec succès pour la vérification de la tenue des exigences par les systèmes complexes (voir la section 4.2 du chapitre 2).

3.4.3 Faire évoluer l'analyse des risques vers la prise en compte des facteurs opérationnels et des conditions environnementales

Les conséquences des pannes fonctionnelles dépendent très fortement du contexte opérationnel de l'avion et en premier lieu de la phase de vol pendant laquelle survient la panne (voir la section 3.2.2). Cependant,

même avec un découpage très fin des scénarios de vol, les phases de vol ne décrivent pas toujours avec assez de précision les différents contextes opérationnels. En effet, la vitesse, l'altitude et d'autres caractéristiques de l'avion sont susceptibles de varier de façon plus ou moins significative. Mais la caractéristique la plus importante et la plus variable est la composante humaine : au sein d'une même phase de vol, les équipages réalisent de nombreuses tâches et, même si leurs actions sont fortement codifiées par des procédures strictes, ils prennent des décisions qui peuvent affecter les fonctions de l'avion. Inversement, les pannes fonctionnelles peuvent avoir des effets sur les équipages, entre autre en terme de charge de travail pour identifier, analyser et pallier les situations dangereuses.

Un autre aspect important du contexte opérationnel est l'environnement de l'avion. En pratique, très peu de scénarios de pannes prennent en compte des pannes sous des conditions environnementales très fortes. Pourtant, il est établi que certaines conditions environnementales, tels qu'une très forte pluie ou un puissant vent de travers, peuvent diminuer les performances de certaines fonctions et donc aggraver des cas de pannes. Les exigences de sécurité découlant de telles combinaisons expriment des indépendances entre des fonctions et des conditions environnementales.

L'évolution de l'analyse des risques pour prendre en compte des informations supplémentaires, telles que des détails sur les actions humaines et des conditions environnementales, est rendue difficile du fait des nombreuses autres contraintes identifiées sur cette analyse. En effet, de telles évolutions risquent d'augmenter la complexité de cette analyse.

3.4.4 Mieux traiter les combinaisons de pannes fonctionnelles

La problématique d'intégration des systèmes évoquée ci-dessus peut remettre en cause l'indépendance entre les fonctions. Par conséquent, de nouvelles exigences de sécurité sont susceptibles d'être identifiées à partir des combinaisons de plusieurs pannes affectant différentes fonctions. D'ailleurs, le document [ARP4754A / ED-79A] recommande bien que tous les scénarios de pannes pertinents soient analysés, y compris les scénarios de pannes multiples.

En pratique, comme le rappelle [Wilkinson et al., 1998], l'analyse des risques est parfaitement adaptée à l'analyse indépendante de toutes les fonctions via tous les modes de panne. Cette tâche est intégralement réalisée chez Airbus, où les analystes de la sécurité étudient exhaustivement toutes les pannes unitaires. En revanche, l'analyse des risques ne fournit aucun support pour étudier les interactions entre les fonctions et exprimer des combinaisons de plusieurs pannes fonctionnelles. Il serait très inefficace d'étudier exhaustivement toutes les combinaisons possibles, du fait de leur trop grand nombre. De plus, la majorité de ces combinaisons de pannes sont, de manière évidente, non pertinentes, dans le sens où aucune nouvelle exigence de sécurité ne peut découler de leur analyse.

L'identification des scénarios de pannes pertinents est une tâche qui demande un effort de travail très important. La solution appliquée aujourd'hui consiste à se focaliser sur un petit ensemble de combinaisons de pannes dont les analystes de la sécurité ont la conviction qu'elles sont pertinentes, soit grâce à leur jugement d'expert, soit grâce au retour d'expérience sur les avions en service. Cette méthode est acceptée par les autorités de certification et l'expérience actuelle nous permet d'avoir confiance en elle. Toutefois, il ne s'agit

pas d'une preuve rigoureuse que toutes les combinaisons de pannes pertinentes sont étudiées. Ici la difficulté est de déterminer et tracer des critères de choix pour les combinaisons de pannes à analyser.

3.4.5 Renforcer la communication et la traçabilité des informations entre la conception et la sécurité

L'analyse des risques s'appuie évidemment sur les caractéristiques de l'avion, telles qu'elles sont définies lors des premières étapes de la conception. Il y a donc un échange d'information entre les domaines de la sécurité et celui de la conception de l'avion, que ce soit pour réaliser l'analyse des risques de niveau avion ou les analyses des risques de niveau système. D'ailleurs, ce lien entre la conception et la sécurité existe pour toutes les analyses de sécurité.

Jusqu'à l'apparition, très récentes, des premiers modèles d'architectures opérationnelles et fonctionnelles ([Reynolds, 2006], [ALFA, 2012]), l'analyse des risques s'appuyait exclusivement sur une interprétation des documents de la conception. Les résultats des analyses des risques, eux même, sont capitalisés dans des documents à la structure faiblement formalisée (voir la section 3.2). Ce lourd travail d'interprétation de la documentation souffre de problèmes récurrents.

- La complexité de ces documents en rend la compréhension difficile pour les ingénieurs non-spécialistes du domaine concerné.
- L'archivage des documents et la gestion de leurs différentes versions compliquent fortement l'organisation du travail.

De futures évolutions de l'analyse des risques, notamment pour mieux prendre en compte le contexte opérationnel et les conditions environnementales risquent d’accroître davantage la quantité de donnée manipulées par les analystes de la sécurité. Par conséquent, l'effort de gestion documentaire risque d'être plus important. C'est pourquoi il est dès aujourd'hui intéressant de renforcer les moyens de communication et de traçabilité de l'information entre la conception et la sécurité.

4 Bilan

La complexité de l'analyse des risques met en évidence plusieurs sources d'inefficacité. Les difficultés sont palliées aujourd'hui par différents moyens dont l'exploitation du retour d'expérience et une importante gestion documentaire. Depuis plusieurs décennies, l'ingénierie s’intéresse aux travaux de modélisation pour supporter cette analyse. Dans le chapitre 2, nous présentons différentes familles de modèles pouvant avoir un lien avec l'analyse des risques ; l'objectif, détaillé au chapitre 3, est d'assister la réalisation de l'analyse des risques par des modèles adaptés.

Chapitre 2

Intérêts des modèles et de l'Ingénierie des Modèles

Sommaire

Résumé. Face aux besoins d'amélioration de l'analyse des risques, les analystes de la sécurité souhaitent utiliser des modèles en support. Les activités de modélisation s'appuient sur les techniques de l'Ingénierie Dirigée par les Modèles (IDM) afin d'assurer une bonne définition des modèles et leur interopérabilité. Ce chapitre présente l'approche IDM et quelques une de ses applications courantes dans les domaines de la conception des systèmes embarqués et des analyses de la sécurité.

1 Introduction

La complexité croissante des systèmes embarqués met en évidence la nécessité de disposer de processus, de méthodes et d'outils performants pour assister leur développement, depuis les premiers concepts à l'implémentation. La fin du premier chapitre a montré que les analyses préliminaires des risques n'échappent pas à ce constat.

Ces dernières décennies ont vu l'émergence de nouveaux moyens pour représenter et étudier les interactions entre les fonctions, promus par l'évolution rapide de l'informatique : les **modèles**. Ils sont aujourd'hui couramment utilisés dans des domaines aussi variés que l'économie, la météorologie ou la conception aéronautique. Ces modèles sont très différents, tant du point de vue de leur sémantique, que de leur support de modélisation ou de leur exploitation, mais le concept fondateur est toujours le même : un modèle est une représentation abstraite du réel ([Jézéquel et al. 2012]). La construction d'un modèle répond à plusieurs problématiques dont les trois principales sont listées ci-dessous.

- Quel est son but ? Tout modèle est développé pour répondre à un besoin bien spécifique, tel qu'une prévision météorologique, une simulation de l'évacuation d'un avion ou encore une estimation de la sécurité d'un avion.
- Que représente-t-il ? Pour atteindre ses objectifs de modélisation, un modèle est défini par un périmètre bien précis à l'intérieur duquel il doit représenter le plus fidèlement possible la réalité.
- Avec quel niveau de détail ? La représentation que chaque modèle fait de la réalité est une représentation abstraite. La notion d'abstraction dans un modèle signifie que ce dernier filtre les informations pour ne garder que celles correspondant à un point de vue orienté vers ses besoins.

Les différents métiers liés à la conception des systèmes embarqués (conception des systèmes, étude des performances, analyse de la sécurité etc.) s'appuient aujourd'hui sur des modèles. Pourtant, à ce jour, il n'existe pas de modèles permettant d'assister l'analyse des risques (FHA).

Ce chapitre présente un état des méthodes de modélisation existantes pouvant guider la définition d'une activité de modélisation pour soutenir l'analyse des risques. La section 2 résume comment modéliser et manipuler les modèles dans le cadre de l'Ingénierie Dirigée par les Modèles. Cette section introduit également une technique pour formaliser les interactions entre différents modèles. Étant donné que l'analyse des risques se base sur une connaissance fonctionnelle du système considéré, nous présentons dans la section 3 des modèles, réalisés par les architectes des systèmes, qui capitalisent ce type de connaissance. Cette section montre également les formalismes couramment utilisés par les architectes. Enfin, la section 4 introduit les méthodes et outils de modélisation utilisés dans le cadre de la sûreté de fonctionnement. Nous montrons que ces formalismes sont particulièrement adaptés pour représenter et analyser les éléments principaux de toute analyse de sécurité : les pannes.

2 L'Ingénierie Dirigée par les Modèles

La conception des systèmes embarqués fait interagir de nombreux métiers différents dont les travaux s'appuient de plus en plus sur des modèles. Au sein du processus de conception, ce sont alors des modèles qui se partagent des informations. Certains modèles sont même automatiquement et intégralement générés à partir de la connaissance contenue dans plusieurs autres modèles existants. Comment alors assurer que les différentes activités de modélisation sont cohérentes et peuvent interagir entre elles ?

L'Ingénierie Dirigée par les Modèles (IDM ou MDE en anglais pour « *Model Driven Engineering* »), est une approche qui offre des techniques pour faciliter la description des modèles et formaliser leurs liens ([Combemale 2008], [Jézéquel et al. 2012]), afin d'en assurer la cohérence. Les bénéfices de l'IDM sont :

- l'intégration de modèles hétérogènes, tant sur le plan syntaxique que sémantique, en mettant à disposition des opérateurs pour manipuler les modèles comme des objets ;
- un support à la définition de domaines de modélisation.

2.1 Modélisation, métamodélisation

Cette première section définit l'activité de modélisation et présente les méthodes proposées dans le cadre de l'IDM afin de concevoir des modèles clairs et cohérents.

2.1.1 La relation entre modèle et instances

Le concept au cœur de l'Ingénierie Dirigée par les Modèles est le **modèle**, c'est-à-dire une représentation abstraite de comportements réels. Le modèle assiste tout particulièrement la conception, la validation et la vérification de différents aspects des systèmes représentés selon le point de vue du modèle et son niveau d'abstraction. Un modèle est souvent conçu pour être exploitable par des outils informatiques pouvant parcourir rapidement et exhaustivement les informations qu'il contient afin de les analyser automatiquement. Pour cela, ces outils, comme les utilisateurs des modèles, doivent être capables de lire ce modèle. La construction du modèle doit alors respecter des **contraintes de modélisation**, qui forment un **langage de modélisation**. Les contraintes de modélisation sont définies comme des règles cohérentes, formées à partir de symboles ou de mots-clés généralement standardisés et interprétables par les ordinateurs.

La relation d'instanciation réunit le modèle et son langage de modélisation.

Définition : instance

> *Un modèle M_1 est une instance d'un langage de modélisation si et seulement si il respecte toutes les contraintes de modélisation du langage de modélisation.*

Évidemment, plusieurs modèles peuvent respecter les mêmes contraintes de modélisation, auxquels cas ils sont tous instances du même langage de modélisation.

2.1.2 Le MOF

Un langage de modélisation peut lui aussi être décrit sous la forme d'un modèle, appelé **métamodèle**. Dans le cadre de la conception de grands systèmes industriels tels que les transports ou l'énergie, de nombreux métiers de modélisation interagissent. Le MOF (« *Meta-Object Facility* ») est apparu comme le moyen de pallier l'émergence de nombreux métamodèles indépendants et incompatibles entre eux. Il offre un langage de description des métamodèles, appelé métamétamodèle et lui-même décrit comme un modèle. Afin de limiter le nombre de niveaux d'abstraction, les métamétamodèles conformes ont la propriété d'être réflexifs, c'est-à-dire de pouvoir se décrire eux-même. Le MOF a été standardisé par l'OMG (« *Object Management Group* »).

Le MOF s'illustre sous la forme d'une pyramide comme sur la figure 2.1 ([OMG MOF 2006]). Le niveau de base, M_0, est le système réel. Les différents modèles permettant l'étude de ce système se trouvent au premier niveau d'abstraction : le niveau M_1. Les métamodèles constituent le niveau M_2. Enfin, le métamétamodèle se situe au niveau M_3.

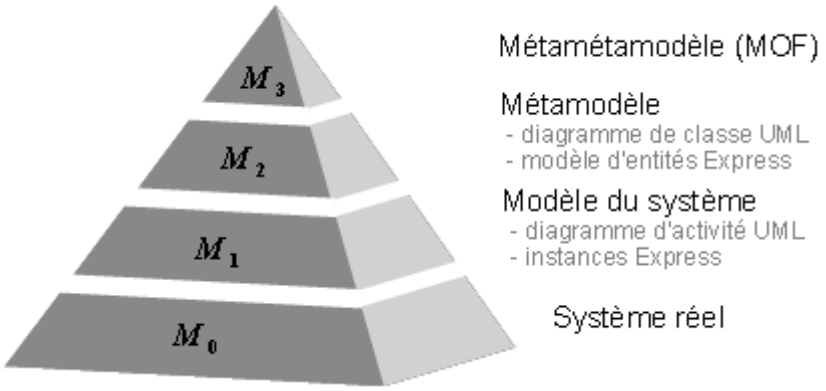


Fig. 2.1 - Pyramide de modélisation de l'OMG

Plusieurs langages de modélisation supportent cette architecture du MOF. Le plus connu est le langage semi-formel UML dont le métamétamodèle peut être trouvé dans [OMG UML 2007]. Le langage Express, que nous introduisons en section 2.3, est également conforme au MOF.

2.1.3 Les langages dédiés de modélisation

L'Ingénierie Dirigée par les Modèles est une approche orientée domaine, qui préconise que chaque métier de modélisation maîtrise ses propres concepts et modèles de données. Les modèles produits ont un périmètre de définition très précis et sont facilement compréhensibles, vérifiables et manipulables par des ingénieurs dont la spécialité n'est pas toujours la programmation.

Les langages dédiés de modélisation (**DSML** pour « *Domain Specific Modeling Language* ») sont, comme leur nom l'indique, des langages de modélisation spécifiques à chaque métier ([Spinellis 2001], [Taha 2008], [Jézéquel et al. 2012]). Leur construction est rigoureuse, basée notamment sur la distinction entre la **syntaxe**, c'est-à-dire les règles d'écriture des modèles, et la **sémantique**, c'est à dire la signification des éléments syntaxiques au sein des modèles.

Définition : DSML

Un DSML est défini par le tuple $< AS , CS^* , M^*_{ac} , SD^* , M^*_{as} >$ *où :*

- AS *est la* ***syntaxe abstraite*** *;*
- CS^* *est l'ensemble des* ***syntaxes concrètes*** *;*
- M^*_{ac} *est l'ensemble des mappings de la syntaxe abstraite vers chaque syntaxe concrète ;*
- SD^* *est l'ensemble des domaines sémantiques ;*
- M^*_{as} *est l'ensemble des mappings de la syntaxe abstraite vers chaque domaine sémantique.*

La syntaxe abstraite est en général décrite en premier. Il s'agit du langage des concepts manipulés par les ingénieurs du domaine. La manipulation réelle est quant à elle réalisée sur une syntaxe concrète, supportée par un langage de programmation et des outils informatiques. Il est important de noter que plusieurs modélisations concrètes peuvent être réalisées à partir d'une unique modélisation abstraite, selon les points de vues privilégiés et les outils de modélisation.

Exemple. L'analyse des risques fait apparaître un certain nombre de concepts, tels que les fonctions et leurs pannes fonctionnelles. Il est possible de modéliser ces concepts via une syntaxe abstraite décrivant la relation d'impact d'une panne fonctionnelle sur une fonction. La section 4 de ce chapitre, consacrée aux modèles de sécurité, montre qu'il existe plusieurs moyens pour modéliser concrètement une telle relation : les langages AltaRica et Figaro par exemple.

La syntaxe abstraite est également liée à une sémantique qui permet de définir un sens à la syntaxe ([Winskel 1993]). Là aussi, plusieurs sémantiques différentes peuvent être exprimées sur une même syntaxe selon les comportements que les analystes souhaitent étudier.

2.2 Transformation de modèle

L'une des problématiques clés de l'Ingénierie Dirigée par les Modèles est la transformation de modèle. Certes, l'utilisation d'un DSML conduit à la conception de modèles adaptés à un métier de modélisation très spécifiques. Néanmoins, bien souvent, ces modèles ont des éléments syntaxiques et/ou sémantiques communs avec des modèles qui sont des instances d'autres DSML et qui sont donc maîtrisés par d'autres spécialistes. La transformation de modèle est l'opération qui consiste à formaliser les liens existant entre des modèles.

2.2.1 Définition de la transformation de modèle

Nous nous intéressons en particulier à un type de transformation de modèle : la génération d'un ensemble de modèles cibles à partir d'un ensemble de modèles sources. Cette transformation prend ses origines dans la programmation de scripts permettant de réécrire des fichiers de données dans différents formats. Elle est

régulièrement utilisée dans divers métiers de l'ingénierie ([Jézéquel et al. 2012]).

La génération de modèle s'appuie sur la définition et l'exploitation d'un modèle de transformation qui décrit les règles de transformation permettant de convertir les modèles sources en modèles cibles ([Combemale 2008]). La figure 2.2 illustre cette définition dans le cadre du MOF.

Définition : Transformation de modèle

Formellement, une transformation de modèle est une opération notée $\boldsymbol{T}$*, telle que :*

$$M_c^* = \boldsymbol{T}\left(MM_s^*, MM_c^*, M_t, M_s^*\right) \text{ avec}$$

– M_s^* *est l'ensemble des modèles sources (modèles d'entrée). L'ensemble de leurs métamodèles correspondants est* MM_s^* ;

– M_c^* *est l'ensemble des modèles cibles (modèles de sortie). L'ensemble de leurs métamodèles correspondants est* MM_c^* ;

– M_t *est le modèle de transformation.*

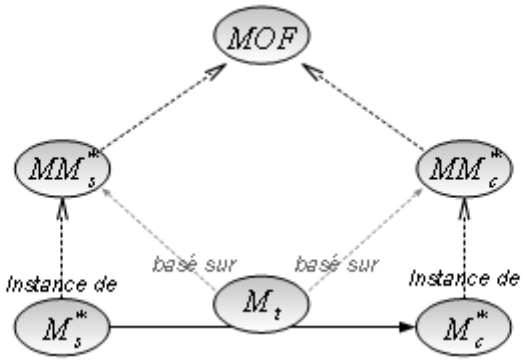


Fig. 2.2 - Génération de modèle, conformément au MOF

2.2.2 Quelques propriétés sur la transformation de modèle

Selon les applications de la transformation de modèle, les modèles sources et cibles peuvent être décrits à partir des mêmes métamodèles. La transformation est alors dite endogène. A l'inverse, quand les métamodèles sont différents, la transformation est exogène.

Les niveaux d'abstraction des modèles entrent également en considération. La transformation de modèle est nécessaire pour changer de niveau d'abstraction : il s'agit de la transformation verticale. La transformation de modèle est horizontale si les modèles sources et les modèles cibles sont au même niveau d'abstraction.

La figure 2.3 présente les différents types de transformations de modèle et leurs cas courants d'utilisation ([Czarnecki et al. 2003], [Sendall et al. 2003], [Combemale 2008], [Jézéquel et al. 2012]).

Nous pouvons constater qu'au sein d'un DSML, les transformations permettant de convertir la syntaxe abstraite vers les différentes syntaxes concrètes sont des transformations exogènes (métamodèles différents) verticales, quand il ne s'agit pas directement de génération de code.

Dans le cas d'une transformation entre des modèles, instances de DSML différents, les règles de transformation s'appuient sur les syntaxes abstraites et les sémantiques. La transformation d'un modèle abstrait vers un autre modèle abstrait est généralement une transformation exogène (métamodèles différents car les DSML sont maîtrisés par des métiers différents) et horizontale.

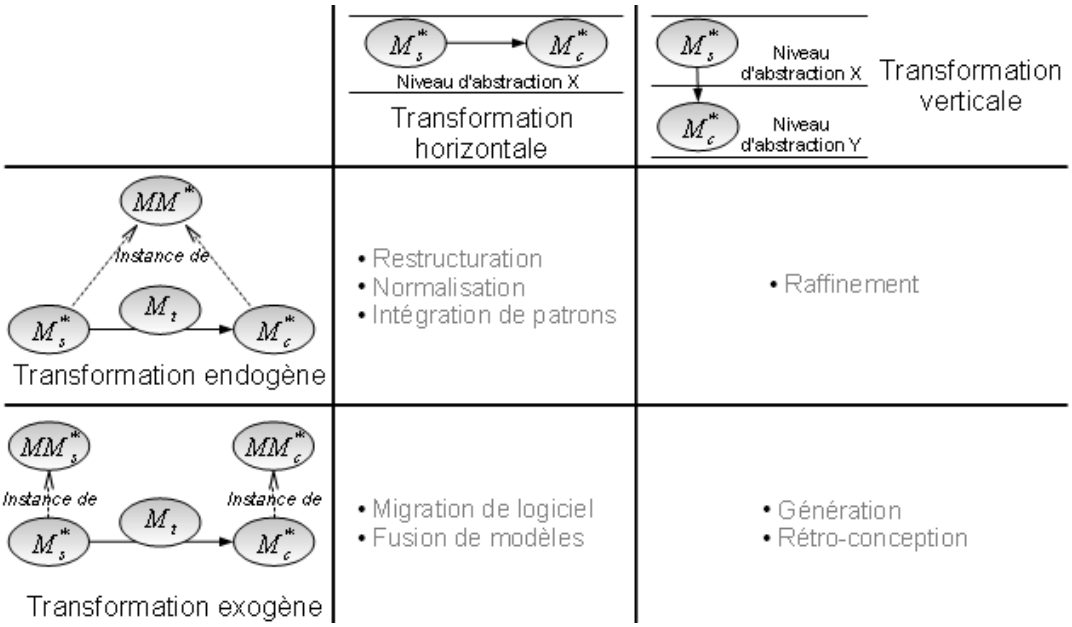


Fig. 2.3 - Types de transformation et leurs principales utilisations ([Jézéquel et al. 2012])

2.3 Express, un langage pour la modélisation et la transformation de modèle

Il existe de nombreux langages pour décrire des modèles et les manipuler par des transformations.

L'un des langages de description de modèle le plus connu est UML (« *Unified Modeling Language* », [OMG UML 2007]). Il s'agit d'un langage de modélisation semi-formel, standardisé par l'OMG (« *Object Management Group* ») depuis 1997, qui trouve ses racines dans la programmation orientée objet. Le langage de contrainte OCL (« Object Constraint Language », [OMG OCL 2010]), fortement lié à UML, permet d'exprimer des propriétés sur les modèles UML. Les diagrammes de classe UML et le langage OCL sont souvent utilisés de pair pour décrire toutes les contraintes d'un langage dédié de modélisation.

Pour la transformation de modèle, les techniques les plus courantes sont l'utilisation de l'API EMF (« *Eclipse Modeling Framework* », [Steinberg et al. 2008]), du langage Kermeta ([Muller et al. 2005]) ou encore du standard QVT ([OMG QVT 2011]) de l'OMG.

Dans cette section, nous présentons le langage Express ([Express 1994]) que nous avons utilisé pour nos travaux. Il a été défini dans le cadre d'un standard nommé STEP (« Standard for Exchange Product model data »). L'objectif initial de ce langage est de représenter des modèles de données dans le secteur de l'ingénierie. Son principal avantage est sa complétude, puisqu'il permet de décrire des concepts structuraux, descriptifs et procéduraux dans un formalisme clair. Ceci a fortement motivé notre choix d'Express comme langage de définition de DSML et comme moyen de spécification de passerelles entre des modèles.

2.3.1 Modélisation avec Express

Un modèle Express est décrit par un diagramme d'entités dans lequel les entités (« *ENTITY* ») sont des classes possédant des attributs. Les attributs sont typés à l'aide des entités de base du langage ou des entités définies dans le modèle. Express est un langage orienté objet permettant l'héritage multiple à l'aide des mots clés « *SUPERTYPE* » et « *SUBTYPE* ». Lorsque l'attribut d'une entité a pour type une autre entité du modèle, un attribut inverse (« *INVERSE* ») peut être déclaré. Enfin, il est possible de définir des attributs dérivés d'autres attributs à partir d'opérations logiques et du mot clé « *DERIVE* ».

La table 2.1 illustre un modèle d'entités dans lequel les avions héritent des propriétés des véhicules et les compagnies aériennes possèdent un ensemble d'avions.

```
1   ENTITY Véhicule SUPERTYPE OF (Avion);
2   Nom: STRING;
3   END_ENTITY;
4
5   ENTITY Avion SUBTYPE OF (Véhicule);
6   Nombre_de_vol: INTEGER;
7   INVERSE
8   possédé_par: Compagnie_aérienne FOR possède;
9   END_ENTITY;
10
11  ENTITY Compagnie_aérienne;
12  Nom: STRING;
13  possède: SET[1:?] OF Avion;
14  DERIVE
15  -- Somme du nombre de vols des avions de la
16  compagnie aérienne
17  total_vol: INTEGER:= […];
18  END_ENTITY;
```

Tab. 2.1 - Entités Express

Les instances des modèles d'entités se présentent sous la forme d'une liste d'instances identifiées par le symbole « # ». La table 2.2 donne un exemple d'instances conformes au modèle de la table 2.1. Nous avons décrit que compagnie aérienne « Air France » possède un avion Airbus A320 ayant réalisé 56 vols.

```
1   #1=Avion('A320',56);
2   #2=Compagnie_aérienne('Air France',[#1]);
```

Tab. 2.2 - Instance des entités Express

2.3.2 Contraintes, fonctions et procédures en Express

L'un des avantages à utiliser Express est de disposer du même langage pour modéliser les entités, décrire des contraintes additionnelles et implémenter des fonctions et des procédures (voir la table 2.3).

Les contraintes sont distinguées en deux familles :

1. des contraintes spécifiques aux entités, déclarées à l'aide du mot clé « *WHERE* » et qui portent sur les attributs de l'entité concemée ;
2. des contraintes globales, déclarées en dehors des entités par le mot clé « *RULE* » et qui portent sur l'ensemble des entités du modèle – l'intérêt étant de pouvoir déclarer des règles sur la taille des instances ou sur des contraintes relevant de plusieurs attributs de différentes entités. Nous utilisons alors « *QUERY* » pour exprimer des ensembles d'instances.

L'implémentation de fonctions et de procédures permet d'externaliser des calculs sur des attributs d'entités. Elles sont déclarées respectivement à l'aide des mot clé « *PROCEDURE* » et « *FUNCTION* ». Express met à disposition toutes les primitives de base d'un langage de programmation : des affectations de variable, des structures conditionnelles (IF, WHILE, CASE) et des opérations mathématiques. Comme elles permettent de manipuler des données, les fonctions et les procédures sont au cœur des opérations de transformation de modèle utilisant le langage Express.

Pour en savoir plus sur le langage Express, se référer à [Express 1994], [Aït-Ameur et al. 2000] ou [Dehainsala et al. 2005]).

```
1   ENTITY Véhicule SUPERTYPE OF (Avion);
2   nom: STRING;
3   END_ENTITY;
4
5   ENTITY Avion SUBTYPE OF (Véhicule);
6   nombre_de_vol: INTEGER;
7   INVERSE
8   possédé_par: Compagnie_aérienne FOR possède;
9   WHERE
10  -- Contrainte spécifique
11  wr1: nombre_de_vol < 10000;
12  END_ENTITY;
13
14  ENTITY Compagnie_aérienne;
15  nom: STRING;
16  possède: SET[1:?] OF Avion;
17  DERIVE
18  -- Somme du nombre de vols des avions de la
19  compagnie aérienne
```

```
20      total_vol: INTEGER:= nb_vol(SELF);
21      END_ENTITY;
22
23      RULE FOR (Compagnie_aérienne)
24      WHERE
25      -- Contrainte globale fixant arbitrairement
26      un nombre limite de compagnies aériennes.
27      Wr1: SIZEOF(QUERY(c <* Compagnie_aérienne))< 100;
28      END_RULE;
29
30      FUNCTION nb_vol(c: compagnie_aérienne):
31      INTEGER;
32      -- la fonction retourne le total du nombre
33      de vols réalisés par tous les avions de la
34      compagnie aérienne
35      LOCAL
36          I : INTEGER;
37          result : INTEGER:=0;
38      END_LOCAL;
39
40      REPEAT I:=LOINDEX(c.possède) TO HIINDEX(c.possède);
41          result := result + c.possède[I].nombre_de_vol;
42      END_REPEAT;
43
44      RETURN (result);
45      END_FUNCTION;
```

Tab. 2.3 - Entités Express avec contraintes et fonctions

2.3.3 Choix d'Express comme langage de modélisation et de transformation de modèle

Nous avons choisi le langage Express pour nos travaux principalement pour les caractéristiques suivantes :

- il permet de développer des modèles, de valider des contraintes et de manipuler des modèles autour d'un formalisme homogène ;
- il est orienté objet et conforme à la philosophie du MOF ;
- il permet un prototypage rapide du fait de sa syntaxe claire ;
- des environnements de développement existent pour assister la conception et la validation de modèles. Ces environnements, tels que ECCO ou JSDAI ([Express 1994]) s'appuient sur l'API SDAI permettant d'accéder aux données contenues dans un modèle Express.

3 Des modèles pour la conception fonctionnelle

Aujourd'hui, les concepteurs de l'avion ne sont plus capables de faire face à la complexité des systèmes sans utiliser de modèles, que ce soit pour la gestion des exigences, pour le développement des architectures des systèmes, pour les analyses (par exemple de la performance ou de la sécurité) ou pour la validation et la vérification des architectures. Cette approche d'ingénierie des systèmes basée sur les modèles découle de l'IDM et est couramment appelée MBSE (« *Model-Based Systems Engineering* », [MBSE]). Nous pouvons distinguer deux familles de modèles de conception :

- les modèles décrivant les fonctionnalités des systèmes ;
- les modèles décrivant les architectures matérielles des systèmes.

En tant qu'analyse fonctionnelle préliminaire, l'analyse des risques est réalisée sans disposer d'une connaissance détaillée des architectures matérielles. En revanche, de nombreuses informations relatives aux fonctions sont capitalisées dans les modèles décrivant les fonctionnalités des systèmes. Ces modèles font l'objet de cette section. Dans la section 3.1, nous introduisons les concepts généraux de la modélisation des architectures opérationnelles et fonctionnelles de systèmes socio-techniques. La section 3.2 présente le diagramme d'activités qui est aujourd'hui la syntaxe concrète la plus couramment utilisée chez divers industriels dans le domaine de la modélisation des architectures fonctionnelles. Puis, la section 3.3 donne la syntaxe abstraite et la sémantique des modèles d'architectures opérationnelles et fonctionnelles réalisés au sein d'Airbus, sur la base des méthodes et outils présentés en sections 3.1 et 3.2, pour assister la conception préliminaire de l'avion. Nous montrons enfin, en section 3.4, ce que ces modèles peuvent apporter à l'analyse des risques et en quoi ils sont également insuffisants.

3.1 Des modèles d'architectures opérationnelles et fonctionnelles

Lors des étapes préliminaires de la conception, ce sont bien les modèles décrivant les fonctionnalités des systèmes qui sont définis et utilisés pour assister diverses analyses. Ces modèles, appelés couramment **modèles d'Architectures Opérationnelles et Fonctionnelles** (ou **modèle A.O.F.**), représentent plusieurs notions générales dont, principalement, les fonctions d'un système socio-technique et leurs interactions au cours de phases d'exploitation du système. Cette section vise à introduire les différentes notions et les principes de modélisation des systèmes socio-techniques dans le cadre général. Une application spécifique à Airbus est présentée dans la section 3.3.

3.1.1 Définitions des activités, des opérations et des fonctions

La définition des termes introduits dans cette section peut être trouvée dans ([Reynolds, 2006], [ALFA, 2012]). Bien que tirée de la conception aéronautique, ces définitions ont leur correspondant dans d'autres domaines industriels.

L'objectif de chaque modèle A.O.F. est la description des tâches réalisées par le système qu'il modélise.

Définition : Activité

Une activité est définie comme une tâche élémentaire réalisée par un système. Toute activité peut être décrite par un ensemble de sous-activités élémentaires qui la composent.

Le développement des modèles A.O.F. s'effectue principalement en trois étapes, qui permettent de caractériser le système socio-technique selon des points de vue complémentaires. Les modèles A.O.F. intègrent la totalité ou une partie de ces trois couches.

1er étape : la couche opérationnelle

Le premier point de vue se concentre sur la question : « comment le produit sera utilisé ? ». En répondant à cette question, les concepteurs construisent la couche dite opérationnelle. Cette couche de développement met en évidence le fait que chaque système est conçu pour une exploitation spécifique. Les utilisateurs interagissent avec le produit à travers les opérations.

Définition : Opération

Une opération est définie comme une activité ou une portion d'activité réalisée exclusivement par des acteurs humains.

Exemple. Les différentes exploitations d'un avion comprennent le transport de passagers, l'acheminement de marchandises, des utilisations militaires etc. Les acteurs de l'avion sont alors principalement les pilotes. Mais nous pouvons identifier aussi les équipes de maintenance, les équipages cabine, des passagers ou encore des parachutistes.

2em étape : la couche fonctionnelle

Le deuxième point de vue répond à la question : « que fait le produit ? ». En réaction aux opérations réalisées par les acteurs, le produit effectue à son tour des tâches programmées, appelées fonctions.

Définition : Fonction

Une fonction est définie comme une activité ou une portion d'activité exclusivement réalisée par le produit.

Remarque. Cette définition est imposée par la conception aéronautique. Elle est plus concrète que la définition donnée au chapitre 1, mais nous constatons que les deux définitions sont conformes. Parfois, dans d'autres cadres industriels, le terme « fonction » désigne et remplace la notion d'activité réalisée conjointement par l'ensemble Homme-Machine.

Exemple. Tous les avions partagent des fonctions de haut niveau, tels que « Se déplacer au sol et dans les airs » et « Transporter une charge utile ». Mais par rapport aux opérations qui seront réalisées avec l'avion, de nouvelles fonctions (en particulier dans le domaine militaire) peuvent s'ajouter.

3em étape : la couche matérielle

Le dernier point de vue est basé sur la question : « comment le produit remplit ses tâches ? ». Les fonctions sont supportées et réalisées par des architectures matérielles décrivant les composants physiques et logiciels du système. Comme pour les activités, cette description admet plusieurs niveaux de détails.

Dans le cadre des modèles A.O.F., cette couche est très rarement détaillée. Lors des phases préliminaires de la conception, l'intérêt se porte uniquement sur les liens entres les fonctions et les composants et non sur les interactions directes entre les composants.

Exemple. Nous identifions un moteur comme composant matériel de l'avion. Mais ce moteur est également composé d'un ensemble d'autres composants dont des turbines, des calculateurs, une tuyère etc.

3.1.2 Définition des dépendances fonctionnelles

L'objectif des modèles A.O.F. est non seulement de représenter les trois couches de développement présentées dans la section précédente, mais aussi de capturer les interactions entre les trois couches et au sein des couches opérationnelles et fonctionnelles. Nous identifions ainsi des interactions :

- entre des opérations dans les cas où des acteurs partagent des informations ;
- entre des fonctions quand ce sont des actions du produit qui communiquent entre elles ;
- entre des opérations et des fonctions quand les acteurs et le produit échangent des informations ;
- entre des fonctions et des composants matériels, représentant ainsi l'allocation des fonctionnalités sur les architectures matérielles.

La connaissance de ces interactions aide à la spécification des fonctions lors des phases préliminaires de la conception. En effet, les trois derniers types d'interactions décrivent les interfaces de chaque fonction avec les acteurs, d'autres fonctions et les architectures matérielles. Les interactions du premier type décrivent l'évolution de l'exploitation du produit, c'est-à-dire le contexte d'exploitation des fonctions. C'est pourquoi ces interactions sont également appelées **dépendances fonctionnelles**.

La compréhension des dépendances fonctionnelles est la problématique fondamentale des phases préliminaires de la conception et des modèles A.O.F.. Des problématiques telles que la propagation des pannes ou le partage de ressources physiques sont directement liées aux dépendances fonctionnelles (pour la propagation des pannes, voir la section 3.4.2 du chapitre 1).

3.1.3 Ingrédients des modèles de systèmes socio-techniques

Même si chaque modèle A.O.F. a son propre niveau d'abstraction, ses propres notations et son propre point de vue, tous les modèles de cette famille respectent des concepts généraux propres à la modélisation des systèmes socio-techniques. La figure 2.4, ci-dessous, présente les éléments qui peuvent composer de tels modèles ([Duval et al. 2012]).

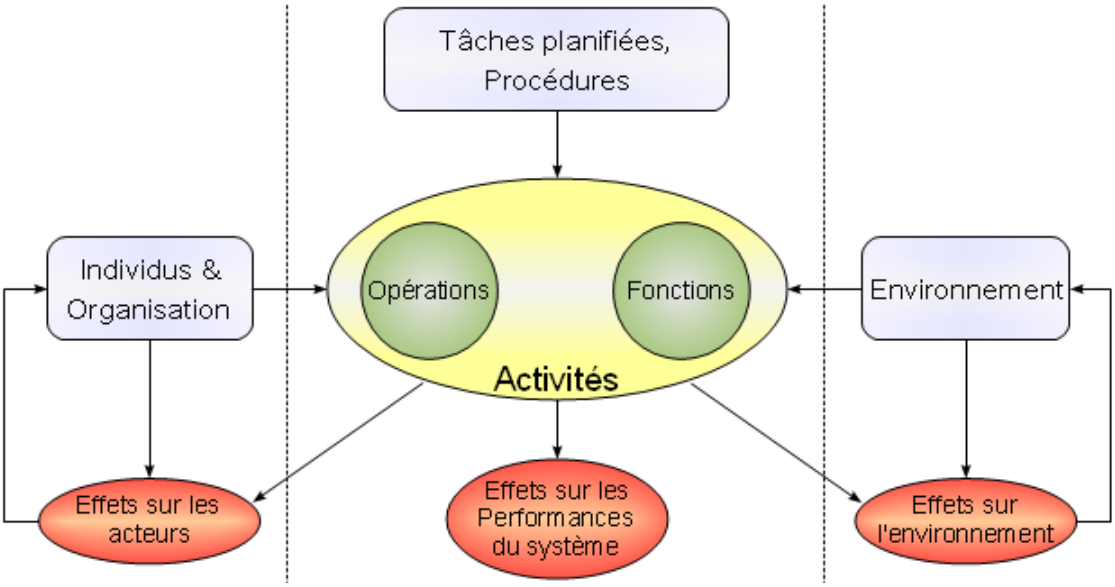


Fig. 2.4 - Architecture d'un système socio-technique

Le cœur d'un modèle A.O.F. est la modélisation des activités réalisées par le système Homme-Machine. Le modèle peut ensuite s'intéresser à plusieurs caractéristiques de ces activités, tels que leur séquencement, leur criticité du point de vue de la sécurité, leur temps d'exécution etc. L'aspect humain peut également être pris en compte avec les connaissances des caractéristiques des acteurs et des organisations. Et enfin, l'environnement du système peut également avoir un impact sur le comportement de certaines activités. Ces trois sources de données (tâches et procédure, individu et organisation, puis environnement) enrichissent la définition des activités.

Le but de tout système socio-technique est la production de résultats. Ces résultats forment l'aboutissement des contributions des activités réalisées par le système. En premier lieu, les concepteurs peuvent s'intéresser à la performance avec laquelle les résultats sont produits et vérifier si cette performance est conforme au cahier des charges du système. Mais selon le point de vue des analyses, un tel modèle A.O.F. peut également servir à étudier les impacts de la réalisation des activités sur les acteurs ou sur l'environnement du système. Ainsi, nous remarquons que la réalisation de certaines activités peut modifier le contexte d'exécution des activités du système.

En pratique, il est extrêmement rare qu'un modèle intègre toutes les caractéristiques présentées sur la figure 2.4. Beaucoup de modèles A.O.F. ne se concentrent que sur la partie centrale pour étudier la performance des systèmes. Les modèles d'étude des Facteurs Humains se concentrent, eux, davantage sur la partie gauche de la figure.

Exemple. Considérons un avion et son équipage, dans sa globalité. Un modèle A.O.F. de cet avion à un haut niveau d'abstraction peut représenter :

- les activités de l'avion, telles que transporter des passagers, décoller, voler d'une position à une autre, atterrir etc. ;

- les procédures, donnant l'enchaînement des activités, du décollage à l'atterrissage ;
- les acteurs de l'avion tels que l'équipage cockpit et l'équipage cabine ;
- l'environnement de l'avion (température, météo etc.) ;
- une évaluation de la capacité de l'avion à réaliser un vol performant (vitesses de vol respectées, sécurité assurée etc.) et de l'impact des activités sur les performances futures (usure des pièces etc.) ;
- une évaluation de l'impact des activités sur les acteurs (fatigue, stress etc.) ;
- une évaluation de l'impact des activités sur l'environnement, telle que la pollution par exemple.

3.2 Langage de modélisation des architectures fonctionnelles – les diagrammes d'activités

Il existe de nombreuses méthodes et outils de modélisation pour assister la conception des systèmes aéronautiques, en décrivant les liens entre les différents équipements qui les composent. Un état de l'art est dressé dans [Estefan 2008]. Les plus courants sont le langage AADL (« *Architecture Analysis and Design Language* », [Feiler et al., 2006]), SCADE ([Dormoy, 2008]) basé sur le langage LUSTRE ([Halbwachs et al., 1991]) et Matlab/Simulink ([Matlab Simulink]). Ces outils ne sont pas propres à l'aéronautique.

Cependant, les interactions nombreuses entre les systèmes embarqués poussent les concepteurs à travailler à un niveau d'abstraction supérieur au niveau matériel et logiciel : le fonctionnel. Lorsqu'ils souhaitent modéliser la structure fonctionnelle d'un système, les concepteurs utilisent les outils de représentation graphique plus simples et interdisciplinaires. Ces outils doivent toutefois être suffisamment expressifs pour capturer correctement les interactions entre les fonctions.

Dans ce but, de nombreuses représentations ont été développées depuis plus d'un demi-siècle. Les plus connues et anciennes sont SADT (« *Structured Analysis and Design Technique* », [Baguet 2005]) et FAST (« *Functional Analysis System Technique* », [Baguet 2005]), qui sont apparues bien avant les modèles d'architectures systèmes pour faire les analyses de besoins. Ces modèles sont toutefois insuffisants pour représenter toute la complexité des interactions entre les fonctionnalités des systèmes.

Les diagrammes EFFBDs (« *Enhanced Functional Flow Block Diagrams* », [Seidner, 2009]), également appelés diagrammes d'activités, développés à la fin des années 70 à partir des FFBDs (« *Functional Flow Block Diagrams* »), sont une version moderne des modèles d'étude fonctionnelle ([Long, 1995]). Ils figurent aujourd'hui parmi les plus utilisés en ingénierie système, en permettant de représenter le comportement fonctionnel de systèmes complexes, distribués, hiérarchiques, concurrents et communicants.

3.2.1 Présentation générale des diagrammes d'activités

Un diagramme EFFBD décrit l'activation de fonctionnalités d'un système dans un ordre précis correspondant à un scénario d'exécution donné. De tels scénarios d'exécution peuvent par exemple être des scénarios de vol dans le cadre de l'aéronautique. L'ordre des fonctions est défini par :

- les attributs dynamiques des fonctionnalités (en particulier leur durée d'exécution) ;
- les structures de contrôle du diagramme, qui transfèrent le flux de contrôle d'un bout à l'autre du scénario d'exécution ;
- les flux de données entre les fonctions.

La figure 2.5, ci-dessous, illustre ces trois caractéristiques.

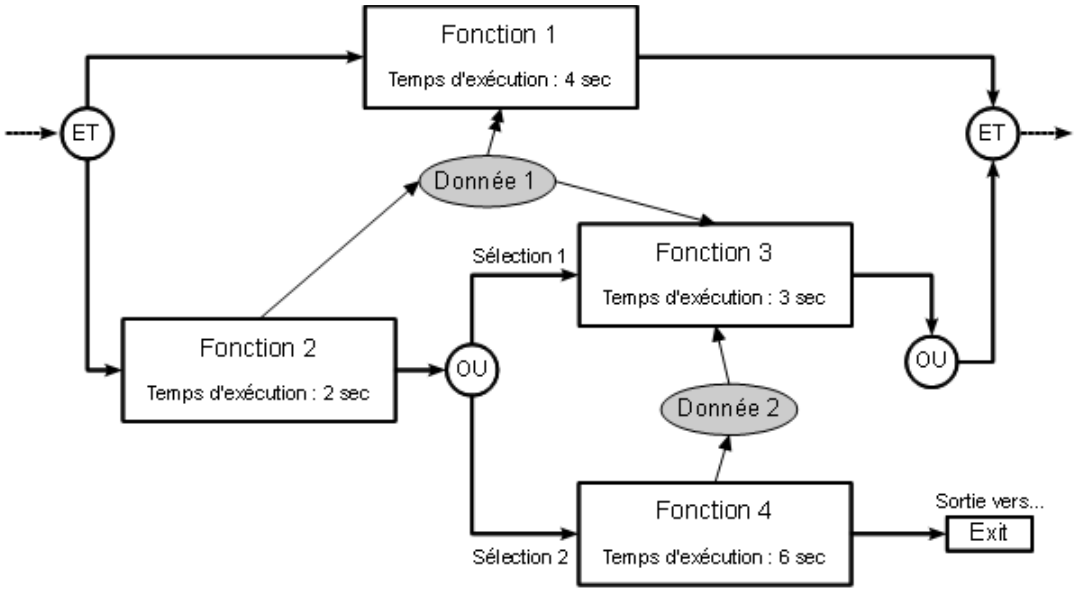


Fig. 2.5 - Exemple de diagramme EFFBD

Un tel diagramme se lit dans le sens d'activation des fonctions, c'est-à-dire de gauche à droite. Les flèches horizontales et verticales guident la lecture. Les deux flèches situées de part et d'autre du diagramme, permettent de repérer ses points d'entrée et de sortie, mais elles n'ont pas de représentation normalisée. Cet exemple théorique est composé de quatre fonctions, toutes caractérisées par leur propre temps d'exécution. Il contient également trois structures de contrôle :

- un « ET » logique signifiant une exécution simultanée ;
- un « OU » logique signifiant une sélection ;
- un « EXIT » signifiant une sortie du modèle.

Il existe plusieurs autres structures de contrôle telles que les boucles ou les itérations.

La figure 2.5 contient enfin trois échanges de données.

- Le premier est un « *trigger* » de la fonction 2 à la fonction 1 : il est représenté par une double flèche et signifie que la fonction 1 attend la donnée pour pouvoir s'exécuter. Il est possible de faire l'analogie avec le domaine de la programmation temps réel dans laquelle une fonction peut représenter une tâche d'un programme. Alors, un « *trigger* » représenterait un transfert d'information avec synchronisation.

– Les deux suivants sont des « *output* » de la fonction 2 à la fonction 3 et de la fonction 4 à la fonction 3 : ce type de donnée est représenté par une simple flèche et signifie que la fonction peut s'exécuter même si elle ne reçoit pas la donnée en entrée. Par analogie avec le domaine de la programmation temps réel, une donnée reçue par « *output* » se comporterait comme une boite au lettre à écrasement.

Nous en déduisons que seul le « *trigger* » contribue à l'ordre d'activation des fonctions du système.

3.2.2 La syntaxe des diagrammes d'activités

La sémantique du formalisme EFFBD est présentée dans le détail dans [Seidner, 2009] sous sa forme mathématique. La structure des diagrammes EFFBDs est proposée sous la forme d'un méta-modèle dans [McInnes et al., 2011]. Bien qu'intuitifs, les diagrammes EFFBDs sont très riches, principalement à cause du nombre important de structures de contrôle disponibles et donc de comportements. Nous présentons ci-dessous une partie de la structure et de la sémantique des EFFBDs. En particulier, nous nous affranchissons des aspects temporels et des structures de contrôle autres que les trois que nous avons illustrés sur la figure 2.5. Nous invitons le lecteur à se référer à la thèse de C. Seidner ([Seidner, 2009]) pour connaître la formalisation complète des diagrammes d'activités.

Définition : structure d'un EFFBD

Un modèle EFFBD est un 4-uplet $\epsilon = < N, D, L, n_0 >$ *où*

– $N = \mathrm{And}_{\mathrm{in}} \cup \mathrm{And}_{\mathrm{out}} \cup \mathrm{Or}_{\mathrm{in}} \cup \mathrm{Or}_{\mathrm{out}} \cup Fct \cup Exit$ *est l'ensemble des nœuds du modèle, composé des sous-ensembles deux à deux disjoints suivants :*

- $\mathrm{And}_{\mathrm{in}}$ *est l'ensemble des portes logiques ET entrantes*
- $\mathrm{And}_{\mathrm{out}}$ *est l'ensemble des portes logiques ET sortantes*
- $\mathrm{Or}_{\mathrm{in}}$ *est l'ensemble des portes logiques OU entrantes*
- $\mathrm{Or}_{\mathrm{out}}$ *est l'ensemble des portes logiques OU sortantes*
- Fct *est l'ensemble des fonctions*
- $Exit$ *est l'ensemble des structures de sortie*

– D *est l'ensemble des données du modèle*

– $L = L_c \cup L_{Fct,c} \cup L_{Fct,l} \cup L_{Fct,p}$ *est l'ensemble des arcs, également appelés « liens ». Il est composé de quatre ensembles deux à deux disjoints :*

- $L_c \subseteq N \times N$ *, l'ensemble non vide des arcs de contrôle entre les nœuds*
- $L_{Fct,c} \subseteq Fct \times D$ *, l'ensemble des arcs de consommation d'une donnée par une fonction (lien « trigger »)*
- $L_{Fct,l} \subseteq Fct \times D$ *, l'ensemble des arcs de lecture d'une donnée par une fonction (lien « output »)*
- $L_{Fct,p} \subseteq Fct \times D$ *, l'ensemble des arcs de production d'une donnée par une fonction*

– $n_0 \in \mathrm{And}_{\mathrm{in}} \cup \mathrm{Or}_{\mathrm{in}} \cup Fct$ *est le nœud initial*

Le métier de la conception implique de respecter quelques règles supplémentaires de modélisation. Nous considérons alors la notion de « EFFBD bien formé » :

Propriété : EFFBD bien formé

Un modèle EFFBD est dit bien formé s'il respecte les propriétés suivantes. Les traductions mathématiques de ces propriétés se trouvent dans [Seidner, 2009].

– *Structures complètes : chaque nœud « ET » et « OU » entrant est lié à un unique nœud « ET » et « OU » sortant respectivement.*

– *Continuité : tous les nœuds doivent pouvoir être atteints à partir du nœud initial.*

– *Imbrication : deux structures de contrôle sont soit en séquence soit entièrement imbriquées l'une dans l'autre.*

– *Modèle borné : les ensembles de la structure et de la sémantique des EFFBDs sont bornés.*

3.2.3 La sémantique des diagrammes d'activités

Le principe sémantique d'un diagramme d'activités est de déterminer l'état d'activité des nœuds du modèle. Une fonction d'évaluation de cet état est donc définie. Puis nous donnons la sémantique d'un modèle EFFBD.

Sémantique d'un EFFBD

Soit $\Delta = \{inactif, activé, exécuté\}$ *l'ensemble des états possibles d'un nœud.*

La fonction d'évaluation de l'état des nœuds et alors définie par : $eval : N \rightarrow \Delta$.

L'état d'un modèle EFFBD est défini par l'évaluation de tous ses nœuds : $s : 2^N \rightarrow 2^\Delta$.

La sémantique d'un modèle EFFBD se présente sous la forme d'un système de transitions, par le triplet $\|\epsilon\| = \{S, s_0, \rightarrow\}$ *où :*

– $S \subseteq \Delta^N$ *est l'ensemble des états du modèle*

– $s_0 = s(n_0) = \left\{ \begin{array}{c} eval(n_0) = activé \\ \forall n \in N / n \neq n_0, \ \ eval(n) = inactif \end{array} \right\}$ *est l'état initial*

– $\rightarrow \subseteq S \times N \times S$ *est la relation de transition discrète caractérisant l'évolution pas à pas de l'état du modèle lors de l'exécution des nœuds. Cette relation peut être trouvée dans [Seidner, 2009]. Elle est au cœur du formalisme EFFBD puisqu'elle décrit le flux d'activation des fonctions.*

3.2.4 Outils intégrant les diagrammes d'activités

Le formalisme EFFBD a été pensé pour n'être employé qu'à travers des outils graphiques. Les trois principaux outils sont présentés ci-dessous.

1. L'atelier logiciel CORE, développé depuis 1992 par Vitech Corporation (actuellement disponible en version 8, [CORE]). Il s'agit principalement d'une base de données robuste, de forte capacité et

collaborative. L'outil propose différentes vues avec les formalismes SADT ou encore FAST. Il propose également des diagrammes d'activités sous la forme FFBD et EFFBD. Son principal atout est son interface, le rendant simple d'utilisation pour les concepteurs.

2. SysML (« *Systems Modeling Language* », [OMG SysML, 2007]) a vu le jour en 2007 à partir du langage UML. A ce titre, SysML hérite du format XMI facilitant l'échange avec d'autres outils et hérite également de la philosophie d'extension du langage UML à l'aide de profils. SysML met à disposition plusieurs nouveaux diagrammes (diagramme des exigences et diagramme d'allocation souvent utilisés dans l'ingénierie) et ne conserve qu'un sous-ensemble des diagrammes d'UML, dont le diagramme d'activités. Plusieurs outils ont été développés sur la base de SysML, dont Papyrus ([Papyrus]) et TopCased ([TopCased]), qui ont l'avantage d'être open-source.
3. L'atelier SCADE Suite ([SCADE Suite]), greffé à l'outil graphique SCADE. Cet outil récent fournit des services équivalents aux diagrammes de CORE ou de SysML pour la conception des architectures fonctionnelles. Il présente un avantage supplémentaire par rapport à CORE ou SysML : la formalisation des interactions entre la conception matérielle des systèmes embarqués dans SCADE et la conception fonctionnelle via SCADE Suite, permettant d'éviter le développement de passerelles entre différents outils.

Ces outils implémentent également un module de vérification dynamique des diagrammes d'activités, qui permet en particulier de faire de la simulation en mode continu ou pas à pas, à partir de la définition de la relation de transition. Le but premier de ce module de vérification est de décrire la capture de ressources partagées et de donner une estimation du temps d'exécution d'un scénario. Le simulateur nécessite trois prérequis : un modèle EFFBD bien formé, la définition des temps d'exécution de chaque fonction (un temps par défaut est proposé dans les outils) et un modèle de liaison entre les fonctions et les ressources (ce modèle est vide par défaut). Les ressources se comportent comme des sémaphores pris et libérés par les fonctions. Les résultats de la simulation se présentent sous la forme de chronogrammes.

3.3 Les modèles A.O.F. chez Airbus

Dans cette section, nous présentons dans le détail les modèles A.O.F., produit chez Airbus dans le cadre d'un projet interne : ALFA (« *Aircraft Level Functional Approach* »). Dans la suite de ce mémoire, lorsque nous évoquons des modèles A.O.F., elles désigneront les modèles ALFA présentés ci-après.

Comme nous le verrons dans le chapitre 6, nous souhaitons manipuler les modèles ALFA comme des objets à des fins de transformation de modèle. Le formalisme EFFBD ne permet pas de telles manipulations, c'est pourquoi dans cette section nous formalisons les modèles ALFA à l'aide du langage Express.

3.3.1 Présentation du projet ALFA

Dans la section 3.4.1 du chapitre 1, nous avons évoqué que les différentes activités de conception de l'avion chez Airbus (choix d'architecture, analyses de performance, analyses de sécurité etc.) se basent, toujours aujourd'hui, sur leur propre vision de l'avion pour ce qui est de la description opérationnelle et

fonctionnelle. Cette diversité s'explique par le fait que chaque activité a des besoins spécifiques et nécessite donc une représentation particulière de ces besoins. Le processus de sécurité lui-même ne s'appuie pas exactement sur la même description fonctionnelle que le processus de conception. Ceci peut conduire à de nombreuses difficultés de communication entre les deux métiers.

Le projet **ALFA (« *Aircraft Level Functional Approach* »)** ([ALFA, 2012]), initié en 2008 chez Airbus, vise à proposer une approche fédérative, en définissant une nouvelle activité de modélisation chargée de décrire le comportement opérationnel et fonctionnel de l'avion (considéré comme un système socio-technique), faisant référence pour toutes les activités de conception. En outre, ALFA doit permettre une gestion plus rigoureuse des exigences opérationnelles et fonctionnelles, en centralisant les exigences définies par toutes les activités de conception, afin de faciliter les choix d'architecture.

L'approche mise en place par le projet ALFA est testée sur plusieurs projets d'étude interne à Airbus, dont le cas d'étude **ESFL (« *Ensure Safe Flight and Landing* »)**. L'objectif de ce cas d'étude est de rechercher les conditions, portant sur les fonctions et les systèmes, permettant d'assurer un vol et un atterrissage sûrs, même en cas de pannes. Il s'intéresse en particulier à la décélération au sol après un atterrissage ou lors d'une interruption du décollage. L'application de nos travaux, présentée au chapitre 7, est en lien étroit avec le cas d'étude ESFL.

La figure 2.6 présente les différentes composantes de la description opérationnelle et fonctionnelle fournies par le projet ALFA.

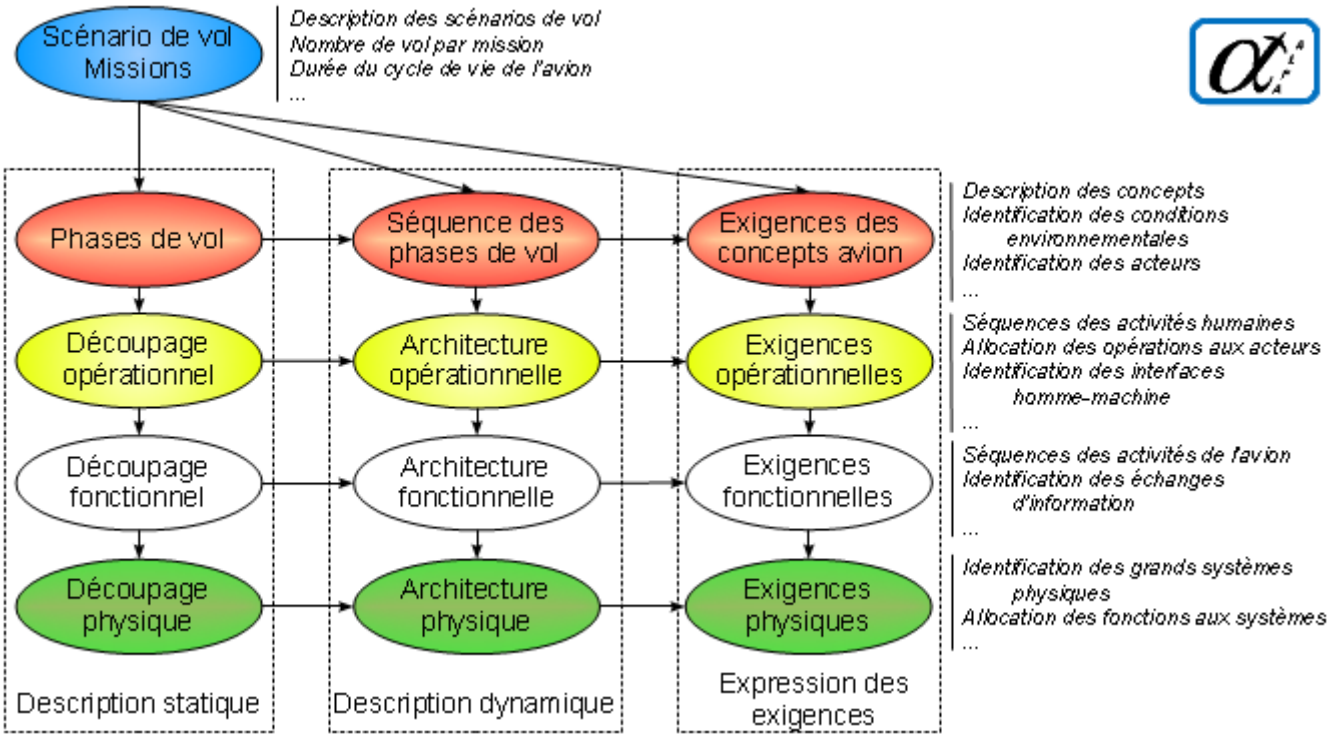


Fig. 2.6 - Périmètre de description opérationnelle et fonctionnelle du projet ALFA

Les lignes de cette figure montrent les différentes couches d'étude de la description opérationnelle et fonctionnelle d'ALFA. Les trois dernières sont les couches opérationnelles, fonctionnelles et matérielles, telles qu'elles sont présentées dans la section 3.1.1. Au dessus de la couche opérationnelle, nous retrouvons la couche « phases de vol » dans laquelle les concepteurs décrivent les contextes d'exploitation du produit. La couche supérieure, « missions » présente les séquences possibles de contextes d'exploitation.

Remarque. Les phases de vol, telles qu'elles sont décrites dans le cadre du projet ALFA, ont exactement la même signification que les phases de vol mises en évidence dans l'analyse des risques.

Les flèches descendantes, sur la figure 2.6, montrent que la description d'une couche sert de base pour la description des couches inférieures. Par exemple, la connaissance des activités de l'équipage (couche opérationnelle) permet de définir les activités que l'avion doit réaliser en réaction (couche fonctionnelle).

Les trois colonnes de la figure 2.6 représentent les différentes contributions du projet ALFA pour la description détaillée des différentes couches de développement.

1. La description statique : chaque couche est décrite initialement avec une description statique (colonne de gauche sur la figure 2.6). Il s'agit d'arbres de décomposition dont les objectifs sont d'identifier les éléments de chaque couche (missions, phases de vol, activités, matériels) et de les ordonner hiérarchiquement. Les éléments feuilles sont plus précis que les éléments en haut de l'arbre. La figure 2.7 donne un exemple d'une telle description fonctionnelle statique pour le contrôle de la trajectoire.

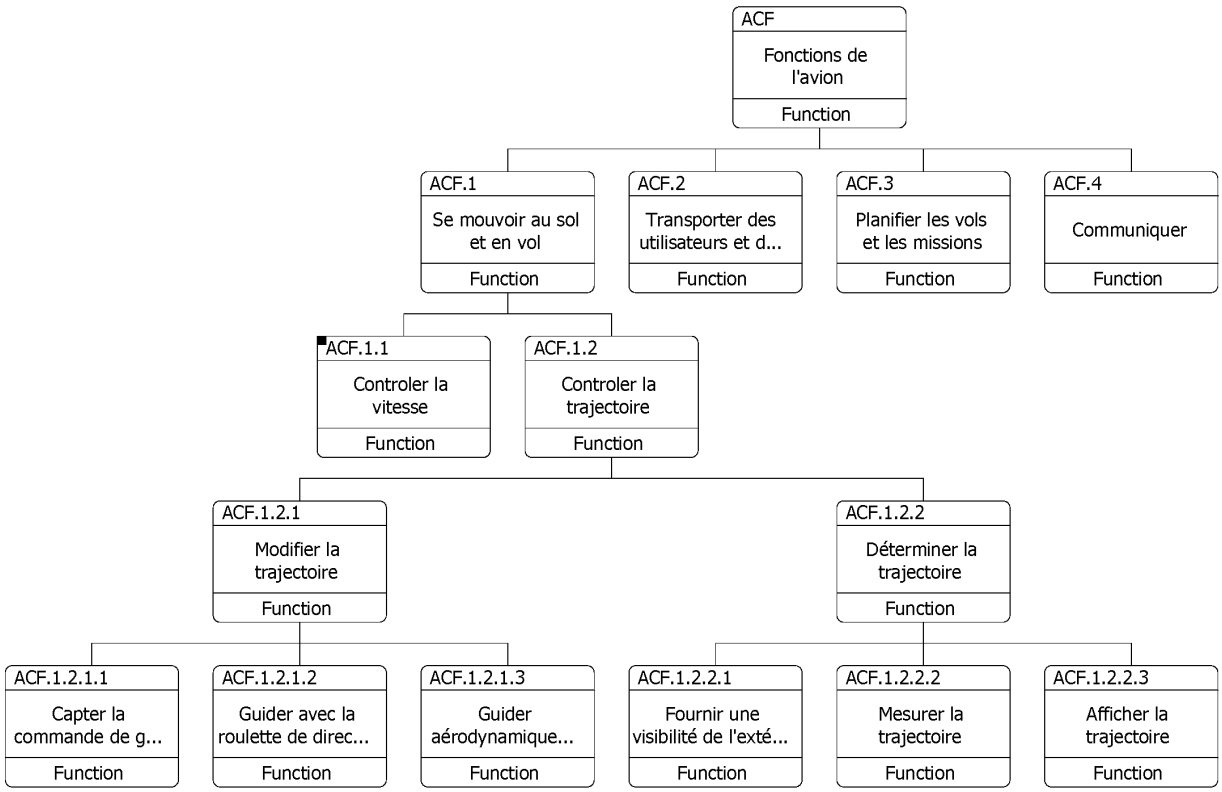


Fig. 2.7 - Arbre de décomposition fonctionnelle partielle de l'avion (exemple théorique)

2. La description dynamique : sur la base des éléments identifiés lors de la phase de description statique, les concepteurs construisent des modèles A.O.F. pour décrire des scénarios d'exploitation de ces éléments, correspondant à l'utilisation prévue du produit en service. La description dynamique de plus haut niveau permet de modéliser des séquences de phases de vol par rapport à des missions bien spécifiques. Puis, pour chaque phase de vol, il est possible de décrire successivement les opérations, les fonctions et le matériel qui sont exploités. Les modèles A.O.F. du projet ALFA se concentrent principalement sur la partie centrale des modèles socio-techniques, telle que décrite sur la figure 2.4.

   En tant que métier de modélisation clairement identifié, les modèles ALFA peuvent être décrits au sein d'un DSML par une syntaxe abstraite, une ou plusieurs syntaxes concrètes et des sémantiques. Les sections suivantes présentent la syntaxe abstraite et la sémantique des modèles ALFA.
3. L'expression des exigences : à partir des deux descriptions précédentes, les spécialistes des différents domaines d'étude de l'avion doivent identifier les exigences de développement attachées aux différents éléments caractérisant le système (missions, phases de vol, activités et matériels). Pour la sécurité, les spécialistes s'appuient sur l'analyse des risques décrite dans le premier chapitre.

3.3.2 Le graphe dirigé acyclique

Les modèles ALFA peuvent être vus comme des graphes dirigés acycliques (« *Directed Acyclic Graph* », DAG). La théorie des graphes propose la définition ci-dessous (inspirée de [Thulasiraman et al., 1992]).

Définition : Graphe DAG

Un graphe dirigé acyclique, plus connu sous le nom de DAG (« Directed Acyclic Graph »), est défini par le doublet $G=(V,E)$ *où :*

- V *est l'ensemble des nœuds (« vertices ») ;*
- $E \subseteq V \times V$ *est l'ensemble des arcs (« edges »).*

Les arcs sont orientés, c'est à dire que $\forall (v_i, v_j) \in V \,/\, e=(v_i, v_j) \in E$, v_i *est le nœud initial et* v_j *est le nœud final.*

Le graphe composé de n *nœuds est acyclique, s'il existe une fonction de l'ensemble des noeuds dans le sous ensemble des* n *premiers entiers,* $ord : V \rightarrow [\![1..n]\!]$, *tel que :*

$\forall (v_i, v_j) \in V \,/\, e=(v_i, v_j) \in E$, $ord(v_i) < ord(v_j)$.

Une telle fonction est appelée ordonnancement topologique.

Nous enrichissons cette définition des deux propriétés ci-dessous.

1. Graphe connexe : il existe un chemin non-dirigé entre tous les nœuds du graphe.
2. Graphe borné : il contient un nombre fini de nœuds.

La table 2.4 présente cette syntaxe des DAG écrite avec le langage Express. Les trois entités qui décrivent les DAG sont déclarées abstraites afin qu'elles ne puissent pas être instanciées directement.

Nous pouvons noter également que les arcs sont définis à partir de leur nœud initial et nœud final et que les nœuds sont tous nommés.

```
1   ENTITY graph ABSTRACT SUPERTYPE;
2         nodes : SET [0:?] OF node;
3         arcs : SET [0:?] OF arc;
4   WHERE
5   -- propriété 1 : chaque graphe contient au
6   moins un nœud
7   -- propriété 2 : graphe connexe
8   -- propriété 3 : graphe orienté
9   -- propriété 4 : graphe acyclique
10  END_ENTITY;
11
12  ENTITY arc ABSTRACT SUPERTYPE;
13        init : node;
14        final : node;
15  INVERSE
16        in_graph : graph FOR arcs;
17  END_ENTITY;
18
19  ENTITY node ABSTRACT SUPERTYPE;
20        name : STRING;
21  INVERSE
22        inputs : SET [0:?] OF arc FOR final;
23        outputs : SET [0:?] OF arc FOR init;
24        in_graph : graph FOR nodes;
25  END_ENTITY;
```

Tab. 2.4 - Extrait du métamodèle des DAG en Express

3.3.3 Introduction de la syntaxe abstraite des modèles ALFA

Un modèle ALFA, noté M_{ALFA}, est un DAG. Il hérite alors de la structure des DAG. La figure 2.8 illustre la structure des modèles ALFA par un diagramme UML. Nous y retrouvons des ingrédients de modélisation des systèmes socio-techniques : les activités (opérations et fonctions) et l'environnement. Un tel modèle ALFA décrit une phase de vol, dans un scénario de vol donné, par les séquences d'activités réalisées pendant cette phase de vol, les échanges de données entre les activités et les impacts environnementaux sur les activités.

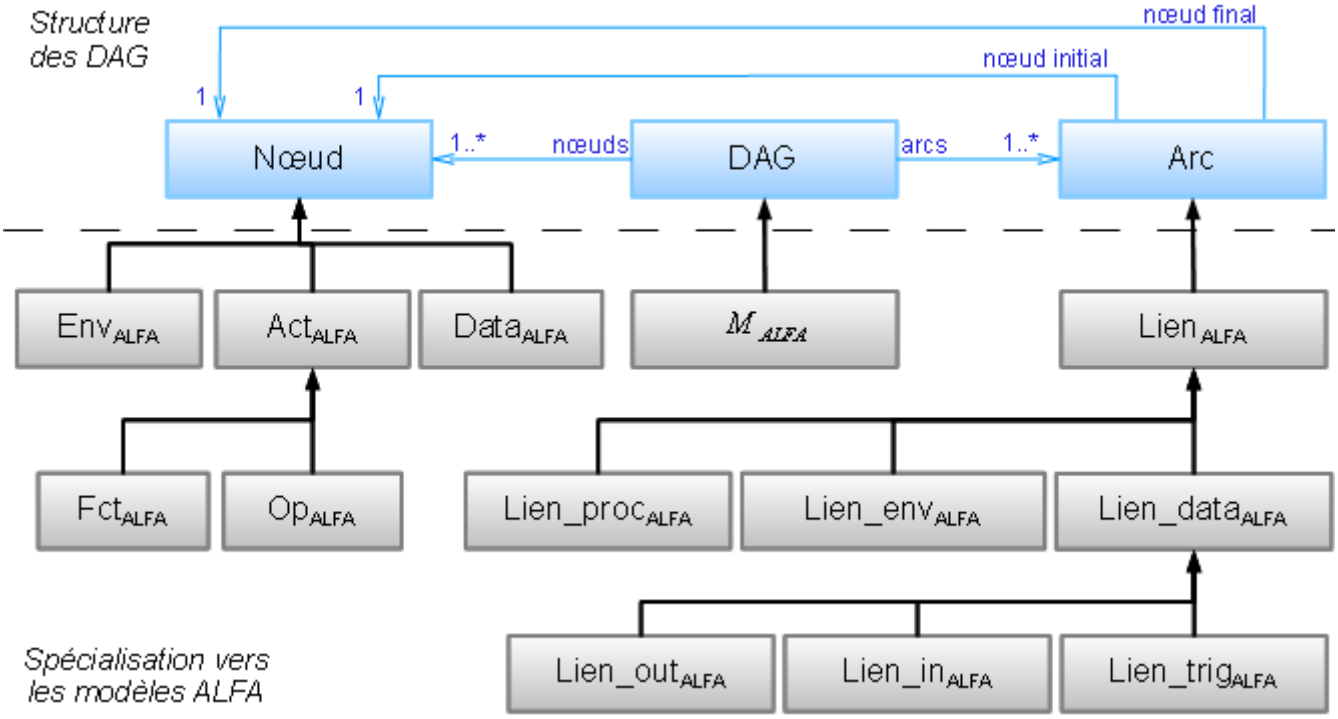


Fig. 2.8 - Structure des modèles ALFA

En Express, la déclaration des modèles ALFA se base sur l'entité « *graph* » vue précédemment (voir la table 2.5).

```
1   ENTITY M_ALFA SUBTYPE OF (graph);
2         activities : SET [0:?] OF Act_ALFA;
3         data : SET [0:?] OF Data_ALFA;
4         envs : SET [0:?] OF Env_ALFA;
5         links : SET [0:?] OF Lien_ALFA;
6   DERIVE
7   -- identification des nœuds et des arcs
8         SELF\graph.nodes : SET[0:?] OF node :=
9   activities + data + envs;
        SELF\graph.arcs : SET OF arc := links;
    WHERE
    -- propriété 1 : le modèle ALFA contient au
    moins une activité.
    END_ENTITY;
```

Tab. 2.5 - Entité des modèles ALFA en Express

Les différents types de nœuds sont définis dans la table 2.6. Comme attendu, les activités sont distinguées en fonctions et opérations. Dans le cadre des modèles ALFA, ces dernières ne possèdent pas d'attributs pertinents pour la sûreté de fonctionnement. En revanche, elles ont plusieurs attributs pertinents pour la conception, tel que leur temps d'exécution.

```
1   ENTITY Act_ALFA ABSTRACT SUPERTYPE
2   SUBTYPE OF (node);
3         running_time : INTEGER;
4   END_ENTITY;
5
6   ENTITY Fct_ALFA SUBTYPE OF (Act_ALFA);
7   END_ENTITY;
8
9   ENTITY Op_ALFA SUBTYPE OF (Act_ALFA);
10  END_ENTITY;
11
12  ENTITY Data_ALFA SUBTYPE OF (node);
13        flight_obj : BOOL;
14  END_ENTITY;
15
16  ENTITY Env_ALFA SUBTYPE OF (node);
17  END_ENTITY;
```

Tab. 2.6 - Entités des nœuds des modèles ALFA en Express

Les échanges de données sont présentés dans le détail dans la section suivante à travers les liens entre les activités et les données. D'ores et déjà nous identifions la notion d'**objectif de vol**. Un objectif de vol est une donnée caractéristique d'une phase de vol, permettant d'estimer sa bonne réalisation. Les données possèdent alors un attribut, nommé « *flight_obj* » qui est un booléen dont la valeur est « vrai » uniquement pour les objectifs de vol.

Exemple. La réussite d'un décollage dépend principalement de la capacité à maintenir l'avion sur l'alignement de la piste et la capacité à atteindre une vitesse suffisante pour décoller. L'un de ces deux objectifs de vol se trouve sur l'exemple de la figure 2.9 sur le contrôle de la trajectoire pendant le décollage. Il est représenté par la donnée « trajectoire contrôlée », étiquetée objectif de vol et identifiée graphiquement par une couleur différente de celle des autres données.

3.3.4 Syntaxe et sémantique des échanges de données

Les activités s'appuient sur des données pour leur bon fonctionnement et transmettent de nouvelles données transformées aux activités suivantes. La table 2.7 présente les entités qui définissent les liens d'échanges de données. Trois familles de liens sont identifiées, en s'inspirant du formalisme EFFBD : les liens sortants, les liens entrants et les liens triggers.

```
1   ENTITY Lien_ALFA ABSTRACT SUPERTYPE SUBTYPE
2   OF (arc);
3   END_ENTITY;
```

```
4       ENTITY Lien_data_ALFA ABSTRACT SUPERTYPE
5       SUBTYPE OF (Lien_ALFA);
6       END_ENTITY;
7       ENTITY Lien_trig_ALFA SUBTYPE OF
8       (Lien_data_ALFA);
9             init : Data_ALFA;
10            final : Act_ALFA;
11      END_ENTITY;
12      ENTITY Lien_in_ALFA SUBTYPE OF
13      (Lien_data_ALFA);
14            init : Data_ALFA;
15            final : Act_ALFA;
16      END_ENTITY;
17      ENTITY Lien_out_ALFA SUBTYPE OF
18      (Lien_data_ALFA);
19            init : Act_ALFA;
20            final : Data_ALFA;
21      END_ENTITY;
```

Tab. 2.7 - Entités des liens de données en Express

Les liens sortants, formalisés « *Lien_out_ALFA* » sont des arcs dont le nœud initial est une activité et le nœud final une donnée. Ces liens représentent la production d'une donnée par une activité. Usuellement, ces liens sont appelés **services** des activités.

Une activité peut délivrer plusieurs services différents. En général le périmètre des activités est construit à partir de la question : « A quoi sert l'activité ? ». Avec ce point de vue, la conception définit les activités par rapport à leurs services et tend alors à décomposer toutes les activités de l'avion de telle sorte qu'à chaque activité est associé un unique service. Ainsi, la majorité des opérations et des fonctions ne fournissent chacune qu'un seul service. Toutefois, il existe quelques d'exceptions, qui ont pour origine les lois de la mécanique et de la mécanique du vol. Par exemple, la vitesse, la trajectoire, l'altitude et la traînée d'un avion sont toutes des données dépendantes les unes des autres. Ainsi, une activité prévue pour agir sur l'un de ces paramètres de l'avion, agit également sur les autres, de façon désirée ou non.

Exemple. La fonction de production de la poussée, « Fournir la poussée », sollicitant les moteurs de l'avion, fournit deux services différents : elle permet de générer une poussée motrice, mais aussi de modifier la trajectoire sur la piste grâce à la poussée différentielle asymétrique.

Les liens triggers et les liens entrants permettent la réception des données par les activités. Leur sémantique est identique à celle définie dans le formalisme EFFBD par les « *triggers* » et les « *outputs* ». Usuellement, les données reçues par liens triggers (respectivement liens entrants) sont appelée **données nécessaires** (respectivement **données utiles**).

3.3.5 Les séquences d'activation des activités

Les activités peuvent se trouver soit au repos, soit elles sont activées et fournissent leurs services. La transition entre ces deux états est appelée l'activation.

Les activités humaines (les opérations) sont activées par les **procédures** et les **prises de décision** par l'équipage. Ces deux moyens sont modélisés différemment et présentés dans les paragraphes ci-dessous. Dans certains cas, une opération peut être en attente d'activation par une combinaison de ces deux moyens d'activation. Alors, toutes les conditions d'activation doivent être remplies pour que l'opération s'exécute.

L'activation des fonctions diffère de l'activation des opérations. En effet, l'avion, de lui-même, ne suit aucune procédure, ni prend la moindre décision. Toutes ses fonctions s'exécutent dans un ordre précis, déterminé lors du développement. Par conséquent, toutes les activations sont forcées. De nombreuses fonctions sont même toujours actives à l'intérieur d'une phase de vol spécifique.

Modélisation des procédures

Lorsque l'opération réalisée suit une planification stricte dans le scénario de vol étudié, elle fait partie des procédures standards. Ces dernières décrivent le séquencement normal d'opérations au cours des différentes phases de vol et en s'adaptant au maximum de scénarios de vol possibles. Les procédures sont propres à chaque type d'appareil, sont formellement décrites, sont enseignées aux pilotes et sont embarquées sur chaque appareil pour être consultables en plein vol. Les procédures sont déterminées en parallèle du développement de l'avion et peuvent servir d'entrées pour certaines analyses.

Les procédures décrivent le séquencement des activités humaines, indépendamment des échanges de données entre les activités. Dans les modèles ALFA, le lien d'activation qui représente un tel séquencement est un lien direct d'une opération à une autre, appelé lien de procédure (voir la table 2.8).

```
1   ENTITY Lien_ALFA ABSTRACT SUPERTYPE SUBTYPE
2   OF (arc);
3   END_ENTITY;
4
5   ENTITY Lien_proc_ALFA SUBTYPE OF
6   (Lien_ALFA);
7         init : Op_ALFA;
8         final : Op_ALFA;
9   END_ENTITY;
```

Tab. 2.8 - Entités des liens de procédure en Express

Dans le cas où une opération reçoit en entrée plusieurs liens d'activation directe, alors cette opération ne s'exécute que si toutes les opérations précédentes ont été exécutées.

Modélisation de la prise de décision

Parfois, les procédures ne permettent pas de sélectionner de façon certaine et objective les opérations à réaliser. Ces cas obligent l'équipage à faire des choix et à s'adapter au contexte opérationnel courant. C'est habituel lorsque des pannes ou des conditions environnementales dégradantes sont identifiées et nécessitent des actions correctives. Il s'agit alors de prises de décision.

Deux moyens sont utilisés pour modéliser la prise de décision.

1. La première méthode est de borner le périmètre du modèle à un scénario de vol précis qui intègre des choix particuliers réalisés par l'équipage. Cette approche a l'avantage de conduire à des modèles de plus petite dimension et donc plus faciles à valider. Les concepteurs la choisissent en priorité quand ils ne s'intéressent pas particulièrement à l'étude de la prise de décision, mais plutôt au séquencement des fonctions dans les différents scénarios de vol.
2. La seconde méthode est de modéliser concrètement la prise de décision dans les modèles ALFA. Les prises de décision s'appuient toujours sur des analyses de la situation opérationnelle (présence de pannes, météo, mode de fonctionnement etc.). Or, les analyses de ces situations sont elles mêmes des activités humaines, éventuellement assistées par des fonctions de l'avion. La prise de décision est alors activée par l'analyse de la situation dans un échange de données nécessaires entre les opérations.

   La prise de décision, elle-même, est une activité humaine ayant un rôle d'aiguillage du flux d'activation opérationnel. Le choix est réalisé par l'équipage à partir d'une certaine connaissance de la situation et d'un jugement humain, par nature imprévisible. Une opération étiquetée « prise de décision » est l'opération initiale de plusieurs liens de procédures (au minimum deux liens). Contrairement aux autres opérations, seul l'un de ces liens, choisi aléatoirement, propage le flux d'activation.

3.3.6 Les conditions environnementales

Les modèles ALFA, ne s'intéressent pas à l'impact des activités sur l'environnement (par exemple en terme de pollution). En revanche, d'éventuelles dégradations d'activités par des conditions environnementales particulières, telles que la pluie ou le fort vent de travers, sont modélisées. Les conditions environnementales sont modélisées par les nœuds « Environnement » et reliées aux activités par les liens d'environnement toujours dirigés de l'environnement vers l'activité (voir la table 2.9).

```
1   ENTITY Lien_ALFA ABSTRACT SUPERTYPE SUBTYPE
2   OF (arc);
3   END_ENTITY;
4
5   ENTITY Lien_env_ALFA SUBTYPE OF (Lien_ALFA);
6         init : Env_ALFA;
7         final : Act_ALFA;
8   END_ENTITY;
```

Tab. 2.9 - Entités des liens d'environnement en Express

La signification de ces liens d'environnement traduit une dégradation potentielle de la performance des activités en cas de présence de la condition environnementale concernée.

Remarque. Le système réel réagit parfois à la présence d'une condition environnementale en déclenchant une procédure de prévention. C'est le cas par exemple en présence de conditions givrantes avant le décollage, nécessitant un dégivrage des surfaces de l'avion. Ce cas n'est considéré dans les modèles ALFA qu'à travers les scénarios de vol. Si nécessaire, l'étude du dégivrage des surfaces de l'avion avant le décollage nécessite le développement d'un modèle distinct qui intègre cette procédure.

3.3.7 Instanciation et implémentation concrète avec les diagrammes d'activités

La figure 2.9 présente un modèle, conforme aux modèles ALFA, décrivant le contrôle de la trajectoire au sol pendant le décollage. Les notations choisies, proches de l'aspect graphique des diagrammes d'activités, ne sont pas normalisées. Cette figure montre que le contrôle de la trajectoire au sol nécessite une activité humaine (l'opération « Gérer la trajectoire ») réalisée par l'équipage cockpit. Si nécessaire, le pilote peut corriger la trajectoire de l'avion en transmettant des commandes de guidage à l'avion. Ce dernier capte les commande et calcule les ordres de guidage (fonction « Calculer le guidage »). Les ordres de guidage sont transmis aux différents actionneurs permettant de modifier la trajectoire : la roulette de direction (fonction « Guider avec la roulette de direction ») et la gouverne de direction (fonction « Guider aérodynamiquement »). Les deux actionneurs participent au contrôle de la trajectoire. Plusieurs conditions environnementales pouvant dégrader les performances de certaines activités ont été identifiées. Par exemple, la pluie ou le givre sur la piste de décollage diminuent les capacités de guidage de la roulette de direction. Cet exemple est un extrait du cas d'étude présenté dans le chapitre 7.

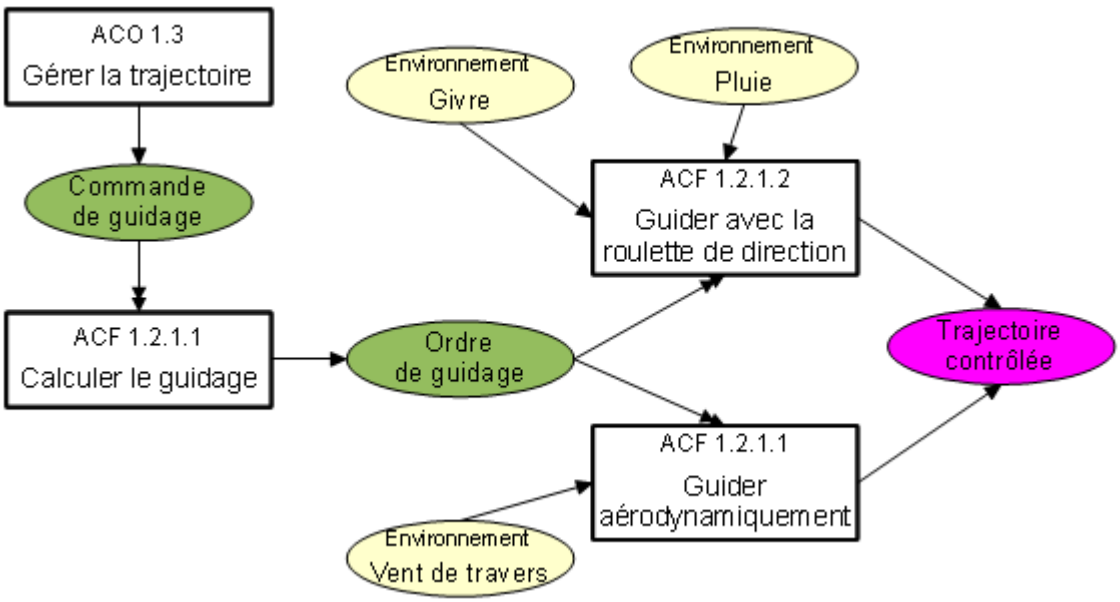


Fig. 2.9 - Contrôle de la trajectoire au sol pendant le décollage (exemple théorique)

Pour des raisons pragmatiques, les architectes de l'avion qui réalisent les modèles ALFA pour décrire les différentes phases de vol des avions, ont choisi le formalisme EFFBD (voir la section 3.2) pour représenter concrètement leurs modèles M_{ALFA}. La figure 2.10 présente le diagramme d'activités équivalent à l'exemple de la figure 2.9.

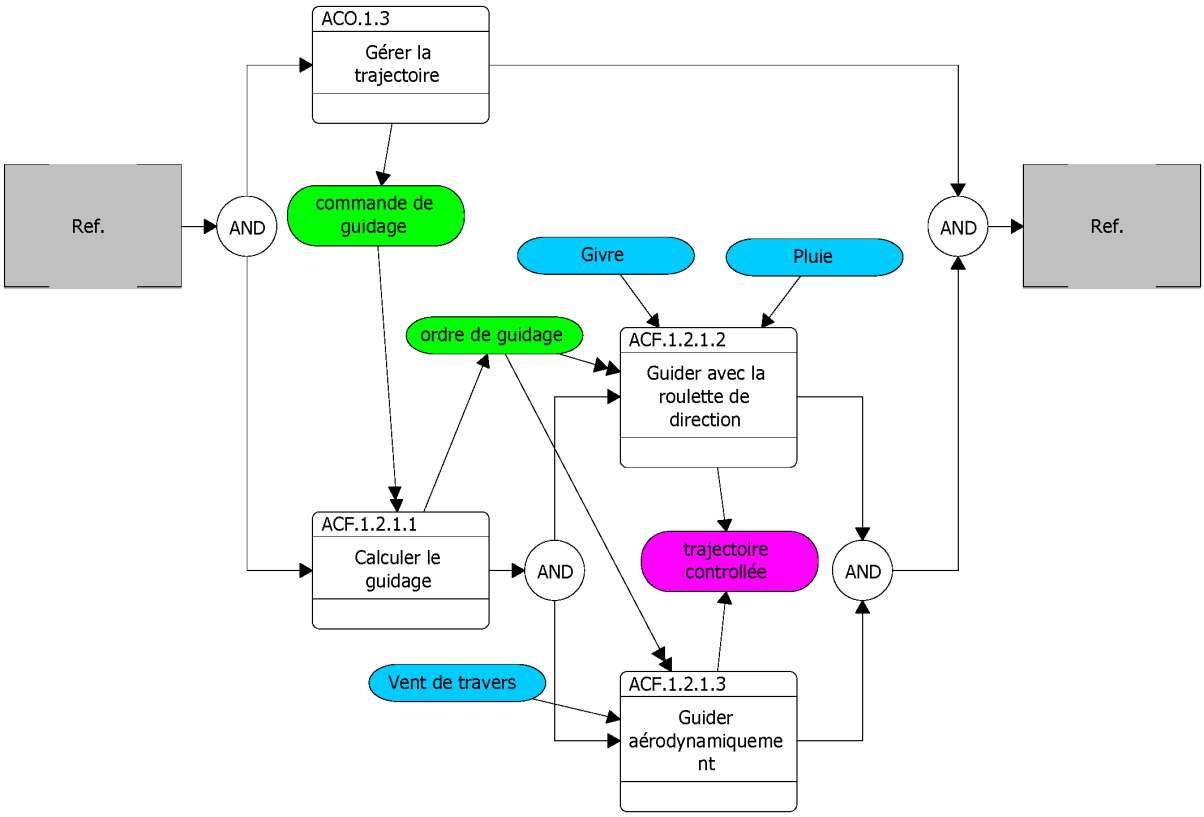


Fig. 2.10 - Modèle EFFBD du contrôle de la trajectoire pendant le décollage

Nous constatons en particulier que les activités sont modélisées par des fonctions EFFBDs alors que les données et les conditions environnementales sont modélisées par des données EFFBDs. Les données nécessaires sont modélisées à l'aide de « *triggers* » et les données utiles à l'aide de « *outputs* ». Lorsqu'elle est modélisée, la prise de décision est représentée par une structure logique en OU. Nous constatons aussi que le formalisme EFFBD nous contraint à ajouter des structure logique en ET pour les liens de procédure.

Les propriétés du DAG imposent aux modèles ALFA la connexité, le caractère borné et l'absence de boucles, nécessaires pour obtenir un modèle EFFBD bien formé (voir la section 3.2.2).

Grâce à la sémantique des diagrammes d'activités, qui exprime l'ordre d'activation des activités, les concepteurs peuvent faire de la simulation et obtenir un chronogramme de l'exécution des activités.

3.4 Apports et insuffisances des modèles ALFA pour l'analyse des risques

De nombreux bénéfices sont attendus des modèles présentés précédemment, en particulier pour résoudre toutes les insuffisances liées à l'analyse des risques (voir la section 3.4 du chapitre 1). La première idée, souvent soutenue par les concepteurs de l'avion, est de réaliser les analyses de sécurité directement à partir des modèles de la conception. Du fait des itérations entre la conception et la vérification des architectures proposées du point de vue de la sécurité, le temps nécessaire pour déterminer une architecture sûre est important. Pourtant, bien souvent, les architectes avion savent quels sont les points de faiblesses des architectures par rapport à leurs précédentes propositions. Ils désirent alors déployer des analyses ciblées et immédiates, et non la mise en œuvre de tout un processus long, difficile et coûteux.

Ce point de vue est compréhensible, car historiquement les activités de sécurité sont contraintes, au niveau de leur support de la phase de conception, par leur objectif final : la certification. Cette exigence de certification requiert la définition et la réalisation d'un processus complexe permettant l'évaluation de la sécurité de l'avion au cours de toutes les phases de son cycle de vie.

Toutefois, malgré leurs apports, les modèles développés par les concepteurs ([Reynolds, 2006], [ALFA, 2012]) sont insuffisants par nature pour réaliser des analyses de sécurité. Les limites sont visibles à deux niveaux et sont détaillées par la suite.

1. Les modèles A.O.F. contiennent de nombreuses informations, telles que les dépendances fonctionnelles, mais n'incluent pas certaines données nécessaires à l'étude des pannes fonctionnelles.
2. Les modèles ALFA s'appuient sur la syntaxe des diagrammes d'activités, dont les outils d'édition et d'analyses ne permettent pas d'étudier la sécurité.

3.4.1 Apports des modèles A.O.F. du projet ALFA

Le chapitre 1 a montré que l'analyse des risques consiste à étudier les conséquences de pannes fonctionnelles sur la réalisation des phases de vol. Or, les modèles ALFA fixent le périmètre de modélisation à chaque phase de vol et met en évidence les paramètres importants de chaque phase de vol à travers les objectifs de vol.

De plus, les propagations de pannes fonctionnelles se font à travers les dépendances fonctionnelles qui sont représentées dans les modèles ALFA. Nous illustrons, ci-dessous, cette relation entre la propagation des pannes et les dépendances fonctionnelles dans les cas de successions fonctionnelles et de fonctionnements parallèles (voir la figure 2.11).

– Nous disposons d'une succession fonctionnelle entre deux fonctions lorsque la première fonction active la seconde. La partie inférieure de la figure 2.11 offre une représentation de cette situation. Son intérêt, du point de vue de la sécurité, est que la défaillance de la fonction 2 risque d'impacter la fonction 3. Avec la perte totale de la fonction 2, par exemple, celle-ci ne serait plus en mesure ni de réaliser ses tâches, ni d'activer la fonction 3.

– Nous disposons d'un fonctionnement parallèle de deux fonctions lorsque les deux fonctions travaillent simultanément dans un même intervalle de temps. Cette situation est représentée sur la figure 2.11 où la fonction 1 est exécutée en parallèle des fonctions 2 et 3. Le fonctionnement parallèle révèle une indépendance fonctionnelle du point de vue des activations. Alors, dans ce cas, nous identifions qu'il n'y a pas de propagation des pannes.

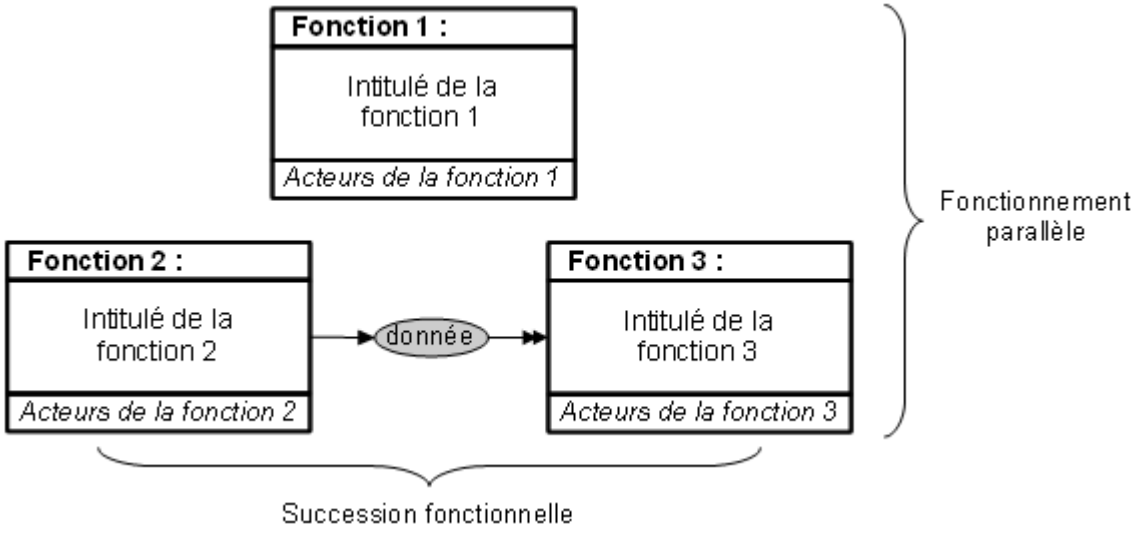


Fig. 2.11 - Succession fonctionnelle et fonctionnement parallèle

3.4.2 Insuffisances des architectures opérationnelles et fonctionnelles

Malgré leurs apports, les modèles ALFA présentent des limitations importantes lorsque nous souhaitons les exploiter pour réaliser des analyses de risques. Toutes ces limitations ont la même origine : les architectes de l'avion, qui conçoivent ces modèles, travaillent sur la conception de l'avion et donc sur sa partie fonctionnelle, alors que les analystes de la sécurité s'intéressent aux comportements dysfonctionnels des architectures. Nous avons identifié deux limitations : limitation dans la description des fonctions et des limitations dans la description de la propagation des pannes fonctionnelles.

Limitation de description des fonctions

Dans les modèles ALFA, les fonctions sont décrites par plusieurs attributs, listés ci-dessous.

– Des identificateurs : toute fonction possède un nom et un numéro qui permettent de la positionner dans le découpage fonctionnel.

– Une durée d'activité : il s'agit du temps d'exécution de la fonction à partir de son activation. Cette durée d'activité est une caractéristique importante de l'étude des performances fonctionnelles de l'avion. Pour les analyses de sécurité, elle peut représenter le temps d'exposition aux risques.

– Une description : il s'agit d'un champ libre permettant de décrire textuellement les fonctionnalités liées à la fonction.

- Des exigences : toutes les activités de l'avion ont leurs propres exigences pour piloter la conception. Les exigences de sécurité sont également enregistrées lorsque l'analyse des risques transmet ses résultats à la conception. Les exigences de performance informent des performances de vol et peuvent donc aider à comprendre les répercussions d'un scénario de pannes.

Bien que certains de ces attributs soient pertinents pour les études de sécurité (les exigences de performance et les identificateurs par exemple), ils ne sont pas suffisants pour l'analyse des risques. En particulier, il manque une description des pannes fonctionnelles. Pour chaque fonction, il est nécessaire d'identifier les pannes fonctionnelles, potentielles et pertinentes, qui peuvent l'affecter. Idéalement, il faudrait formaliser ces pannes fonctionnelles, dans les modèles d'architecture, à partir des quatre modes de pannes de l'analyse des risques (voir la section 3.2.1 au chapitre 1). Pour aller plus loin, d'autres caractéristiques seraient intéressantes pour préciser la propagation des pannes à travers les fonctions : les moyens de détection et de recouvrement des pannes par exemple.

Limitation de description de la propagation des pannes fonctionnelles

Les dépendances fonctionnelles, modélisées dans les modèles ALFA, informent de l'existence de liens entre les fonctions et entre les fonctions et les opérations. Nous avons observé que ces liens permettent de déterminer la propagation des pannes fonctionnelles, mais les modèles ALFA ne définissent pas les domaines de valeurs de ces liens pour permettre d'étudier les impacts de pannes.

Si, à l'aide du formalisme des diagrammes d'activités, les liens de procédure et les liens triggers sont évalués à l'aide de booléens permettant de déterminer les conditions d'activation des activités, les échanges de données ne sont pas du tout évalués. Cela est pourtant nécessaire pour en estimer leur qualité. Dans ce cas, les modes de fonctionnement erronés en particulier ne peuvent pas être propagés dans les modèles.

3.4.3 Limites du formalisme EFFBD pour modéliser les architectures opérationnelles et fonctionnelles

Dans la section précédente, nous avons présenté les limites des modèles ALFA du point de vue des concepts modélisés. A ces limitations s'en ajoutent d'autres, ayant pour origine le choix des diagrammes d'activités comme modélisation concrète. [Seidner, 2009] a montré que la syntaxe des EFFBD peut être liée à celle des graphes sans perte d'informations. En revanche, les diagrammes d'activités ajoutent des contraintes de structure supplémentaires. Les architectes du projet ALFA se sont alors adaptés à ces contraintes faute de développer une syntaxe concrète plus adaptée.

Des dépendances fonctionnelles parasites

La modélisation des flux fonctionnels est obligatoire dans les EFFBD : toutes les fonctions ont exactement une entrée d'activation et une sortie d'activation non vide pour modéliser le flux fonctionnel, en plus de la modélisation des données nécessaires. Les modèles ALFA se sont donc adaptés à ce formalisme. Malheureusement, suite à cette adaptation, de nouvelles dépendances fonctionnelles apparaissent et parasitent les modèles. Elles sont représentées par des lignes horizontales et des portes logiques.

La figure 2.12, ci dessous, représente l'implémentation de la situation de la figure 2.11 du paragraphe 3.4.1 avec le formalisme EFFBD.

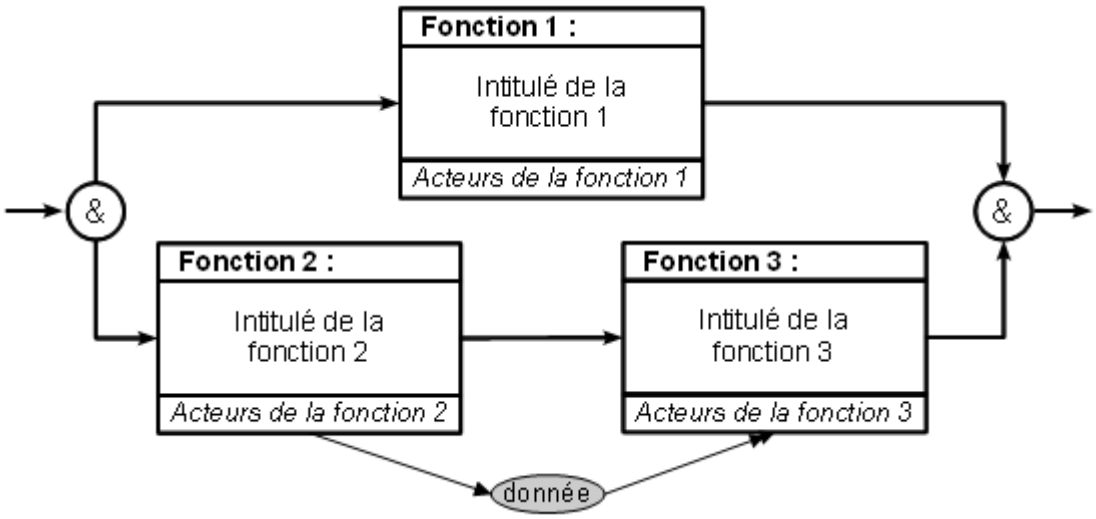


Fig. 2.12 - Dépendances fonctionnelles parasites

Nous constatons sur cette figure que trois nouvelles dépendances fonctionnelles entre les fonctions sont ajoutées :

- les fonctions 1 et 2 sont activées à partir du même flux d'activation. En effet, les modèles EFFBD ont obligatoirement un flux initial et un flux final unique. Ainsi, même au tout début d'un modèle, tous les fonctionnements parallèles sont initialement synchronisés par le flux unique (à l'aide une structure logique en ET) ;
- les fonctions 2 et 3 se succèdent. Bien sûr, cette succession était déjà capturée par l'échange de la donnée nécessaire. L'activation de la fonction 3 par la fonction 2 est simplement dupliquée dans les modèles EFFBD. En pratique, cette dépendance supplémentaire peut avoir un effet très génant ; puisque la succession est acquise par le flux fonctionnel parasite, l'échange de la donnée nécessaire peut être modélisé par erreur comme une donnée utile par les architectes ;
- les fonctions 1 et 3 se synchronisent à la fin de leurs exécutions. Il s'agit de la même problématique que pour la première dépendance fonctionnelle parasite : toutes les activations doivent se terminer par le même flux final.

Ces dépendances fonctionnelles parasites ont un effet sur l'analyse des pannes. En effet, comme nous l'avons vu, le fonctionnement parallèle n'induit normalement aucune propagation des pannes. Mais sur notre figure, avec la synchronisation des fonctions 1 et 3 par une porte logique en « ET », si la fonction 1 est perdue, alors le flux d'activation, en sortie, est perdu. Il y a donc propagation des modes de perte fonctionnelle alors que cela n'est pas désiré.

Remarque. Notons qu'avec une porte « OU », la signification devient : « il suffit que l'une des deux branches s'exécute pour que la seconde ne s'exécute jamais ». Dans ce cas, il y a même une violation du sens fonctionnel du modèle. C'est pourquoi, par défaut, tous les fonctionnements parallèles sont modélisés par des portes logique « ET ». Les portes « OU » ne sont pas utilisées pour les dépendances fonctionnelles mais pour différencier plusieurs scénarios de pannes issus d'une même situation (voir par exemple le modèle du décollage au chapitre 7).

Manque de formalisme pour les conditions environnementales

Les conditions environnementales ont été identifiées comme des paramètres pertinents pour diverses analyses et pas seulement celles dédiées à la sécurité (voir la section 3.3.6). Toutefois, le formalisme EFFBD n'offre aucune classe ni type adaptés pour prendre en considération les conditions environnementales pouvant dégrader les performances d'une activité. Par conséquent, les conditions environnementales sont modélisées comme des données échangées. Elles ne sont distinguées des données que sur la couche graphique des modèles, grâce à un changement de couleur non formalisé.

4 Des modèles pour les analyses de sécurité

Puisque les modèles de la conception sont jugés insuffisants pour assister les analyses des risques (voir la section 3.4), les analystes doivent se tourner vers d'autres techniques de modélisation. Quelques analyses du processus de sécurité font déjà l'objet d'un support par le développement et l'exploitation de modèles. Cette section présente quelques unes de ces techniques, dont AltaRica, que nous avons utilisé.

4.1 Les modèles des pannes

Pour chacun des systèmes qu'ils souhaitent étudier, les analystes de la sécurité commencent toujours par identifier les risques liés à ce système. Puis, à partir de la connaissance des architectures systèmes, ils sont en mesure de vérifier si ces architectures sont robustes au regard des risques. Il s'agit des analyses de type PSSA et SSA présentées à la section 2.2 du chapitre 1. Pour cela, ils étudient les impacts des pannes sur les éléments composant les architectures : ils recherchent alors les pannes susceptibles de conduire à une situation à risque. Dans le cadre de cette activité, les risques sont aussi appelés **événements redoutés**.

Les analystes de la sécurité peuvent s'appuyer sur différentes techniques d'analyse des architectures systèmes. Dans le monde aéronautique, quatre techniques sont proposées par les recommandations ([ARP4761 / ED-135]) :

- l'arbre de défaillances ;
- le diagramme de dépendances ;
- l'analyse markovienne ;
- l'analyse de modèles formels d'architectures dysfonctionnelles.

Les deux premières techniques ont en commun la production d'un modèle des pannes du système considéré, à partir duquel il est possible de réaliser des vérifications autant quantitatives que qualitatives. Ces techniques sont présentées dans le détails à la suite de cette section.

L'analyse markovienne ([Howard, 1971]) est une technique purement quantitative. Or, l'analyse des risques est une évaluation qualitative. Par conséquent, nous ne la présentons pas dans ce mémoire.

Enfin, la dernière technique n'a été acceptée par les autorités de certification que très récemment et fait l'objet de la section 4.2.

4.1.1 Les Arbres de Défaillances

Les Arbres de Défaillances (AdD, appelés « *Fault Tree Analysis* », FTA, en anglais) ([Chatelet, 2000], [Villemeur, 1988]) ont été créés par Watson, de Bell Telephone, au début des années 60s, puis ont été améliorés par Boeing. En 1965, Boeing et l'université de Washington ont sponsorisé la première édition de la « *System Safety Conference* », une conférence dédiée à la sûreté de fonctionnement des systèmes. Lors de cet événement, plusieurs papiers présentèrent les Arbres de Défaillances [Ericson, 1999], ce qui marqua le début de l'intérêt de nombreux industriels pour cette méthode de modélisation.

Le concept fondamental des Arbres de Défaillances est de représenter le comportement défaillant d'un système dans un modèle logique. Comme son nom l'indique, un Arbre de Défaillances est une structure arborescente dont les feuilles représentent les pannes des éléments du systèmes et dont le sommet modélise un événement redouté. Il se construit du sommet vers les feuilles à l'aide des portes logiques ET et OU, comme l'illustre la figure 2.13.

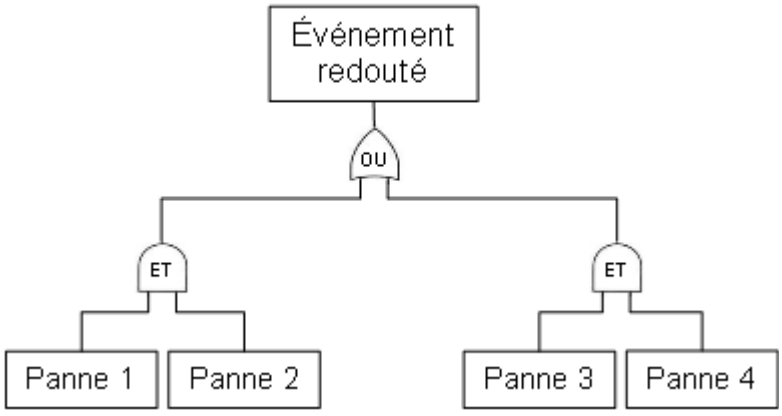


Fig. 2.13 - Arbre de Défaillances

L'objectif de la représentation par Arbres de Défaillances est de modéliser et d'expliciter la formule logique booléenne donnant l'ensemble des combinaisons de pannes sur les éléments du système, conduisant à l'événement redouté. Les ensembles de combinaisons de pannes conduisant à ces événements redoutés sont des impliquants de ces événements. Formellement, un événement redouté est représenté par une formule booléenne, notée F, dont les variables, notées $(p_1, \ldots, p_n)$ sont les n pannes des éléments du système

([Rauzy, 2001]). Une combinaison de pannes est un ensemble de variables. Alors, une telle combinaison, notée π, est un impliquant de l'événement redouté lorsque si et seulement si, lorsque les variables de π sont vraies, la formule F est également vraie. Nous notons alors : $\pi \models F$.

La notion de **coupes minimales** est définie et formalisée dans [Rauzy, 2001], à partir de la définition des impliquants : un impliquant est une coupe minimale s'il est impossible d'en extraire un sous-ensemble strict qui soit lui aussi un impliquant.

Exemple. Sur l'arbre de la figure 2.13, nous identifions deux impliquants : $\{Panne1, Panne2\}$ et $\{Panne3, Panne4\}$. Ces impliquants sont des coupes minimales. En revanche, si $\{Panne1, Panne2, Panne3\}$ est également un impliquant de l'arbre, cet ensemble de panne n'est pas une coupe minimale. Des deux coupes minimales, nous remarquons qu'aucune panne unitaire ne conduit à l'événement redouté et que les deux combinaisons de pannes sont indépendantes.

Les coupes minimales permettent de faire des calculs quantitatifs sur les risques. [Thomas, 2002] présente comment, connaissant la probabilité d'occurrence des pannes d'un système ainsi que les coupes minimales liées à un événement redouté, il est possible de calculer la probabilité de chaque impliquant puis la probabilité d'occurrence de l'événement redouté.

D'autres méthodes existent pour exploiter qualitativement et quantitativement les Arbres de Défaillances, telles que la simulation de Monte-Carlo ([Hammersley et al., 1964]) ou encore le calcul d'un Diagramme de Décision Binaire issu d'un arbre ([Bryant, 1986], [Rauzy, 1993]). L'aspect quantitatif n'étant pas détaillé dans ce mémoire, le lecteur intéressé pourra se tourner vers les documents cités pour plus de détail.

4.1.2 Les Arbres de Défaillances dynamiques

L'utilisation des Arbres de Défaillances a l'avantage d'être facilement compréhensible par différents utilisateurs et pas seulement les analystes de la sécurité. Néanmoins, la version, dite statique, de l'Arbre de Défaillances, présentée à la section précédente, souffre de plusieurs limitations :

- pas de prise en compte de l'ordre d'occurrence des pannes ;
- pas de modélisation de la configuration du système ni de son environnement ;
- pas de représentation des aspects temporels (variation temporelle de la configuration et de l'environnement).

Des Arbres de Défaillances, dits dynamiques, ont été développés pour pallier ces limitations. Ainsi, J. B. Dugan a proposé dans [Dugan et al., 1998] de décomposer l'arbre initial :

- en une partie statique, traitée comme un Arbre de Défaillances statique, par exemple à partir d'un Diagramme de Décision Binaire ;
- en une partie dynamique traitée par des graphes de Markov ([Howard, 1971]).

Une approche similaire a été déployée dans [Bouissou et al., 2003]. Le principal inconvénient de cette approche réside dans le fait que la notion de coupes minimales n'est pas définie dans les Arbres de Défaillances dynamiques, si bien que les études qualitatives des dépendances entre les pannes ne sont plus possibles. Seules les analyses quantitatives sont réalisées.

[Cepin et al., 2002] présente une approche différente de l'arbre dynamique, basée sur le principe de la composition de sous-arbres qui correspondent à différentes configurations du système considéré. Le passage de manière discrète dans le temps d'un ensemble de configurations à un autre est encodé dans une matrice. Cette dernière permet, lors du traitement de l'arbre, d'activer ou de désactiver certaines branches de l'arbre pour se placer dans un état de configurations précis du système. La figure 2.14 donne une vue d'un tel Arbre de Défaillances dynamique. L'avantage est que la notion de coupes minimales est préservée à l'intérieur de chaque configuration.

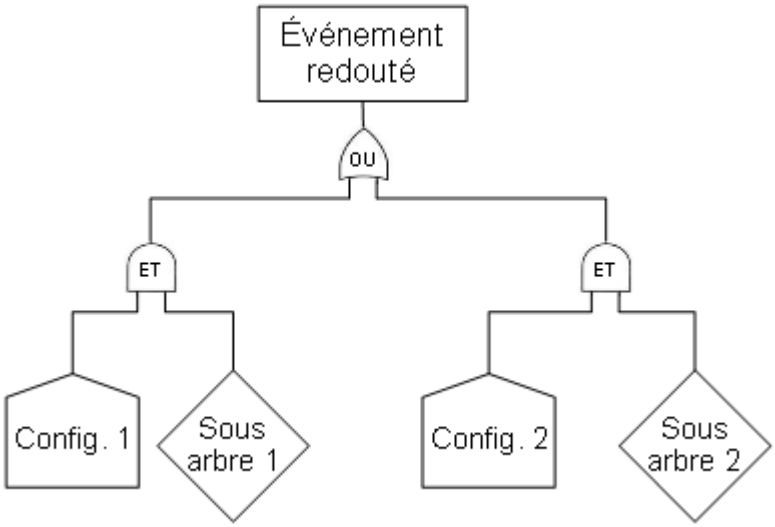


Fig. 2.14 - Arbre de Défaillances dynamique

4.2 Les modèles d'architectures dysfonctionnelles

Cette fois, le principe n'est plus de modéliser les liens entre les pannes, mais de modéliser comment les pannes impactent les équipements d'un système et comment les équipements sont liés entre eux. La propagation des pannes découle alors des liens entre les équipements. Ces modèles sont donc à un niveau d'abstraction supérieur aux Arbres de Défaillances.

Comme pour les Arbres de Défaillances ou les graphes de Markov, les analyses des risques basées sur les modèles d'architectures dysfonctionnelles sont dédiés aux activités de PSSA et de SSA pour la vérification des architectures des systèmes aéronautiques. Cette approche de modélisation est appelée **« *Model-Based Safety Assessment* » (MBSA)**. Elle est adoptée par de nombreux industriels dans divers secteurs à risques, développant des systèmes embarqués, dont, entre autre, Airbus ([MBSA Guideline, 2011]), Rockwell Collins ([Joshi et al., 2006]) ou encore EDF ([Bouissou et al., 1991]).

La figure 2.15, inspirée du guide d'Airbus ([MBSA Guideline, 2011]), détaille les principes de l'approche MBSA. La partie supérieure de cette figure (en bleu et en noir) représente le processus réalisé en dehors de l'approche MBSA. Dans ce processus, pour chaque analyse de sécurité, les spécialistes commencent par interpréter les données issues des documents de conception afin de comprendre le comportement du système sur lequel est réalisé l'analyse. Ensuite, à l'aide de cette connaissance du système, ils réalisent les analyses nécessaires et produisent des résultats permettant de vérifier les architectures systèmes : création d'arbres de défaillances, détermination des coupes minimales, calculs de probabilités d'occurrences d'événements redoutés etc. (voir la section 4.1 de ce chapitre). Cependant, l'analyse du comportement du système est encore majoritairement aujourd'hui une expertise informelle, fortement dépendante des compétences des analystes de la sécurité.

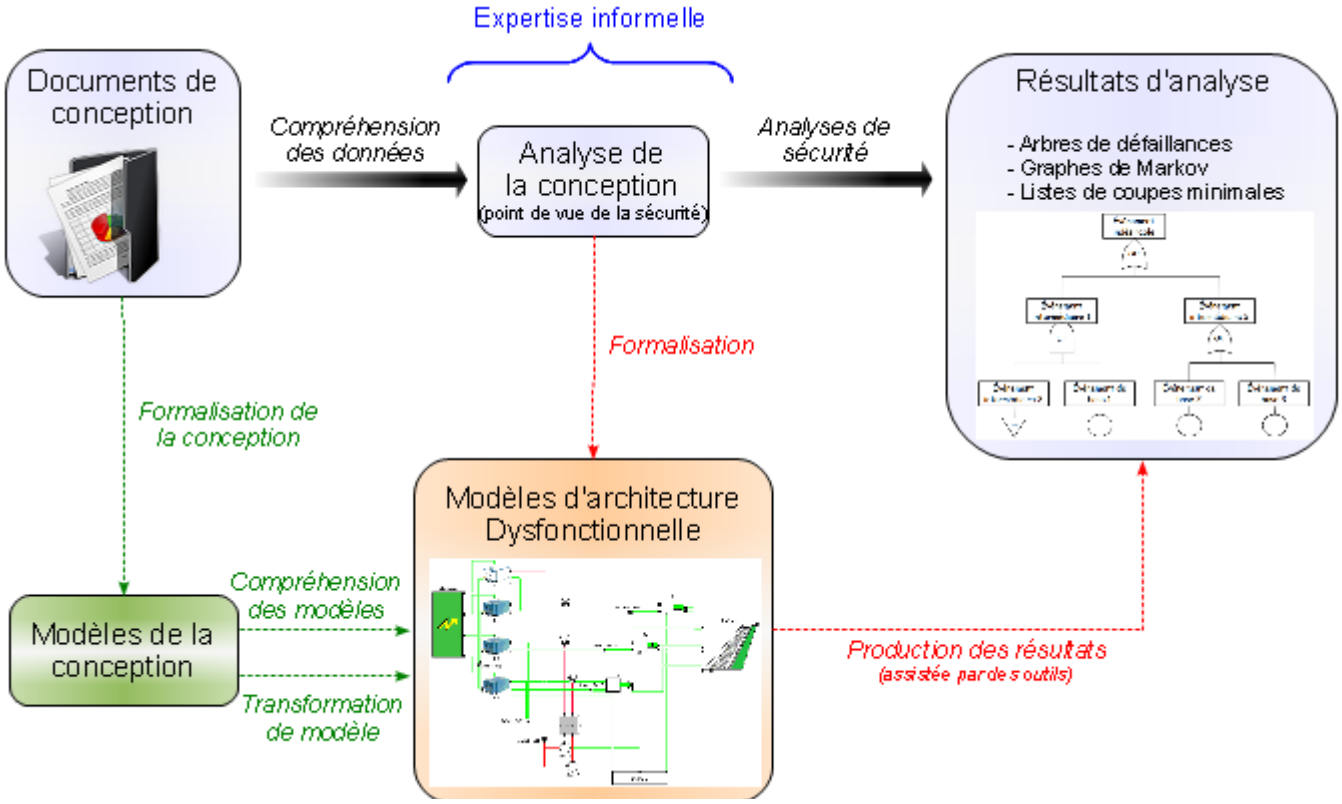


Fig. 2.15 - L'approche « *Model-Based Safety Assessment* » (MBSA)

L'approche MBSA actuellement appliquée chez Airbus, mais aussi chez d'autres industriels, est représentée en rouge sur la figure 2.15. Elle pallie l'expertise informelle des systèmes à l'aide d'une étape de formalisation ayant pour but de produire des modèles de ces systèmes. Cette étape de modélisation renforce la confiance que les analystes ont dans leur compréhension des comportements des systèmes. En outre, ces modèles intègrent les caractéristiques dysfonctionnelles des équipements de ces systèmes, permettant d'assister les analyses de sécurité à l'aide de divers outils de production de résultats, tels que des générateurs d'arbres de défaillances ou des outils de calcul de probabilité.

De plus en plus, l'approche MBSA se couple avec des approches de formalisation propres à la conception (en vert sur la figure 2.15). Les concepteurs réalisent des modèles de conception, liés plus ou moins

formellement aux modèles d'architectures dysfonctionnelles. L'interprétation des données de la conception est en effet plus aisée (plus rapide et avec plus de confiance) à partir de modèles plutôt que de documents. De plus, grâce à l'ingénierie des modèles, des passerelles peuvent être réalisées entre les modèles de la conception et les modèles de sécurité.

4.3 Le langage formel AltaRica

De nombreux langages de modélisation permettent de développer des modèles de sécurité dans le cadre de l'approche « *Model-Based Safety Assessment* », dont les principaux sont : AltaRica, Figaro, Hip-Hops, les modèles d'erreur en AADL, les modèles d'erreur en EFFBD et les modèles d'erreur en Scade ou Matlab/Simulink. Cette section est consacrée uniquement au langage AltaRica. Les autres langages sont présentés dans la section 4.4.

Le langage AltaRica ([Point et al., 1999], [Point, 2000] et [Arnold et al., 2000]) est né dans les années 90 à partir de projets communs entre des industriels, tels que Total ou Dassault Aviation, et des chercheurs du LaBRI, le « Laboratoire Bordelais de Recherche en Informatique », dans le but de disposer d'un nouveau langage de modélisation des architectures dysfonctionnelles, avec les exigences suivantes :

- un langage hiérarchique et compositionel permettant de modéliser des architectures complexes ;
- un langage formel fortement spécialisé pour assister les analyses des risques de systèmes embarqués aussi bien sur le plan quantitatif que qualitatif ;
- un langage facile d'utilisation pour les ingénieurs de la sûreté de fonctionnement.

4.3.1 Utilisation du langage AltaRica

Nos travaux s'appuient sur ce langage pour les raisons suivantes :

- AltaRica permet de modéliser des architectures aussi bien dans leur vue fonctionnelle que dans leur vue dysfonctionnelle. La représentation dysfonctionnelle du comportement d'un système est simplifiée grâce à l'explicitation d'événements, modélisant les pannes potentielles des éléments du système.
- AltaRica permet de réaliser des modèles compositionnels et hiérarchiques. Cette capacité est une grande aide pour maîtriser la complexité des systèmes (découpage d'un système en plusieurs sous-systèmes communicants, différents niveaux de détails etc.)
- AltaRica est puissamment outillé par des outils graphiques d'édition et de visualisation, favorisant la lecture des modèles, et des outils de calcul, favorisant l'exploitation des modèles (simulation, génération automatique d'Arbres de Défaillances, etc.)

Le langage AltaRica a déjà donné lieu à de nombreuses études pour différentes problématiques, en particulier :

- la vérification des architectures à l'aide de l'approche MBSA. Permettre de vérifier les architectures

systèmes du point de vue de la sûreté de fonctionnement est le but originel du langage AltaRica. Chez Airbus, pour les avions récents, l'analyse de plusieurs systèmes aéronautiques a été accompagnée de modèles AltaRica de ces systèmes. Plusieurs projets européens ont accompagné ces démarches de vérification ces dernières années, dont :

- « *Enhanced Safety Assessment for Complex Systems* » (ESACS) ([Bozzano et al., 2003])
- « *Improvement of Safety Activities on Aeronautical Complex systems* » (ISAAC) ([Bozzano et al., 2006])
- « *More Integrated and cost efficient System Safety Assessment* » (MISSA) ([Bozzano et al., 2011])

Plusieurs travaux de thèse se sont également concentrés sur cette problématique, dont :

- [Kehren, 2005] a défini des motifs d'architecture en AltaRica pour aider à la conception de systèmes sûrs.
- [Humbert, 2008] a proposé une utilisation complémentaire d'AltaRica et de Scade pour aider à la définition d'exigences de sécurité d'un système embarqué.
- [Sagaspe, 2008] a développé un outil d'aide à l'allocation des équipements sur les architectures fonctionnelles et spatiales dans le cadre des systèmes aéronautiques.
- [Bernard, 2009] a proposé une méthodologie de modélisation AltaRica et d'analyse de la sécurité de systèmes multiples et en interaction, à l'aide de la notion de raffinement.
- [Adeline, 2011] a travaillé sur des modèles de sécurité pour de grands systèmes complexes et proposé une méthode pour la validation de ces modèles (voir le paragraphe ci-dessous).
- [Chaudemar 2012] a présenté des méthodes et des outils pour étudier l'organisation des mécanismes de sécurité dans les architectures.

– la validation des modèles AltaRica. Dès les premières utilisations du langage, une question s'est posée : quelle confiance pouvons nous avoir envers les modèles AltaRica développés ? Un des principaux avantages de ce langage est sa polyvalence. En effet, les modèles AltaRica peuvent se traduire en divers autres formalismes pour lesquels des outils de vérification de propriétés existent. Par exemple, l'Onera a développé une passerelle vers le langage d'entrée du vérificateur de modèle SMV (voir [Bozzano et al., 2007] pour plus de détails). En outre, en 2003, un vérificateur de modèles AltaRica, nommé MecV, a été développé au LaBRI ([Vincent, 2003]).

Il existe plusieurs variations du langage AltaRica. Nous nous concentrons par la suite sur le formalisme AltaRica Dataflow qui est une version allégée d'AltaRica dans laquelle les flux sont unidirectionnels, et les ensembles sont finis (voir la sémantique dans la section suivante).

4.3.2 La sémantique du langage AltaRica Dataflow

La sémantique du langage AltaRica Dataflow découle de la théorie classique des systèmes états transitions ([Arnold, 1994], [Hopcroft et al., 2006]).

Définition d'un automate de mode

Un modèle AltaRica est défini comme un automate de mode ([Rauzy, 2002]).

Définition : La sémantique du langage AltaRica (automate de mode)

Un modèle AltaRica est un automate de mode, défini par le 7-uple :

$A=(D, V, dom, \Sigma, \delta, \sigma, I)$, *où*

– D *est un domaine fini de valeurs,*

– V *est l'ensemble fini des variables. Nous distinguons trois catégories de variables. Les trois sous-ensembles suivants sont deux à deux disjoints :*

- S *est l'ensemble fini des variables d'état,*
- F^{in} *est l'ensemble fini des variables de flux entrants,*
- F^{out} *est l'ensemble fini des variables de flux sortants,*

– $dom : V \rightarrow 2^D$, *telle que* $\forall v \in V, dom(v) \neq \emptyset$ *associe à une variable son domaine de valeur.*

– $dom : 2^V \rightarrow 2^{2^D}$, *est surchargée de telle façon que :*

$$\forall U \subseteq V, dom(U) = \prod_{u \in U} dom(u)$$

– Σ *est l'ensemble fini des événements,*

– δ *est l'ensemble des transitions, définies par la fonction partielle :*

$$dom(S) \times dom(F^{\text{in}}) \times \Sigma \rightarrow dom(S)$$

Chaque événement de Σ *est lié à, au moins, une transition.* $dom(S) \times dom(F^{\text{in}})$ *forme les gardes des transitions.*

– σ *est l'ensemble des assertions, définies par la fonction totale :*

$$dom(S) \times dom(F^{\text{in}}) \rightarrow dom(F^{\text{out}})$$

– $I \subset \sigma$ *est l'ensemble des fonctions partielles décrivant l'état initial (i.e. une évaluation initiale de* $dom(S)$*).*

Cette sémantique correspond bien à un automate à états-transitions. Les états sont déterminés à partir des assertions σ. Les transitions sont définies par δ. Nous pouvons constater qu'il existe un nombre fini d'états et de transitions pour chaque modèle AltaRica, du fait que les ensembles D et V sont finis (voir [Point et al., 1999] pour plus de détails).

Exemple. Nous définissons le « bloc booléen » comme un automate de mode composé d'un état, d'une entrée et d'une sortie booléennes (voir la figure 2.16). Ce bloc est soumis à un événement représentant une panne du bloc.

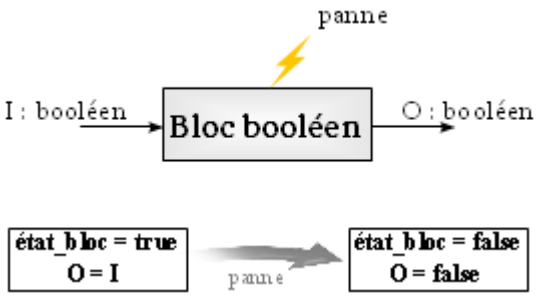


Fig. 2.16 - Bloc booléen

La sémantique de ce bloc booléen est :

- $D_{bloc} = Booléen$
- $S_{bloc} = \{état\_bloc\}$, $F^{in}_{bloc} = \{I\}$ et $F^{out}_{bloc} = \{O\}$
- $dom_{bloc} : S_{bloc} \cup F^{in}_{bloc} \cup F^{out}_{bloc} \rightarrow Booléen$
- $\Sigma_{bloc} = \{panne\}$
- $\delta_{bloc} : Booléen \times Booléen \times \{panne\} \rightarrow Booléen$, tel que :
  - $\delta_{bloc}(?, ?, panne) = false$ avec « ? », booléen indéterminé.
- $\sigma_{bloc} : Booléen \times Booléen \rightarrow Booléen$, tel que :
  - $\sigma_{bloc}(true, true) = true$
  - $\sigma_{bloc}(true, false) = false$
  - $\sigma_{bloc}(false, ?) = false$ avec « ? », booléen indéterminé.
- $I_{bloc}(état\_bloc) = true$

Composition des nœuds AltaRica

Le principe de la composition est de considérer chaque modèle AltaRica comme pouvant être le sous-modèle, appelé **nœud**, d'un modèle englobant. La règle de composition suivante, tirée de [Rauzy, 2002] et [Kehren, 2005], montre que si chaque nœud respecte la sémantique des modèles AltaRica, à savoir d'être un automate de mode, alors le modèle issu de la composition est également un automate de mode et donc un modèle AltaRica sémantiquement acceptable. La règle de composition est présentée sur deux nœuds (voir la figure 2.17) ; par récurrence, elle peut être étendue à un modèle composé de plus de deux nœuds.

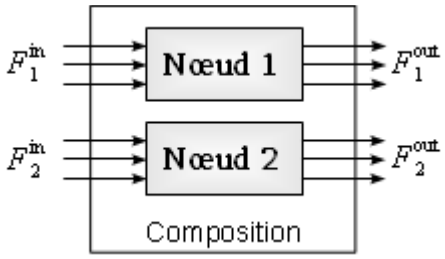


Fig. 2.17 - Composition de deux nœuds

Composition parallèle. Soit 2 automates de mode, notés $A_i=(D_i, V_i, dom_i, \Sigma_i, \delta_i, \sigma_i, I_i)$, $\forall i\in[1,2]$, disjoints, c'est-à-dire que $F_1^{in}\cap F_2^{in}=\emptyset$ et $F_1^{out}\cap F_2^{out}=\emptyset$. Alors la composition de ces nœuds est un automate de mode noté $A=(D, V, dom, \Sigma, \delta, \sigma, I)$ où :

- $V=S\cup F^{in}\cup F^{out}$ avec $S=S_1\cup S_2$, $F^{in}=F_1^{in}\cup F_2^{in}$ et $F^{out}=F_1^{out}\cup F_2^{out}$ (union des variables) ;
- $dom: V \rightarrow 2^D$, tel que : $\forall v\in V, dom(v)=\{dom_1(v) si\ v\in V_1\}$
- $\Sigma=\Sigma_1\cup\Sigma_2$ (union des événements) ;
- δ est obtenu en remontant les δ_1 et δ_2 dans A :

$\forall e\in\Sigma_1,\ \delta(dom(S), dom(F^{in}), e)=\langle\delta_1(dom_1(S_1), dom_1(F_1^{in}), e), dom_2(S_2)\rangle$ et

$\forall e\in\Sigma_2,\ \delta(dom(S), dom(F^{in}), e)=\langle dom_1(S_1), \delta_2(dom_2(S_2), dom_2(F_2^{in}), e)\rangle$

- σ est obtenu en remontant les σ_1 et σ_2 dans A :

$\sigma(dom(S), dom(F^{in}))=\langle\sigma_1(dom_1(S_1), dom_1(F_1^{in})), \sigma_2(dom_2(S_2), dom_2(F_2^{in}))\rangle$

- $I=\langle I_1, I_2\rangle$.

Plusieurs nœuds peuvent alors se retrouver et interagir dans la même vue. Il est possible de réaliser des connexions entre les nœuds. Cela consiste à imposer des contraintes à certaines variables d'entrée d'un nœuds pour que leurs valeurs soient égales à celle d'un même nombre de variables de sortie d'autres nœuds. Ces contraintes sur les valeurs des variables ne remettent pas en cause la sémantique des automates de mode, à condition que les variables d'entrée et de sortie liées soient indépendantes au niveau des assertions. Cette condition signifie que les variables d'entrée ne doivent pas intervenir dans le calcul des valeurs des variables de sorties auxquelles elles sont connectées. La figure 2.18, ci-dessous, représente ce cas de figure avec deux nœuds où une des sorties du premier nœud (« out ») est une entrée du second nœud (« in »).

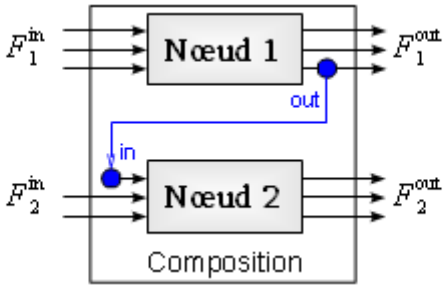


Fig. 2.18 - Composition de deux nœuds avec lien

La règle de composition, pouvant être étendue par récurrence, devient alors :

Composition en séquence. Soit 2 automates de mode, notés $A_i=(D_i, V_i, dom_i, \Sigma_i, \delta_i, \sigma_i, I_i)$, $\forall i\in[1,2]$, disjoints, c'est-à-dire que $F_1^{in}\cap F_2^{in}=\emptyset$ et $F_1^{out}\cap F_2^{out}=\emptyset$, mais avec le lien entre la sortie $out\in F_1^{out}$ et

l'entrée $\text{in} \in F_2^{\text{in}}$. Alors la composition de ces deux nœuds est un automate de mode noté $A=(D, V, dom, \Sigma, \delta, \sigma, I)$ où :

- $V=S \cup F^{\text{in}} \cup F^{\text{out}}$ avec $S=S_1 \cup S_2$, $F^{\text{in}}=F_1^{\text{in}} \cup F_2^{\text{in}} - \{\text{in}\}$ et $F^{\text{out}}=F_1^{\text{out}} \cup F_2^{\text{out}}$ (union des variables).
- $dom : V \rightarrow 2^D$, tel que : $\forall v \in V, dom(v)=\begin{cases} dom_1(v) \ si \ v \in V_1 \\ dom_1(\text{out}) \ si \ v=\{\text{in}\} \\ dom_2(v) \ si \ v \in V_2 - \{\text{in}\} \end{cases}$
- $\Sigma=\Sigma_1 \cup \Sigma_2$ (union des événements) ;
- Soit $valF_2 : F_2^{\text{in}} \rightarrow 2^D$, définie par : $\forall v \in F_2^{\text{in}}, \ valF_2(v)=\begin{cases} dom_2(v) \ si \ v \in F_2^{\text{in}} - \{\text{in}\} \\ dom_1(\text{out}) \ si \ v=\{\text{in}\} \end{cases}$
- δ est obtenu en remontant les δ_1 et δ_2 dans A :
  $\forall e \in \Sigma_1, \ \delta(dom(S), dom(F^{\text{in}}), e)=\langle \delta_1(dom_1(S_1), dom_1(F_1^{\text{in}}), e), dom_2(S_2) \rangle$ et
  $\forall e \in \Sigma_2, \ \delta(dom(S), dom(F^{\text{in}}), e)=\langle dom_1(S_1), \delta_2(dom_2(S_2), valF_2, e) \rangle$
- σ est obtenu en remontant les σ_1 et σ_2 dans A :
  $\sigma(dom(S), dom(f^{\text{in}}))=\langle \sigma_1(dom_1(S_1), dom_1(f_1^{\text{in}})), \sigma_2(dom_2(S_2), valF_2) \rangle$
- $I=\langle I_1, I_2 \rangle$.

Exemple. La figure 2.19 illustre un exemple de composition et de connexion avec un modèle AltaRica contenant deux blocs booléens, tels que décrits dans l'exemple précédent.

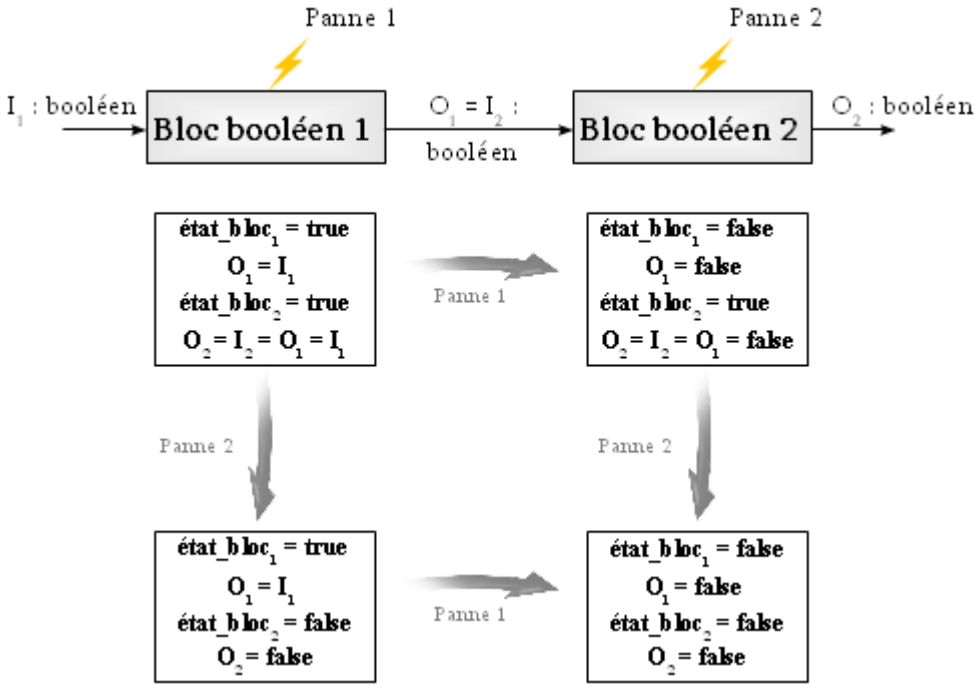


Fig. 2.19 - Un modèle composé de deux blocs booléens liés

Dans un modèle AltaRica composé de plusieurs nœuds, il est possible de contraindre plusieurs événements à se produire en même temps. Concrètement, cette opération de synchronisation permet de tirer plusieurs transitions simultanément alors que ces dernières sont définies asynchrones dans les automates de mode. Une synchronisation est définie par son vecteur des événements synchronisés et par sa stratégie pour tirer les transitions. La démonstration qu'un automate de mode composé de synchronisations d'événements respecte toujours la sémantique des automates de mode peut être trouvée dans [Kehren, 2005] ou [Sagaspe, 2008]. Trois types de synchronisations sont présents dans les modèles AltaRica :

1. Synchronisation forte : les événements composant une synchronisation forte ne peuvent pas avoir lieu indépendamment. De plus, toutes les gardes de leurs transitions doivent être satisfaites pour que ces transitions puissent être tirées.
2. Diffusion : les événements composant une diffusion ne peuvent pas avoir lieu indépendamment. En revanche, il n'est pas nécessaire que toutes les gardes de leurs transitions soient satisfaites pour que des transitions puissent être tirées. Seules les transitions dont les gardes sont satisfaites sont tirées simultanément.
3. Défaillance de Cause Commune (DCC)[2] : les événements composant une DCC peuvent avoir lieu indépendamment de la synchronisation. Et, comme pour la diffusion, lorsque la synchronisation a lieu, le sous-ensemble des transitions dont les gardes sont satisfaites sont tirées simultanément.

4.3.3 La syntaxe du langage AltaRica Dataflow

La déclaration d'un nœud AltaRica commence avec le mot clé *node*, suivi du nom du nœud, et se termine avec le mot clé *edon*. Un nœud est composé de :

- Variables d'états : la déclaration des variables d'état commence avec le mot clé *state*. Chaque variable est ensuite définie par son nom et l'ensemble des valeurs dans lequel elle prend ses valeurs. Les variables d'état peuvent être initialisées à l'aide du mot clé *init*.
- Variables de flux : la déclaration des variables de flux commence avec le mot clé *flow*. Chaque définition de variables de flux se base sur le nom de la variable, son domaine de valeurs et sa direction (entrant : *in*, sortant : *out* et local : *local*). Ces variables de flux sont également nommées les **interfaces** des nœuds.
- Événements : le nom de chaque événement du nœud est déclaré sous la forme d'une liste après le mot clé *event*.
- Transitions : les transitions sont déclarées après le mot clé *trans*. Chaque événement intervient au moins dans une transition. La syntaxe d'une transition est : *garde* |- *événement* → *affectation ;* . La garde est une assertion booléenne basée sur les valeurs des variables d'états et de flux entrants. L'affectation est une assertion qui modifie les valeurs des variables d'états du nœud.

2 Le type de synchronisation « Défaillance des Causes Commune » a pour origine une propriété de la sûreté de fonctionnement : certains équipement, définis comme différents, peuvent en réalité avoir un point commun (même concepteur, même emplacement dans l'avion, même compilateur logiciel etc.) à partir duquel une unique panne peut avoir un impact simultané sur un sous-ensemble de ces équipements.

- Assertions : les assertions sont déclarées après le mot clé *assert*. Elles modifient les valeurs des variables de flux sortant en fonction des valeurs des variables d'états, des variables de flux entrant et des variables de flux locaux.
- Sous-nœuds : lorsque le nœud considéré est composé d'autres nœuds, ces derniers sont déclarés après le mot clé *sub*. Le champs des assertions permet d'établir les contraintes de liaison entre les sous-nœuds.
- Synchronisation d'événements : les synchronisations sont déclarées après le mot clé *sync*. La syntaxe des synchronisation varie en fonction de leur type (voir la section 4.3.2), mais se base sur la déclaration du vecteur des événements synchronisés.

Pour chaque nœud une partie de ces caractéristiques peuvent être vide, auquel cas il n'est pas nécessaire de les déclarer. [Point et al., 1999] et [Point, 2000] présentent plus de détails sur la syntaxe du langage AltaRica.

Exemple. La table 2.10 donne le code AltaRica correspondant au bloc booléen de l'exemple précédent.

```
1   node Bloc_booleen
2         state
3         etat_bloc : bool ;
4         flow
5         I : bool : in ;
6         O : bool : out ;
7         event
8         panne ;
9         trans
10        etat_bloc = true |- panne -> etat_bloc:=
11  false ;
12        assert
13        O = case{(etat_bloc = true and I = true) :
14  true, else false} ;
15  edon
```

Tab. 2.10 - Code AltaRica du bloc booléen

4.3.4 Les outils d'édition graphique des modèles AltaRica

Les outils d'édition graphique facilitent le travail de modélisation des analystes de la sécurité. Comme ces derniers sont rarement des experts en programmation, la majorité des outils de modélisation proposent des surcouches de développement permettant de masquer la syntaxe du langage AltaRica. Les principaux éditeurs graphiques sont les plate-formes OCAS de Dassault Système et RAMSES d'Airbus, dans lesquelles les nœuds AltaRica sont enrichis à l'aide d'icônes pouvant changer en fonction de leurs entrées et de leurs états

internes. Les liens entre les nœuds sont représentés par des arcs orientés pour montrer la direction des flux et colorée en fonction des valeurs transportées par chaque flux.

Ces deux outils intègrent également des gestionnaires de composants sous la forme de librairies, permettant d'instancier, dans différents modèles, plusieurs nœuds à partir d'un seul type de nœud.

Bien que le code AltaRica soit au cœur des modèles réalisés dans les outils OCAS et RAMSES, ces derniers ajoutent de nombreuses informations (en particulier des informations graphiques) et utilisent leur propre langage de données. Le code OCAS ou RAMSES s'appuie sur le standard de documentation XML et est décrit par une DTD[3]. La table 2.11 présente quelques caractéristiques du langage d'OCAS et de RAMSES en langage Express pour une compréhension plus aisée du code.

```
1   -- Les entités « IO », « STATE », « EVENT »
2   et « ICON » ne sont pas présentées dans cet
3   extrait
4
5   ENTITY COMPONENT_FAMILY;
6         modèles : SET OF COMPONENT_MODEL;
7   END_ENTITY;
8
9   ENTITY COMPONENT_MODEL;
10        interfaces : SET OF IO;
11        états : SET OF STATE;
12        événements : SET OF EVENT;
13        icônes : SET OF ICON;
14  END_ENTITY;
15
16
17  ENTITY COMPONENT;
18        nom : STRING;
19        type : COMPONENT_MODEL;
20  END_ENTITY;
21
22  ENTITY LINK;
23        init : IO;
24        final : IO;
25  END_ENTITY;
26
27  ENTITY ARCHITECTURE;
28        composants : SET OF COMPONENT;
29        liens : SET OF LINK;
30  END_ENTITY;
```

Tab. 2.11 - Extrait du métamodèle du langage des outils OCAS et RAMSES avec Express

3 Une DTD (« *Document Type Definition* ») décrit la structure autorisée d'un fichier XML.

Cet extrait présente la principale caractéristique du langage des outils OCAS et RAMSES : la bibliothèque de composants. Un composant est le terme usuel dans les outils OCAS et RAMSES pour désigner un nœud d'un modèle AltaRica. Il est nommé et décrit par ses variables d'entrée, de sortie et d'état, ses événements, ses transitions et ses assertions (conformément à la sémantique du langage AltaRica, voir la section 4.3.2 du chapitre 2) ainsi que par des caractéristiques graphiques, telles que ses icônes. Cependant, dans les outils OCAS et RAMSES, un composant n'est pas directement définis par ces attributs ; il hérite d'un « modèle de composants ». L'avantage est alors de pouvoir réutiliser un unique comportement pour définir plusieurs nœuds, par instanciation. Le modèle de composants est sauvegardé dans une famille de composants ; il s'agit d'un moyen basique d'archivage.

Les liens n'étant pas des nœuds, ils n'ont pas de « modèle » correspondants. Les liens sont donc décrits directement avec une interface initiale et une interface finale (i.e. des variables de flux entrants ou sortants de nœuds).

Enfin, dans les outils OCAS et RAMSES, un modèle AltaRica est une architecture composée entre autre de composants et de liens.

4.3.5 Outils d'analyses des modèles AltaRica

Les outils d'analyse offrent des moyens de calcul de l'état d'un modèle AltaRica par rapport aux nœuds qui le composent. Les paragraphes suivants présentent différents outils d'analyses. Plus de détails peuvent être trouvés dans [Boulanger, 2012].

La simulation

Un simulateur interactif des modèles AltaRica est proposé dans les plate-formes graphiques OCAS et RAMSES. Cet outil commence par calculer la configuration initiale du modèle simulé. Puis, le simulateur propose de déclencher des événements du modèle. Plus précisément les simulateurs d'OCAS et de RAMSES ne proposent que le sous-ensemble des événements pour lesquels il existe au moins une transition dont la garde est vraie. Lorsque l'utilisateur sélectionne un événement, le simulateur tire les transitions pouvant l'être et recalcule le nouvel état courant.

Le simulateur offre l'avantage de faciliter la compréhension du comportement dysfonctionnel d'un modèle, puisque la majorité des événements d'un modèle AltaRica représentent des pannes. En particulier, il devient plus aisé de valider un modèle avec l'équipe de conception d'un système. Il s'agit, pour chaque modèle de valider le comportement de chacun des nœuds indépendamment, puis de valider le comportement de leurs interactions. En particulier, les analystes de la sécurité doivent vérifier que la propagation des pannes dans le modèles est cohérente avec le comportement du système réel. Cela peut être fait entre autre en jouant des scénarios de pannes particuliers avec le simulateur.

Le générateur de séquences SeqGen

Dans la section 4.1, nous avons présenté la notion de coupe minimale et son intérêt du point de vue de la vérification des architectures systèmes. SeqGen est un outil proposé dans OCAS et RAMSES dont l'objectif est de générer automatiquement les coupes minimales d'un modèle d'architecture par rapport à un événement

redouté préalablement défini. Cet événement redouté peut être n'importe quelle valeur correspondant à une sortie d'un nœud du modèle.

SeqGen se base sur le simulateur de modèles AltaRica. Il joue automatiquement tous les scénarios de pannes possibles (jusqu'à un nombre de combinaisons défini par l'utilisateur) et vérifie si l'état obtenu du modèle conduit à l'événement redouté ou non. Il sauvegarde finalement les coupes minimales dans un fichier de données compatible avec d'autres outils tels que les générateurs d'arbres de défaillances (voir les paragraphes suivants). SeqGen a l'avantage de sauvegarder l'ordre des combinaisons des pannes. Parfois certaines pannes n'ont pas les mêmes effets quand d'autres pannes les ont précédées et ont modifié l'état du système. En revanche, la limite du nombre de combinaisons, qui est nécessaire pour contrôler le temps d'exécution de l'outil, induit de fait que les coupes minimales d'ordre supérieur ne sont pas identifiées, ce qui peut pénaliser des calculs de fiabilité dans la mesure où les coupes d'ordre supérieur peuvent être très nombreuses.

Le générateur d'Arbres de Défaillances

Le générateur d'Arbres de Défaillances est un outil indépendant d'OCAS ou de RAMSES, mais qui existe en version intégrée à OCAS. Il permet, pour certains modèles AltaRica, de produire automatiquement un Arbre de Défaillances à partir du modèle et d'un événement redouté sélectionné ([Rauzy, 2002]). Cet arbre représente une formule booléenne dont les variables sont les événements du modèle AltaRica. La formule décrit les chemins qui mènent le système de l'état initial à un état qui représente l'événement redouté. Les coupes minimales peuvent ensuite être calculées à l'aide d'autres outils tel qu'Aralia.

La limitation du générateur d'Arbres de Défaillances réside dans les modèles qu'il peut traiter : des modèles, pour lesquels toutes les permutations de tous les sous-ensemble d'événements conduisent au même état. Les modèles ne satisfaisant pas cette exigence peuvent en revanche être traités par le générateur de séquences présenté précédemment. Le générateur d'Arbres de Défaillances présente par contre un avantage de taille : il ne nécessite pas de borner la taille des combinaisons de pannes. Cet outil est par conséquent susceptible de fournir des résultats quantitatifs plus précis que la génération de séquences.

4.4 Comparaison du langage AltaRica avec d'autres langages de modélisation

Dans les paragraphes suivants, nous dressons un panorama des autres langages de modélisation existants, permettant d'assister les analyses de sécurité dans le cadre de l'approche MBSA. Ces langages sont comparés à AltaRica à l'aide de la table 2.12, qui présente une vue globale des avantages et des inconvénients de chacun de ces langages de modélisation dans le cadre de l'approche MBSA. Cette étude s'est concentrée sur cinq critères.

1. Modularité et hiérarchie (Oui/Non) : le langage de modélisation permet-il de représenter des architectures hiérarchisées et divisées en module ?
2. Exclusif pour modéliser la sécurité (Oui/Non) : le langage de modélisation ne permet-il que de concevoir des modèles dédiés uniquement aux analyses de sécurité ?

3. Expressivité de la sécurité (Bon/Moyen/Mauvais) : le langage de modélisation est-il assez riche pour permettre la modélisation de toutes les informations relatives à la sécurité ?
4. Couplage avec les modèles de la conception (Bon/Moyen/Mauvais) : est-ce que des passerelles existent (ou peuvent aisément être développées) entre les modèles de sécurité et les modèles de la conception ?
5. Outils d'analyse de la sécurité (Bon/Moyen/Mauvais) : est ce que le langage de modélisation est supporté par des outils permettant l'analyse des modèles du point de vue de la sécurité ?

	Modularité et hiérarchie	**Exclusif pour modéliser la sécurité**	**Expressivité de la sécurité**	**Couplage avec les modèles de la conception**	**Outils d'analyse de la sécurité**
AdD	Non	Oui	Moyen	Mauvais	Bon
AltaRica	Oui	Non	Bon	Bon	Bon
Figaro	Oui	Oui	Bon	Moyen	Bon
HiP-HOPS	Oui	Oui	Bon	Moyen	Bon
AADL	Oui	Non	Moyen	Bon	Moyen
EFFBD	Oui	Non	Mauvais	Bon	Mauvais
Scade Simulink	Oui	Non	Moyen	Bon	Moyen

Tab. 2.12 - Évaluation des langages de modélisation de la sécurité des systèmes

Chez Airbus, le choix du langage de modélisation pour l'approche MBSA s'est porté sur le langage AltaRica, du fait de sa forte expressivité, tant pour représenter les informations dédiées aux analyses de sécurité que pour représenter les architectures systèmes elles-mêmes, dans leur fonctionnement nominal.

4.4.1 Le langage Figaro

Le langage Figaro a été créé par EDF en 1990 [Bouissou et al., 1991]. Il s'agit d'un langage orienté objet : une base de connaissance définit des types et des comportements génériques, à partir desquels les équipements d'un système sont créés par héritage. Ces équipements ne peuvent pas être organisés hiérarchiquement, ni même être encapsulés. Figaro permet de décrire des modèles probabilistes pouvant être simplifiés en modèles qualitatifs. Il n'est en revanche pas possible de développer un modèle purement qualitatif pour éventuellement le compléter par la suite avec des attributs quantitatifs.

Les langages Figaro et AltaRica ont été comparés dans [Bouissou et al., 2006]. Il apparaît que la syntaxe de Figaro est très expressive, proche du langage naturel. Néanmoins, le manque d'outils d'aide à l'édition des

modèles en fait un langage difficile à appréhender par les analystes de la sécurité « non-informaticiens ». La difficulté d'apprentissage de Figaro est compensée par sa puissance de modélisation : modèles génériques et capitalisation forte des données d'expertises fiabilistes. La conclusion de l'article [Bouissou et al., 2006] s'attache à montrer que Figaro et AltaRica diffèrent au niveau de leur philosophie. En effet, Figaro s'adresse à des spécialistes du langage, dont la tâche serait de concevoir des outils graphiques à destination des analystes de la sécurité, qui eux n'auront pas une ligne de code à écrire. Les liens entre Figaro et les modèles de la conception sont plus difficiles à formaliser. A l'inverse, AltaRica est un langage plus accessible, qui s'adresse donc plus directement aux analystes de la sécurité, et plus proche des modèles de la conception.

4.4.2 HiP-HOPS

HiP-HOPS (« *Hierarchically Performed Hazard Origin and Propagation Studies* ») est un langage de modélisation développé par Y. Papadopoulos dans les années 2000 ([Papadopoulos et al., 1999]). Son principal objectif est d'assurer la cohérence entre les activités d'analyse des risques, de vérification d'architecture par les Arbres de Défaillances et les AMDEC (« Analyse des Modes de Défaillance, de leurs Effets et de leur Criticité », voir [ARP4754A / ED-79A] pour plus de détails), en centralisant les données autour d'une modélisation hiérarchique d'un système et de ses dysfonctionnements, suite à des pannes. A chaque niveau de la modélisation hiérarchique du système, des diagrammes de flux de données sont utilisés pour décrire les activités du système et de ses sous-systèmes jusqu'au niveau des équipements.

Les informations des pannes sont décrites à travers une annotation du modèle du système (pouvant être modélisée avec Scade ou Matlab/Simulink entre autre). Il s'agit alors d'expressions logiques qui décrivent les déviations des sorties de chaque composant du système par rapport aux combinaisons des entrées (éventuellement déviées) et des pannes internes au composant. Dans la sémantique de HiP-HOPS, les pannes sont considérées comme des entrées non-nominales des composants. La description d'un comportant défaillant peut être réutilisée pour annoter plusieurs composants, mais en pratique, les composants étant souvent différents, leurs comportements défaillants le sont aussi. Le langage HiP-HOPS est décrit avec davantage de détail dans [Papadopoulos et al., 2008]. Nous constatons qu'il est moins expressif que le langage AltaRica. En effet, dans HiP-HOPS, les pannes sont des variables des assertions permettant de décrire l'état des sorties de chaque composant par rapport à l'état de leurs entrées. Le modèle obtenu est alors dit « statique » par rapport à un scénario de panne donné. Dans AltaRica, les pannes sont modélisées par des événements susceptibles de modifier l'état d'un modèle (voir le détail de la sémantique d'AltaRica dans la section 4.3.2). La capacité à activer des événements confère à AltaRica la possibilité de décrire des modèles dits « dynamiques ».

Cependant, HiP-HOPS a l'avantage d'intégrer un processus d'analyse de la sécurité. Il est en effet implémenté dans l'outil SAM (« *Safety Argument Manager* »), qui supporte une méthode consistant à détailler pas à pas les niveaux hiérarchiques inférieurs du système. Il propose aussi des outils permettant de générer des Arbres de Défaillances pour la vérification du système et des AMDEC pour chaque composant. Bien qu'il soit moins expressif que le langage AltaRica, HIP-HOPS fait aujourd'hui partie des meilleurs moyens de modélisation de la sécurité des systèmes.

4.4.3 Les modèles d'erreurs en AADL

Le langage AADL (« *Architecture Analysis and Design Language* » (voir [Feiler et al., 2006] pour plus de détails) est un langage de conception d'architecture standardisé par la SAE (« *Society of Automotive Engineers* »). Il est destiné à la conception et à l'analyse de systèmes embarqués temps réel, à partir de la définition de composants. Ces derniers sont classés en dix catégories :

- les composants de description de la plateforme d'exécution (catégories *Processor*, *Memory*, *Device* et *Bus*), permettant de décrire les équipements physiques ;
- les composants de description du logiciel applicatif (catégories *Process*, *Thread*, *Thread Group*, *Subprogram* et *Data category*), permettant de décrire les équipements logiciels ;
- le composant de composition, *System*, qui représente le système global.

Chaque composant est défini par un type et une implémentation. Le type décrit les interfaces du composant avec l'ensemble du modèle (données d'entrée et de sortie, mode de fonctionnement interne etc.). L'implémentation représente la structure interne du composant par description des interactions des sous-composants qui le composent.

Il existe de nombreuses passerelles entre AADL et d'autres formalismes. De plus, ce langage est enrichi de plusieurs annexes. L'une d'elles, « *AADL Error Model Annex* » ([SAE-AS5506/1]), définit la notion de modèle d'erreurs pour compléter les modèles d'architectures et exploiter ces derniers sous les différents points de vue de la sûreté de fonctionnement. Cette annexe représente un sous-langage dont le but est de spécifier en AADL les caractéristiques de la sûreté de fonctionnement. A l'instar des modèles d'architecture en AADL, les modèles d'erreurs sont construits sur la base de composants spécifiques ayant un type et une implémentation. Le type permet d'exprimer les pannes, et d'autres événements perturbateurs, et les états défaillants d'un élément du système. Avec l'implémentation, les modeleurs définissent les transitions, de l'état de fonctionnement nominal aux états défaillants par rapport aux pannes, et les probabilités d'occurrence des pannes. Les composants du modèle d'erreur sont liés par composition aux composants du système.

Le principal avantage de ce modèle d'erreur est qu'il permet ainsi l'annotation des caractéristiques défaillantes du système directement sur le modèle AADL original. Ainsi, les analystes de la sûreté de fonctionnement sont assurés que leurs analyses prennent en compte les interactions nominales entre les composants du système. Le duo composé d'un modèle d'architecture système en AADL et de son modèle d'erreurs est décrit dans la norme AADL et est parfaitement formalisé. Ainsi, ces modèles peuvent être exploités à l'aide d'outils de model-checking (voir [Feiler et al., 2007] pour plus de détails).

Dans sa thèse, A.-E. Rugina a proposé une passerelle des modèles AADL enrichis vers des réseaux de Petri afin de réaliser des calculs de fiabilité ([Rugina, 2007]). Dans [Joshi et al. 2007], une autre passerelle a été déployée vers des Arbres de Défaillances (voir la section 4.1.1).

Les modèles d'erreur en AADL ont l'avantage d'être directement issus des modèles de la conception. Cependant, leur expressivité et leur exploitation du point de vue de la sûreté de fonctionnement n'est pas aussi bon qu'avec des langages tels qu'AltaRica, Figaro ou HiP-HOPS.

4.4.4 Modèle d'erreurs avec le formalisme EFFBD

[Seidner, 2009] propose la sémantique d'un EFFBD étendu pour prendre en compte des pannes sur les fonctions ou des données du système. Sept catégories de pannes ont été définies :

1. non activation de la fonction : la fonction recevant correctement tous ses flux d'activation ne s'active pas ;
2. exécution sans fin de la fonction : une fois activée, la fonction ne se termine jamais ;
3. durée d'exécution modifiée de la fonction : le temps d'exécution de la fonction change ;
4. pas de production des données de la fonction : à l'issue de son exécution, la fonction ne produit pas ses données, mais transmet quand même son flux d'activation vers le nœud suivant ;
5. pas de transfert du flux d'activation de la fonction : à l'issue de son exécution, la fonction produit ses données mais ne transmet pas son flux d'activation vers le nœud suivant ;
6. flux de données initial modifié : à l'état initial, certain flux de données « trigger » sont déjà transmis ;
7. flux de données produit modifié : la quantité de flux produit par la fonction est modifiée.

Ces catégories de pannes sont étroitement liées à la sémantique des EFFBDs (voir la section 3.2.3). Il est possible dans une certaine mesure de les lier aux modes de panne de l'analyse des risques. (1) et (2) simulent la perte totale d'une fonction. (3), (4), (5) et (7) peuvent représenter des pertes partielles d'une fonction. Pour (3), nous avons une perte partielle uniquement si le temps de réponse d'une fonction est plus long que celui attendu, mais toujours dans un ordre de grandeur acceptable. (4), (5) et (7) représentent quand à eux des pertes totales de sous-fonctions de la fonction considérée. Enfin, (6) peut permettre de simuler un fonctionnement intempestif en activant un « trigger » qui ne devrait pas l'être.

Cependant, toutes les causes de perte partielle, fonctionnement erroné et fonctionnement intempestif ne sont pas modélisées. Le fonctionnement erroné d'une fonction est d'ailleurs un mode de panne complètement absent de la sémantique proposée pour les modèles d'erreur en EFFBD. De tels modèles ne sauraient donc assister correctement l'analyse des risques. De plus, à notre connaissance, ces modèles d'erreur ne sont toujours pas implémentés dans les outils industriels. Par conséquent, l'utilisation du formalisme EFFDB pour les analyses de sécurité n'est pas mature.

4.4.5 Modèles d'erreurs en Scade et Matlab/Simulink

Dans le monde industriel, on ne différencie pas le langage formel Lustre ([Halbwachs et al., 1991]) de l'outil de modélisation graphique SCADE ([Dormoy, 2008]) dont le langage de programmation constitue le cœur. SCADE a été développé en 1995 par la société Esterel Technologies pour aider à la conception de systèmes critiques. Scade réunit un ensemble de diagrammes de type flots de données et de machines à états. Les premiers sont basés sur le langage synchrone Lustre, conçu pour la programmation sûre de systèmes réactifs et temps réel. Les langages synchrones sont dotés d'une sémantique mathématique précise et leur comportement est déterministe. Les modèles SCADE s'interfacent alors parfaitement avec d'autres modèles de l'ingénierie système tels que les modèles Matlab/Simulink.

L'un des atouts de SCADE est la génération de code C ou Ada pour le développement de systèmes temps réel sur différents supports d'exécution temps réel comme OSEK (utilisé dans l'automobile) et MicroC/OS-II (utilisé principalement dans l'avionique).

Concurrent de SCADE, Matlab/Simulink ([Matlab Simulink]) est également un langage à flots de données synchrone, enrichi d'une couche graphique, permettant la modélisation d'architectures des systèmes embarqués. Matlab/Simulink est particulièrement dédié à l'étude des lois de contrôle/commande.

Scade et Matlab/Simulink sont des langages très utilisés dans l'ingénierie pour la modélisation des systèmes et en particulier des lois de contrôle commande. Plusieurs travaux se sont donc intéressés à l'exploitation directe de ces modèles de conception pour assister les activités de sûreté de fonctionnement. Les similarités des approches déployées sur chacun de ces deux langages nous ont incités à les présenter dans la même section.

Dans [Papadopoulos et al., 2001], l'approche choisie est une extension de Matlab/Simulink, appelée « *Hazard analysis editor* », qui permet d'annoter les composants d'un modèle Matlab/Simulink avec des informations sur les conditions de pannes et de propagations de pannes. Ainsi, les informations propres à la sécurité sont greffées directement sur les modèles d'architecture du système. Ces informations sont des modes de défaillance, qui viennent modifier les valeurs calculées par les fonctions. Le modèle du système est donc étendu et comme le modèle original, il est simulable et vérifiable à l'aide d'outils de model-checking. En particulier, un outil a été développé pour générer automatiquement les coupes minimales à partir d'événements redoutés choisis.

Une autre famille d'approches s'appuie sur le prouveur SCADE Design Verifier d'Esterel Technologies, lié à SCADE, afin de vérifier des propriétés de sûreté de fonctionnement sur les modèles SCADE. En particulier, il est possible d'en extraire les coupes minimales et des calculs de probabilités d'occurrences de pannes, sans avoir à générer des Arbres de Défaillances ([Abdulla et al., 2004]). Cette approche peut également être déployée sur des modèles Matlab/Simulink en transformant au préalable ces modèles en modèles Scade ([Joshi et al., 2005]) ou en utilisant directement la plate-forme de vérification FSAP/NuSMV ([Bozzano, 2003]).

Toutefois, les modèles produits dans ces deux langages sont de trop bas niveau (couche matérielle) pour que ces approches puissent être adaptées pour assister l'analyse des risques. De plus, l'expressivité de ces modèles d'erreur est limité du point de vue de la sécurité.

5 Bilan

En vue de développer une nouvelle activité de modélisation pour assister l'analyse des risques, nous avons présenté des techniques permettant de définir des modèles cohérents et respectueux des processus de conception et de sécurité grâce aux DSML et au MOF.

Ce chapitre a également dressé une vue globale des techniques de modélisation mises en œuvre pour, d'une part, représenter les dépendances fonctionnelles dans des modèles A.O.F. et, d'autre part, étudier les

dysfonctionnement des architectures pour assister des analyses de sécurité. En particulier, nous avons présenté les diagrammes d'activités et le langage AltaRica qui sont tous les deux utilisés chez Airbus. Nos travaux s'appuient sur ces deux formalismes.

Chapitre 3

Objectifs et approche de nos travaux

Sommaire

Résumé. Ce chapitre introduit nos travaux en présentant nos objectifs de modélisation et l'approche générale que nous avons choisie. Les objectifs et les exigences que nous y définissons guident nos travaux décrits dans les chapitres 4, 5 et 6.

1 L'analyse des risques basée sur les modèles : les objectifs

1.1 La problématique de notre étude

Dans le chapitre 1, nous avons montré en section 3.4 que l'analyse des risques est une étape du processus de sécurité qui pourrait être optimisée, en particulier au niveau du format de capitalisation des données de l'analyse et vis-à-vis de la complexité des systèmes aéronautiques. Parmi les axes d'amélioration, l'identification des fonctions à analyser peut être consolidée grâce à l'approche méthodologique du projet ALFA. Les autres axes d'amélioration sont toujours des problématiques ouvertes, rappelées ci-après :

- détermination des conséquences des pannes ;
- prise en considération de données complémentaires, telles que la charge de travail de l'équipage ou les conditions environnementales ;
- identification de combinaisons de pannes pertinentes à analyser ;
- communication et traçabilité entre les domaines de la conception et de la sécurité.

Or, récemment, les concepteurs de l'avion ont développé des activités de modélisation permettant de gérer la complexité des systèmes aéronautiques. Cependant, ces modèles ne sont pas adaptés pour soutenir directement les activités de sûreté de fonctionnement (voir la section 3.4 du chapitre 2). Alors, pour répondre aux quatre limitations, rappelées ci-dessus, nous proposons de définir une nouvelle approche de modélisation. Des recherches récentes (voir la section 4.2 du chapitre 2) ont montré que l'utilisation de modèles formels, et en particulier des modèles en langage AltaRica, permet de remédier au problème de complexité des systèmes et de leur intégration dans l'avion ([MBSA Guideline, 2011], [Bozzano et al., 2011]). Le chapitre 2 présente entre autre les avantages des modèles de sécurité et la méthodologie associée, dans le cadre de l'approche dite « *Model-Based Safety Assessment* » (MBSA) permettant de vérifier les architectures systèmes. Ces modèles offrent :

- un cadre formel de capitalisation des données, dont la nature structurée vient renforcer la confiance envers les analyses de sécurité ;
- des moyens de calcul permettant d'exploiter les modèles de façon à leur faire porter le poids de la complexité des systèmes et de leur intégration dans l'avion ;
- des vues, pouvant être soutenues par des couches graphiques ergonomiques, dont le principal bénéfice est d'assister la compréhension des systèmes et la communication avec différents acteurs de la conception. Ces vues sont davantage conformes aux diagrammes utilisés pour décrire les architectures, que les formats des documents de FHA ou de SSA ou que des Arbres de Défaillances.

Connaissant ces avantages, les analystes de la sécurité chez Airbus ont souhaité mettre en place une approche similaire à l'approche MBSA, pour assister les analyses des risques. Les nouveaux modèles de sécurité doivent être concrétisés à l'aide du langage AltaRica (voir la section 4.3.1 du chapitre 2), sur la base conceptuelle de modèles A.O.F.. Notre problématique d'étude s'est adaptée à ce besoin :

Problématique de l'étude. Comment passer des modèles A.O.F. du projet ALFA à des modèles d'analyse des aspects dysfonctionnels s'appuyant concrètement sur le langage AltaRica ? Et quelles sont les contraintes de ces modèles dysfonctionnels et les moyens mis à disposition pour les exploiter, dans le but de pallier les difficultés identifiées dans l'analyse des risques ?

1.2 Les objectifs pour une analyse des risques basée sur les modèles

Trois objectifs découlent de la problématique précédente. Ils sont présentés dans les paragraphes ci-dessous et dans [Maitrehenry et al. 2011 (2)].

1.2.1 Définition d'une modélisation pour soutenir l'analyse des risques

Notre premier objectif est de proposer de nouveaux modèles et méthodes d'exploitation des modèles pour assister l'analyse des risques. Ces modèles formels devront, en particulier, faciliter l'étude des conséquences des pannes fonctionnelles sur l'avion et ses occupants (deuxième limitation, à la section 3.4.2 du chapitre 1), prendre en considération de nouvelles données telles que les dépendances fonctionnelles et les conditions environnementales (troisième limitation, à la section 3.4.3 du chapitre 1) et enfin permettre l'identification des scénarios de pannes pertinents à analyser (quatrième limitation, à la section 3.4.4 du chapitre 1). Les travaux répondant à ce premier objectif font l'objet du chapitre 4.

Objectif 1. Formaliser les concepts d'une nouvelle approche de modélisation pour assister l'analyse des risques.

Ce premier objectif est suivi d'une étape d'implémentation des concepts à l'aide du langage AltaRica. Les travaux répondant à ce second objectif font l'objet du chapitre 5. Un cas de déploiement industriel présenté au chapitre 7 permet de juger de la difficulté à modéliser concrètement nos concepts ainsi que de l'utilité des modèles d'analyse des aspects dysfonctionnels au regard des analyses des risques réalisées lors de ces dernières décennies.

Objectif 2. Implémenter et expérimenter les concepts, formalisés précédemment, à l'aide du langage de modélisation AltaRica.

1.2.2 Passerelle entre les modèles ALFA et les modèles dédiés à l'analyse des risques

Comme ALFA est un projet fédérateur de toutes les activités pendant la conception d'un nouvel avion, le meilleur moyen d'améliorer la communication entre les analyses des risques et la conception est d'interfacer les nouveaux modèles de sécurité avec les modèles ALFA.

Les deux objectifs précédents se concentrent sur la définition de nouveaux modèles AltaRica, sur la base des éléments des architectures opérationnelles et fonctionnelles. Mais à ce stade, les deux modélisations ne sont connectées que de façon informelle : les modèles AltaRica sont générés à partir des modèles d'architecture et les résultats de leurs exploitations sont transmis aux concepteurs (voir la figure 3.1).

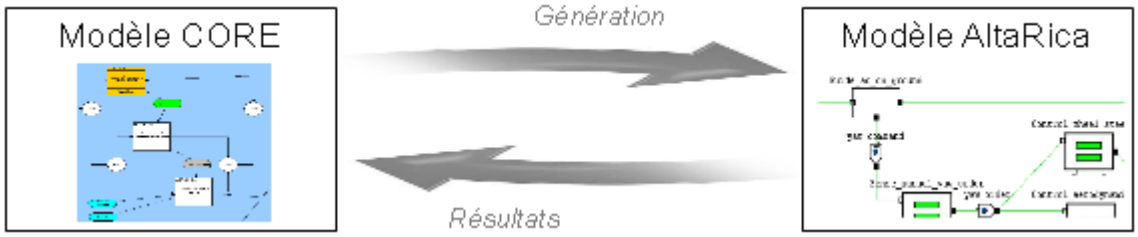


Fig. 3.1 - Liens entre les modèles de conception et les modèles de sécurité

Le troisième et dernier objectif de nos travaux est de formaliser les liens entre les modèles concrets du projet ALFA et les modèles AltaRica que nous proposons. Nous nous appuyons pour cela sur les techniques de transformation de modèle de l'Ingénierie Dirigée par les Modèles (voir la section 2.2 du chapitre 2). Les travaux correspondant à ce troisième objectif font l'objet du chapitre 6.

Objectif 3. Formaliser la transformation de modèle entre les modèles de sécurité et les modèles ALFA. Implémenter et expérimenter cette formalisation à travers le développement d'un démonstrateur.

1.2.3 Les exigences liées aux objectifs définis

Nos trois objectifs sont accompagnés de plusieurs exigences afin que nos travaux soient exploitables dans plusieurs contextes et cas d'étude industriels, en particulier chez Airbus.

Indépendance par rapport aux outils et aux langages de modélisation

Les applications réalisées au sein de nos travaux s'appuient sur les outils CORE et SysML pour la conception des modèles ALFA, et sur le langage AltaRica pour les modèles de sécurité. Pour la conception, SysML s'impose de plus en plus, mais manque toujours de maturité industrielle. AltaRica, quand à lui, est déjà un langage de modélisation de référence dans le cadre de l'approche MBSA (voir la section 4.2 au chapitre 2), mais il subit régulièrement des évolutions syntaxiques parfois conséquentes.

Nous constatons que les outils et langages de modélisation utilisés actuellement sont susceptibles de changer ou d'être remplacés au cours de futurs projets et futurs programmes avion. Notre approche doit alors pouvoir s'adapter à différents outils et langages de modélisation.

Préservation des processus et modèles existants pour la conception et la sécurité

La rigueur des processus de développement interdit aux analystes de la sécurité d'imposer un changement dans les processus et les documents de la conception. Même si leurs avis sont pris en compte lors de la conception et de différentes revues des modèles A.O.F., leurs activités ne doivent en aucun cas remettre en cause les référentiels. Cette exigence contraint particulièrement le troisième objectif concernant la passerelle entre la conception et la sécurité : la passerelle ne doit pas être intrusive dans les modèles d'entrée.

Mais le processus de sécurité doit également être préservé. En effet, comme ce processus est validé par les autorités de certification, toute modification doit être étudiée par différents industriels aéronautiques. Il faut en démontrer la robustesse et les gains par rapport au processus actuellement agréé. Or, notre approche de modélisation s'inscrit dans un projet de recherche, n'ayant pas pour objectif de remettre en cause les processus actuels. Notre approche agit alors comme un soutien à l'analyse des risques, sans impacter le processus en lui même. Notamment, le format du document de l'analyse des risques ne doit pas être modifié.

Approche évolutive

Nous avons identifié plusieurs perspectives liées aux modèles que nous proposons. La définition de nos modèles de sécurité n'est donc pas figée. Il est nécessaire que cette définition puisse évoluer de façon modulaire. L'exigence est double : non seulement l'approche proposée doit pouvoir être aisément complétée par l'ajout de nouveaux concepts, mais en plus, elle doit être assez ouverte pour que les différents éléments de base qui composent cette approche, puissent être réutilisables indépendamment les uns des autres.

Traçabilité entre la conception et la sécurité

La traçabilité est au cœur du développement d'un nouvel avion car elle assure la cohérence entre toutes les analyses réalisées au cours de la conception. En particulier, toutes les exigences de niveau avion, système et équipement sont tracées.

Dans le cadre de notre proposition, cela signifie que chaque élément des modèles de sécurité doit être lié explicitement aux éléments de la conception dont il est issu. La traçabilité prend tout son sens avec les techniques d'ingénierie des modèles qui nous servent à formaliser la passerelle entre les modèles ALFA et nos modèles de sécurité.

2 Présentation de l'approche proposée

En réponse aux trois objectifs et à leurs exigences, présentés précédemment, nous proposons l'approche illustrée par la figure 3.2 ci-dessous.

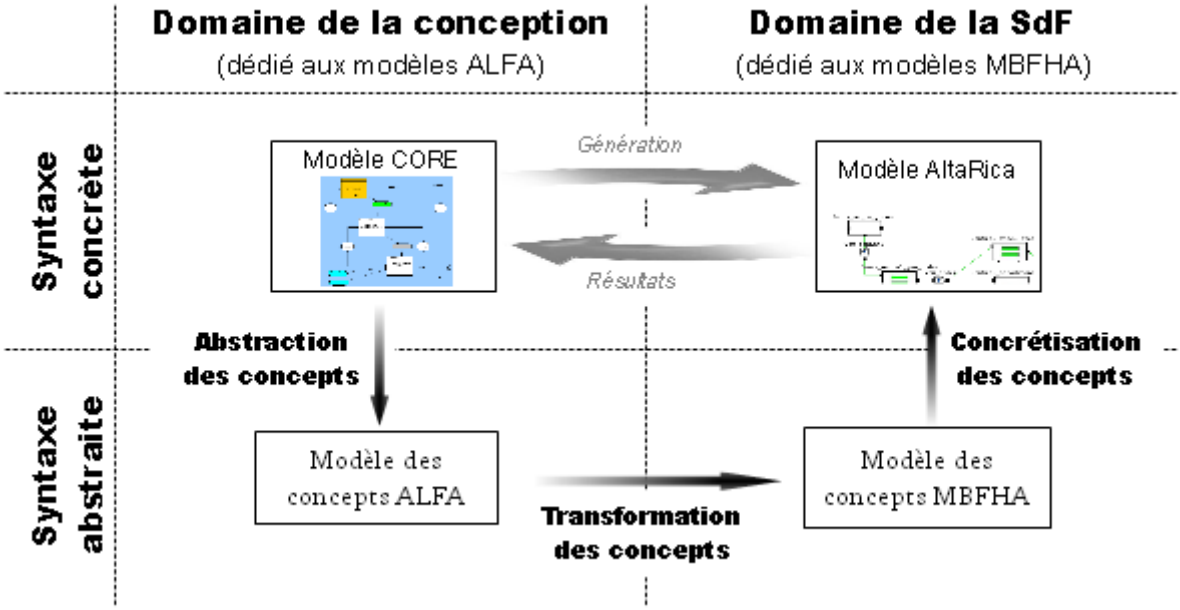


Fig. 3.2 - Approche pour la synthèse de nos modèles de sécurité

2.1 Description détaillée de l'approche

Il s'agit d'une approche classique, inspirée par les techniques de l'Ingénierie des Modèles. Les domaines des modèles ALFA et de nos modèles de sécurité sont clairement distingués à l'aide de deux DSML (voir la section 2.1.3 du chapitre 2).

1. La séparation des deux vues (conception et sécurité) est conforme au besoin de préservation des processus de développement de l'avion. Les concepts de modélisation des architectures opérationnelles et fonctionnelles sont définis par les concepteurs dans le « domaine de la conception » à gauche de la figure 3.2. De l'autre coté, les concepts des modèles de sécurité sont maîtrisés par les analystes de la sécurité dans le domaine dit « domaine de la SdF ».

2. Il existe de nombreux modèles A.O.F., se focalisant sur différents scénarios de vol et des analyses spécifiques de performance. Ces modèles sont construits sur les mêmes concepts, mais selon les besoins d'analyse, ils peuvent être écrits dans des langages de modélisation différents. Un DSML permet de réunir tous ces modèles autour de leurs concepts. Ainsi, à l'intérieur de chaque DSML, nous distinguons pour chaque modèle sa représentation abstraite (par exemple le modèle ALFA, M_{ALFA}) et sa représentation concrète (par exemple le modèle EFFBD correspondant).

   La même logique s'applique au domaine des modèles de sécurité. Les modèles de concepts sont

appelés modèles MB_{FHA} , pour « *Model-Based FHA* ». Ils peuvent ensuite être concrétisés à l'aide de différents outils tels que le langage Figaro ou le langage AltaRica.

3. L'ingénierie des modèles propose des méthodes de transformation de modèle basées sur la définition de plusieurs DSMLs. Dans notre approche, nous choisissons de lier les deux DSMLs au niveau de leur syntaxe abstraite, c'est-à-dire au niveau des concepts de modélisation. Ainsi, pour générer un modèle de sécurité en AltaRica à partir d'un modèle ALFA en CORE ou en SysML, trois étapes de transformation de modèle sont nécessaires.

   - Première étape : une transformation de l'architecture opérationnelle et fonctionnelle concrète vers son modèle ALFA propre. Cette transformation est appelée « abstraction ».
   - Deuxième étape : puis une transformation du modèle ALFA obtenu vers un modèle MBFHA. Ce dernier intègre des concepts du modèle ALFA auxquels il ajoute des informations de sécurité. Cela s'appelle la « transformation des concepts ».
   - Troisième étape : enfin, le modèle MBFHA est traduit en syntaxe AltaRica. Il s'agit d'une étape de « concrétisation » des concepts.

2.2 Notre approche au regard des exigences

Cette approche respecte les exigences établies dans le paragraphe 1.2.3 .

1. L'indépendance de l'approche par rapport aux outils et aux langages de modélisation est assurée grâce à la distinction des vues concrètes et abstraites. Même si à l'avenir les outils et les langages sont susceptibles de changer, les concepts manipulés resteront stables. En particulier, les interactions entre les domaines de la conception et de la sécurité sont indépendantes des outils, car elles ont lieu entre les définitions abstraites des différents modèles.
2. Notre approche préserve les processus existants. La séparation des deux domaines de modélisation offre une protection supplémentaire au processus de la conception : les analystes de la sécurité n'ayant accès en écriture qu'au domaine des modèles de sécurité.
3. Notre approche n'entrave pas les évolutions. Il est possible de modéliser de nouveaux concepts dans l'un ou l'autre des domaines, sans pour autant être contraint de concrétiser ces concepts. De possibles évolutions peuvent donc être réalisées au niveau des représentations abstraites, sans perturber les analyses effectuées sur les modèles concrets.
4. Enfin, l'approche renforce le besoin de traçabilité, puisque la traçabilité doit non seulement être garantie entre les éléments des modèles de la conception et des modèles de sécurité, mais aussi au sein de chaque domaine entre les concepts et leurs représentations concrètes.

Deuxième partie

Définition et synthèse de modèles pour l'analyse des risques

Chapitre 4

Définition des modèles conceptuels pour la sécurité

Sommaire

Résumé. Ce chapitre définit les concepts des modèles pour assister l'analyse des risques. Nous y introduisons la notion d'efficacité permettant d'évaluer la performance des activités. Nous proposons une formalisation des modèles dégradés, basée sur ces valeurs d'efficacité et nous montrons que ces modèles permettent l'étude des pannes fonctionnelles.

1 Introduction : Formalisation des modèles conceptuels de sécurité

Les modèles d'analyse des aspects dysfonctionnels, dédiés à l'analyse des risques, notés MB_{FHA}, reposent sur la modélisation des pannes fonctionnelles, telles qu'elles sont définies pour l'analyse des risques, et sur leur propagation à travers les fonctions et les opérations. Or, comme nous l'avons vu, les propagations des pannes fonctionnelles sont fortement liées aux dépendances fonctionnelles. Ces dépendances sont déjà capturées dans des modèles A.O.F., dans le cadre du projet ALFA. La structure des modèles MB_{FHA} est similaire à celle des modèles A.O.F. M_{ALFA}, présentée à la section 3.3.2 du chapitre 2.

Mais, comme nous l'avons vu dans la section 3.4 du chapitre 2, les modèles M_{ALFA} ne fournissent pas une formalisation assez riche des dépendances fonctionnelles pour l'étude des pannes fonctionnelles. Nous proposons de détailler la définition des concepts des modèles de sécurité.

La section 2 de ce chapitre introduit les principes de la sémantique de la couche nominale des modèles de sécurité, dans laquelle la modélisation des pannes est omise. La section 3 présente les concepts permettant d'étudier des fonctionnements dégradés par des pannes ou des conditions environnementales. Enfin, la section 4 présente les principes d'analyses.

2 Sémantique de la couche nominale des modèles de sécurité

Le fonctionnement nominal d'un système représente son comportement lorsqu'il ne subit aucune dégradation par des pannes. En revanche, les dégradations par les conditions environnementales sont prises en compte. Dans cette section, nous définissons les modèles de sécurité dans ce fonctionnement nominal.

Bien qu'aucune analyse des risques ne puissent être réalisée sur un modèle qui n'intègre pas une représentation des pannes, il est primordial que le comportement nominal du modèle soit conforme à celui de l'architecture conçue par les architectes de l'avion. Pour cette raison, nous avons choisi de repartir de la formalisation des modèles M_{ALFA}.

2.1 La syntaxe abstraite de la couche nominale des modèles de sécurité

De la même façon que pour les modèles ALFA, nous définissons la syntaxe abstraite des modèles MB_{FHA} à partir de la structure des graphes dirigés acycliques, ayant les propriétés d'être continus et bornés (voir la section 3.3.2 du chapitre 2). La table 4.1 présente les bases de la structure des modèles d'analyse des aspects dysfonctionnels, à l'aide du langage Express. Nous retrouvons logiquement que les modèles MB_{FHA} sont composés d'opérations, de fonctions, de données échangées, de conditions environnementales et de différents liens entre les nœuds.

```
1   ENTITY MB_FHA SUBTYPE OF (graph);
2         activities : SET [0:?] OF Act_MBFHA;
3         data : SET [0:?] OF Data_MBFHA;
4         envs : SET [0:?] OF Env_MBFHA;
5         links : SET [0:?] OF Lien_MBFHA;
6   -- attributs dérivés et contraintes
7   similaires à ceux de l'entité M_ALFA
8   END_ENTITY;
```

Tab. 4.1 - Entités des modèles de sécurité et de leurs liens en Express

Le squelette des entités qui composent ce modèle est celui de la table 4.2. Cette définition est en tout point conforme aux entités présentes dans les modèles M_{ALFA} (voir la section 3.3.3 du chapitre 2). La définition de ces entités se précise tout au long de ce chapitre.

```
1   ENTITY Act_MBFHA ABSTRACT SUPERTYPE SUBTYPE
2   OF (node);
3   END_ENTITY;
4   ENTITY Fct_MBFHA SUBTYPE OF (Act_MBFHA);
5   END_ENTITY;
6   ENTITY Op_MBFHA SUBTYPE OF (Act_MBFHA);
7   END_ENTITY;
8
9   ENTITY Data_MBFHA SUBTYPE OF (node);
10  END_ENTITY;
11
12  ENTITY Env_MBFHA SUBTYPE OF (node);
13  END_ENTITY;
14
15  ENTITY Lien_MBFHA ABSTRACT SUPERTYPE
16  SUBTYPE OF (arc);
17  END_ENTITY;
18
19  ENTITY Lien_data_MBFHA ABSTRACT SUPERTYPE
20  SUBTYPE OF (Lien_MBFHA);
21  END_ENTITY;
22  ENTITY Lien_trig_MBFHA SUBTYPE OF
23  (Lien_data_MBFHA);
24        init : Data_MBFHA;
25        final : Act_MBFHA;
26  END_ENTITY;
27  ENTITY Lien_in_MBFHA SUBTYPE OF
28  (Lien_data_MBFHA);
29        init : Data_MBFHA;
```

```
30          final : Act_MBFHA;
31      END_ENTITY;
32      ENTITY Lien_out_MBFHA SUBTYPE OF
33      (Lien_data_MBFHA);
34          init : Act_MBFHA;
35          final : Data_MBFHA;
36      END_ENTITY;
37
38      ENTITY Lien_proc_MBFHA SUBTYPE OF
39      (Lien_MBFHA);
40          init : Op_MBFHA;
41          final : Op_MBFHA;
42      END_ENTITY;
43
44      ENTITY Lien_env_MBFHA SUBTYPE OF
45      (Lien_MBFHA);
46          init : Env_MBFHA;
47          final : Act_MBFHA;
48      END_ENTITY;
```

Tab. 4.2 - Entités composant les modèles de sécurité

2.2 Introduction de la sémantique de la couche nominale des modèles de sécurité

Dans la section 3.3.4 du chapitre 2, nous avons introduit la notion de service d'une activité. Elle permet d'évaluer la contribution de chaque activité à la réalisation de la phase de vol représentée par le modèle MB_{FHA} considéré. Cependant, la propagation de ces valeurs à travers les liens (et donc à travers les dépendances fonctionnelles) n'est pas formalisée dans les modèles M_{ALFA}. Cette section pallie ce manque.

2.2.1 Quelques définitions préliminaires

La sémantique des modèles MB_{FHA} nécessite de nouvelles définitions, applicables aussi bien à la couche nominale des modèles qu'aux modèles intégrant des pannes fonctionnelles. Premièrement, l'étude des dépendances fonctionnelles n'est possible qu'en disposant d'une évaluation des liens.

Exemple. Nous avons montré, dans la section 3.4.1 du chapitre 2, que la perte totale d'une fonction censée transmettre une donnée nécessaire à une autre fonction conduit à la non-production de la donnée et donc à un arrêt du flux d'activation fonctionnel. Une évaluation des liens de donnée, allant de la première fonction vers la seconde fonction, pourrait par exemple utiliser l'ensemble composé de deux valeurs : $\{donnée\ produite,\ donnée\ non\ produite\}$.

Dans le cadre général, nous introduisons le domaine de valeurs D, qui contient toutes les valeurs utilisées pour évaluer les attributs des entités des modèles MB_{FHA}.

Définition : Évaluation des liens dans les modèles de sécurité

D : domaine des valeurs du modèle MB_{FHA}.

$Lien_{MBFHA}$: ensemble des liens du modèle MB_{FHA}, tel que défini dans la table 4.2.

Le domaine des valeurs permettant d'évaluer les liens dans les modèles de sécurité est :

– $dom : Lien_{MBFHA} \rightarrow 2^D$, *telle que* $\forall l \in Lien_{MBFHA},\ dom(l) \neq \emptyset$.

De la même façon que cela a été fait pour la sémantique du langage AltaRica (voir la section 4.3.2 du chapitre 2), nous surchargeons la fonction dom par :

– $dom : 2^{Lien_{MBFHA}} \rightarrow 2^D$, *telle que :* $\forall L \subseteq Lien_{MBFHA},\ dom(L) = \prod_{l \in L} dom(l)$.

Il s'agit du vecteur des valeurs admissibles de chaque lien d'un ensemble défini de liens.

L'évaluation des liens dans les modèles de sécurité est défini par :

– $val : Lien_{MBFHA} \rightarrow D$, *telle que* $\forall l \in Lien_{MBFHA},\ val(l) \in dom(l)$.

Nous en définissons l'évaluation d'un ensemble de lien par :

– $val :\ 2^{Lien_{MBFHA}} \rightarrow D^{2^{Lien}}$, *telle que :* $\forall L \subseteq Lien_{MBFHA},\ val(L) = \prod_{l \in L} val(l)$.

Par commodité, nous définissons également les liens d'entrée et de sortie des nœuds par :

Définition : Liens entrants et sortants d'un nœud

– *L'ensemble des liens entrants dans un nœud est* $input :\ Noeud \rightarrow 2^{Lien_{MBFHA}}$, *tel que :*

$$\forall n \in Noeud,\ input(n) = \{(n', n) \in Lien_{MBFHA} / n' \in Noeud\}.$$

– *L'ensemble des liens sortants d'un nœud est* $output :\ Noeud \rightarrow 2^{Lien_{MBFHA}}$, *tel que :*

$$\forall n \in Noeud,\ output(n) = \{(n, n') \in Lien_{MBFHA} / n' \in Noeud\}.$$

2.2.2 Les lois de comportement

Pour tous les nœuds des modèles MB_{FHA} nous identifions des lois de comportement pour en évaluer les liens entrants et sortants. La définition de ces lois dépend du type de nœud.

Loi de comportement des activités

Dans la section 3.3.4 du chapitre 2, nous avons défini la notion de service d'une activité comme étant un objectif de cette activité. La valeur des services produits dépend de l'évaluation des liens entrants et des liens trigger dans l'activité, à partir desquels l'activité apporte une valeur ajoutée (voir la figure 4.1).

Fig. 4.1 - Services d'une activité

Chaque activité est donc régie par une loi de comportement, nommée Loi_{act}. Cette loi de comportement s'appuie également sur des variations dans le comportement interne des activités. En effet, nous prévoyons qu'au cours d'une même phase de vol, certaines activités peuvent voir leur fonctionnement modifié.

Exemple. Les changements de comportement des activités ne sont pas rares. Pour les fonctions, nous pouvons citer les changements de configuration de l'avion lors de la préparation du vol voire en plein vol. Par exemple, la plupart des fonctions de calcul, qui pilotent les actionneurs, ont plusieurs modes de calcul, dont un mode complet prenant en compte un grand nombre de variables et plusieurs modes dégradés dans lesquels les lois de calcul sont plus approximatives voire transmettent directement et sans vérification les commandes des pilotes aux actionneurs. Ces modes de fonctionnement ne sont pas forcement liés à des situations de pannes fonctionnelles : parfois, pour économiser l'énergie, une loi de calcul dégradé est utilisée car moins coûteuse, tout en restant aussi acceptable et sûre que la loi la plus riche. Cependant, les pannes fonctionnelles sont également des exemples de variations de comportement. Ce cas est traité dans la section 3.1.1.

Définition : Loi de comportement des activités

Soit $C_{act} \subset D$ *l'ensemble des variations comportementales potentielles pour les activités.*

Soit $c: \ Act_{MBFHA} \rightarrow C_{act}$ *la fonction qui associe à chaque activité son mode de comportement actif.*

La propagation des liens à travers chaque activité $a \in Act_{MBFHA}$ *est définie formellement par la loi de comportement :*

– $Loi_{act}(a): \ C_{act} \times dom(input(a)) \rightarrow dom(output(a))$, *qui calcule les valeurs des liens sortants de l'activité par rapport aux valeurs de ses liens entrants et de son comportement interne* $c(a) \in C_{act}$.

La loi de comportement de chaque activité définit donc la façon dont les liens se propagent à travers les activités. Du fait que chaque activité représente des opérations et des fonctions différentes, il y a dans l'absolu autant de comportements possibles que d'activités et donc tout autant de lois de comportement. Toutefois, comme les activités représentent une abstraction de la réalité, tant pour les actions humaines que pour les systèmes avion, il est possible dans une certaine mesure de mettre en évidence des comportements génériques. Nous revenons sur cette problématique dans la section 2.5.4.

Loi de comportement des données

Les échanges de données nécessaires et de données utiles entre les activités sont réalisées à travers les nœuds de type $Data_{MBFHA}$. Comme pour les activités, nous identifions une propagation de liens à travers ces nœuds : plusieurs services de différentes activités contribuent à la production d'un flux de donnée qui est transmis à d'autres activités (voir la figure 4.2, ci-dessous).

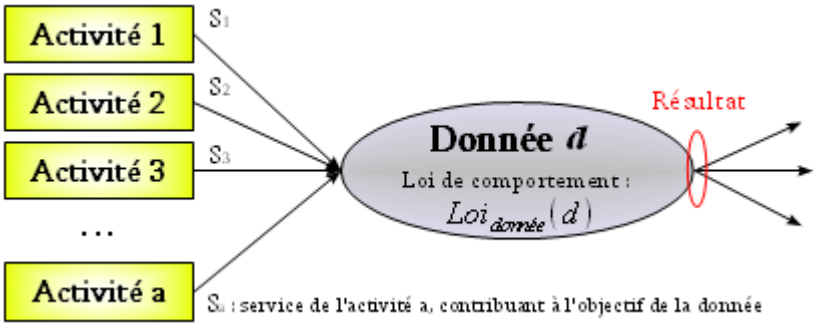


Fig. 4.2 - Contribution d'activités à la réalisation d'un résultat

Exemple. Dans l'exemple du contrôle de la trajectoire au sol, la donnée « trajectoire contrôlée » combine les services des fonctions « Guider avec la roulette de direction » et « Guider aérodynamiquement ». Chacun de ces deux services contribue en effet au contrôle de la trajectoire.

Nous employons le terme de **résultat** d'une donnée pour définir le travail conjoint de différentes activités. Le résultat d'une donnée est donc la valeur transmise aux activités qui reçoivent cette donnée en entrée, en tant que donnée utile ou nécessaire à travers leurs liens entrants.

Définition : Loi de comportement des données

Soit $r: Data_{MBFHA} \rightarrow D$ *la fonction qui associe à chaque donnée la valeur de son résultat.*

La combinaison des services au sein d'une donnée $d \in Donnée$ *est définie formellement par la loi de comportement :*

– $Loi_{donnée}(d): dom(input(d)) \rightarrow dom(output(d))$, *qui calcule les valeurs des liens sortants de la donnée* d *(égales à la valeur de son résultat) en fonction des valeurs de ses liens entrants.*

Comme pour la loi de comportement des activités, nous montrons par la suite qu'il est possible de définir des lois génériques pour déterminer le résultat produit par un ensemble évalué de services.

Loi de comportement des conditions environnementales

Contrairement aux autres familles de nœuds, les conditions environnementales n'ont aucun lien entrant. Les valeurs de leurs liens sortants ne dépendent alors que de leur comportement interne, qui caractérise leur puissance d'impact. La section 2.3.2 donne plus de détail sur la signification de ces valeurs.

Définition : Loi de comportement des conditions environnementales

Soit $P_{env} \subset D$ *l'ensemble des puissances environnementales potentielles des conditions environnementales.*

Soit $c: Env_{MBFHA} \rightarrow P_{env}$ *la fonction qui associe à chaque condition environnementale sa puissance d'impact.*

La loi de comportement de chaque $n_{env} \in Env_{MBFHA}$ *est définie formellement par :*

– $Loi_{env}(n_{env}): P_{env} \rightarrow dom(output(n_{env}))$ *, qui décrit les valeurs prises par les liens sortants de la condition environnementale* n_{env} *par rapport à sa puissance d'impact.*

2.2.3 Définition de la sémantique de la couche nominale des modèles de sécurité

Les valeurs observées et analysées dans nos modèles de sécurité sont celles qui représentent les objectifs attendus par le système socio-technique modélisé dans la phase de vol considérée. Ce sont les nœuds de la famille « Donnée » qui représentent ces objectifs à travers l'évaluation de leur résultat. En particulier, certaines données, appelées « Objectif de vol », caractérisent directement la phase de vol (voir la section 3.3.4 du paragraphe 2).

Sémantique des modèles de sécurité

La sémantique d'un modèle MB_{FHA} *est le triplet :*

$\|MB_{FHA}\| = (D, C, \sigma)$ *, où*

– D *est l'ensemble des valeurs du domaine,*

– $C = c(Act_{MBFHA}) \times c(Env_{MBFHA})$ *est le vecteur des comportements internes du modèle, avec :*

- $c(Act_{MBFHA}) = \prod_{a \in Act_{MBFHA}} c(a) \in (C_{act})^{card(Act_{MBFHA})}$ *est le vecteur des modes de fonctionnement des activités du modèle,*
- $c(Env_{MBFHA}) = \prod_{n_{env} \in Env_{MBFHA}} c(n_{env}) \in (P_{env})^{card(Env_{MBFHA})}$ *est le vecteur des valeurs de puissance d'impact des conditions environnementales du modèle,*

– $\sigma: (C_{act})^{card(Act_{MBFHA})} \times (P_{env})^{card(Env_{MBFHA})} \rightarrow D^{card(Data_{MBFHA})}$ *est la fonction déterminant la valeur de tous les résultats du modèles par rapport aux modes de fonctionnement de ses activités et aux puissances d'impact de ses conditions environnementales.*

La sémantique de nos modèles de sécurité est construite sur la base des trois lois de comportement présentées à la section 2.2.2. En particulier, la fonction σ est obtenue par composition des nœuds des modèles MB_{FHA} , selon un principe similaire à celui présenté à la section 4.3.2 du chapitre 2 pour le langage de modélisation AltaRica. Cette composition peut être trouvée en annexe A.

2.2.4 Problématiques liées à la sémantique

La définition de la sémantique des modèles MB_{FHA} , initiée dans la section précédente, présente deux problématiques majeures.

1. La définition du domaine des valeurs *D* : nous avons vu que ce domaine inclut les valeurs représentant les différents modes de fonctionnement des activités et les puissances d'impact des conditions environnementales. Mais cet ensemble permet également d'évaluer tous les liens du modèle. L'évaluation de ces différents types de liens ne se fait pas sur les mêmes ensembles de valeurs. La première problématique est donc d'expliciter les sous-ensemble des valeurs des modèles MB_{FHA}. Cette problématique fait l'objet de la section 2.3.
2. La sémantique des éléments : dans la section 2.2.2, nous avons montré que la sémantique des éléments des modèles MB_{FHA} est construite sur la base de lois de comportements pour les activités, les données et les conditions environnementales. Nous avons donné l'interface de ces deux lois sans préciser ni les domaines de valeurs (voir la première problématique) ni les opérations mathématiques réalisées. Cette problématique est traitée dans la section 2.4.

2.3 Définition du domaine des valeurs

Étant donné que les liens sont au cœur de la modélisation des dépendances fonctionnelles, leur évaluation est nécessaire dans les modèles de sécurité pour pouvoir calculer la propagation des pannes. Les trois familles de liens définies dans la structure des modèles MB_{FHA}, dans la section 2.1, jouent des rôles différents dans ces dépendances fonctionnelles ; leur évaluation n'est donc pas forcément identique. Les sections suivantes se focalisent sur chaque famille de liens et propose un domaine d'évaluation adapté à chacune. Cela répond à la première problématique identifiée dans la section 2.2.4.

2.3.1 Les valeurs des liens de procédure

D'après la section 3.3.5 du chapitre 2, les liens de procédure transmettent le flux d'activation d'opération à opération pour modéliser la séquence des activités humaines planifiées et réalisées au cours d'une phase de vol dans un scénario de vol donné. Les opérations, comme toutes les activités, peuvent être soit activées, soit non-activées. L'activation est réalisée à partir du moment où tous les flux d'activation sont reçus, c'est à dire :

- les opérations initiales de tous les liens de procédure entrant dans l'opération sont toutes activées ;
- les données nécessaires (trigger) entrant dans l'opération sont toutes reçues.

Pour les liens de procédure, cela signifie que la seule information à transmettre est le statut d'activité de leur opération initiale.

Définition : Domaine d'évaluation des liens de procédure

Le domaine des valeurs permettant l'évaluation des liens de procédure, noté $V_{proc} \subset D$, *est composé des deux éléments :* $V_{proc} = \{activé, non-activé\}$.

Alors, $\forall l \in Lien\_proc_{MBFHA}$, $dom(l) = V_{proc}$.

L'opposition sémantique des valeurs $activé$ et $non-activé$ rappelle le comportement booléen. C'est pourquoi, concrètement, il est possible de modéliser V_{proc} par des booléens.

2.3.2 La puissance environnementale dégradante

Dans la section 3.3.6 du chapitre 2, nous avons montré que les nœuds « Environnement », représentant les conditions environnementales, sont liés aux activités par des liens d'environnement. Les activités peuvent alors voir leurs services modifiés en présence d'une condition environnementale, sous la forme d'une diminution de performance. Chaque lien d'environnement doit donc transmettre l'état de son nœud initial, qui est de type Env_{MBFHA}, c'est à dire sa puissance environnementale.

Définition : Domaine d'évaluation des liens d'environnement

Nous précisons la puissance environnementale comme l'ensemble $P_{env} \subset D$ *des valeurs ordonnées comprises dans l'intervalle* $P_{env} \subseteq [\, puissance\,nulle\,,\, puissance\,infinie\,[$.

Alors, $\forall l \in Lien\_env_{MBFHA},\ dom(l) = P_{env}$.

Dans le cas général, l'ensemble P_{env} n'est pas contraint à être fini. Il admet en revanche une borne minimale, notée $p_{nul} \in P_{env}$ qui signifie l'absence de la condition environnementale. C'est un ensemble ordonné puisqu'il est possible de définir une échelle des puissances des conditions environnementales.

En première approche, il est possible de le définir par un booléen dont la valeur « *true* » signifie la présence de la condition environnementale considérée. De façon plus précise, la puissance environnementale peut également être définie dans l'ensemble des réels positifs : $P_{env} = \mathbb{R}_+$.

Exemple. Pour la pluie, sa puissance environnementale peut être une fonction croissante qui dépend de la quantité de pluie tombée. La puissance nulle signifie alors qu'il ne pleut pas du tout.

2.3.3 L'efficacité opérationnelle et fonctionnelle

Enfin, la dernière famille de liens sont les liens de données, modélisés par des liens sortants des activités, qui sont des services, et des liens entrants ou trigger dans des activités, qui sont des résultats des données. Notre choix de domaine d'évaluation est motivé par le fait que les résultats sont définis comme la combinaison de services qui y contribuent (voir la section 3.3.4 du chapitre 2). Or, nous pouvons remarquer que la contribution de différents services n'est pas forcément identique : certaines activités fournissent un service contribuant davantage au résultat que d'autres. De plus, ces différentes contributions sont également susceptibles d'évoluer en fonction du contexte opérationnel et donc des phases de vol.

Exemple. Dans l'exemple du contrôle de la trajectoire au sol, la donnée « trajectoire contrôlée » est modélisée avec deux services en entrée, correspondant aux contributions respectives des deux fonctions « Guider

avec la roulette de direction » et « Guider aérodynamiquement » (voir la figure 2.9 du chapitre 2). Cependant, à basse vitesse, le service rendu par la roulette de direction contribue beaucoup plus au contrôle de la trajectoire que le guidage aérodynamique. En effet, les forces aérodynamiques appliquées sur la gouverne de direction sont faibles quand la vitesse de l'avion est faible. Mais le rapport de force entre les deux fonctions penche en faveur du contrôle aérodynamique au fur et à mesure que la vitesse de l'avion augmente, grâce à l'augmentation des forces aérodynamiques sur la gouverne de direction et la résistance de braquage de la roulette de direction.

Pour représenter cette évaluation, nous introduisons la notion d'**efficacité**. L'efficacité opérationnelle est une notion qui existe déjà dans la littérature dans le domaine des Facteurs Humains, plus souvent sous le nom de « performance humaine » ([Rasmussen 1983], [Doguc et al. 2009]). Elle est définie comme une mesure de l'accomplissement avec succès d'un ensemble de tâches par un acteur humain, dans un temps requis si une exigence temporelle est spécifiée.

Nous introduisons l'efficacité fonctionnelle pour les fonctions, en partant de la définition de l'efficacité opérationnelle précédente.

Définition : Efficacité

L'efficacité d'un service d'une activité est le taux de contribution de l'activité à la production du résultat auquel le service participe.

L'efficacité d'un résultat d'une donnée est le taux d'aboutissement des objectifs représentés par la donnée.

L'ensemble des valeurs d'efficacité $E \subset D$ *représente des taux.* E *est un ensemble ordonné et borné par le taux maximal* e_{max} *et le taux minimal* e_{min} *.*

Nous définissons $e_{nul} \in E$ *la valeur d'efficacité correspondant à un taux nul.*

Alors, $\forall l \in Lien\_data_{MBFHA}$, $dom(l) = E$.

Dans le cas général, E n'est pas contraint à être un ensemble fini, bien qu'il soit borné. Les objectifs que représente chaque donnée sont finis. Par conséquent, le taux d'aboutissement de ces objectifs est nécessairement majoré par le taux maximal signifiant que tous les objectifs sont correctement remplis. De même, la contribution d'un service ne peut pas excéder ce taux maximal de participation.

A l'inverse, les objectifs représentés par une donnée possèdent leurs objectifs opposés, qui peuvent être déterminés en se demandant quels objectifs faut-il atteindre pour s'éloigner le plus possible du résultat attendu, quitte à s'y opposer complètement ? Ce maximum d'opposition correspond à la borne inférieure de l'efficacité. Bien sur, aucune activité n'est définie, dans son mode de fonctionnement nominal, avec pour but de s'opposer aux résultats désirés. En revanche, nous verrons que ce comportement est possible dans certains cas de panne.

Exemple. Si, pendant la phase de décollage, l'équipage commande à l'avion un virage à droite de 30°, alors l'efficacité du résultat associé à la donnée « trajectoire contrôlée » vaut e_{max} si le virage à droite a bien été réalisé, dans les temps et avec un angle suffisamment proche des 30° requis. En revanche, si, à cause d'un scénario de pannes, le virage réalisé est vers la gauche, alors l'avion s'est opposé aux objectifs qui lui ont été fixés. Dans le cas où cette opposition est considérée maximale, l'efficacité du résultat de la donnée « trajectoire contrôlée » vaudrait e_{min}. Enfin, si l'avion ne vire pas (fonctionnement non nominal), l'efficacité à contrôler la trajectoire peut être jugée nulle : e_{nul}.

En mode de fonctionnement nominal, les résultats produits sont par définition toujours égaux au maximum d'efficacité. Le cas contraire signifierait que les objectifs d'un résultat ne pourraient jamais être atteints. Dans ce cas, l'architecture proposée ne satisferait pas les exigences de performance. Bien sûr, dans une architecture correcte en fonctionnement nominal, certains résultats peuvent ne pas être produits. Mais si leur production est requise, alors ils sont évalués à e_{max}.

En première approche, les taux peuvent être modélisés plus précisément comme des pourcentages dans $\mathbb{R}$. Dans ce cas, l'ensemble des valeurs d'efficacité est défini comme : $E=[-100,100]\subset\mathbb{R}$.

2.4 Détails sur la sémantique des données et des conditions environnementales

Grâce à la définition complète du domaine des valeurs $D=Booléen\cup E\cup P_{env}\cup C_{act}$, les opérations mathématiques réalisées au sein des lois de comportement des nœuds peuvent être définies plus précisément. Cela répond à la deuxième problématique identifiée dans la section 2.2.4.

Dans cette section nous traitons le cas des conditions environnementales et des données.

2.4.1 Sur la sémantique des conditions environnementales

Le nœud Env_{MBFHA} est le plus simple des nœuds des modèles MB_{FHA} car il n'a aucun lien entrant. Il accepte en revanche un nombre indéfini de liens sortants vers des activités dégradées par la présence de la condition environnementale. Ces liens appartiennent tous à la famille des liens d'environnement et sont donc évalués sur l'ensemble P_{env}. La loi de comportement des conditions environnementales affecte alors à tous ses liens sortants la valeur de puissance environnementale de la condition environnementale considérée.

Définition : Loi de comportement des conditions environnementales

La sémantique d'un nœud $n_{env}\in Env_{MBFHA}$, est définie par le triplet $\|n_{env}\|=(D, c(n_{env}), Loi_{env}(n_{env}))$ avec :

- $D=Booléen\cup E\cup P_{env}\cup C_{act}$ *est l'ensemble des valeurs du domaine,*
- $c(n_{env})\in P_{env}$ *est une valeur caractéristique de puissance environnementale dégradante de la condition environnementale n_{env}. Elle représente son comportement.*
- $Loi_{env}(n_{env})$: $P_{env}\rightarrow dom(output(n_{env}))$, *qui affecte la valeur $c(n_{env})$ à tous les liens sortants du nœud n_{env}, c'est à dire que :* $\forall l\in output(n_{env}),\ val(l)=c(n_{env})$.

2.4.2 Sur la sémantique des données

Les caractéristiques des nœuds de type « Donnée » sont présentées dans la section 2.2.2. Nous y avons défini l'interface de la loi de comportement des données, qui agrège les services en entrée d'une donnée pour en calculer le résultat. Les données ne sont pas des variables d'un modèle de sécurité, comme le sont les conditions environnementales ou les activités par leurs modes de fonctionnement. Par conséquent, le comportement des données est unique. L'agrégation des services suit deux principes :

- équité : la donnée ne favorise aucun service d'une activité au dépend d'une autre ;
- addition : nous considérons que les services d'activités différentes ne sont pas directement liés, c'est à dire que la performance d'une activité ne dépend pas de la performance d'une autre sans que cette dépendance fonctionnelle ne soit capturée entre les activités elles-même. Dans ce cas, la combinaison des services est l'addition des taux de contribution à la production du résultat.

Il ne nous est pas possible de définir la loi d'agrégation des services au sein des données dans le cadre général. Nous nous plaçons alors le cas particulier $E=[-100,100]\subset\mathbb{R}$ introduit à la section 2.3.3. Nous définissons alors une loi de composition interne dans cet ensemble E : la somme tronquée, notée $\oplus$ et construite sur la base de l'addition dans les réels.

$$\forall(x,y)\in E \qquad x\oplus y=\begin{cases} e_{max} & si\ (x+y)\geqslant e_{max} \\ e_{min} & si\ (x+y)\leqslant e_{min} \\ x+y & sinon \end{cases}$$

Dans le cas particulier $E=[-100,100]\subset\mathbb{R}$, la sémantique des données est définie par :

Définition : Loi de comportement des données

La sémantique d'un nœud $d\in Data_{MBFHA}$, *est définie par* $\|d\|=(D, Loi_{donnée}(d))$, *avec :*

- $D=Booléen\cup E\cup P_{env}\cup C_{act}$ *est l'ensemble des valeurs du domaine,*
- $Loi_{donnée}(d):\ dom(input(d))\rightarrow dom(output(d))$, *tel que :*

$$\forall l\in output(d),\ \ val(l)=\bigoplus_{d_i\in input(d)} val(d_i)$$

2.5 Détails sur la sémantique des activités

Dans cette section, nous précisons la loi de comportement des activités, introduite dans la section 2.2.2. Comme nous l'avons vu, le fait que chaque activité a une signification bien particulière dans le monde réel conduit naturellement à ne pas pouvoir définir une loi de comportement unique. Néanmoins, par abstraction et simplifications, nous mettons en évidence quelques caractéristiques génériques :

- le calcul de l'activation des activités ;

- l'identification de différents types de service ;
- la définition de la valeur nominale et de la pire valeur des services ;
- l'utilisation de valeurs agrégées pour le calcul des différents services ;

2.5.1 Agrégation des liens entrants

Les liens en entrée des activités peuvent être de quatre types : des liens de donnée nécessaires, utiles, des liens d'environnement ou des liens de procédure. Face à cette complexité et afin de pouvoir définir une loi de comportement pour les activités, nous proposons quatre règles de combinaison de liens entrants. Les avantages de ces agrégations sont mis en évidence dans les sections suivantes.

Combinaison des données nécessaires

Les données nécessaires à une activité sont représentées par les liens de donnée de type liens triggers (voir la section 3.3.4 du chapitre 2). Ces liens sont évalués à l'aide de l'efficacité. Nous proposons d'agréger leur valeur à l'aide des deux critères suivants :

- évaluation pessimiste : les données nécessaires d'une activité sont définies comme étant les plus importantes pour l'activité. A ce titre, toute donnée dont l'efficacité ne serait pas maximale est susceptible de dégrader les services de l'activité concernée. C'est pourquoi, dans le cadre d'une évaluation pessimiste, la valeur d'efficacité agrégée de l'ensemble des données nécessaires est égale à la valeur minimale des efficacités des données nécessaires ;
- l'attente d'une valeur : les données nécessaires sont également définies comme requises par leurs activités. La valeur d'efficacité nulle, e_{nul}, qui représente un résultat inexistant, bloque l'exécution de l'activité concernée. Cette dernière ne peut pas s'exécuter et renvoyer ses services tant qu'une de ses entrées de données nécessaires est inexistante. Par conséquent, la valeur d'efficacité de la combinaison des données nécessaires est égale à e_{nul} si au moins un des flux de donnée nécessaire a la valeur e_{nul}, même si des valeurs d'efficacité négative sont présentes.

Définition : Agrégation des données nécessaires

Soit $val_n : Act_{MBFHA} \rightarrow E$ *la valeur représentant l'agrégation des données nécessaires, définie pour tout* $a \in Act_{MBFHA}$ *par :*

$$val_n(a) = \begin{cases} e_{nul} \quad si \quad \exists l \in input(a) / l \in Lien\_trig_{MBFHA} \wedge val(l) = e_{nul} \\ e_{max} \quad si \quad not(\exists l \in input(a) / l \in Lien\_trig_{MBFHA}) \\ \min\limits_{l \in input(a) / l \in Lien\_trig_{MBFHA}} val(l) \quad sinon \end{cases}$$

Combinaison des données utiles

De façon similaire, nous agrégeons les données utiles de chaque activité. Nous proposons une agrégation moins pessimiste que pour les données nécessaires afin de tenir compte du fait que si une donnée utile est manquante ou dégradée, il est toujours possible de s'appuyer sur des données par défaut ou des données précédentes. Nous choisissons alors d'utiliser la moyenne arithmétique des efficacités.

Définition : Agrégation des données utiles

Soit $val_u : Act_{MBFHA} \rightarrow E$ *la valeur représentant l'agrégation des données utiles, définie pour tout* $a \in Act_{MBFHA}$ *par :*

$$val_u(a) = \left\{ \begin{array}{l} \frac{1}{card\,(input(a))} \cdot \sum_{l \in input(a) / l \in Lien\_in_{MBFHA}} val(l) \quad si \quad \exists l \in input(a) / l \in Lien\_in_{MBFHA} \\ e_{max} \quad sinon \end{array} \right\}$$

Agrégation des puissances environnementales

Nous proposons également une agrégation des puissances environnementales en entrée des activités. D'après la section 3.3.6 du chapitre 2, les conditions environnementales dégradent les services des activités.

En considérant que les conditions environnementales sont indépendantes, leurs impacts peuvent être additionnés. Pour disposer de l'addition, nous nous plaçons dans le cas particulier $P_{env} = \mathbb{R}_+$. Alors :

Définition : Agrégation des puissances environnementales

Soit $val_e : Act_{MBFHA} \rightarrow P_{env}$ *la valeur représentant l'agrégation des puissances environnementales, définie pour tout* $a \in Act_{MBFHA}$ *par :*

$$val_e(a) = \left\{ \begin{array}{l} \sum_{l \in input(a) / l \in Lien\_env_{MBFHA}} val(l) \quad si \exists l \in input(a) / l \in Lien\_env_{MBFHA} \\ p_{nul} \quad sinon \end{array} \right\}$$

Agrégation des activations par procédures

Enfin, nous proposons une agrégation des valeurs booléennes des liens de procédure entrants dans les opérations. Puisqu'une opération ne peut s'activer que si tous ses liens de procédure sont évalués à « *true* », nous définissons :

Définition : Agrégation des activations par procédure

Soit $val_p : Op_{MBFHA} \rightarrow Booléen$ *la valeur représentant l'ensemble des liens de procédure, définie pour tout* $a \in Op_{MBFHA}$ *par :*

$$val_p(a) = \left\{ \begin{array}{l} \bigwedge_{l \in input(a) / l \in Lien\_proc_{MBFHA}} val(l) \quad si \exists l \in input(a) / l \in Lien\_proc_{MBFHA} \\ true \quad sinon \end{array} \right\}$$

2.5.2 Loi d'activation

Le chapitre 2 montre que les activités ont une activation séquencée. Les successions et parallélismes entre les activités sont même au cœur de deux des trois dépendances fonctionnelles identifiées. Les activités peuvent donc se trouver soit au repos, soit elles sont activées et fournissent leurs services. La figure 4.3 ci-dessous illustre ce principe d'activation.

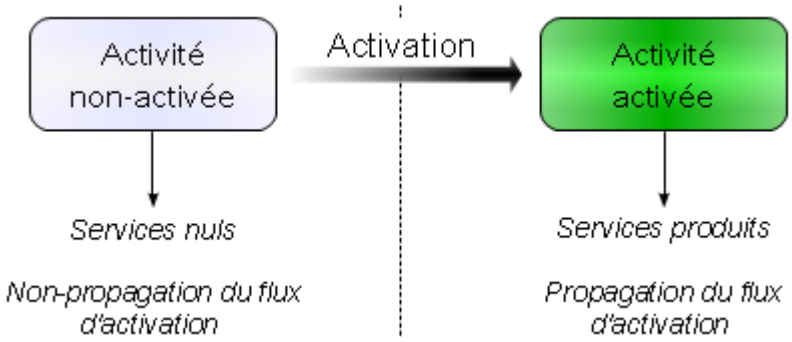


Fig. 4.3 - État d'activation d'une activité

Cela signifie que les valeurs des services des activités dépendent tout d'abord de l'état d'activation. D'après la section 3.3.5 du chapitre 2, une activité est activée si elle réunit toutes les conditions suivantes :

- tous ces liens de procédure sont reçus (uniquement pour les opérations) ;
- toutes les données nécessaires sont reçues.

Par défaut, ces conditions sont vraies si l'activité ne reçoit aucun des liens correspondants.

Définition : Principe d'activation

Pour toute activité $a \in Act_{MBFHA}$, *soit* $Activation(a)$: $dom(input(a)) \rightarrow Booléen$ *la formule booléenne décrivant l'activation de l'activité. Elle est définie par :*

$$Activation(a) = (val_p(a) \wedge (val_n(a) \neq e_{nul}))$$

2.5.3 Types, valeurs nominales et valeurs limites des services

La valeur d'un service dépend de sa signification dans le monde réel. Bien que chaque service soit unique il est possible de mettre en évidence trois caractéristiques fondamentales pour mieux les définir. La table 4.3 illustre ces trois attributs, rattachés aux liens sortants.

```
1    ENTITY Lien_out_MBFHA SUBTYPE OF
2    (Lien_data_MBFHA);
3          init : Act_MBFHA;
4          final : Data_MBFHA;
```

```
5          service_t : type_service;
6          service_nominal : efficacité;
7          service_pire : efficacité;
8      END_ENTITY;
```

Tab. 4.3 - Entité Express représentant les services des activités

Le type de service

Comme nous l'avons indiqué dans le chapitre 2, la majorité des activités ne délivre qu'un seul service, aussi appelé le service principal. Ceci vient du fait que le périmètre de définition des activités est déterminé par les concepteurs à partir des objectifs principaux de chaque activité de l'avion. Toutes les activités ont un service principal.

Cependant, comme les lois de la mécanique du vol relient entre elles la vitesse, l'altitude, la traînée et d'autres variables, certaines activités dédiées au contrôle de l'une de ces variables ont également des impacts sur d'autres variables. Elles offrent donc d'autres services, qui peuvent être regroupés en différentes familles, telles que :

- les services asymétriques, qui correspondent à une contribution d'une activité à partir de son fonctionnement asymétrique. Par exemple, le contrôle asymétrique des moteurs peut aider à contrôler la trajectoire, au dépend de la pleine poussée ;
- les services parasites, quand ils sont considérés comme des effets secondaires indésirables, qu'il faut palier à l'aide d'autres services. C'est le cas par exemple dans la phase de montée de l'avion, avec le train d'atterrissage, qui, avant d'être rentré, provoque une traînée importante s'opposant à l'objectif de montée et de vitesse.

A chaque service est alors identifié un type de service parmi un ensemble de type modélisé. La table 4.4 illustre un tel ensemble, suffisant pour distinguer les services principaux des services asymétriques.

```
1      TYPE type_service = ENUMERATION OF
2      (service_principal, service_asymétrique);
3      END_TYPE;
```

Tab. 4.4 - Ensemble des types de service en Express

La valeur nominale du service

Comme nous l'avons vu lors de la définition de l'efficacité, à la section 2.3.3, même en fonctionnement nominal, certains services contribuent plus que d'autres à la réalisation d'un résultat. Ce constat est visible même pour les services appartenant à la même famille. Notre proposition de modélisation consiste à définir une valeur nominale pour chaque service et dégrader peu à peu cette valeur en fonction des dégradations

des valeurs des données entrantes, de l'accroissement de la puissance des conditions environnementales et de la sévérité des pannes.

La pire valeur du service

D'après le paragraphe précédent, la borne supérieure d'efficacité pour un service est sa valeur nominale. Inversement, il faut également définir une borne inférieure. Bien souvent il s'agit de la borne inférieure de l'ensemble des valeurs d'efficacité, e_{min}. Mais parfois, les valeurs d'efficacité négatives n'ont aucun sens réel pour l'activité, auquel cas sa pire efficacité est la valeur nulle, e_{nul}.

Exemple. Pour la fonction qui régit la visibilité à travers la vitre du cockpit, la pire valeur d'efficacité est e_{nul}, qui correspond à l'incapacité de fournir une visibilité. Pour cette fonction, une efficacité négative signifierait que la fonction induit en erreur la visibilité des pilotes en fournissant une vue différente de la réalité. Ce serait possible à partir d'un écran numérique, mais pas par une simple vitre.

2.5.4 Sur le calcul de chaque service

D'après la section 2.2.2, la loi de comportement des activités détermine la valeur de chacun des services par rapport à un nombre indéfini de variables entrantes. Une simplification peut être réalisée en utilisant les quatre valeurs agrégées. Pour chaque service d'une activité $a \in Act_{MBFHA}$, la loi ne se base alors plus sur un nombre indéfini de variables, mais sur les cinq variables suivantes :

1. $c(a) \in C_{act}$: le mode de fonctionnement de l'activité ;
2. $val_n(a) \in E$: la valeur agrégée des données nécessaires ;
3. $val_u(a) \in E$: la valeur agrégée des données utiles ;
4. $val_p(a) \in Booléen$: la valeur agrégée des liens de procédure ;
5. $val_e(a) \in P_{env}$: la valeur agrégée des puissances environnementales.

Il est important de noter que cette proposition n'est valable que si les agrégations de données et de puissances environnementales sont possibles, c'est à dire dans des ensembles E et P_{env} où les opérations arithmétiques (addition, multiplication et division) sont définies. Nous pouvons par exemple nous placer dans le cas particulier que nous avons présenté : $E = [-100,100] \subset \mathbb{R}$ et $P_{env} = \mathbb{R}_+$.

Nous ne présentons pas les lois de comportement définies pour tous les types de services. Dans le chapitre 5, nous illustrons la loi permettant de déterminer les valeurs du service principal d'une activité, à l'aide du langage de modélisation AltaRica. Il est toutefois possible de dégager d'ores et déjà quelques propriétés des lois de comportement des activités.

1. Dans le mode de fonctionnement nominal, l'efficacité de tous les services est positive.
2. La valeur d'efficacité des services décroit lorsque l'efficacité des données nécessaires décroit, ou lorsque

l'efficacité des données utiles décroit ou lorsque la puissance environnementale croit.

3. La décroissance de l'efficacité des services est plus importante avec la décroissance de l'efficacité des données nécessaires qu'avec la décroissance de l'évaluation des données utiles.
4. Quelle que soit la puissance environnementale dégradante, si les données nécessaires et utiles ont une valeur d'efficacité positive, l'efficacité des services rendus restent positives.

3 Formalisation des modèles dégradés

Les modèles conceptuels, dédiés à l'analyse des risques, ont été définis au niveau de leur comportement nominal, c'est à dire sans pannes. Nous nous intéressons à présent au comportement, dit dégradé, de ces modèles en modélisant les pannes fonctionnelles. Ces dernières impactent en premier lieu les fonctions, mais elles se propagent à travers les liens jusqu'aux opérations et aux données échangées.

3.1 Impacts des pannes fonctionnelles sur le comportement des fonctions

En cas de panne, les fonctions ne sont plus à même de délivrer leurs services comme en fonctionnement nominal. Dès lors, leur contribution à la production de résultats devient moins importante. Les pannes fonctionnelles agissent donc sur l'efficacité des fonctions à fournir leurs services. A ce titre, il s'agit d'une variable importante de la loi de comportement de chaque activité. Dans la section 2.2.2, nous avons présenté l'ensemble des variations comportementales C_{act}. Dans la suite, nous proposons une définition plus précise de cet ensemble afin de prendre en compte les variations de comportement survenant du fait de pannes fonctionnelles.

3.1.1 Introduction à la modélisation des pannes fonctionnelles

La première étape de la formalisation des pannes fonctionnelles dans les modèles MB_{FHA} est la définition formelle des pannes elles-même.

Prise en compte des pannes fonctionnelles

Soit C_{act} l'ensemble des modes de fonctionnement potentiels des activités, contenant :

- $nominal \in C_{act}$ *est le mode de fonctionnement sans panne ;*
- $\forall p \in (C_{act} - \{nominal\})$, p *est une panne fonctionnelle.*

Soit c : $Activité \rightarrow C_{act}$, la fonction qui associe à chaque activité son mode de fonctionnement, telle que :

- $\forall f \in Fonction$, $c(f)$ *est le mode de fonctionnement de la fonction f, i.e. cette fonction est soit en fonctionnement nominal, soit subit une panne fonctionnelle ;*
- $\forall o \in Opération$, $c(o) = \{nominal\}$ *signifie que les opérations ne subissent aucune panne fonctionnelle.*

Remarque. Dans la section 2.2.2, nous avons indiqué que l'ensemble des variations comportementales C_{act} représente également les changements de mode de fonctionnement même dans le cadre nominal, sans pannes. La définition précédente exclut cette situation. Par choix, nous avons en effet considéré que les changements nominaux de certains modes de fonctionnement sont représentés par des scénarios de vol spécifiques. Les modes nominaux ne sont donc pas considérés comme des variables des modèles de sécurité. Ce choix est motivé par le fait que nous souhaitons limiter les variables de la loi de comportement des modèles MB_{FHA} aux pannes fonctionnelles et aux puissances des conditions environnementales.

La figure 4.4 ci-dessous, représente l'impact des pannes sur une fonction générique. Elle fait référence à la figure 4.1 illustrant la loi de comportement des activités.

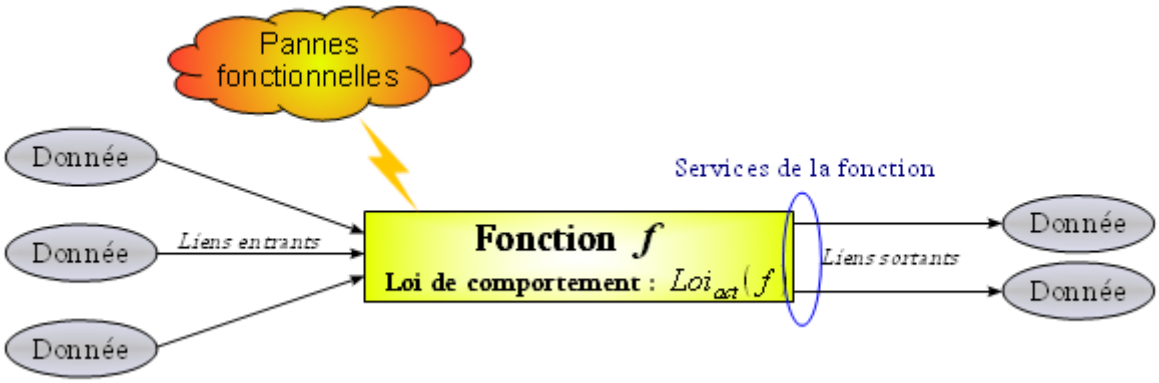


Fig. 4.4 - Impact de pannes fonctionnelles sur une fonction

3.1.2 Impacts sur la syntaxe et la sémantique des fonctions

Du fait des pannes fonctionnelles, la structure et la sémantique des modèles d'analyse des aspects dysfonctionnels est modifié. La table 4.5 introduit les évolutions de la structure. Nous constatons logiquement que les pannes fonctionnelles affectent en premier lieu les fonctions, à travers une définition des pannes potentielles et d'éventuelles capacités de détection et de recouvrement.

```
1   ENTITY Fct_MBFHA SUBTYPE OF (Act_MBFHA);
2   -- Liste des pannes fonctionnelles pouvant
3   affecter la fonction
4         pannes : SET OF mode_panne;
5
6   -- Présence ou non de moyens de détection
7   des entrées dégradées
8         detect : BOOL;
9   END_ENTITY;
```

Tab. 4.5 - Entité Express représentant les fonctions des modèles de sécurité

Sur le plan sémantique, la prise en compte des pannes modifie clairement la loi de comportement des fonctions. Toutefois, il est important de noter que la sémantique globale des modèles de sécurité n'est aucunement modifiée par la prise en compte des pannes ; les variations de comportement des fonctions ont en effet déjà été prises en considération.

Identification des pannes fonctionnelles

La notion de panne fonctionnelle a été définie précisément à la section 3.2.1 du chapitre 1 comme l'association d'un mode de panne (perte totale, perte partielle, fonctionnement erroné ou fonctionnement intempestif) à une fonction. Mais nous avons également vu qu'à l'exception de la perte totale, chacun des modes de pannes peut être raffiné pour décrire différentes conséquences directes sur la fonction impactée. De façon générale, à chaque fonction est associé son ensemble de modes de panne qu'elle peut subir. Il existe des modes de panne qui n'ont aucun sens pour certaines fonctions. L'identification de l'attribut « pannes » des fonctions permet de limiter la modélisation aux seules pannes fonctionnelles qui ont du sens.

Le type « mode_panne » est une liste de tous les modes de panne qu'il est possible de modéliser. Cette liste n'est pas figée au niveau des concepts des modèles de sécurité. La section 2.1.4 du chapitre 5 présente une proposition de modélisation concrète des pannes à l'aide du langage AltaRica.

Les capacités de détection des activités

La loi de comportement des activités montre que les dégradations de la valeur d'efficacité des liens entrants et triggers d'une activité, ayant pour origine des pannes fonctionnelles ou des conditions environnementales, se propagent à travers cette activité jusqu'à ses services. Mais cette modélisation est peut-être trop pessimiste car elle occulte toutes les capacités que pourraient avoir certaines activités pour détecter et recouvrir les dégradations.

Dans les modèles M_{MFHA}, nous considérons que la détection des situations dégradées et leur recouvrement par des opérations spécifiques sont un cas particulier de prise de décision (voir la section 3.3.5 du chapitre 2).

Mais les fonctions elles-même disposent de capteurs et autres outils de sécurité pour identifier de potentielles dégradations et éventuellement les corriger. Toutes les fonctions ne sont pas équipées de tels moyens, d'autant plus qu'ils sont souvent coûteux autant au développement qu'à la production. Ainsi, seules certaines fonctions, dites à risques, sont ainsi instrumentées. Il semble alors que la décision d'instrumenter une fonction pour qu'elle puisse détecter et recouvrir les situations dégradées ait pour origine l'analyse des risques. Mais en pratique, les architectures sont souvent suffisamment définies pour que l'analyse des risques puisse prendre en compte certaines capacités de détection. Les problématiques de détection et de recouvrement des situations dégradées dans les modèles de sécurité prédictifs ont fait l'objet de plusieurs études et sont toujours des questions largement ouvertes aujourd'hui, surtout au niveau de la couche fonctionnelle ([Humbert, 2008]).

Dans le cadre de notre approche, nous distinguons deux cas de détection par les fonctions.

- Détection du fonctionnement dégradé : nous considérons que les fonctions n'ont pas les moyens de détecter leurs propre défaillances car leurs pannes peuvent tout à fait impacter leurs capacités de détection tout autant que leurs services. Dans le pire cas, ces moyens de détection deviennent inopérants.
- Détection d'entrées dégradées : si une fonction est en mode de fonctionnement nominal, alors nous pouvons considérer que les moyens de détection qu'elle peut posséder lui permettent d'identifier les dégradations sur ses données d'entrée. Cette caractéristique fait donc partie de son comportement intrinsèque c'est à dire de la définition de la loi de comportement des activité. Elle est identifiée dans la syntaxe des fonctions par l'attribut « detect ».

Prise en compte des pannes dans la loi de comportement des fonctions

Les services d'une fonction ne dépendent pas seulement des valeurs des liens entrants, mais sont également évalués à partir du comportement interne de la fonction qui représente la façon dont la fonction transforme ses entrées pour produire ses services. Or, c'est justement ce comportement qui est dégradé par les pannes fonctionnelles. Nous avons déjà évoqué la difficulté à définir précisément la loi de comportement des activités. Nous en dégageons toutefois une propriété supplémentaire (voir la section 2.5.4).

- Les valeurs d'efficacité des services d'une fonction subissant une panne sont toujours inférieures aux valeurs d'efficacité de ces mêmes services lorsque la fonction est dans état nominal de fonctionnement.

3.1.3 Les types de pannes fonctionnelles

La section 3 du chapitre 1 a montré que le moyen de décrire des pannes sur la couche fonctionnelle est d'étudier les modes de pannes fonctionnelles (perte totale, perte partielle, fonctionnement erroné et fonctionnement intempestif). S'appuyer sur ces modes de pannes permet aux modèles dégradés M_{FHA} d'être au plus près de l'analyse des risques.

La figure 4.5 illustre tous les modes de fonctionnement d'une fonction donnée. Les transitions de l'automate, qui permet d'atteindre ces différents modes, sont des pannes fonctionnelles. Nous constatons que les différents états dépendent de l'état d'activation, tel qu'il a été présenté sur la figure 4.3 de la section 2.5.2.

Remarque. Les fourchettes de valeurs des services présentées sur la figure 4.5 sont issues de l'étude réalisée à la section 3.2 qui suit. Nous notons la valeur nominale des services :

$$\forall f \in Fonction,\ \forall l \in output(f) / l \in Lien\_sortant,\\ val_{nominal}(l) = Loi_{act,l}(nominal, val(input(f)))$$

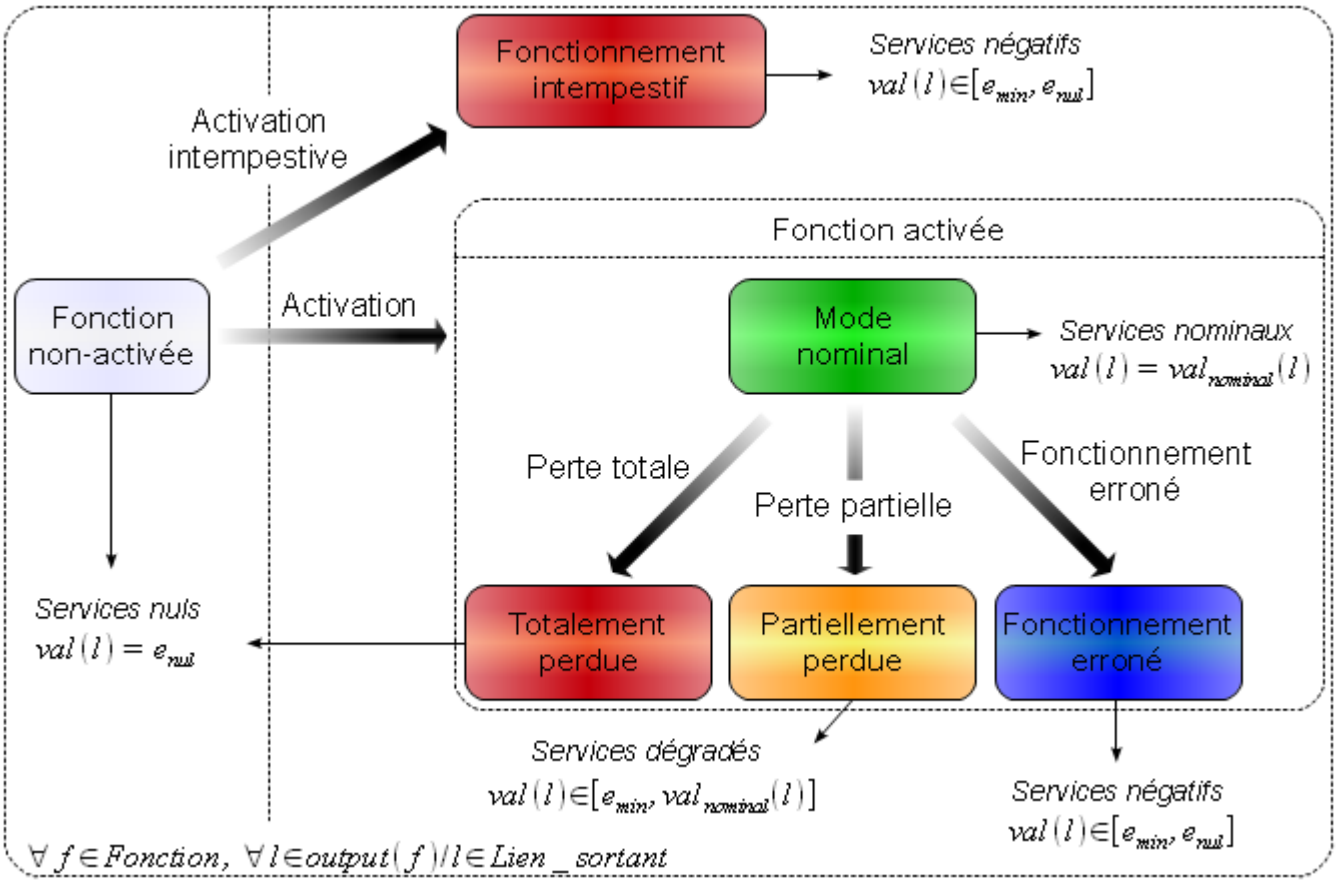


Fig. 4.5 - Automate des pannes fonctionnelles

Lorsqu'elle est non activée, une fonction peut malgré tout subir une activation intempestive, qui peut éventuellement impacter négativement certains résultats du modèle. Dans la section 3.2.4 nous montrons que seuls les services contribuant aux résultats attendus sont modélisés.

La fonction peut également être activée normalement, auquel cas, initialement, son fonctionnement est nominal. Mais ce fonctionnement nominal peut être dégradé par différents types de pannes, construit sur la base des modes de pannes de perte totale et partielle et du fonctionnement erroné. Les impacts de ces trois modes de panne sur les services sont présentés aux sections 3.2.1, 3.2.2 et 3.2.3 respectivement.

Nous pouvons trouver dans la littérature plusieurs automates des pannes proches de celui de la figure 4.5. Dans ses modèles d'analyse des commandes de la gouverne de direction, [Bernard et al., 2002] a mis en place un automate à trois états : correct, erroné et perdu. L'état correct peut conduire aux états erronés et perdu et l'état erroné lui même peut être dégradé davantage jusqu'à être perdu. Dans notre proposition, nous avons choisi de ne pas modéliser de transition entre les états dérivés de pannes. Pourtant, il pourrait sembler logique que la perte totale d'une fonction puisse également découler de sa perte partielle, comme une aggravation. La justification de ce choix est présentée dans la section 4.3 de ce chapitre, consacrée à la recherche automatique des causes d'une situation redoutée à l'aide des modèles de sécurité.

Dans [Castel et al., 2001], les auteurs présentent un automate de pannes dédié à modéliser des équipements. Il met en évidence des caractéristiques qui ont leur équivalent au niveau fonctionnel, telles que les capacités de détection d'une panne ou de recouvrement d'une panne. Nous montrons dans la section 3.1.2 comment ces caractéristiques sont intégrées dans les modèles dégradés.

3.2 Effets des pannes fonctionnelles sur les services des fonctions

Les paragraphes suivants décrivent comment les différents modes de pannes considérés impactent les services des fonctions. Comme nous l'avons vu dans le chapitre 1, certains modes de pannes doivent être raffinés pour différencier les divers comportements de pannes. Cela peut donner lieu à différents niveaux de détail dans la modélisation.

Dans la suite de cette section, nous nous intéressons aux pannes d'une fonction notée f.

3.2.1 La perte totale

Le premier mode de panne étudié au cours de l'analyse des risques est la perte totale fonctionnelle. Elle se traduit par le fait que la fonction concernée n'est plus en mesure de fonctionner. La fonction n'est donc plus en état de délivrer le moindre de ses services et ce quelque soit le taux d'efficacité de ses entrées et la puissance dégradante des conditions environnementales qui l'impactent. Ce mode de panne ne présente aucune ambiguïté et c'est d'ailleurs la raison pour laquelle de nombreuses modélisations ont déjà été réalisées chez différents industriels pour faciliter les analyses des pertes totales ([Louis et al. 2012]).

Le tableau 4.6 présente l'effet de la panne fonctionnelle sur le statut de la fonction puis sur ses services :

Panne fonctionnelle	Effets sur les services de la fonction	Valeur des services
Perte totale	Services non délivrés	$\forall l \in output(f) / l \in Lien\_sortant,$ $val(l) = e_{nul}$

Tab. 4.6 - Modélisation de la perte totale d'une fonction

3.2.2 La perte partielle

Dans le cas de la perte partielle, la fonction concernée ne fonctionne que partiellement, c'est à dire qu'elle renvoie des services, mais qui sont produits de façon incomplète ou avec une moindre performance (contrainte de temps non respectée ou légère imprécision dans les résultats par exemple). Il est tentant de dire que le pire cas de la perte partielle semble être celui où tous les services sont nuls, nous ramenant ainsi à la modélisation de la perte totale. Mais, comme nous l'avons vu dans le chapitre 1 au paragraphe 3.2.1, les pires cas de la perte partielle sont en réalité ceux qui induisent des asymétries, impactant ainsi de façon négative certains résultats, tout en en dégradant d'autres. Lorsqu'une fonction a plusieurs asymétries possibles, alors il faut décrire une perte partielle distincte pour chacun des différents cas. Lorsque la perte

partielle fonctionnelle ne conduit à aucune asymétrie, alors le pire cas de la perte partielle est effectivement la perte totale. En première approche, il peut ne pas être pertinent de modéliser la perte partielle.

Le tableau 4.7 formalise les effets de la perte partielle. Notez que les valeurs données pour le vecteur des services ne s'appliquent que si seul le *Statut* varie, c'est-à-dire que le degré d'efficacité des entrées est maximal et que la puissance environnementale dégradante est nulle.

Panne fonctionnelle	**Effets sur les services de la fonction**	**Valeur des services**
Perte partielle sans asymétrie	Tous les services sont positifs mais légèrement inférieurs à leur valeur nominale.	$\forall l \in output(f) / l \in Lien\_sortant,$ $val(l) \in [e_{nul}, val_{nominal}(l)]$
Perte partielle avec asymétrie	Certains services s'opposent à la bonne réalisation de leur résultat. D'autres services sont légèrement dégradés, mais positifs.	$\forall l \in output(f) / l \in Lien\_sortant,$ $val(l) \in [e_{min}, val_{nominal}(l)]$ $\exists l \in output(f) / l \in Lien\_sortant,$ $val(l) \in [e_{min}, e_{nul}[$

Tab. 4.7 - Modélisation de la perte partielle d'une fonction

3.2.3 Le fonctionnement erroné

La fonction, qui a un fonctionnement erroné, transmet ses services, mais de façon incorrecte par rapport à ce qui est attendu. Le pire cas est lorsque la fonction considérée transmet ses services avec une efficacité négative pour s'opposer à la bonne réalisation des résultats auxquels elle contribue. L'intensité de ces contributions négatives est variable ce qui peut amener à considérer différents types de fonctionnement erroné. Parfois l'efficacité fonctionnelle négative n'a aucun sens pour certains services fonctionnels. Dans ce cas, nous constatons en pratique que les pires effets du fonctionnement erroné sont similaires à ceux qui surviennent avec la perte totale ou la perte partielle asymétrique de la fonction. Alors, dans le cadre de l'analyse des risques, de tels scénarios de fonctionnement erroné sont généralement liés aux scénarios de perte totale et/ou de perte partielle de la même fonction. Dans ce cas, le fonctionnement erroné n'est pas pertinent à modéliser.

Exemple. Considérons la fonction de contrôle de la roulette de direction, « Guider avec la roulette de direction ». Le fonctionnement erroné de la roulette peut, dans le pire cas, la conduire à tourner dans le sens opposé à celui qui est souhaité, aggravant toute déviation de trajectoire. Dans ce cas, le service de la fonction a clairement une efficacité négative par rapport au résultat « trajectoire contrôlée ». Mais pour la fonction « Fournir la poussée » par exemple, l'efficacité négative n'a aucun sens au regard de la production de la poussée motrice : au pire, cette production est nulle. Alors, le fonctionnement erroné provoque dans le pire une perte totale ou une perte partielle avec asymétrie ; des pannes déjà considérées à partir d'autres modes de pannes. En effet, si nous analysons cette fonction par rapport à son autre

service, contribuant au résultat « trajectoire contrôlée », l'efficacité négative de ce service existe et est modélisée par la perte partielle asymétrique de la fonction.

Le tableau 4.8, ci-dessous, se place dans la première configuration où le fonctionnement erroné conduit à des contributions négatives à la production des services.

Panne fonctionnelle	Effets sur les services de la fonction	Valeur des services
Fonctionnement erroné	Les services de la fonction contribuent négativement à la production des résultats.	$\forall l \in output(f) / l \in Lien\_sortant,$ $val(l) \in [e_{min}, e_{nul}]$

Tab. 4.8 - Modélisation du fonctionnement erroné d'une fonction

3.2.4 Le fonctionnement intempestif

Le fonctionnement intempestif traduit une activation imprévue d'une fonction, qui a pour effet de perturber le bon fonctionnement courant. Nous identifions deux situations distinctes.

1. Souvent, la fonction, qui s'active intempestivement, n'a aucun impact sur les objectifs du scénario de vol analysé. Cette situation est identifiée lorsque la fonction ne participe à aucun des résultats présents dans le modèle. Dans ce cas, il n'est pas pertinent de modéliser la fonction et son mode intempestif.
2. Parfois la fonction contribue intempestivement à certains résultats du modèle. Le cas où elle y contribue positivement n'est pas intéressant car cela signifierait qu'au lieu de dégrader la situation, la panne la renforcerait. Les situations pertinentes sont celles où la fonction s'oppose à la bonne réalisation des objectifs d'un ou de plusieurs de ses résultats, qui sont représentés dans le modèle. Cette opposition peut varier en intensité et être au mieux une non-contribution (efficacité e_{nul}). Plusieurs types de fonctionnements intempestifs peuvent alors être identifiés.

Exemple. Lors du décollage, le fonctionnement intempestif de la fonction de contrôle des freins de roue, « Freiner avec les roues », a un impact très fort sur le contrôle de la vitesse de l'avion. Puisque l'objectif pendant le décollage est d'accélérer l'avion et que la panne tend au contraire à le freiner, l'activation intempestive de la fonction s'oppose à la bonne réalisation de l'objectif. Mais la fonction « Freiner avec les roues » participe également au contrôle de la trajectoire de l'avion grâce au freinage différentiel. Si le freinage s'oppose avec vigueur à l'objectif d'accélération, alors le freinage doit être maximal sur toutes les roues. Cela signifie que les écarts de trajectoire dus au freinage différentiel sont nuls. Mais un autre fonctionnement intempestif peut être modélisé : un freinage différentiel important du fait d'un freinage asymétrique. Dans ce dernier cas, l'opposition à l'objectif d'accélération est moins intense.

Par définition, le fonctionnement intempestif modifie le principe d'activation, en plus de propager des valeurs de services négatifs.

Sémantique : Principe d'activation d'une fonction soumise à un fonctionnement intempestif

Soit $intemp: C_{act} \rightarrow Booléen$ *la formule qui teste la présence d'un déclenchement intempestif par rapport aux pannes fonctionnelles.*

$\forall f \in Fonction,\ Activation(f) = (val_n(f) \neq e_{nul}) \vee intemp(c(f))$.

Cette précision sur l'activation des fonctions ne remet pas en cause la sémantique générale des modèles MB_{FHA}. La nouvelle formule d'activation traduit le fait qu'une fonction s'active soit normalement (condition $(val_n(f) \neq e_{nul})$ soit intempestivement si la condition $intemp(c(f))$ est satisfaite.

Le tableau 4.9, ci-dessous, présente la valeur des services d'une fonction activée intempestivement.

Panne fonctionnelle	Effets sur les services de la fonction	Valeur des services
Fonctionnement intempestif	Les services de la fonction s'opposent à un certain nombre de résultats et sont nuls pour les autres.	$\forall l \in output(f) / l \in Lien\_sortant,$ $val(l) \in [e_{min}, e_{nul}]$

Tab. 4.9 - Modélisation du fonctionnement intempestif d'une fonction

4 Principes d'analyse des modèles dégradés

L'un des principaux avantages des modèles formels est de rendre possible le calcul de propriétés sur les modèles. Dans le cadre de l'analyse des risques, nous avons identifié plusieurs besoins dont, en particulier : assister l'évaluation des conséquences des pannes fonctionnelles sur l'avion et aider à l'identification de combinaisons pertinentes de pannes fonctionnelles. Ces deux problématiques font l'objet des sections 4.2 et 4.3 respectivement.

4.1 Lien entre la modélisation et les analyses

Il est important de noter que la construction d'un modèle doit être faite par rapport aux analyses que les utilisateurs souhaitent faire de ce modèle. Il est donc nécessaire d'adapter les modèles à leurs exploitations. Les deux sources d'adaptation identifiées sont le niveau de détail dans la modélisation et la définition de situations redoutées observées.

4.1.1 Définition d'un niveau de détail

La sémantique que nous avons proposée contient deux sources d'imprécision, présentées ci-dessous.

1. Les ensembles E, P_{env} et C_{act} sont encore indéfinis, bien que, pour les deux premiers, nous avons proposé des analogies avec des sous-ensembles de $\mathbb{R}$ et que, pour le dernier, nous avons proposé une construction à partir des modes de panne de l'analyse des risques.
2. L'évaluation des modèles par la fonction σ est elle aussi indéfinie car la loi de comportement des activités et plus particulièrement le comportement des activités ne peut pas être défini dans le cadre général.

Au niveau des concepts, ces imprécisions dans la définition est volontaire, permettant ainsi plusieurs concrétisations des concepts. Ainsi, il est possible de construire des modèles avec différents niveaux de détails, adaptés aux analyses souhaités.

Exemple. Si le besoin est d'évaluer précisément des effets de pannes mais de façon plus grossière les conditions environnementales, alors il est possible d'évaluer l'efficacité avec $E=[-100,100]\subset\mathbb{R}$ et la puissance environnementale avec $P_{env}=Booléen$. Si le besoin est d'étudier exclusivement la perte totale alors l'ensemble C_{act} se trouve réduit à $C_{act}=\{nominal, perte\_totale\}$.

4.1.2 Définition des situations redoutées

La notion de **situation redoutée** rappelle la définition des événements redoutés de l'approche « *Model-Based Safety Assessment* » (voir la section 4.2 du chapitre 2). Les événements redoutés représentent, dans les modèles MBSA, les conditions de pannes (« *Failures Conditions* », voir le chapitre 1) déterminées grâce à l'analyse des risques. Mais l'analyse des risques elle-même ne dispose pas d'une définition claire des situations redoutées déterminées par une autre analyse du processus de sécurité.

Par conséquent, le choix des situations redoutées observées est laissé aux analystes de la sécurité chargés de l'analyse des risques. Ils peuvent ainsi maîtriser les analyses réalisées sur les modèles. Nous proposons deux pistes pour exprimer des situations redoutées pertinentes à analyser.

Situations de dépassement de valeurs seuils des objectifs de vol

Comme nous l'avons vu dans la section 4.2.1, les résultats des données d'un modèle et plus particulièrement les résultats des objectifs de vol montrent les conséquences des pannes et des conditions environnementales sur les marges de sécurité et les capacités de l'avion à réaliser ses phases de vol. Comme ces résultats sont évalués à l'aide des valeurs d'efficacité (un ensemble ordonné de valeur), il est possible de définir des valeurs limites pour chaque résultat, en dessous desquelles les marges de sécurité et/ou les capacités de vol ne sont plus suffisantes. En effet, une valeur d'efficacité en dessous du seuil signifie que les objectifs ne sont pas suffisamment atteints.

Définition : Situation redoutée modélisée par des seuils d'efficacité

Soit $seuil : Donnée \rightarrow E$ *la fonction qui à chaque donnée associe une valeur seuil.*

Nous définissons également le vecteur des seuils d'un sous-ensemble des données d'un modèle par le produit cartésien : $\forall D \subseteq Donnée$, $Seuil(D) = \prod_{d \in D} seuil(d)$.

Alors une situation redoutée est une formule booléenne définie sur un modèle par :

$S_{MB_{FHA}} : E^{card(Donnée)} \rightarrow Booléen$ *, tel que,* $\forall Seuil(Donnée)$ *:*

$S_{MB_{FHA}}(Seuil(Donnée)) = \{\exists d \in Donnée / r(d) \leqslant seuil(d)\}$ *, signifiant que la situation redoutée est atteinte (vraie) si une des données du modèle a un résultat en dessous de son seuil.*

En définissant un seuil d'efficacité pour chaque donnée d'un modèle, les analystes de la sécurité construisent directement des situations redoutées. La pertinence de ces situations est en revanche laissée à l'appréciation des analystes de la sécurité et dépendent des seuils choisis.

En pratique, les seuils ne sont définis que sur un sous-ensemble des données du modèle. Pour définir des valeurs seuil sur la totalité des données, il faut affecter une valeur de seuil sans impact au sous-ensemble complémentaire des données. Cette valeur sans impact est la valeur d'efficacité minimale.

Situations où des pannes perturberaient leur propre séquence d'actions correctives

Dans le chapitre 1, nous avons indiqué que l'étude des scénarios de pannes s'intéresse également aux moyens de détection des pannes et aux moyens d'actions correctives pouvant être réalisées par l'équipage. Une situation redoutée qui nous apparaît très pertinente est le cas où les actions correctives envisagées sont elles-mêmes perturbées par le scénario de panne considéré. La mise en évidence de cette problématique a été initiée dans le cadre de l'étude de cas « *Ensure Safe Flight and Landing* » (voir la section 3.3.1 du chapitre 2) lié au projet ALFA, où les concepteurs des architectures fonctionnelles souhaitent déterminer avec précision les fonctions nécessaires pour assurer un vol et un atterrissage en toute sécurité. En particulier, ils souhaitent identifier le sous-ensemble des fonctions utilisées dans le cadre des actions correctives.

Notre approche peut répondre à ce besoin, et même avec davantage de précision : au lieu d'identifier les fonctions dont la défaillance peut être à l'origine d'une séquence d'actions correctives et qui est également exécutée dans le cadre de la réalisation de cette série d'actions, nous proposons d'identifier les pannes fonctionnelles à l'origine d'une séquence d'actions correctives et qui empêchent la bonne réalisation de cette séquence d'actions.

Exemple. Dans le chapitre 7 de ce mémoire, nous présentons notre cas d'étude sur la phase de décollage. L'une des principales particularités de cette phase est la possibilité pour les pilotes d'interrompre le décollage à partir du moment où la situation ne leur semble pas sûre. Par exemple, si un mauvais contrôle de la trajectoire est détecté par les pilotes pendant le décollage, ces derniers doivent décider (ce cas est clairement décrit au niveau des procédures de pilotage) de freiner l'avion tout en le maintenant autant que possible sur l'alignement de la piste jusqu'à l'arrêt de l'appareil. Mais comment être certain qu'un mauvais

contrôle de la trajectoire ne rende pas impossible l'arrêt du décollage en restant sur la piste? Dans le cadre de l'étude de cas ESFL, la fonction « Guider avec la roulette de direction » a été identifiée comme à la fois nécessaire au décollage, dont la défaillance peut conduire à arrêter le décollage et contributrice à la réussite de cet arrêt du décollage dans de bonnes conditions. Nous pouvons identifier les pannes fonctionnelles de la fonction « Guider avec la roulette de direction » pouvant conduire à arrêter le décollage et analyser les conséquences de ces pannes fonctionnelles lors de la phase d'arrêt du décollage. La fonction est certes contributrice dans les deux phases de vol, mais les contextes opérationnels étant différents, sa défaillance ne dégrade peut-être pas autant les résultats dans une phase que dans l'autre. Dans ce cas précis, traité plus en détail dans le chapitre 7, les pilotes disposent du freinage différentiel par les roues pour compenser en partie la défaillance de la roulette de direction pendant la phase d'arrêt du décollage.

Comme l'a illustré l'exemple précédent, cette approche fait intervenir deux phases de vol ou du moins une séquence d'actions supplémentaires au sein d'une phase de vol.

1. Cas de deux phases de vol : l'approche fait alors intervenir deux modèles, le premier dans lequel la panne survient et amène à considérer la seconde phase de vol, non nominale, pour retourner dans une situation sûre. Pour la première phase de vol, la situation redoutée choisie correspond aux conditions menant à la seconde phase de vol. Ces conditions traduisent des capacités de vol dégradées. La première situation redoutée peut donc être définie à l'aide de seuils d'efficacité. Pour la seconde phase de vol, la situation redoutée choisie correspond à l'incapacité de réaliser convenablement cette phase. Là aussi, cette situation redoutée peut être définie avec des seuils d'efficacité sur les résultats du second modèle. Ce cas fait donc intervenir deux évaluations de situations redoutées.
2. Cas d'une séquence d'actions supplémentaires : le choix de réaliser une séquence d'actions supplémentaires est une prise de décision (voir la section 3.3.5 du chapitre 2). Nous avons vu que la façon la plus courante de modéliser la prise de décision dans les modèles ALFA est de réaliser deux modèles correspondant aux deux scénarios de vol différents, le premier sans les actions supplémentaires du fait de la détection de la panne et le second avec. Nous nous retrouvons alors de nouveau avec l'évaluation de deux situations redoutées, la première pour évaluer la capacité à réaliser la phase de vol et la seconde pour évaluer la capacité à réaliser les actions correctives.

Dans tous les cas, nous travaillons sur le principe de seuils d'efficacité sur les résultats d'un modèle.

4.2 Calcul des conséquences des pannes fonctionnelles

Dans la section 3.4.1 du chapitre 2, nous avons montré que l'étude, du point de vue de la sécurité, des dépendances fonctionnelles permet d'identifier les chemins de propagation des pannes fonctionnelles. Nos modèles de sécurité représentent ces dépendances fonctionnelles. Ils sont donc dédiés à l'étude de la propagation des pannes fonctionnelles.

4.2.1 Identification des conséquences étudiées lors d'une analyse des risques

Comme nous l'avons vu dans la section 3.2.3 du chapitre 1 et plus particulièrement dans la table 1.2, qui met en perspective la criticité des scénarios de pannes par rapport à leurs conséquences ([CS-25 1309]), les conséquences des scénarios de panne sont mesurées à plusieurs niveaux, tels que :

- la santé des acteurs humains (équipage, passagers, personnes au sol etc.) ;
- le confort des équipages et des passagers ;
- les marges de sécurité et les capacités de l'avion à assurer un vol et un atterrissage sûr ;
- la charge de travail de l'équipage ;

Dans nos modèles qui décrivent la propagation des pannes fonctionnelles, nous pouvons mesurer les marges de sécurité et les capacités de l'avion à assurer un vol et un atterrissage sûr, grâce aux résultats. En effet, par définition, les résultats d'un modèle sont les taux d'aboutissement des objectifs à atteindre au cours de la phase de vol considérée. En particulier, les résultats des objectifs de vol, qui forment le sous-ensemble des données caractérisant fondamentalement chaque phase de vol, sont des valeurs prépondérantes pour déterminer la bonne conduite d'un vol.

Exemple. Dans l'exemple du contrôle de la trajectoire au sol pendant le décollage, plusieurs données ont été identifiées, tels que « commande de guidage » et « ordre de guidage ». Ces objectifs intermédiaires sont importants dans le cadre du contrôle de la trajectoire, mais la finalité de cet extrait de modèle est la capacité à contrôler la trajectoire, correspondant à l'objectif de vol « trajectoire contrôlée ».

Si les conséquences des pannes sur une phase de vol peuvent être mesurées à partir des résultats du modèle correspondant, tous les objectifs n'ont pas le même impact sur la sécurité. Une telle mesure des conséquences est donc sujette à interprétation par les spécialistes de l'analyse des risques. A ce titre, les modèles de sécurité offrent une assistance aux analystes pour mesurer les conséquences de scénarios de pannes fonctionnelles sur les objectifs réalisés au cours d'une phase de vol mais ils ne permettent pas d'en déduire directement leur criticité.

4.2.2 Principe de calcul des conséquences

Une fois le modèle structurel formé et les lois de comportement des activités définies, la sémantique des modèles MB_{FHA} ne présente plus que deux variables :

- $c(Activité)$, qui est le vecteur composé de l'état de fonctionnement de toutes les activités du modèle. Étant donné que l'état de fonctionnement des opérations est défini strictement nominal, seul $c(Fonction)$ varie ;
- $c(Environnement)$, qui est le vecteur composé de la puissance environnementale de toutes les conditions environnementales du modèle.

En faisant varier ces deux paramètres, nous fixons des états de pannes fonctionnelles sur certaines fonctions et des conditions environnementales particulières. Nous sommes alors en mesure de modéliser des scénarios de pannes fonctionnelles incluant des conditions environnementales.

Or, la sémantique des modèles MB_{FHA} est également définie par une loi de comportement à l'échelle d'un modèle : $\sigma : (C_{act})^{card(Activité)} \times (P_{env})^{card(Environnement)} \rightarrow E^{card(Donnée)}$, telle que :

- soit $R(Donnée) = \prod_{d \in Donnée} r(d)$ le vecteur composé des résultats de toutes les données du modèle, alors
- $R(Donnée) = \sigma(c(Activité), c(Environnement))$ est la fonction déterminant la valeur de tous les résultats du modèles par rapport aux comportements des activités, à leur état de fonctionnement et aux puissances des conditions environnementales.

Ainsi, par définition, la sémantique de nos modèles de sécurité relie directement les pannes et les impacts de conditions environnementales aux résultats et donc à la mesure des conséquences des pannes, du point de vue des marges de sécurité et des capacités à voler et atterrir de façon sûre. Les analystes de la sécurité peuvent s'appuyer sur la définition de situations redoutées au sein des modèles pour établir des points de comparaison.

4.3 Calcul des causes de situations redoutées

Le calcul des conséquences s'apparente à une simulation de nos modèles de sécurité. A l'inverse, la recherche d'états de panne à l'origine d'un ensemble de conséquences donné fait appel à un parcours des modèles selon les méthodes du model-checking. Nous nous appuyons pour cela sur la notion de situation redoutée en déterminant toutes les combinaisons de pannes fonctionnelles et de conditions environnementales qui conduisent à une situation redoutée définie.

La recherche des causes suit les quatre principes suivantes.

1. Il faut définir une situation redoutée en fixant des seuils d'efficacité aux résultats du modèle. Il suffit qu'un seul de ces seuils soit franchit pour que la situation redoutée soit atteinte.
2. Une situation initiale doit ensuite être choisie. Généralement, la situation initiale sélectionnée est le mode de fonctionnement nominal, sans pannes ni dégradations par des conditions environnementales. Toutefois, la situation initiale peut également être un état global du modèle où des fonctions sont d'ores et déjà en panne et/ou des conditions environnementales sont présentes.
3. La situation peut évoluer, à partir de la situation initiale, par l'ajout de nouvelles pannes fonctionnelles et la présence ou l'aggravation de conditions environnementales. La loi de comportement du modèle permet de calculer les résultats de toutes les données du modèle. En comparant ces valeurs aux seuils de la situation redoutée précédemment définie, nous déduisons si les nouvelles combinaisons de pannes et de conditions environnementales conduisent à la situation redoutée ou non.

4. Par model-checking, toutes les situations peuvent être automatiquement testées à partir de la situation initiale. Ainsi, la liste de toutes les combinaisons de pannes et de conditions environnementales conduisant à la situation redoutée peut être dressée. Ces combinaisons sont les causes de la situation redoutée, également appelée « implicant » (voir la section 4.1.1 du chapitre 2).

Remarque 1. L'ensemble des coupes minimales correspondant à la situation redoutée peut être déterminé à partir de l'ensemble des implicants selon le principe présenté à la section 4.1.1 du chapitre 2. Si la situation initiale elle-même conduit à la situation redoutée, alors l'ensemble des coupes minimales est l'ensemble vide. Ce cas n'est évidemment pas pertinent.

Remarque 2. L'analyse des risques consiste en l'étude de scénarios de pannes et non de l'évolution potentielle d'une panne fonctionnelle au sein d'une même fonction. C'est pourquoi la recherche des implicants doit exclure le sous-ensemble des fonctions initialement soumises à des pannes. C'est également la raison pour laquelle l'automate des pannes présenté à la section 3.1.3 et plus particulièrement sur la figure 4.5 ne représente pas l'évolution d'une perte partielle ou d'un fonctionnement erroné vers une perte totale.

5 Bilan

Dans ce chapitre, nous avons présenté les concepts de la nouvelle approche de modélisation basée sur les modèles ALFA. Ces concepts intègrent une représentation des dépendances fonctionnelles, des pannes fonctionnelles considérées classiquement lors de l'analyse des risques et des impacts de conditions environnementales. Nous avons montré que de tels modèles permettent l'analyse des conséquences des pannes fonctionnelles du fait de leur propagation à travers les différentes dépendances fonctionnelles.

Lors des phases de conception préliminaires, les éléments manipulés sont abstraits et sont définis sans présumer des choix réalisés lors de la conception des architectures matérielles. A notre connaissance, des approches similaires existent dans différents secteurs industriels, tels que l'aéronautique ([Chaudemar 2012]), le ferroviaire ([Belmonte 2009]) ou encore l'automobile ([Sandberg et al. 2010]), mais avec une formalisation mathématique moins détaillée des concepts. En cela, l'introduction de la notion d'efficacité et des lois de comportement des activités a été fondamentale, en permettant le calcul de toutes les variables des modèles de sécurité selon les états de pannes fonctionnelles et de conditions environnementales. L'efficacité opérationnelle et fonctionnelle a fait l'objet de la publication suivante : [Maitrehenry et al. 2012].

Dans le chapitre suivant nous proposons une concrétisation des concepts à l'aide du langage AltaRica.

5.1 Comparaison avec d'autres approches

Nous pouvons trouver dans la littérature quelques initiatives de modélisation ayant inspiré notre proposition de modélisation.

Dans [Chaudemar et al., 2009] et [Chaudemar 2012], J.-C. Chaudemar présente un modèle AltaRica construit à partir d'une architecture d'un drone aérien (UAV, « *Unmanned Aerial Vehicle* »), en trois couches (opérationnelle, fonctionnelle et physique), conforme à la modélisation des systèmes socio-techniques (voir la section 3.1.3 du chapitre 2). L'objectif de ces travaux de modélisation est de développer une activité de la famille des analyses de risques à partir de modèles AltaRica. Une telle analyse est en effet nécessaire pour prétendre insérer un drone dans le trafic aérien ; à ce jour, aucune analyse des risques n'a été certifiée pour les drones d'où l'ambition dès le départ de concevoir une analyse basée directement sur des modèles formels. Nous listons quelques points communs et différences entre nos deux approches de modélisation.

1. Les séquences d'exécution des fonctions ne sont pas explicites dans la modélisation proposée par J.-C. Chaudemar. En effet, ses séquences d'exécution sont représentées par des nœuds de la couche opérationnelle, liés à des fonctions de la couche fonctionnelle et pilotant l'activation des fonctions par rapport à un déroulement conditionné par des événements et des transitions. Il s'agit de nœuds faisant de la supervision et de la gestion de phases de vol. Cette approche ne modélise donc pas dans le détail les activités humaines ; ceci ne perturbe pas l'étude des modèles du point de vue de l'analyse fonctionnelle de la sécurité. Nous pouvons d'ailleurs noter que dans notre proposition, les opérations sont des nœuds très simples ayant peu d'impact sur le comportement général du modèle.
2. Les pannes fonctionnelles modélisées par J.-C. Chaudemar sont la perte et le fonctionnement erroné. Les nœuds représentant les fonctions retournent leur état de fonctionnement, qui est interprété par les fonctions s'appuyant sur leurs services. Les données échangées ne sont pas explicitement identifiées, si bien que chaque fonction traite ses données d'entrée par rapport à leur signification. Cela implique que toutes les fonctions ont leur traitement propre des données d'entrée. Notre approche de modélisation présente alors l'avantage de mettre en évidence des comportements génériques, applicables pour toutes fonctions et données échangées dans n'importe quel modèle socio-technique représentant une phase de vol. Ceci est rendu possible grâce à la notion d'efficacité.
3. L'activation des fonctions est en revanche beaucoup plus explicite dans les modèles de J.-C. Chaudemar que dans les nôtres. Cet aspect ne nous a pas semblé pertinent à expliciter pour l'étude des pannes. Dans [Chaudemar et al., 2009], les fonctions ont un état interne spécifique pour l'activation, en plus de l'état de fonctionnement. Cet état bascule de l'activation à la non-activation et vice versa lorsque la fonction reçoit un flux d'activation de la part d'un nœud de supervision. Le changement d'état est réalisé à travers des événements spécifiques. Ces derniers présentent la particularité de se déclencher automatiquement quand la garde devient vraie et de ne pas être pris en compte par le générateur de séquences.

Dans [Belmonte et al. 2010] et [Belmonte 2009], la problématique présentée est d'identifier les scénarios de pannes à l'origine d'une situation redoutée, qui sont également la cause de l'échec des moyens correctifs mis en place. Le cadre d'étude est le ferroviaire. Le traitement de cette problématique nous a inspiré pour l'exploitation de nos modèles de sécurité. L'approche de F. Belmonte consiste à réaliser plusieurs arbres de défaillances (voir la section 4.1.1 du chapitre 2) successifs :

- le premier arbre de défaillances est focalisé sur une situation redoutée (exemple : le risque de perte de contrôle de la trajectoire de l'avion pendant le décollage) ;
- d'autres arbres de défaillances sont construits pour tous les moyens préventifs (le terme de « barrière préventive » est également utilisé) pouvant être appliqués une fois que la situation redoutée est identifiée (exemple : interrompre le décollage et stopper l'avion sur la piste) ;
- enfin, d'autres arbres de défaillance sont construits pour tous les moyens de protection (ou « barrière protectrice ») pouvant réduire les conséquences de la situation redoutée initiale (exemple : maintenir fermement les passagers et les membres d'équipages sur leur siège).

En comparant les coupes minimales de ces différents arbres de défaillances, il est possible d'identifier d'éventuels points communs de défaillance entre l'occurrence du danger, la défaillance des moyens préventifs et la défaillance des moyens de protection. Dans le cadre de l'analyse des risques, nous décrivons également des moyens d'actions correctifs (voir la section 3.2.2 du chapitre 1). D'où notre intérêt pour mettre en place cette approche à partir de la recherche des causes de situations redoutées dans nos modèles. Le chapitre 7 présente une application de ce principe d'analyse.

5.2 Critiques sur l'approximation lors de l'agrégation des données utiles et nécessaires

Nous pouvons émettre une critique sur notre approche : l'agrégation des données utiles et nécessaires peut conduire à des biais de comportement des modèles.

Face à la difficulté d'exprimer les opérations mathématiques réalisées pour décrire la loi de comportement des activités, du fait du nombre aléatoire et important de variables, nous avons proposé des simplifications, au niveau des concepts de nos modèles de sécurité, permettant de ramener les variables au nombre de quatre :

- la valeur agrégée des flux d'activation (pour les opérations) ;
- la valeur agrégée des liens d'environnement ;
- la valeur agrégée des liens de données nécessaires ;
- la valeur agrégée des liens de données utiles.

Or, les deux dernières valeurs agrégées sont obtenues au prix de trois importantes approximations. Nos travaux actuels ne nous ont pas permis de résoudre ces problématiques tout en conservant des comportements génériques pour les activités.

1. Pour l'agrégation des données nécessaires, nous avons justifié notre choix de prendre la valeur minimale parmi les valeurs d'efficacité des données nécessaires par le fait que les données nécessaires sont plus importantes pour la fonction que les données utiles. Mais pour choisir si une donnée est nécessaire ou utile, nous analysons en fait si la fonction concernée doit attendre la donnée pour s'exécuter ou non. Nous constatons en pratique que certaines données, définies comme nécessaires se révèlent certes nécessaires du point de vue de l'activation, mais moins importante du point de vue de l'information échangée que certaines données utiles. La séparation des données en deux sous-ensembles, qui prend

son origine dans les modèles ALFA, présente ses limites et mériterait d'être raffinée.

2. Nous supposons qu'au sein des données nécessaires ou au sein des données utiles, toutes les données ont le même poids vis à vis de la fonction concernée. En pratique, c'est rarement le cas, même si l'abstraction de la couche fonctionnelle ne permet pas de définir précisément le poids de chaque donnée. Le poids des données pourraient être une caractéristique mise en avant dans de futures approches.
3. Du fait des valeurs agrégées, le poids des données varient d'une fonction à l'autre de façon non maîtrisée. Prenons par exemple le cas d'une fonction recevant une donnée nécessaire et dix données utiles en entrée. La valeur agrégée des données nécessaires prend alors la valeur de l'unique donnée nécessaire. La valeur agrégée des données utiles prend la valeur moyenne des valeurs des données utiles. Le poids de la donnée nécessaire est donc clairement plus important que celui de toutes les données utiles, puisque les valeurs de ces dernières sont noyées parmi les neufs autres valeurs. Heureusement, en pratique, une fonction reçoit rarement plus de deux données, limitant ainsi cette forte imprécision.

5.3 Perspective : liens avec d'autres modèle de sécurité

Dans la section 4.2 du chapitre 2, nous avons présenté une approche existante de modélisation pour étudier les comportements dysfonctionnels des architectures pour vérifier que ces dernières respectent les exigences de sécurité. Il s'agit de l'approche MBSA pour « Model-Based Safety Assessment ». nous identifions une perspective de modélisation : formaliser les éventuels liens entre les modèles de sécurité que nous proposons et les modèles existants dans le cadre de l'approche MBSA.

Les modèles MBSA décrivent les interactions entre les équipements d'un système quand notre proposition vise à décrire les interactions entre les fonctions de ce système. Le lien entre les deux approches de modélisation est celui de l'allocation des fonctions aux équipements. L'allocation des fonctions aux équipements est une problématique importante de la conception. Du point de vue de la sûreté de fonctionnement, il s'agit de garantir que les critères de sécurité, sur lesquels s'appuient la conception et la vérification des architectures matérielles, sont conformes aux exigences exprimées par les analyses des risques. De futur travaux sont nécessaires pour renforcer le lien d'allocation des fonctions aux équipements, notamment grâce aux modèles.

Chapitre 5

Mise en œuvre de l'analyse des risques avec AltaRica

Sommaire

Résumé. Dans le chapitre 4, nous avons défini formellement les modèles de sécurité rendant possible l'analyse des pannes fonctionnelles pour assister les analyses des risques. Toutefois, pour que ces modèles soient exploitables, il est nécessaire de pouvoir les écrire dans un langage suffisamment outillé pour nos besoins d'analyse. Dans le cadre de nos travaux, nous avons choisi le langage AltaRica enrichi de la couche graphique RAMSES/OCAS. Dans la suite, nous montrons comment les modèles de sécurité peuvent s'écrire avec le langage AltaRica.

1 Compatibilité des modèles de sécurité avec la sémantique du langage AltaRica

La première question qui se pose pour concrétiser les modèles de sécurité à l'aide du langage AltaRica est : la sémantique de ce langage est-elle capable de supporter le formalisme des modèles MB_{FHA} ?

La sémantique et la syntaxe du langage AltaRica ont été présentées dans le détail au paragraphe 4.3 du chapitre 2. La sémantique de ce langage est celle des automates de mode, caractérisée principalement par le système état-transition et le calcul des assertions reliant les sorties d'un automate à ses entrées.

Dans cette section, nous montrons que les concepts que nous avons présentés au chapitre 4 peuvent être décrits avec la sémantique du langage AltaRica. Nous rappelons, d'après la section 2.2 du chapitre 4, que les nœuds des modèles d'analyse des aspects dysfonctionnels ont une sémantique de la forme :

$\forall n \in Noeud$, $\|n\| = (D, c(n), \sigma(n))$, avec :

– $c(n)$ le comportement intrinsèque du nœud, évalué par un vecteur de x éléments de D ;

– $\sigma(n)$: $D^x \times dom(input(n)) \rightarrow dom(output(n))$ la loi de comportement du nœud, tel que : $val(output(n)) = \sigma(n)(c(n), val(input(n)))$.

Un modèle MB_{FHA} a aussi une telle sémantique par composition des nœuds qui le constituent.

1.1 Les variables du modèle de sécurité

Les modèles de sécurité font intervenir plusieurs variables. Dans cette section, nous montrons la correspondance entre ces variables conceptuelles et les variables de la sémantique d'AltaRica. Il s'agit dans un premier temps de définir l'ensemble des variables, V, de la sémantique d'AltaRica. Puis dans un second temps, nous nous intéressons à l'évaluation de ces variables à l'aide de la fonction dom .

1.1.1 Identification des variables

La sémantique du langage AltaRica distingue trois familles de variables : les variables d'état, les variables de flux entrant et les variables de flux sortant.

Les variables d'état

L'ensemble S des variables d'état est composé des variables du modèle qui caractérisent son comportement interne, susceptible d'évoluer grâce aux transitions et pris en compte dans le calcul des assertions. La correspondance pour chaque nœud n est la valeur $c(n)$.

A l'échelle d'un modèle MB_{FHA} , par composition, l'ensemble des variables d'état est le vecteur des comportements internes de toutes les activités et toutes les conditions environnementales. Nous obtenons alors : $S = C = c(Activité) \times c(Environnement)$.

Les variables de flux

Dans la sémantique d'AltaRica, les variables de flux entrant et sortant sont définies comme les **interfaces** des nœuds et des modèles. La figure illustre l'effort de concrétisation entre une modélisation sans les interfaces et une modélisation qui les met en évidence.

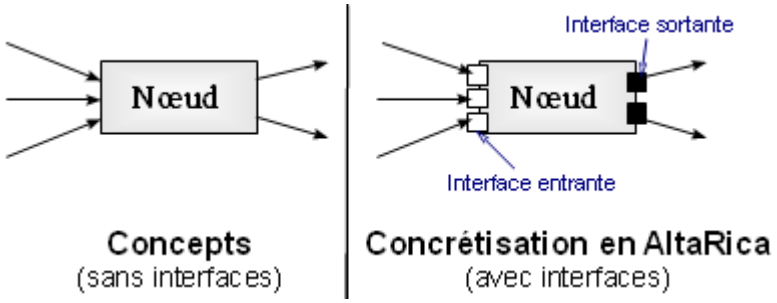


Fig. 5.1 - Modélisation des interfaces

Cette notion d'interface n'a pas son équivalent direct dans les concepts des modèles de sécurité. Toutefois, dans la section 2.2.1 du chapitre 4, nous avons présenté deux fonctions pour déterminer les liens entrants et sortants d'un nœud :

- $input: \ Noeud \rightarrow 2^{Lien}$, telle que $\forall n \in Noeud, input(n) = \{(n', n) \in Lien / n' \in Noeud\}$;
- $output: \ Noeud \rightarrow 2^{Lien}$, telle que $\forall n \in Noeud, output(n) = \{(n, n') \in Lien / n' \in Noeud\}$.

L'unique différence entre ces définitions et la notion d'interface est que l'interface n'est pas dépendante d'un lien. Deux liens sortant d'un nœud et dont la valeur est toujours identique correspondent à deux éléments sortant de ce nœud, mais peuvent être issus de la même interface. Par commodité, nous avons décidé que tous les liens entrants et sortants seraient connectés aux nœuds par des interfaces indépendantes, même si elles ont toujours la même valeur. Nous définissons alors l'interface comme : $interface: \ Noeud \times Lien$ (son évaluation est traitée à la section 1.1.2).

Nous définissons alors les variables de flux entrant et sortant par :

- $F^{\text{in}} = \{interface(n, l) / \ n \in Noeud, \ l \in input(n)\}$ et
- $F^{\text{out}} = \{interface(n, l) / \ n \in Noeud, \ l \in output(n)\}$

Les interfaces du modèle global MB_{FHA} découlent de la composition des nœuds (voir l'annexe A).

1.1.2 Domaine de valeur et évaluation dans le domaine de valeur

Le domaine des valeurs des modèles de sécurité est $D = Booléen \cup E \cup P_{env} \cup C_{act}$. Sa définition correspond bien au domaine des valeurs de la sémantique d'AltaRica. De même nous avons déjà défini la fonction dom pour associer à chaque lien un domaine de valeur (voir la section 2.2.1 du chapitre 4).

Cependant, cette définition ne couvre pas les variables d'état ni les variables d'entrée et de sortie des

nœuds et des modèles. Nous proposons alors d'étendre la définition de la fonction dom .

Domaine d'évaluation des variables :

Soit $V = S \cup F^{in} \cup F^{out}$ l'ensemble des variables défini à partir des trois sous-ensembles présentés à la section précédente. Les sous-ensembles sont deux à deux disjoints.

Alors nous définissons $dom : V \rightarrow 2^D$ tel que :

– $\forall v \in V,\ v \in S \Rightarrow dom(v) = (C_{act} \cup P_{env})$ *, signifiant que les états internes sont soit des états de panne soit des puissances de conditions environnementales ;*

– $\forall v \in V,\ v \in F^{in} \Rightarrow dom(v) = (E \cup P_{env} \cup Booléen)$ *, signifiant que les entrées d'un nœud sont soit des données évaluées par l'efficacité, soit des conditions environnementales soit des flux d'activation opérationnelle ;*

– $\forall v \in V,\ v \in F^{out} \Rightarrow dom(v) = (E \cup P_{env} \cup Booléen)$ *, avec la même signification que pour le domaine d'évaluation de l'ensemble F^{in} .*

Nous définissons ensuite $dom : 2^V \rightarrow 2^D$, tel que :

– $\forall U \subseteq V,\ dom(U) = \prod_{u \in U} dom(u)$.

Cette définition est conforme en tout point à la sémantique du langage AltaRica.

1.2 Les assertions du modèle de sécurité

Nous avons montré dans la section 1.1 que les concepts des modèles de sécurité respectent la sémantique du langage AltaRica pour ce qui est de la déclaration et l'évaluation des variables.

La dernière étape, pour rendre compatible la sémantique des modèles de sécurité avec celle du langage AltaRica, est la modélisation des lois de comportement de chaque nœud et la modélisation de la loi de calcul des résultats d'un modèle MB_{FHA} . Dans les sections suivantes, nous présentons deux approches : l'utilisation exclusive des assertions et l'enrichissement avec les transitions. Comme nous le verrons dans la section 1.2.2, la seconde approche exploite davantage la richesse du langage AltaRica, mais est également complémentaire de la première approche. C'est cette solution que nous retiendrons par la suite.

1.2.1 Approche 1 : utilisation exclusive des assertions

Par définition, les lois de comportement $\sigma(n):\ D^x \times dom(input(n)) \rightarrow dom(output(n))$, telles que $val(output(n)) = \sigma(n)(c(n), val(input(n)))$, de chaque nœud et du modèle global, sont des assertions, permettant de calculer la valeur des liens sortants de chaque nœud par rapport aux liens entrants et aux comportements internes.

Or, les comportements internes $c(n) \subseteq c(Activité) \times c(Environnement)$ sont par définition les états internes (éléments de l'ensemble S) du nœud considéré (voir la section 1.1.1). D'où :

$\sigma(n):\ dom(S) \times dom(input(n)) \rightarrow dom(output(n))$.

A présent, nous proposons une évaluation des interfaces :

$val:\ interface(n,l) \rightarrow E \cup P_{env} \cup Booléen$ telle que :

- $\forall n \in Noeud,\ \forall l \in (input(n) \cup output(n)),\ val(interface(n,l)) = val(l)$, i.e. l'interface relative à un lien prend la valeur de son lien.

Alors :

- $val(input(n)) = val(F^{\text{in}})$ avec $val(F^{\text{in}}) = \prod_{interface \in F^{\text{in}}} val(interface) \in dom(F^{\text{in}})$;
- $val(output(n)) = val(F^{\text{out}})$ avec $val(F^{\text{out}}) = \prod_{interface \in F^{\text{out}}} val(interface) \in dom(F^{\text{out}})$.

Ainsi, la loi de comportement de chaque nœud et des modèles MB_{FHA} s'écrit :

$\sigma:\ dom(S) \times dom(F^{\text{in}}) \rightarrow dom(F^{\text{out}})$.

Nous en concluons donc que les assertions d'AltaRica permettent de modéliser les lois de comportement pour chaque nœud du modèle ainsi que pour le modèle global. Ainsi, nous capturons bien le comportement qui décrit l'impact sur les sorties des pannes fonctionnelles et des conditions environnementales.

1.2.2 Approche 2 : enrichissement avec les transitions

La première approche, centrée exclusivement sur les assertions, suffit pour établir que les concepts présentés au chapitre 4 peuvent être implémentés à l'aide du langage AltaRica. En revanche, nous n'exploitons pas la caractéristique fondamentale des automates de mode à savoir le système d'états-transitions. De plus, comme nous le verrons dans la section 3, relative à l'exploitation des modèles de sécurité pour assister l'analyse des risques, les outils d'analyse basés sur le langage AltaRica, s'appuient sur le principe des systèmes à états-transitions.

Nous proposons alors d'enrichir les concepts des modèles de sécurité, à l'aide d'une meilleure expression des états internes des nœuds (les éléments de S). Afin de capturer leurs variations, nous proposons d'utiliser la notion d'événement du langage AltaRica.

Soit alors $\Sigma = \{Fonction \times (C_{act} - \{nominal\})\} \cup \{Environnement \times (P_{env} - \{p_{nul}\})\}$ l'ensemble des événements du modèle, composé de :

- $\{Fonction \times (C_{act} - \{nominal\})\}$, l'ensemble des pannes fonctionnelles des fonctions du modèle ;
- $\{Environnement \times (P_{env} - \{p_{nul}\})\}$, l'ensemble des valeurs non nulles des conditions environnementales du modèle.

Alors, pour tous les nœuds d'un modèle MB_{FHA}, nous définissons les transitions par $\delta:\ dom(S) \times dom(F^{\text{in}}) \times \Sigma \rightarrow dom(S)$, telles que :

- pour une fonction f, si son état est nominal et qu'une panne fonctionnelle (f, p) survient, avec $p \in (C_{act} - \{nominal\})$, alors son état est modifié : il prend la valeur p.
- pour une condition environnementale n_{env}, si son état est p_{nul} et qu'un événement (n_{env}, p) survient, avec $p \in (P_{env} - \{p_{nul}\})$, alors son état est modifié : il prend la valeur p.

Les opérations et les données ne sont concernées ni par les événements, ni par les transitions. Nous pouvons également constater que les interfaces d'entrée n'interviennent pas dans l'affectation des nouveaux états internes.

Cette approche de modélisation via les événements et les transitions, s'appuie largement sur l'approche présentée à la section précédente puisque le calcul des interfaces sortantes est toujours réalisé à l'aide des assertions. Les événements et les transitions nous permettent simplement de mettre en évidence les variables fondamentales de chaque modèle MB_{FHA}.

2 Modélisation concrète des nœuds des modèles de sécurité

Une fois établie la compatibilité des modèles de sécurité avec la sémantique du langage AltaRica, nous nous intéressons à la modélisation syntaxique. Cette étape finale de la concrétisation des concepts nous contraint à achever la définition des domaines de valeur et des lois de comportement. Nous devons utiliser les types de valeurs et les expressions logiques supportées par le langage AltaRica.

A partir de cette section, nous proposons une modélisation concrète des concepts présentés au chapitre 4. De nombreux choix de modélisation ont été faits.

2.1 Domaine des valeurs en AltaRica

Le domaine des valeurs manipulées dans les modèles de sécurité est la réunion de quatre sous-ensembles disjoints : $D = Booléen \cup E \cup P_{env} \cup C_{act}$. La modélisation des booléens est facile avec AltaRica puisqu'il s'agit d'un des types de base du langage. Les trois autres ensembles doivent en revanche être construits. Les modélisations de l'efficacité et de la puissance environnementale présentent des problématiques communes. Elles font l'objet des sections 2.1.1, 2.1.2 et 2.1.3. La modélisation des modes de fonctionnement des activités fait l'objet de la section 2.1.4.

2.1.1 Modélisation concrète de l'efficacité et de la puissance environnementale

Nous avons identifié trois contraintes concernant la modélisation de l'efficacité et de la puissance environnementale. Pour les deux premières, il s'agit de donner un sens concret à ces valeurs d'efficacité et de puissance environnementale afin qu'elles soient exploitables par les analystes de la sécurité et par les concepteurs. La dernière contrainte présente une limitation du langage AltaRica. Elles sont détaillées toutes les trois ci-après :

La notion d'efficacité dans la conception

Comme nous l'avons vu dans le chapitre 4, une évaluation des services de chaque activité doit être réalisée à l'aide de la loi de comportement. Dans un modèle nominal, sans panne ni présence de condition environnementale, les services de toutes les activités sont nominaux. Mais ces services nominaux n'ont pas tous la même valeur d'efficacité (voir la section 2.3.3 du chapitre 4). A qui incombe alors la responsabilité de définir l'efficacité nominale de tous les services des activités ? Aux analystes de la sécurité qui sont en charges des modèles MFHA ? Ou aux concepteurs de l'avion qui travaillent déjà sur des exigences de performance avion dont la définition présente des similitudes avec la notion d'efficacité ?

Cette problématique est encore largement ouverte, mais elle met en évidence une contrainte pour la modélisation de l'efficacité en AltaRica : la définition concrète de l'efficacité doit avoir du sens pour les concepteurs autant que pour les experts de la sécurité.

L'évaluation de la puissance environnementale

La puissance environnementale permet en particulier d'évaluer les variables correspondantes aux états des conditions environnementales modélisées. Au niveau des concepts, ces états sont infinis, bien qu'admettant une borne inférieure, qui est la puissance nulle p_{nul}. Alors, comment mesurer la puissance des conditions environnementales ?

Pour la pluie, par exemple, il semble possible de la mesurer à partir de la quantité de pluie. Pour le brouillard, la mesure peut se baser sur sa densité. Ces exemples illustrent la difficulté de la mesure de la puissance environnementale, puisque la pluie et le brouillard doivent être évalués sur le même ensemble de valeur et que les valeurs de puissance environnementale ont un impact sur les services des activités, indépendamment de la connaissance des conditions environnementales à l'origine de ces valeurs de dégradation. La seconde problématique concerne donc la cohérence des différentes mesures des conditions environnementales, en prenant en considération le fait qu'il s'agit en fait de mesures d'impacts sur les activités. Or, comme pour l'efficacité, ces impacts doivent avoir du sens pour les analystes de la sécurité en charge de l'analyse des risques.

Les limites de modélisation du langage AltaRica

Dans la sémantique du langage AltaRica, l'ensemble des valeurs du domaine, D, est défini comme un ensemble fini. Il peut alors s'agir, par exemple, de la réunion de booléens, d'intervalles bornés d'entier ou d'ensembles énumérés. L'utilisation d'ensembles infinis de valeurs rend impossible l'analyse des modèles AltaRica par des outils comme le générateur d'arbres de défaillance ou le générateur des coupes minimales.

Par conséquent, nous n'utilisons pas des intervalles de réels pour représenter l'efficacité ou la puissance environnementale.

2.1.2 Discrétisation de l'efficacité en AltaRica

Dans la section 2.3.3 du chapitre 4, nous avons défini la notion d'efficacité par l'ensemble borné E. Nous avons proposé dans une première approche la concrétisation de cet ensemble sous la forme d'un intervalle de

réels : $E=[-100,100]\subset\mathbb{R}$. Ce choix est motivé par la définition de l'efficacité comme un taux de contribution à un résultat ou un taux d'aboutissement à des objectifs. De tels taux peuvent aisément être représentés par des pourcentages, soit des intervalles tels que $[-1,1]$ ou $[-100,100]$ par exemple. Ces taux présentent l'avantage d'être facilement compréhensibles et interprétés sans ambiguïté par les concepteurs et les analystes de la sécurité. Ils répondent donc à la première problématique évoquée dans la section précédente.

Nous proposons de discrétiser les valeurs d'efficacité en une énumération de valeurs qualitatives. La table 5.1 ci-dessous présente les codes AltaRica et Express correspondant à une énumération de cinq valeurs. Dans la section 3.3, nous montrons que cette énumération est un compromis satisfaisant entre la complexité de modélisation et la précision des résultats. Bien sur, ce compromis peut-être remis en question par rapport à des besoins d'analyse précis.

```
1   -- Code AltaRica
2   domain MBFHA_types_Effectiveness_5 = {High,
3   Hneg, Lneg, Low, None};
4
5   -- Code Express
6   TYPE efficacité = ENUMERATION OF (High, Low,
7   None, LNeg, HNeg);
8   END_TYPE;
```

Tab. 5.1 - Codes AltaRica et Express de l'ensemble des valeurs d'efficacité

Le sens de chacune de ces valeurs d'efficacité est lié directement à la notion de taux de contribution ou d'aboutissement d'objectifs.

1. « High » : cette valeur est la borne maximale de l'ensemble des valeurs d'efficacité. Le pourcentage correspondant est 100%. Pour un service, cette valeur signifie que le service contribue totalement à la réalisation de son résultat, même s'il est le seul contributeur. La présence d'autres services contributeurs ne peut pas améliorer l'aboutissement qui est déjà total ; en revanche, ces autres services renforcent la robustesse face à des dégradations telles que des pannes ou des conditions environnementales dégradantes.
2. « Low » : l'efficacité faible correspond à une contribution de 50% à la réalisation des objectifs d'un résultat. La présence d'autres services contributeurs est requise afin que les objectifs puissent être totalement atteints.
3. « None » : cette valeur correspond à l'efficacité nulle e_{nul} et au pourcentage 0%. Concrètement, un service à la valeur nulle ne participe aucunement à la réalisation de l'objectif auquel il est lié.
4. « LNeg » (« Low Negative ») : la valeur faiblement négative est la valeur symétrique de l'efficacité faible (« Low ») par rapport à la valeur nulle. Elle correspond à un taux de contribution négatif, valant -50%. Cela signifie que le service évalué « LNeg », s'oppose faiblement à la réalisation de son résultat.

5. « HNeg » (« High Negative ») : la valeur fortement négative est la valeur symétrique de l'efficacité élevée (« High »). Il s'agit également de la borne minimale des valeurs d'efficacité. Elle correspond à un taux de contribution négatif de -100%. Un service évalué « HNeg » s'oppose donc très fortement à la réalisation de son résultat.

La correspondance entre les critères d'efficacité précédents et les pourcentages de contribution, permet de définir l'addition dans l'ensemble E.

2.1.3 Modélisation concrète de la puissance environnementale

Comme pour la modélisation de l'efficacité, nous n'utilisons pas la modélisation de la puissance environnementale par $P_{env} = \mathbb{R}_+$, comme nous l'avons proposé en première approche dans le chapitre 4. Une telle modélisation serait difficile à interpréter pour les concepteurs et les analystes de la sécurité.

Nous choisissons, comme pour l'efficacité, de définir des valeurs qualitatives. Or, il n'existe pas à notre connaissance de valeurs précises. Les conditions environnementales pertinentes à prendre en compte dans les analyses de sécurité sont listées dans un document, appelé « *Common Data Document* », avec leur probabilité d'occurrence. Mais cette probabilité d'occurrence n'est liée à aucune valeur de puissance environnementale. Cela signifie que les analyses de sécurité ne s'intéressent qu'à la présence ou l'absence totale de chaque condition environnementale. Nous proposons alors l'énumération de la table 5.2.

```
1   -- Code AltaRica
2   domain MBFHA_types_Environmental_power =
3   {Null, Strong, Very_strong};
4
5   -- Code Express
6   TYPE puissance_env = ENUMERATION OF (Null,
7   Strong, Very_strong);
8   END_TYPE;
```

Tab. 5.2 - Codes AltaRica et Express de l'ensemble des puissances environnementales

Le sens de chacun des trois critères composant cette énumération sont :

1. « Null » : cette valeur correspond à la puissance nulle, notée p_{nul} de puissance environnementale dégradante. Concrètement, il s'agit d'une absence de la condition environnementale considérée.
2. « Strong » : cette valeur correspond simplement à la présence de la condition environnementale considérée.
3. « Very_strong » : cette valeur permet d'identifier des combinaisons de présence de plusieurs conditions environnementale. Elle correspond à l'impact combiné de deux puissances « Strong » sur une même activité.

Notre choix est motivé par l'intérêt que nous portons à l'étude de combinaisons de deux conditions environnementales. Comme cela a été évoqué dans la section 4 du chapitre 4, cet ensemble peut être modifié selon les besoins d'analyse. Il peut par exemple être réduit à ses deux premières composantes si les combinaisons de deux conditions environnementales avec des pannes ne sont pas jugées pertinentes.

2.1.4 Modélisation des états de fonctionnement des activités

Les états de fonctionnement des activités, formant l'ensemble C_{act}, sont toutes les pannes fonctionnelles pouvant affecter une fonction, auquel est ajouté l'état de fonctionnement nominal (voir la section 3.1.3 du chapitre 4). Les pannes fonctionnelles sont exprimées à l'aide des modes de panne, mais, comme nous l'avons vu, ces modes de panne peuvent conduire à des pannes différentes. En particulier, nous distinguons les pannes symétriques et asymétriques pour les fonctionnements erronés et les pertes partielles (voir la section 3.2 du chapitre 4).

La table 5.3, ci-dessous, présente l'ensemble des états de fonctionnement à partir des modes de panne.

```
1   -- Code AltaRica
2   domain MBFHA_types_Failure_modes =
3   {Erroneous_asym, Erroneous_sym, Nominal,
4   Partially_lost_asym, Partially_lost_sym,
5   Totally_lost};
6
7   -- Code Express
8   TYPE mode_panne = ENUMERATION OF
9   (Total_loss, Partial_loss_sym,
10  Partial_loss_asym, Erroneous_sym,
11  Erroneous_asym);
12  END_TYPE;
```

Tab. 5.3 - Codes AltaRica et Express de l'ensemble des modes de fonctionnement

Nous avons représenté tous les modes de panne à l'exception du déclenchement intempestif des fonctions, que nous avons choisi de traiter comme une perspective (voir la section 4.2.1).

2.2 Modélisation des nœuds « Donnée » et « Environnement »

A présent que l'ensemble des valeurs du domaine est modélisé, nous pouvons modéliser concrètement les différents nœuds des modèles de sécurité. Dans cette section, nous nous focalisons sur les deux nœuds les plus simples, à savoir les données et les conditions environnementales. Les activités sont traitées dans la section 2.3.

2.2.1 Modélisation concrète des données du modèle

Dans la section 2.4.2 du chapitre 4, nous avons donné la sémantique des données échangées au sein des modèles MB_{FHA}. Cette formalisation met en évidence les caractéristiques des données, à savoir :

- pas d'état interne ;
- plusieurs interfaces d'entrée pour chacun des liens entrants ;
- plusieurs interfaces de sortie pour chacun des liens sortants ;
- la valeur de toutes les interfaces de sorties est la valeur du résultat de la donnée, obtenu grâce à la somme tronquée des valeurs d'efficacité des interfaces d'entrée.

Comme tous les nœuds, et d'après la section 1.1.1, les données sont construites avec autant d'interface d'entrée et de sortie que de liens entrants et sortants. Cependant, comme par définition tous les liens sortants de chaque donnée ont la même valeur (le résultat de la donnée), la sémantique du langage AltaRica nous permet de ne définir concrètement qu'une seule interface de sortie pour tous les liens sortants. Alors :

$\forall d \in Donnée,\ F^{\text{out}} = \{interface\},\ avec\ \forall l \in output(d),\ interface(d, l) = interface$.

La figure 5.2 illustre l'instanciation d'une donnée ayant deux liens entrants, avec les graphismes choisis pour la concrétisation à l'aide des outils de visualisation et d'édition graphique OCAS et RAMSES.

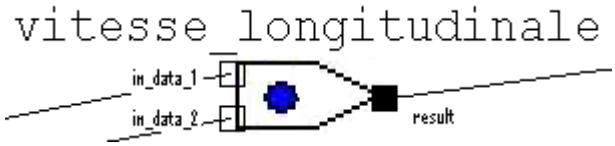


Fig. 5.2 - « Donnée » dans l'outil OCAS

La table 5.4, ci-dessous, présente le code AltaRica d'un nœud de type « Donnée », ayant deux liens entrants. L'assertion de ce nœud fait appel à un opérateur, noté $add\_eff\_2 : E \times E \rightarrow E$, qui calcule la somme tronquée de deux valeurs d'efficacité.

```
1   node MBFHA_Dataflow_2
2    flow
3     in_data_1:MBFHA_types_Effectiveness_5:in;
4     in_data_2:MBFHA_types_Effectiveness_5:in;
5     result:MBFHA_types_Effectiveness_5:out;
6    assert
7     result = add_eff_2(in_data_1,in_data_2);
8   edon
```

Tab. 5.4 - Code AltaRica des données à deux services entrants

Concrètement, l'opérateur add_eff_2 convertit les valeurs qualitatives d'efficacité en entiers dans

l'intervalle $[-2,2]$, afin de pouvoir utiliser l'addition, tel que « High » vaut « 2 », Low vaut « 1 » (nous constatons bien que l'efficacité faible est une valeur deux fois plus faible que l'efficacité élevée), « None » vaut « 0 », « LNeg » vaut « -1 » et « HNeg » vaut « -2 » (nous constatons bien que les valeurs d'efficacité négatives sont opposées aux valeurs positives).

Ce code peut être instancié pour modéliser toutes les données ayant exactement deux services contribuant à son résultat. Les données ayant plus de deux services contributeurs peuvent être construites comme le cas de la table 5.4, à l'aide d'opérateurs prenant en compte davantage de variables et réalisant la somme tronquée de toutes leurs variables.

2.2.2 Modélisation concrète des conditions environnementales

La sémantique des conditions environnementales met en évidence leurs caractéristiques :

- un état interne évalué par une puissance environnementale ;
- aucune interface d'entrée ;
- plusieurs interfaces de sortie pour chacun des liens sortant en direction d'activités ;
- la valeur de toutes les interfaces de sortie est la valeur de l'état interne.

Comme pour les données, les interfaces de sortie peuvent être regroupées autour d'une unique interface :

$\forall n_{env} \in Environnement,\ F^{\text{out}} = \{interface\},\ avec\ \forall l \in output(n_{env}),\ interface(n_{env}, l) = interface$.

Graphiquement, une condition environnementale se présente telle que sur la figure 5.3. Son code AltaRica est celui de la table 5.5. Nous constatons en premier lieu que l'état interne prend une valeur de puissance environnementale, qui est soit « Null » soit « Strong ». L'icône à gauche de la figure 5.3 représente la valeur nulle, à savoir une absence de la condition environnementale. A l'inverse, à droite de la figure, l'icône rouge représente la présence de la condition environnementale, soit la valeur « Strong » affectée à son état interne. Via l'événement « Strong » et une transition, l'état interne de la condition environnementale peut passer de la valeur « Null » à la valeur « Strong ». L'état initial définit la situation initiale à partir de laquelle nous souhaitons réaliser nos analyses (voir la section 4.1 du chapitre 4). La plupart du temps, cette situation initiale est l'état nominal du modèle. Dans le code de la table 5.5, nous avons décrit l'absence de la condition environnementale à l'état initial.

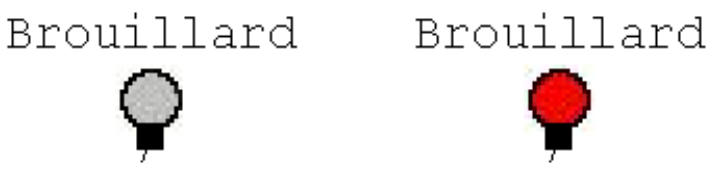


Fig. 5.3 - « Condition environnementale » dans OCAS

```
1   node MBFHA_Environmental_condition
2     flow
3       out_env:MBFHA_types_Environmental_power:out;
4     state
5       Status:MBFHA_types_Environmental_power;
6     event
7       strong;
8     trans
9       (Status = Null) |- strong -> Status :=
10  Strong;
11    assert
12      out_env = Status;
13    init
14      Status := Null;
15  edon
16
```

Tab. 5.5 - Code AltaRica des conditions environnementales

Ce code AltaRica peut être instancié sans modification pour toutes les conditions environnementales du modèle de sécurité.

2.3 Modélisation des activités

La troisième et dernière famille de nœuds à modéliser sont les activités. Les opérations et les fonctions, bien que différentes, présentent beaucoup de points communs :

- plusieurs interfaces de données nécessaires en entrée ;
- plusieurs interfaces de données utiles en entrée ;
- plusieurs interfaces de puissances environnementales en entrée ;
- plusieurs interfaces de service en sortie ;

Les opérations ont en plus des interfaces d'entrée et de sortie pour les flux d'activation. Les fonctions, elles, ont en plus un état de fonctionnement interne, ainsi que des transitions pour modéliser les pannes fonctionnelles.

2.3.1 Agrégation des valeurs d'entrée des activités

Comme nous l'avons vu au chapitre 4, la première étape du calcul de la loi de comportement est l'agrégation des données d'entrée, ainsi que des puissances environnementales et des flux d'activation. Le squelette de code AltaRica d'une activité est donné par la table 5.6, avec deux liens de chaque type.

```
1   node Squelette_activité
2     flow
3   //Variables de flux d'activation (opération)
4    in_act_1:bool:in;
5    in_act_2:bool:in;
6   //Variables de données nécessaires
7    E_n_1:MBFHA_types_Effectiveness_5:in;
8    E_n_2:MBFHA_types_Effectiveness_5:in;
9   //Variables de données utiles
10   E_u_1:MBFHA_types_Effectiveness_5:in;
11   E_u_2:MBFHA_types_Effectiveness_5:in;
12  //Variables de puissances environnementales
13   env_1:MBFHA_types_Environmental_power:in;
14   env_2:MBFHA_types_Environmental_power:in;
15  //Variables d'agrégation
16   val_P:bool:private;
17   val_N:MBFHA_types_Effectiveness_5:private;
18   val_U:MBFHA_types_Effectiveness_5:private;
19   val_E:MBFHA_types_Environmental_power:private;
20    assert
21  //Calcul des valeurs agrégées
22      val_P =agr_p_2(in_act_1,in_act_2);
23      val_N =agr_n_2(E_n_1,E_n_2);
24      val_U =agr_u_2(E_u_1,E_u_2);
25      val_E =agr_e_2(env_1,env_2);
26  edon
```

Tab. 5.6 - Code AltaRica des valeurs d'entrée des activités et de leur agrégation

Notons que les lignes de code relatives aux flux d'activation ne sont pas présentes dans le code des fonctions. Nous avons déclaré toutes les variables d'entrée, puis déclaré quatre variables internes qui représente les agrégations présentées à la section 2.5.1 du chapitre 4 :

1. val_P, noté val_p dans le chapitre 4, est l'agrégation des flux d'activation ;
2. val_N, noté val_n dans le chapitre 4 est l'agrégation des données nécessaires ;
3. val_U, noté val_u dans le chapitre 4, est l'agrégation des données utiles ;
4. val_E, noté val_e dans le chapitre 4 est l'agrégation des puissances environnementales.

Les lois de composition, permettant de calculer ces valeurs agrégées, ont également été spécifiées à la section 2.5.1 du chapitre 4. Ces spécifications doivent être adaptées aux ensembles des valeurs concrètement manipulées.

1. Pour les flux d'activation, la loi de composition reste inchangée car elle est toujours définie dans l'ensemble des booléens. Il s'agit d'un « ET » logique entre les valeurs de tous les liens de procédure.

2. De même, le calcul de l'efficacité globale des données nécessaires ne dépend pas de la façon dont l'efficacité est modélisée. La valeur reste nulle si l'un des liens de donnée nécessaire a une valeur nulle. Sinon la valeur est égale à la plus faible valeur des liens de donnée.
3. En revanche, avec les critères qualitatifs ordonnés concrétisant les valeur d'efficacité, nous ne somme plus en mesure de réaliser directement le calcul de l'efficacité globale des données utiles, du fait des opérations d'addition et de division :

$$val_u(a)=\left\{\begin{array}{l}\frac{1}{card\,(input(a))}\cdot\sum_{l\in input(a)/l\in Lien\_entrant} val(l) \quad si \quad \exists l\in input(a)/l\in Lien\_entrant \\ e_{max} \quad sinon\end{array}\right\}.$$

Nous proposons, comme pour l'assertion des données, de convertir les critères d'efficacité en entiers dans l'intervalle $[-2,2]$. L'addition des valeurs des données utiles peut alors être réalisée. Puis, en convertissant la somme entière obtenue en nombre réel, il devient possible de réaliser la division. Le résultat de la division appartient forcément à l'intervalle des réels $[-2,2]$. Nous retournons alors dans de domaine des entiers à l'aide de l'arrondi. Afin d'avoir une approche pessimiste, cet arrondi est égal à l'entier inférieur ; il s'agit de la partie entière, notée $E: \mathbb{R} \rightarrow \mathbb{Z}$.

4. L'agrégation concrète des puissances environnementales est également légèrement différentes de celle qui s'applique aux concepts, car ces derniers se basent sur un domaine infini de puissance environnementale. Du fait du nombre limité de valeurs qualitatives permettant d'évaluer concrètement la puissance environnementale et que les entrées ne peuvent prendre que les valeurs « Null » et « Strong », désignant la présence ou l'absence de chaque condition environnementale, nous pouvons traduire cette loi de composition par des formules logiques :
   - si toutes les puissances environnementales sont « Null » alors le résultat vaut « Null » ;
   - si deux puissances environnementales ou plus prennent la valeur « Strong » alors le résultat vaut « Very_Strong » ;
   - sinon le résultat vaut « Strong ».

2.3.2 Modélisation de l'activation des activités

Les valeurs agrégées calculées précédemment nous permettent de déterminer l'état d'activation de chaque activité.

A partir du squelette des activités, présenté dans la table 5.6, nous illustrons la modélisation de l'activation dans la table 5.7. Il s'agit du code AltaRica correspondant à l'activation d'une opération. L'activation d'une fonction peut-être déduit de cette table en considérant que val_p prend toujours la valeur « true ». La table suivante montre bien qu'une activité est activée si ses liens de procédure sont tous actifs et si toutes les données nécessaires sont reçues.

```
1   node Opération_activation
2     flow
3   //Variables d'entrée
4   […]
5   //Variables d'agrégation
6    val_P:bool:private;
7    val_N:MBFHA_types_Effectiveness_5:private;
8    val_U:MBFHA_types_Effectiveness_5:private;
9    val_E:MBFHA_types_Environmental_power:private;
10  //Variable d'activation
11  Act:bool:private;
12    assert
13  //Calcul des valeurs agrégées
14  […]
15  //Calcul de l'activation
16  Act = (val_P AND (val_N!=None));
17  edon
18
```

Tab. 5.7 - Code AltaRica de l'activation d'une opération

2.3.3 Modélisation des pannes fonctionnelles

D'après la section 2.1.4, les pannes fonctionnelles modélisées sont la perte totale, les pertes partielles symétriques et asymétriques ainsi que les fonctionnements erronés symétriques et asymétriques. L'automate décrivant les transitions entre le mode de fonctionnement nominal et ces états de panne est présenté à la section 3.1.3 du chapitre 4 et rappelé à la section 1.2.2 de ce chapitre. Nous avons montré que les pannes sont modélisées à l'aide d'événements évalués dans l'ensemble $C_{act} - \{nominal\}$ et de transitions.

Le code AltaRica correspondant est celui de la table 5.8. Les pannes sont modélisées sous la forme d'événements et des transitions permettent de modifier l'état interne de fonctionnement de la fonction selon les pannes qu'elle subit.

```
1   node Fonction_pannes
2     flow
3   […]
4     state
5   //Variable interne
6       status:MBFHA_types_Failure_modes;
7     event
8       total_loss,
9       partial_loss_sym,
10      partial_loss_asym,
11      error_sym,
```

```
12          error_asym;
13        trans
14       (status = Nominal) |- partial_loss_sym ->
15      status := Partially_lost_sym;
16       (status = Nominal) |- partial_loss_asym ->
17      status := Partially_lost_asym;
18       (status = Nominal) |- total_loss ->
19      status := Totally_lost;
20       (status = Nominal) |- error_sym ->
21      status := Erroneous_sym;
22       (status = Nominal) |- error_asym ->
23      status := Erroneous_asym;
24        assert
25      […]
26        init
27          status := Nominal;
28      edon
```

Tab. 5.8 - Code AltaRica de la modélisation des pannes fonctionnelles

Remarque. Nous constatons qu'initialement l'état interne de la fonction est nominal. Ce choix est validé si la fonction considérée est bien en fonctionnement nominal dans la situation initiale. L'initialisation dépend en effet des analyses (voir la section 4.1 du chapitre 4).

2.3.4 Loi de comportement

A présent que les valeurs agrégées des variables d'entrée et les pannes sont définies, il ne reste plus qu'à concrétiser les lois de comportement à l'aide d'assertions : il s'agit de calculer la valeur des services de chaque activité. Comme nous l'avons vu dans la section 2.5.3 du chapitre 4, les services se caractérisent par un type (service principal ou asymétrique par exemple), une valeur nominale d'efficacité et une pire valeur d'efficacité. De plus, les activités ont éventuellement des capacités de détection et de recouvrement des pannes (voir la section 3.1.2 du chapitre 4). Il faut donc définir une loi de comportement pour chacun des cas rencontrés. La table 5.9 illustre la loi de comportement pour le service principal d'une fonction, qui contribue intégralement à la réalisation du résultat considéré dans le mode nominal (valeur nominale « High »), dont la pire valeur de service est égale à la pire valeur d'efficacité (valeur « Hneg »), et qui n'a aucun moyen de détection ni de recouvrement des pannes. Pour simplifier la présentation, la fonction représentée ci-dessous est supposée ne pas pouvoir être dégradée par des conditions environnementales.

```
1     node Fonction_service_principal
2       flow
3     //Variables d'entrée
4     //Variables d'agrégation
5     //Variable d'activation
6     //Variables de sortie – Service principal
7      Sp:MBFHA_types_Effectiveness_5:out;
8     […]
9       assert
10    //Calcul des valeurs agrégées
11    //Calcul de l'activation
12    //Calcul du service principal Sp
13  Sp := case {
14  not(Act) : None,
15  (((status = Nominal) and (val_N = High)) and
16        ((val_U = High) or (val_U = Low))) : High,
17  ((((status = Partially_lost_sym) or (status =
18        Partially_lost_asym)) and (val_N = Low)) and
19        ((val_U = None) or (val_U = Lneg))) : None,
20  (((status = Nominal) and (val_N = High)) and
21        (val_U = Hneg)) : None,
22  ((((status = Nominal) and (val_N = Low)) or
23        (((status = Partially_lost_sym) or (status =
24        Partially_lost_asym)) and (val_N = High))) and
25        (val_U = Lneg)) : None,
26  (((status = Erroneous_sym) or (status =
27        Erroneous_asym)) or (val_N = Hneg)) : Hneg,
28  (((((status = Nominal) or (status =
29        Partially_lost_sym)) or (status =
30        Partially_lost_asym)) and (val_N = Lneg)) and
31        ((val_U = Lneg) or (val_U = Hneg))) : Hneg,
32  (((status = Nominal) and (val_N = High)) and
33        ((val_U = None) or (val_U = Lneg))) : Low,
34  ((((status = Nominal) and (val_N = Low)) or
35        (((status = Partially_lost_sym) or (status =
36        Partially_lost_asym)) and (val_N = High))) and
37        (((val_U = High) or (val_U = Low)) or (val_U =
38        None))) : Low,
39  ((((status = Partially_lost_sym) or (status =
40          Partially_lost_asym)) and (val_N = Low)) and
41          ((val_U = High) or (val_U = Low))) : Low,
42  else Lneg};
43        init
44          status = Nominal;
45     edon
```

Tab. 5.9 - Code AltaRica de la déclaration du service principal

Remarque 1. La valeur du service principal découle des propriétés logiques sur les valeurs agrégées et l'état interne de la fonction. Les principes qui régissent ces propriétés sont propres aux services principaux :

- si la fonction n'est pas activée, alors son service principal vaut « None ». Ce cas résulte du principe fondamental de l'activation ;
- si la fonction est en mode nominal alors son service principal vaut « High ».
- si la fonction est totalement perdue, alors son service principal vaut « None », d'après l'impact de la perte totale, quelles que soient les valeurs des données d'entrée ;
- si la fonction fonctionne de façon erronée alors son service principal est le moins efficace possible (i.e. « Hneg »). Comme il n'est pas possible de diminuer davantage cette efficacité, le service principal garde cette pire valeur quelles que soient les valeurs des données d'entrée ;
- si la valeur agrégée des données nécessaires est inférieure ou égale à la pire efficacité, alors le service prend comme valeur sa pire efficacité. Avec ce choix, nous considérons dans les modèles AltaRica que les données nécessaires sont des prérequis indispensables à la bonne exécution de l'activité, en leur donnant un poids de dégradation très important ;
- dans les autres cas où la fonction est dans un état nominal ou partiellement perdue ou que la valeur agrégée de ses données nécessaires est supérieure à la pire efficacité, l'efficacité du service principal a une valeur comprise entre la meilleure et la pire efficacité ; la dégradation étant progressive par rapport aux dégradations des données. Ces combinaisons sont nombreuses comme en atteste la table 5.9.

Remarque 2. La complexité de cette loi de comportement montre bien la difficulté pour la maîtriser et la valider. Aujourd'hui, seuls les résultats d'analyses, et en particulier la simulation, permettent de réaliser cette validation.

2.3.5 Représentation graphique des activités

Dans les modèles d'analyse des aspects dysfonctionnels, nous nous focalisons sur les fonctions. Par conséquent, les opérations ont une représentation graphique simple, dans OCAS et RAMSES, ne trahissant cependant aucune de leurs caractéristiques (voir la figure 5.4), à part leurs interfaces potentielles :

- l'interface de gauche (carré creux) est une entrée de flux d'activation ;
- l'interface de droite (carré plein) est l'interface de sortie des flux d'activation ;
- les interfaces du bas (carré creux ou carré plein respectivement) sont les entrées ou sorties des données ;
- les interfaces du haut (carré creux) sont les entrées de puissance environnementale.

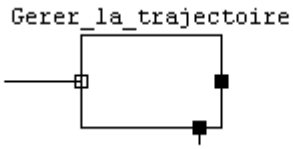


Fig. 5.4 - Représentation graphique d'une opération

Graphiquement, dans les outils OCAS ou RAMSES, les fonctions changent d'icône par rapport à leur état interne et de l'efficacité nominale de leur service principal.

Si l'efficacité de leur service principal est élevé (i.e. « High »), alors les fonctions sont représentées avec deux barres horizontales (voir la figure 5.5). Quand les deux barres sont noires, alors la fonction n'est pas activée ; dans les autres cas, la fonction est activée. Les deux barres vertes signifient que la fonction est dans son état nominal de fonctionnement. Une barre rouge traduit la perte partielle symétrique ou asymétrique. Les deux barres rouges signifient la perte totale de la fonction. Enfin, les deux barres bleues signifient que la fonction fonctionne de façon erronée, symétriquement ou asymétriquement.

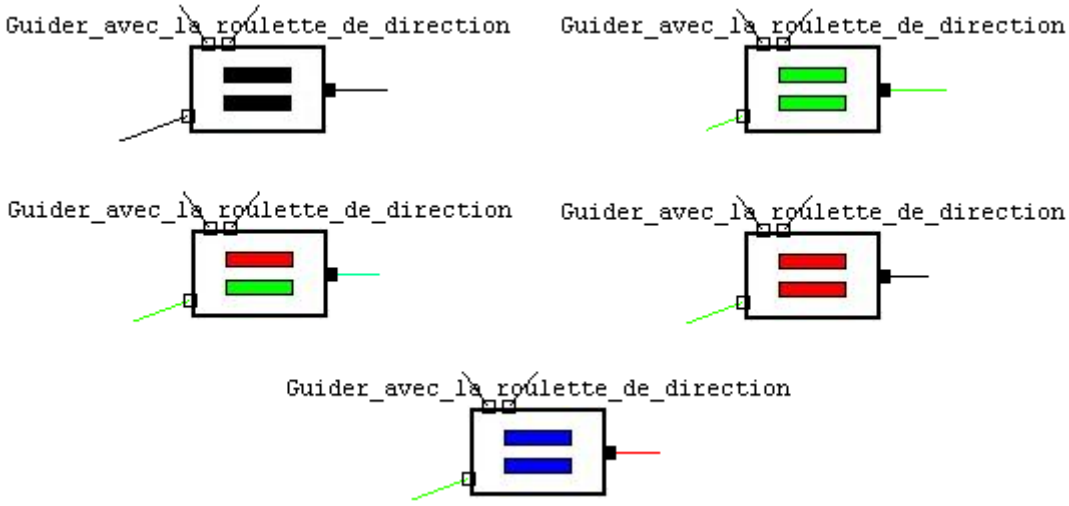


Fig. 5.5 - Représentation graphique d'une fonction (1)

Si l'efficacité de leur service nominal est faible (i.e. « Low »), alors les fonctions sont représentées avec une unique barre horizontale (voir la figure 5.6). Le code couleur a ensuite la même signification que précédemment. Simplement, les pertes partielles et totales ne sont plus distinguées graphiquement et conduisent toutes à une représentation avec une barre rouge.

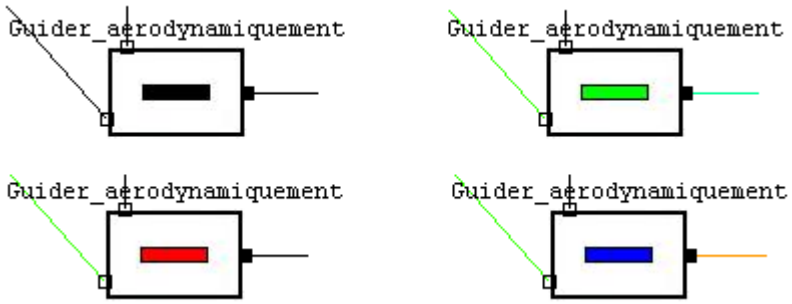


Fig. 5.6 - Représentation graphique d'une fonction (2)

Ces représentations montrent également les interfaces des fonctions :

- les interfaces de gauche (carré creux) sont les données d'entrée ;
- les interfaces de droite (carré plein) sont les services ;
- les interfaces du haut (carré creux) sont les entrées de puissance environnementale.

2.4 Représentation du modèle global

Les modèles AltaRica correspondant aux concepts MB_{FHA} sont obtenus par compositions des nœuds représentant les activités, les données et les conditions environnementales.

Le modèle AltaRica nous permettant d'illustrer les moyens d'analyses est celui du contrôle de la trajectoire au sol (voir la figure 5.7). Ce modèle est une partie du modèle plus général présenté au chapitre 7.

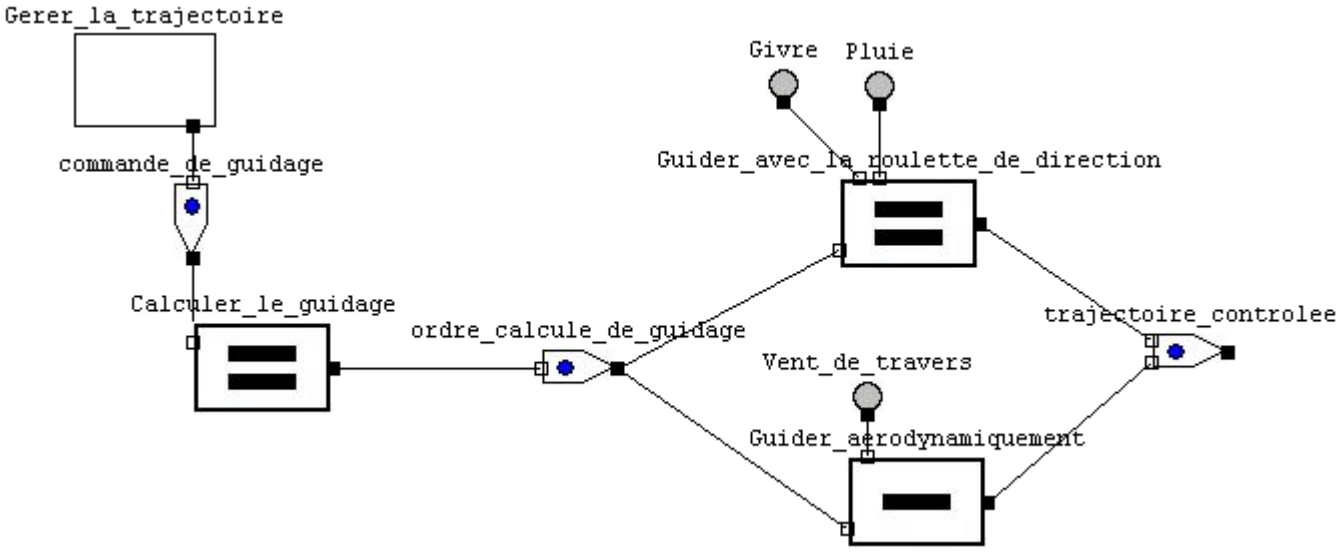


Fig. 5.7 - Modèle AltaRica du contrôle de la trajectoire

Comme nous l'avons vu dans la section 4.1.2 du chapitre 4, un tel modèle peut être préparé en vue des

analyses en identifiant des situations redoutées pertinentes. Ces situations redoutées sont des vecteurs particuliers de valeur des résultats du modèle. La situation redoutée est atteinte si au moins une de ces valeurs est inférieure à sa valeur seuil. La définition des seuils est réalisée dans une nouvelle famille de nœuds, appelée « Observateur » ([MBSA Guideline, 2011]).

Définition : Observateur

Un observateur dans un modèle AltaRica est un composant décrivant les situations redoutées. Dans l'approche MBSA (voir la section 4.2 du chapitre 2), un observateur représente la formalisation d'une condition de pannes (« Failure Condition », FC, voir la section 3.3.1 du chapitre 1).

Concrètement, un observateur est un nœud AltaRica composé d'une sortie booléenne. Cette sortie est vraie si la situation redoutée est atteinte. Les observateurs possèdent un grand nombre d'entrées représentant la capture de différentes variables du modèle, nécessaires pour analyser la situation redoutée. Dans le cadre de nos modèles de sécurité, les observateurs s'intéressent aux valeurs des résultats du modèle. La table 5.10 présente un observateur pour le modèle du contrôle de la trajectoire pendant le décollage. Un seul résultat est observé : le résultat de l'objectif de vol « trajectoire contrôlée ». La valeur de seuil que nous avons choisie est « None » : nous recherchons donc les situations conduisant à un contrôle de la trajectoire négatif ou nul.

```
1   node Observateur_controle_trajectoire
2     flow
3   //Variables d'entrée – trajectoire contrôlée
4    traj_contr:MBFHA_types_Effectiveness_5:in;
5   //Variables de sortie
6    situation_redoutee:bool:out;
7     assert
8   //Calcul de la situation – la trajectoire
9   contrôlée est inférieure ou égale au seuil
10   situation_redoutee =(traj_contr = Hneg OR
11  traj_contr = Lneg OR traj_contr = None);
12  edon
```

Tab. 5.10 - Code AltaRica d'un observateur sur le contrôle de la trajectoire

Nous pouvons constater que contrairement aux activités, aux données et aux conditions environnementales, les nœuds « Observateur » ont des assertions dépendantes des entrées observées. Ils sont donc propres à leur modèle. Nous constatons également que ces nœuds n'ont pas leur équivalent dans les modèles A.O.F. ; leur définition est alors entièrement à la charge des analystes de la sécurité.

A ce stade, nous disposons de modèles complets et outillés pour les analyses.

3 Exploitation et analyse des modèles AltaRica

L'objectif des modèles concrets en AltaRica est de disposer d'outils d'analyses pour exploiter les modèles de sécurité afin d'assister l'analyse des risques. Dans le chapitre 2, nous avons présenté deux outils d'analyse, implémentés dans les outils d'édition graphique OCAS et RAMSES : un simulateur et un générateur de séquences. Dans la section 4 du chapitre 4, nous avons développé les principes d'analyse que nous souhaitons mettre en œuvre concrètement. La recherche des conséquences d'un scénario de pannes fait l'objet de la section 3.1 et s'appuie sur le simulateur. La recherche des causes de situations redoutées fait quant à elle l'objet de la section 3.2 et s'appuie sur le générateur de séquences. Dans la section 3.3, nous montrons comment les résultats d'exploitation de nos modèles varient par rapport aux ensembles des valeurs du domaine. En particulier, nous justifions notre choix d'une modélisation de l'efficacité par cinq critères qualitatifs et d'une modélisation de la puissance environnementale à partir de trois autres critères qualitatifs.

3.1 Simulation des modèles de sécurité

Comme la modélisation concrète en AltaRica que nous avons proposée est conforme à la formalisation des concepts présentés au chapitre 4, la recherche des conséquences d'un scénario de pannes peut être réalisée sur les modèles AltaRica. Or, un outil existe déjà dans OCAS et RAMSES pour faire une telle recherche de conséquences des pannes : il s'agit du simulateur, introduit à la section 4.3.4 du chapitre 2.

L'autre bénéfice attendu de la simulation est de permettre la validation du comportement de nos modèles de sécurité par des campagnes de tests.

Les différentes étapes permettant de réaliser un calcul de conséquences d'un scénario de pannes à l'aide du simulateur sont les suivantes. Elles sont regroupées en trois familles : lancement de la simulation, simulation des pannes fonctionnelles et des conditions environnementales et manipulation des simulations.

3.1.1 Lancement de la simulation dans OCAS et RAMSES

1. Configuration des états internes initiaux : dans les modèles de sécurité, les états internes sont les états de fonctionnement des fonctions et les puissances environnementales des conditions environnementales. Ces états internes doivent être définis avec une valeur initiale afin que le simulateur puisse calculer la situation initiale (voir l'étape 2). Les états initiaux sont déclarés dans les nœuds du modèle à l'aide du mot clé « init » (voir les tables 5.9 et 5.11 par exemple).
2. Lancement de la simulation : le principe de calcul de la simulation est la détermination de la valeur de toutes les variables de sortie du modèle en s'appuyant sur les états initiaux. A ce stade, aucune transition n'est déclenchée. Le calcul des variables de sortie s'effectue grâce aux assertions et est rendu possible du fait que les modèles sont dirigés, acycliques et bornés. La figure 5.8 illustre le lancement d'une simulation à partir de la situation nominale comme situation initialement choisie. Les liens vert et bleu signifient respectivement la transmission d'une efficacité « High » et « Low ».

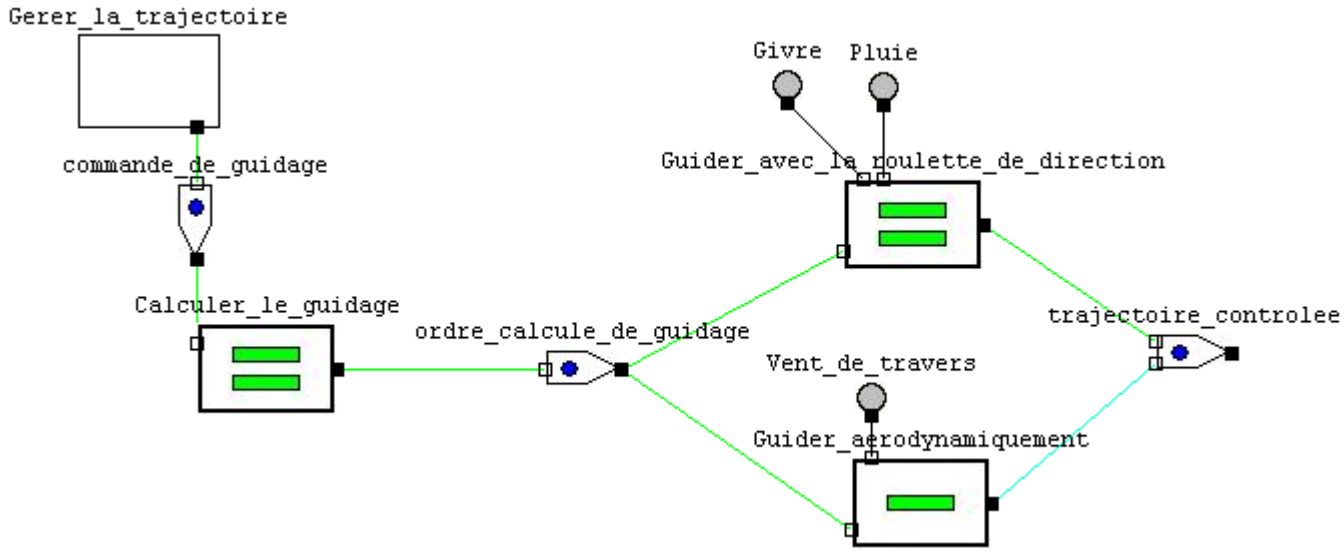
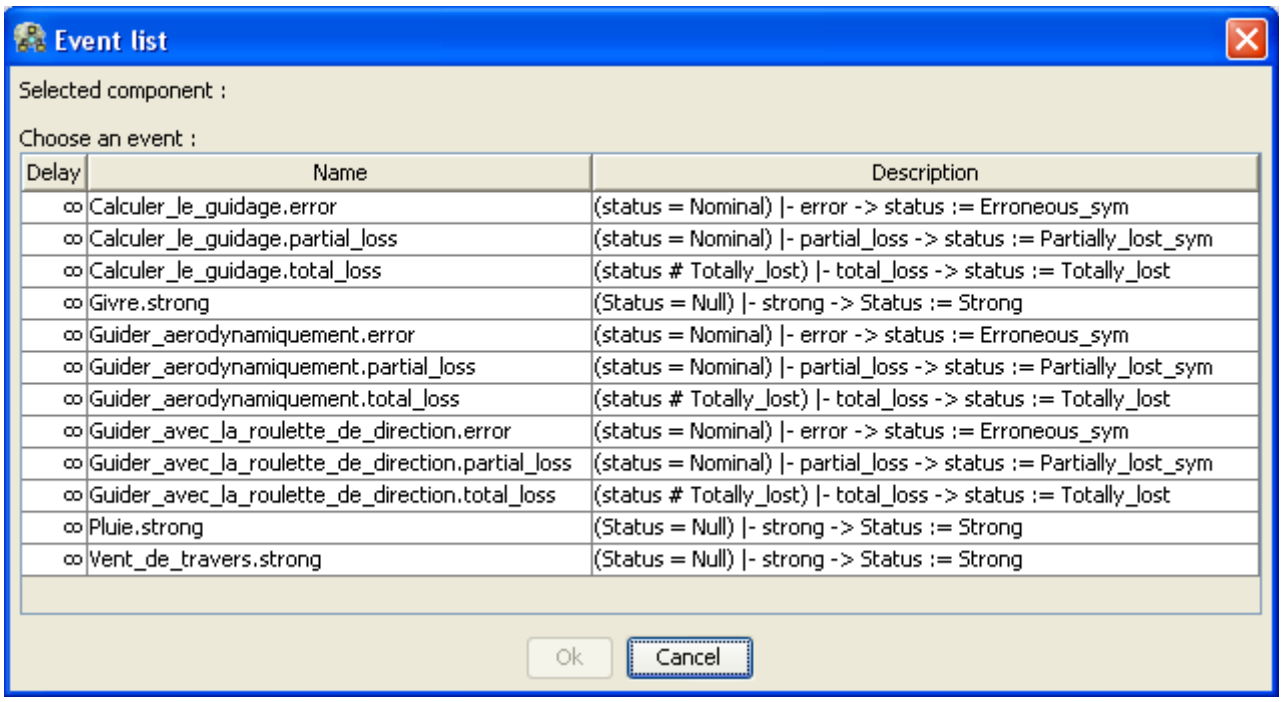


Fig. 5.8 - Simulation : situation nominale du modèle

3.1.2 Simulation des pannes fonctionnelles et des conditions environnementales

3. Choisir un événement : une fois la situation initiale calculée, le simulateur permet de déclencher des événements parmi une liste mise à disposition par le simulateur. Cette liste contient tous les événements du modèle (pannes fonctionnelles et présence de conditions environnementales) qui sont liés à au moins une transition dont la garde est satisfaite. Cette liste (voir la figure 5.9) réduite évite ainsi que l'utilisateur déclenche des événements qui ne modifient aucune valeur du modèle.



Delay	Name	Description
∞	Calculer_le_guidage.error	(status = Nominal) |- error -> status := Erroneous_sym
∞	Calculer_le_guidage.partial_loss	(status = Nominal) |- partial_loss -> status := Partially_lost_sym
∞	Calculer_le_guidage.total_loss	(status # Totally_lost) |- total_loss -> status := Totally_lost
∞	Givre.strong	(Status = Null) |- strong -> Status := Strong
∞	Guider_aerodynamiquement.error	(status = Nominal) |- error -> status := Erroneous_sym
∞	Guider_aerodynamiquement.partial_loss	(status = Nominal) |- partial_loss -> status := Partially_lost_sym
∞	Guider_aerodynamiquement.total_loss	(status # Totally_lost) |- total_loss -> status := Totally_lost
∞	Guider_avec_la_roulette_de_direction.error	(status = Nominal) |- error -> status := Erroneous_sym
∞	Guider_avec_la_roulette_de_direction.partial_loss	(status = Nominal) |- partial_loss -> status := Partially_lost_sym
∞	Guider_avec_la_roulette_de_direction.total_loss	(status # Totally_lost) |- total_loss -> status := Totally_lost
∞	Pluie.strong	(Status = Null) |- strong -> Status := Strong
∞	Vent_de_travers.strong	(Status = Null) |- strong -> Status := Strong

Fig. 5.9 - Liste des événements déclenchables du modèle

4. Calcul de l'effet du déclenchement d'un événement : Une fois un événement déclenché, le simulateur recalcule les sorties des nœuds ayant une transition tirée par l'événement, puis recalcule de proche en proche les sorties des nœuds suivants par itération pour mettre à jour les variables du modèle qui sont modifiées par la propagation des impacts des changements d'état. La figure suivante montre les effets du fonctionnement erroné de la fonction « Calculer le guidage ». Les liens rouges et oranges signifient respectivement la propagation d'efficacité « HNeg » et « LNeg ».

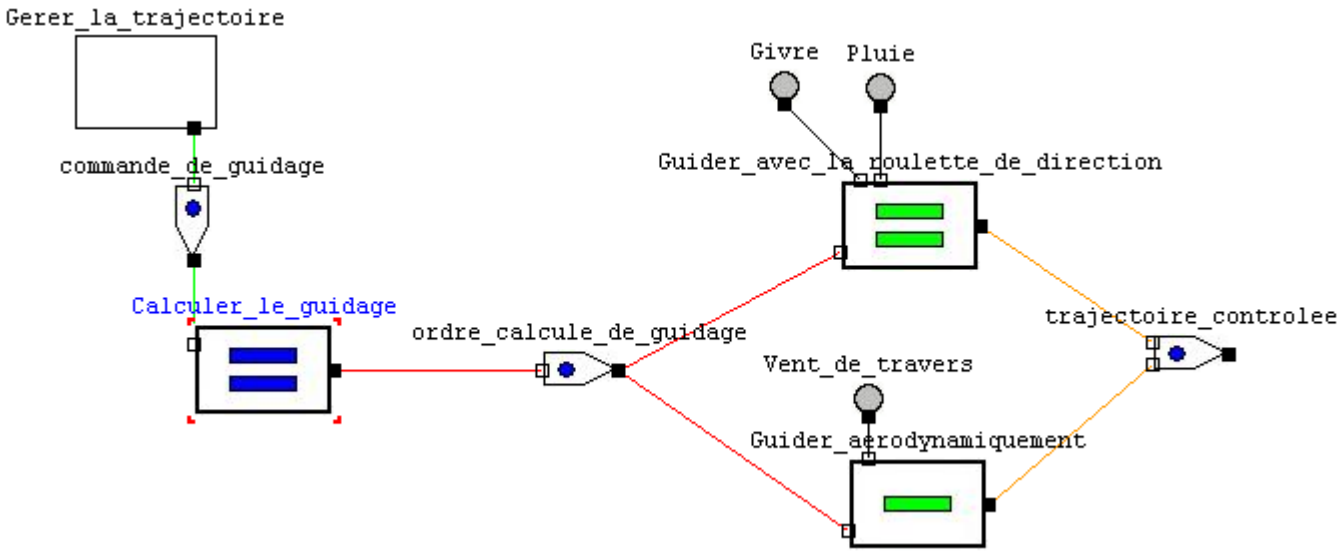


Fig. 5.10 - Simulation : situation avec le fonctionnement erroné du calcul du guidage

3.1.3 Manipulation des simulations

Le simulateur offre plusieurs services annexes, tels que :

5. Sauvegarde de campagne de simulation : toutes les simulations réalisées peuvent être sauvegardées. Le simulateur offre la possibilité de charger un tel fichier de sauvegarde pour rejouer une campagne de simulation. Il nous est également possible de construire ces fichiers manuellement ou à l'aide d'autres outils.
6. Recherche des variables modifiées : le simulateur offre diverses vues des valeurs des variables du modèle dont la vue graphique de la figure 5.10, mais également des tables de valeurs avec des filtres permettant à l'analyste de se focaliser sur des variables précises. Il est en particulier possible d'extraire automatiquement et de visualiser le sous ensemble des variables qui ont subi un changement de valeur depuis la situation initiale ou depuis le dernier pas de simulation (i.e. le dernier événement déclenché).

3.2 Recherche de causes de situations redoutées en AltaRica

Le concept de la recherche des causes de situations redoutées est présenté dans la section 4.3 du chapitre 4. Ce principe de recherche se base tout d'abord sur la définition d'une situation initiale à partir des états initiaux des nœuds « Fonction » et « Environnement ».

3.2.1 Utilisation du générateur de séquence

Le fonctionnement d'un observateur peut être analysé à l'aide de la simulation. Il peut également assister l'utilisation de la simulation. Mais il permet surtout d'utiliser le générateur de séquence d'OCAS ou de RAMSES pour rechercher les causes d'une situation redoutée. Cet outil s'appuie sur quatre informations.

1. Une valeur d'une variable représentant la situation redoutée : d'après la définition d'un observateur, la situation redoutée est atteinte quand la variable de sortie de l'observateur prend la valeur booléenne « true ».
2. Choix d'un ordre : l'outil va rechercher les événements qui conduisent à la valeur précédente. Dans les modèles de sécurité, les événements sont des pannes fonctionnelles ou des présences de conditions environnementales. L'ordre est une valeur entière représentant le cardinal maximal des ensembles d'événements recherchés. En pratique, comme nous le verrons dans la chapitre 7, nous nous intéressons principalement aux pannes doubles avec éventuellement la présence d'une condition environnementale. L'ordre 3 est donc suffisant. Notons que plus l'ordre est important, plus la recherche des causes est longue.
3. Choix des séquences retenues : l'outil dispose de plusieurs algorithmes de recherche des séquences d'événements conduisant à la situation redoutée choisie. La base de cet algorithme est similaire pour les différentes variations : il s'agit du parcours des événements du modèle sous la forme d'un arbre en arrêtant la recherche en profondeur quand la situation redoutée est atteinte ou quand l'ordre est atteint. Il existe trois variations :

   - Combinaison : dans ce mode, les événements ne peuvent être déclenchés qu'une seule fois. De plus, l'ordre de déclenchement des événements n'est pas pris en considération. Ce mode ne doit donc être utilisé que lorsque l'ordre d'apparition des pannes n'entre pas en compte dans le calcul des situations courantes d'un modèle.
   - Permutation : dans ce mode également, les événements ne sont déclenchés qu'une seule fois dans une même séquence. En revanche, l'ordre de déclenchement des événements est pris en compte. Ainsi, toutes les permutations d'événements seront testées par l'outil.
   - Répétition : ce mode, en plus de s'intéresser aux permutations d'événements, étudie également le déclenchement répétitif d'un même événement. Ce mode permet par exemple de prendre en compte plusieurs « reset » d'un composant.

Dans le cadre de nos modèles MB_{FHA}, la répétition n'a aucun sens car nos automates de panne ne contiennent aucune boucle. Quand une fonction subit une panne fonctionnelle, nous n'avons pas modélisé une potentielle aggravation supplémentaire de cette fonction. De même, l'ordre d'apparition des pannes n'a pas d'impact sur le modèle. Pour s'en convaincre, il suffit de relire la section 1.2.1 dans laquelle nous avons montré que la sémantique de nos modèles de sécurité ne nécessite pas a priori l'expression d'événements ou de transitions. Nous sommes donc en présence de modèles statiques. Nous pouvons constater en revanche que l'utilisation des événements et des transitions facilite grandement l'exploitation des modèles en AltaRica. Nous choisissons alors le mode de recherche

nommé « Combinaison ».

4. Format de sauvegarde des séquences : à partir du choix du point 3, l'outil identifie toutes les séquences d'événements conduisant à la situation redoutée. Mais il est également capable, par option, de limiter les résultats qui renvoient aux seules coupes minimales (voir la section 4.1.1 du chapitre 2). Généralement, ce sont les coupes minimales qui nous intéressent. Les résultats sont enregistrés dans un fichier de données sous différents formats.

3.2.2 Résultats du générateur de séquences

La table 5.11 présente un extrait du résultat de la recherche des causes à l'ordre 3 sur l'exemple du contrôle de la trajectoire au sol, pour la situation redoutée exprimée par l'observateur de la table 5.10, à partir du fonctionnement nominal. Nous remarquons alors que pour ce modèle, la situation redoutée est atteinte par deux pannes simples et huit pannes doubles. Parmi les combinaisons de pannes présentées dans la table ci-dessous, nous constatons que, comme prévu, le fonctionnement erroné de la fonction de calcul de la trajectoire conduit bien à la situation redoutée. Nous avions vu ce cas lors de la simulation (voir la figure 5.10 de la section 3.1.2).

```
{Calculer_le_guidage.error}
{Guider_avec_la_roulette_de_direction.error}
{Calculer_le_guidage.total_loss ;
        Guider_aerodynamiquement.error}
{Givre.strong ; Guider_aerodynamiquement.error}
[…]
```

Tab. 5.11 - Résultat de la recherche de séquences d'événements

Dans le chapitre 7, nous présentons sur un cas d'étude comment ces résultats peuvent assister l'analyse des risques.

3.3 Étude de la granularité de l'efficacité

La pertinence des analyses de recherche des conséquences et de recherche des causes, présentée précédemment, dépend fortement de la confiance que nous pouvons avoir en notre modélisation de l'efficacité et de la puissance environnementale par des critères qualitatifs. Nous avons justifié l'utilisation de valeurs qualitatives par la nécessité que les concepteurs de l'avion et les analystes de la sécurité puissent interpréter ces critères. En revanche, nous pouvons nous interroger sur le nombre de valeurs qualitatives à modéliser.

3.3.1 Principe d'équilibre pour la recherche des causes de situations redoutées

Pour l'efficacité, l'utilisation des valeurs qualitatives présente trois inconvénients majeurs.

- Approximations dans la définition des valeurs des services nominaux : pour le service principal par exemple, les analystes de la sécurité n'ont le choix qu'entre la valeur « Low » (50% d'efficacité) ou la valeur « High » (100% d'efficacité). Alors, afin d'être pessimiste quand à la sécurité, s'ils souhaitaient modéliser un service principal comme contributeur à 70%, ils choisiront la valeur « Low ».
- Approximation dans les conséquences des pannes sur les services : prenons la perte partielle par exemple. Pour le service principal, une telle panne conduit à une division par deux de son efficacité. Ainsi, la valeur « High » devient « Low » très logiquement, mais la valeur « Low » devient « None » car il n'existe pas de valeur quantitative signifiant une contribution de 25%. La perte partielle est alors modélisée comme la perte totale du fait de l'imprécision de la définition concrète de l'efficacité.
- Approximation dans les conséquences des données d'entrées sur les services : comme pour les pannes, une dégradation du fait de données utiles ou nécessaires dégradées n'est pas toujours modélisée parfaitement, mais systématiquement avec une valeur pessimiste.

Pour bien se rendre compte des conséquences de ces approximations, nous avons d'étudié une nouvelle modélisation de l'ensemble des efficacités, à partir de sept et non plus cinq valeurs qualitatives. Nous définissons alors $E=\{\text{High, Medium, Low, None, Lneg, Mneg, Hneg}\}$ avec :

- « High » correspondant au taux de 100% ;
- « Medium » correspondant au taux de 50% ;
- « Low » correspondant au taux de 25% ;
- « None » correspondant au taux de 0% ;
- « Lneg » correspondant au taux de -25% ;
- « Mneg » correspondant au taux de -50% ;
- « Hneg » correspondant au taux de -100%.

Nous avons réalisé une étude comparative entre cette modélisation en sept valeurs et la modélisation en cinq valeurs telle que décrite tout au long de ce chapitre. Nous présentons nos résultats sur deux modèles :

- le modèle sur le contrôle de la trajectoire au sol que nous avons utilisé comme exemple dans ce chapitre. Il s'agit d'un modèle composé de neuf nœuds (observateur compris) et de douze événements. Nous l'appelons par la suite « Modèle simple ».
- un modèle plus important concernant le décollage, composé de 31 nœuds et 43 événements. Nous l'appelons par la suite « Modèle complet ».

Pour ces deux modèles, nous avons construit deux versions, l'une avec cinq niveaux d'efficacité et l'autre

avec les sept niveaux d'efficacité. Bien sûr, nous y retrouvons les mêmes activités, données, conditions environnementales et observateurs. En revanche, le code AltaRica des nœuds s'adapte aux différentes modélisation de l'efficacité. Nous réalisons alors une série de génération de séquences et nous montrons dans la table 5.12 le nombre de séquences de chaque ordre. Nous précisons que les conditions environnementales ont été écartées dans le cadre de cette analyse (elles font l'objet de la section 3.3.2). Ainsi, sur le modèle du contrôle de la trajectoire, les cinq coupes minimales d'ordre 2 correspondent bien aux huit coupes identifiés dans la table 5.11 car trois des huit coupes étaient construites à partir de conditions environnementales. Il en va de même pour les coupes d'ordre 3.

	Coupes minimales d'ordre 1	**Coupes minimales d'ordre 2**	**Coupes minimales d'ordre 3**	**Total des coupes minimales**
Modèle simple 5 niveaux d'efficacité	2	5	0	7
Modèle simple 7 niveaux d'efficacité	2	4	0	6
Modèle complet 5 niveaux d'efficacité	0	46	46	92
Modèle complet 7 niveaux d'efficacité	0	29	18	47

Tab. 5.12 - Nombre des coupes minimales jusqu'à l'ordre 3 pour différents modèles

Le nombre de coupes minimales à chaque ordre n'est pas important en soi. En revanche, nous constatons une diminution du nombre de coupes minimales identifiées pour les mêmes observateurs lorsque nous augmentons les niveaux d'efficacité. Cette conséquence est logique puisque nos approximations sont toujours pessimistes ; une modélisation plus fine permet alors d'être un peu moins pessimiste. Toutefois, la diminution du nombre de coupes minimales identifiées comporte deux risques :

- nous risquons en premier lieu de ne pas identifier une combinaison de pannes qui pourrait potentiellement être significative pour l'analyse des risques ;
- puis, l'augmentation du nombre de valeurs qualitatives accroît de façon significative la difficulté de modélisation des lois de comportement des activités en AltaRica, puisque les formules logiques doivent prendre en compte des valeurs supplémentaires pour les entrées et les pannes. Il peut également être plus difficile pour les analystes de déterminer les valeurs d'efficacité nominales des services.

Le problème de granularité dans la modélisation de l'efficacité est donc un problème de compromis entre la précision des résultats et la difficulté de modélisation. Des travaux supplémentaires de modélisation sont

nécessaires pour identifier la granularité adéquate. Il semble que la modélisation basée sur les cinq niveaux d'efficacité est un bon compromis, comme nous le verrons dans le chapitre 7.

3.3.2 Principe d'équilibre pour l'étude des impacts des conditions environnementales

Pour l'analyse des risques, le plus important est d'identifier et d'analyser les combinaisons de pannes les plus pertinentes. L'étude des impacts des conditions environnementales dans les cas de pannes est de l'ordre des perspectives d'amélioration de l'analyse des risques. La prise en compte des conditions environnementales ne doit donc pas occulter l'objectif initial de l'analyse des risques vis à vis des pannes fonctionnelles. Dans les modèles de sécurité, ce principe signifie que l'équilibre entre les impacts des pannes et les impacts des conditions environnementales doit pencher de façon mesurée du coté des pannes fonctionnelles. Nous justifions ainsi notre modélisation des impacts des conditions environnementales que nous pouvons considérer comme :

- une modélisation grossière du fait du faible nombre de valeurs qualitatives pour évaluer la puissance des conditions environnementales. Nous nous focalisons en effet exclusivement sur la présence ou l'absence des différentes conditions environnementales ;
- une modélisation légèrement optimiste au niveau de l'effet de la puissance environnementale dégradante dans la loi de comportement des activités. Une telle modélisation a pour effet de limiter fortement la présence des conditions environnementales dans les coupes minimales d'ordre 1 ou d'ordre 2.

Là aussi il est difficile de juger sans davantage de tests que notre proposition de modélisation est satisfaisante. La table 5.13 montre le nombre de coupes minimales de chaque ordre pour l'exemple complet avec ou sans les conditions environnementales. Ce test, comme plusieurs autres réalisés au cours de nos travaux, indique que :

- aucune coupe minimale d'ordre 1 n'est composée de la présence d'une condition environnementale ;
- peu de coupes minimales d'ordre 2 sont composées d'une condition environnementale.
- La majorité des coupes minimales composées d'une condition environnementale sont d'ordre 3.

	Coupes minimales d'ordre 1	**Coupes minimales d'ordre 2**	**Coupes minimales d'ordre 3**	**Total des coupes minimales**
Modèle complet avec l'environnement	0	49	72	121
Modèle complet sans l'environnement	0	46	46	92

Tab. 5.13 - Nombre des coupes minimales du modèle complet avec ou sans l'environnement

4 Bilan et perspectives sur les modèles dédiés à l'analyse des risques

4.1 Bilan

Dans ce chapitre, nous avons proposé une modélisation concrète des concepts introduits dans le chapitre 4 à l'aide du langage AltaRica. Nous avons montré en particulier que la sémantique des modèles AltaRica proposés respecte les concepts. Puis, grâce aux outils d'exploitation des modèles, basés sur le langage AltaRica, nous avons mis en place des méthodes d'analyse des modèles grâce à la simulation et à la recherche automatique des coupes minimales correspondant à une situation redoutée. Même si la simulation a un intérêt évident pour assister l'analyse des risques grâce aux calculs de la propagation des pannes, il reste encore à démontrer que nos modèles de sécurité peuvent aider efficacement les analystes de la sécurité en charge de l'analyse des risques. A l'exception de ce point, qui fait l'objet du chapitre 7, nous pouvons considérer que les deux objectifs, présentés à la section 1.2.1 du chapitre 3 et rappelés ci-dessous, sont atteints par les chapitres 4 et 5.

Objectif 1. Formaliser les concepts d'une nouvelle approche de modélisation pour assister l'analyse des risques.

Objectif 2. Implémenter et expérimenter les concepts, formalisés précédemment, à l'aide du langage de modélisation AltaRica.

4.2 Perspectives sur la concrétisation des modèles de sécurité

Sur l'étape de concrétisation des concepts introduits au chapitre 4, nous avons identifié deux voies d'amélioration de nos travaux : la modélisation du mode de panne intempestif et l'utilisation d'autres langages de modélisation.

4.2.1 Retour sur le mode de panne intempestif

Dans la section 2.3 du chapitre 5, nous avons présenté notre proposition de modélisation des activités de nos modèles de sécurité. Nous avons volontairement écarté le mode de panne intempestif parmi les modes de pannes fonctionnelles modélisés. Nous avons vu que les modèles proposés sont néanmoins riches en information et aptes à assister efficacement l'analyse des risques. Toutefois, il est évident que certaines combinaisons de pannes ne sont pas traitées par nos modèles de sécurité du fait de la non modélisation de l'activation intempestive potentielle. Nous identifions donc la prise en compte de ce mode de panne comme une importante perspective future. L'annexe C présente les travaux préliminaires réalisés pour prendre en compte le fonctionnement intempestif dans nos modèles AltaRica. Les paragraphes ci-dessous expliquent brièvement en quoi la modélisation de ce mode de panne est problématique.

1. Le fonctionnement intempestif d'une fonction présente dans un modèle A.O.F. traduit une activation de la fonction alors que cette dernière ne devait pas être exécutée. Cela signifie qu'avant la panne fonctionnelle, la fonction ne reçoit pas une ou plusieurs de ses données nécessaires. Or, en fonctionnement nominal, cela est impossible : les résultats de toutes les données ont la valeur d'efficacité maximale. Le fonctionnement intempestif ne peut alors intervenir qu'après une ou plusieurs autres pannes fonctionnelles. Nous nous rendons alors compte que l'impact et l'intérêt de ce type de fonctionnement intempestif sur l'ensemble d'un modèle sont plutôt limités.

2. Le fonctionnement intempestif d'une fonction non modélisée dans le modèle A.O.F. signifie que la fonctionnalité n'est normalement pas utilisée dans la phase de vol considérée, et que son activation et son exécution dégrade les résultats de cette phase de vol. C'est le cas par exemple du freinage intempestif des roues pendant le décollage. Dans ce cas, le fonctionnement intempestif peut être dimensionnant. Malheureusement, les dépendances fonctionnelles liées à ces fonctions ne sont pas modélisées dans les modèles de la conception puisque ces derniers ne considèrent que le fonctionnement nominal : les fonctions non-utilisées nominalement dans les différentes phases de vol ne sont pas représentées ! L'ajout de ces fonctions et de leurs dépendances doit donc se faire manuellement après la génération des modèles AltaRica. Les informations relatives à ces fonctions potentiellement intempestives ne sont actuellement ni complètement maîtrisées par les concepteurs, ni complètement maîtrisées par les analystes de la sécurité. Des travaux supplémentaires sont requis pour mieux modéliser ce mode de panne.

4.2.2 Alternatives au langage AltaRica pour concrétiser les modèles de sécurité

Le choix du langage AltaRica est clairement validé par le fait que les concepts des modèles de sécurité sont compatibles avec la sémantique de ce langage.

Mais, comme nous l'avons vu dans la section 1.2.1, les modèles de sécurité peuvent être modélisés sans utiliser les événements. AltaRica apparaît alors comme un langage un peu trop expressif par rapport à nos besoins de formalisation. Cependant, l'utilisation des événements dans AltaRica permet toutefois d'exploiter plus facilement les modèles à l'aide du simulateur et du générateur de séquences existant dans les outils d'édition graphique OCAS et RAMSES.

Avec une analyse de compatibilité sémantique similaire à ce que nous avons réalisé dans la section 1, nous pouvons montrer que la sémantique du langage HiP-HOPS (voir la section 4.4.2 du chapitre 2) est suffisante pour représenter les modèles de sécurité. Nos travaux n'ont pas exploré l'utilisation concrète de ce langage et des outils d'analyses associés. Des travaux supplémentaires devraient être entrepris pour être certain qu'il puisse remplacer efficacement le langage AltaRica pour des applications industrielles.

Chapitre 6

La transformation de modèle

Sommaire

Résumé. A présent que les modèles de la conception et ceux permettant l'analyse des aspects dysfonctionnels sont définis, il convient de s'intéresser à la formalisation des liens entre ces deux familles de modèles. Ce chapitre rappelle l'approche choisie et présente dans le détail les différentes étapes de la transformation de modèle qui concernent les analystes de la sécurité.

1 Introduction : notre approche de transformation de modèle

Les chapitres 4 et 5 ont permis de définir une syntaxe et une sémantique pour la modélisation abstraite et une modélisation concrète en AltaRica des modèles de sécurité MB_{FHA}, permettant d'assister l'analyse des risques. Conformément à notre approche, la syntaxe de ces modèles de sécurité s'appuie fortement sur celle des modèles ALFA présentés dans le chapitre 2. C'est pourquoi nous avons mis en évidence dans le chapitre 3 l'intérêt de formaliser le lien entre les domaines de modélisation (DSML) des architectures fonctionnelles et de l'analyse des risques dont les modèles abstraits sont M_{ALFA} et MB_{FHA} respectivement.

Rappel de l'objectif 3. Formaliser les interactions entre nos nouveaux modèles de sécurité et les modèles ALFA. Implémenter et expérimenter cette formalisation pour développer un démonstrateur.

Nous nous appuyons pour cela sur les techniques de transformation de modèles issues de l'Ingénierie Dirigée des Modèles. La figure 6.1, ci-dessous, détaille la figure 3.2 du chapitre 3 pour illustrer les différentes étapes de transformation (les flèches pleines) et de métamodélisation au sein de chacun des deux domaines de modélisation.

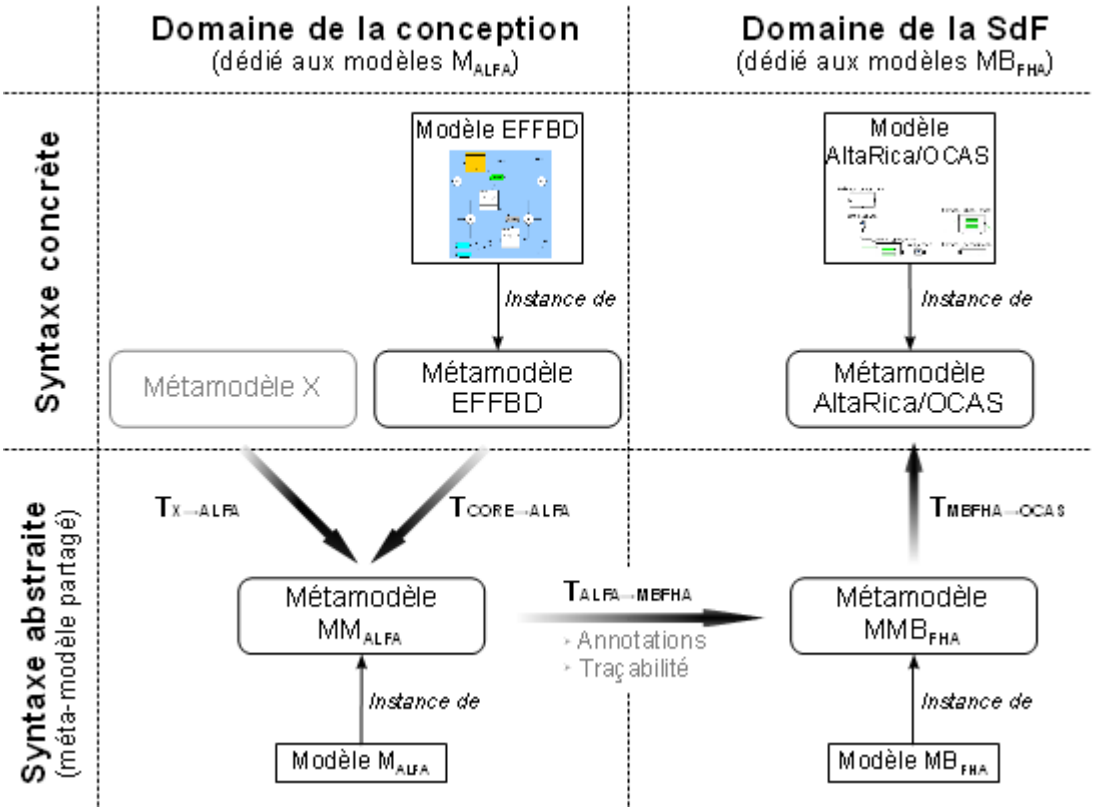


Fig. 6.1 - Application de la transformation de modèle par métamodélisation

Nous retrouvons sur cette figure les deux DSML avec dans la partie supérieure les modèles concrets, sur lesquels les architectes et les analystes de la sécurité font leurs analyses, et dans la partie inférieure les modèles abstraits. Comme nous l'avons vu dans la section 2.2.2 du chapitre 2, le moyen de relier les deux DSML est de réaliser une transformation de type « *model to model* » entre les deux métamodèles de leur syntaxes abstraites. Cette transformation, notée $T_{ALFA \rightarrow MBFHA}$ est endogène horizontale et représente à elle seule les échanges d'informations entre la conception et l'analyse de la sécurité. Cette transformation de modèle fait l'objet de la section 3 de ce chapitre.

Au sein de chacun des DSML, il existe des représentation en syntaxe abstraites des différentes syntaxes concrètes. Pour le domaine des modèles ALFA, ce sont les architectes avion qui sont en charge de la maîtrise de ce DSML. En nous plaçant dans le point de vue de l'analyste de la sécurité, nous ne nous intéresserons donc qu'à la génération des modèles AltaRica au sein du second DSML. Cette transformation de modèle, notée $T_{MBFHA \rightarrow AltaRica}$, fait l'objet de la section 3.2.

2 Principes de nos transformations de modèles

L'approche choisie pour définir les deux transformations de modèle $T_{ALFA \rightarrow MBFHA}$ et $T_{MBFHA \rightarrow AltaRica}$ est incrémentale :

1. définition des règles de transformation ;
2. annotation des modèles sources ;
3. sauvegarde des traces de la transformation.

Les transformations de modèles sont définies à l'aide du langage Express qui permet d'exprimer des procédures en plus des modèles de données que nous avons déjà présentés dans ce mémoire. Elles pourraient être exprimées via d'autres langages de transformation.

2.1 Définition des règles de transformation

Les deux transformations que nous étudions ont la particularité de transformer chaque modèle source en un modèle cible. Le périmètre de ces transformations de modèle est donc limité à une source et à sa cible. De plus, les modèles M_{ALFA}, MB_{FHA} et les modèles AltaRica sont tous décrits sur la base de graphes dirigés acycliques.

2.1.1 La déclaration de la transformation de modèle

Il existe deux façons de définir les modèles de transformation.

1. La première méthode est de définir une génération des modèles cibles à partir de fonctions informatiques. Il s'agit d'une approche fonctionnelle. Le langage Express permet de calculer le modèle cible à l'aide d'attributs dérivés et de fonctions (voir la table 6.1).

```
1     ENTITY Transformation;
2           source : Modèle_source;
3     DERIVE
4     cible : Modèle_cible:=transform(SELF.source);
5     END_ENTITY;
6
7     FUNCTION transform(s:Modèle_source):Modèle_cible;
8     […]
9     END_FUNCTION;
10
11    ENTITY Modèle_source SUBTYPE OF (graph);
12    END_ENTITY;
13
14    ENTITY Modèle_cible SUBTYPE OF (graph);
15    END_ENTITY;
```

Tab. 6.1 - Entité de la transformation de modèle (génération du modèle cible)

2. La seconde méthode est de vérifier des propriétés sur deux modèles pour assurer que les modèles cibles sont bien des modèles générés à partir des modèles sources. Il s'agit d'une approche prédicative, qui nécessite de définir le modèle source et le modèle cible. Avec le langage Express, cette méthode consiste en l'ajout d'une contrainte (voir la table 6.2).

```
1     ENTITY M_T_ALFA_MBFHA;
2           source : Modèle_source;
3           cible : Modèle_cible;
4     WHERE
5           wr1:transform(SELF.source,SELF.cible);
6     END_ENTITY;
7
8   FUNCTION transform(s:Modèle_source,c:Modèle_cible):boolean;
9     […]
10    END_FUNCTION;
```

Tab. 6.2 - Entité de la transformation de modèle (propriétés entre les modèles)

Il est important de noter que ces deux approches appliquent des règles de transformation qui sont identiques. La première approche génère le modèle cible alors que la seconde vérifie que le modèle cible est bien la transformé du modèle source. Nous choisissons de nous placer dans le premier cas.

2.1.2 La procédure de transformation

Les règles de transformation sont présentes dans la fonction « transform » des tables 6.1 et 6.2.

Les modèles ALFA et les modèles d'analyse des aspects dysfonctionnels partagent un squelette syntaxique commun. Ces familles de modèles représentent en effet les mêmes phases de vol, les mêmes activités réalisées, les mêmes données, les mêmes conditions environnementales et les mêmes liens. Tous les nœuds et tous les liens présents dans un modèle M_{ALFA} se retrouvent dans le modèle MB_{FHA} correspondant à la même phase de vol dans le même scénario de vol puis dans le modèle AltaRica correspondant. La génération des modèles de sécurité peut donc être réalisée pas à pas à travers la transformation indépendante de chacun des nœuds et des arcs du modèle source.

La table 6.3 illustre les règles de notre transformation de modèle. La première fonction est la fonction « transform » qui génère un modèle cible à partir d'un modèle source, comme nous l'avons vu dans la section 2.1.1. Elle consiste à traiter un à un tous les nœuds et tous les arcs du modèle source et d'appeler des fonctions de transformation dédiée pour chacun de ces éléments. La table ci-dessous présente la procédure de transformation de modèle en se focalisant plus particulièrement sur les fonctions du modèle source.

```
1   FUNCTION transform(source: Modèle_source):
2   Modèle_cible;
3         LOCAL
4               cible : Modèle_cible := ?;
5               I: INTEGER;
6         END_LOCAL;
7
8   -- pour chaque activité du modèle source
9   REPEAT I:=LOINDEX(source.activités) TO
10  HIINDEX(source.activités);
11    -- si l'activité est une fonction
12    IF 'Fonction_source' IN
13    TYPEOF(source.activités[I]) THEN
14    -- alors la fonction est transformée et sa
15    -- transformée intègre le modèle cible
16        cible.activités := cible.activités +
17    transform_fct(source.activités[I]);
18    END_IF;
19
20  -- si l'activité est une opération
21  […]
22  END_REPEAT;
23
24  -- de même pour les autres nœuds et arcs
25  -- du modèle source
26  […]
27
```

```
28      RETURN (cible);
29      END_FUNCTION;
30
31
32      -- Transformation des fonctions
33      FUNCTION transform_fct(f_input :
34      Fonction_source): Fonction_cible;
35            LOCAL
36                  f_output : Fonction_cible := ?;
37            END_LOCAL;
38
39      -- affectation des attributs et des
40      relations de la fonction cible par rapport à
41      ceux de la fonction source et des
42      annotations (voir section 3.1.1)
43      -- sauvegarde de la trace (voir la section
44      3.1.3)
45      […]
46      RETURN (f_output);
47      END_FUNCTION;
48
49      -- Transformation des opérations
50      FUNCTION transform_op(o_input :
51      Opération_source): Opération_cible;
52      […]
53      END_FUNCTION;
54
55      -- de même, une fonction dédiée pour chaque
56      -- autre famille de nœuds et d'arcs
```

Tab. 6.3 - Procédure de la transformation de modèle, focalisée sur les fonctions

2.1.3 Validation de la transformation de modèle

La manière la plus intuitive de valider une transformation de modèles est de réaliser des tests. Les tests consistent à générer de nombreux modèles à partir de modèles sources différents par leurs éléments instanciés et de vérifier que les comportements des modèles générés sont conformes à ceux attendus. Le chapitre 7 présente un déploiement industriel de notre prototype de transformation de modèle sur le cas d'étude du décollage. Il s'agit d'un test particulier de la transformation de modèle. L'exploitation du modèle de sécurité résultant de la transformation automatique montre qu'il est conforme au modèle transformé manuellement. Avec de nombreux tests comme celui ci, il est possible d'avoir une grande confiance dans la transformation de modèle.

La validation peut également être réalisée directement au niveau de la transformation de modèle en vérifiant la préservation de propriétés sémantiques ([Jézéquel et al. 2012]). De telles propriétés peuvent être codées à l'aide de contraintes dans le langage Express, à l'image de contraintes OCL (voir la table 6.4). Nous avons identifié plusieurs propriétés permettant de valider la transformation des concepts :

- la préservation des dépendances fonctionnelles. Il s'agit d'extraire du modèle ALFA source la liste de toutes les dépendances fonctionnelles et de vérifier que ces dépendances sont toujours présentes dans le modèle de sécurité correspondant puis dans le modèle AltaRica correspondant ;
- la préservation de l'ordre d'activation des activités. Ce flux d'activation des activités est la sémantique, mise en évidence dans les modèles ALFA, grâce à l'utilisation concrète des diagrammes d'activités (voir la section 3.2.3 du chapitre 2). Il s'agit alors de vérifier sur les modèles de sécurité et les modèles AltaRica que le flux d'activation en fonctionnement nominal n'a pas été modifié par la transformation.

```
1   ENTITY Transformation;
2         source : Modèle_source;
3   DERIVE
4   cible : Modèle_cible:=
5   transform(SELF.source);
6   WHERE
7   -- Préservation des dépendances
8   -- fonctionnelles
9   wr1: dep_fct(SELF.source,SELF.cible);
10  […]
11  END_ENTITY;
12
13  […]
14
15  FUNCTION dep_fct(s:Modèle_source,
16  c:Modèle_cible):BOOL;
17  -- la fonction retourne « true » si toutes
18  -- les dépendances fonctionnelles sont
19  -- préservées, « false » sinon.
20  RETURN […];
21  END_FUNCTION;
```

Tab. 6.4 - Validation d'une transformation de modèle par des contraintes

Lors de l'instanciation, le langage Express informe de la violation ou non des contraintes.

2.2 Annotation des modèles sources

La section 3.4 du chapitre 2 et le chapitre 4 ont montré que la syntaxe abstraite des modèles ALFA est insuffisante pour représenter les modèles abstraits d'analyse des aspects dysfonctionnels. La transformation

de modèle $T_{ALFA \rightarrow MBFHA}$ exploite alors des informations contenues dans les modèles ALFA, telles que les dépendances fonctionnelles, mais requiert également d'autres informations propres à la sécurité, telles que la connaissance des pannes fonctionnelles. Afin de préserver les modèles sources de toute information indésirable maîtrisée par d'autres domaines de modélisation, ces informations supplémentaires ne sont pas intégrées directement dans les modèles sources. L'opération réalisée est alors l'annotation ([Whittle et al. 2005], [Vara et al. 2009]).

2.2.1 Principes de l'annotation de modèles

L'annotation de modèles consiste à ajouter des informations aux éléments d'un modèle sans modifier ce dernier. Il existe deux méthodes pour réaliser des annotations ([Whittle et al. 2005]).

1. Coder l'annotation dans la transformation de modèle : dans certains cas, il est possible d'ajouter les données annotées dans le code de la procédure de génération de modèle.
2. Produire un modèle d'annotation : la seconde approche est que la transformation de modèle s'appuie sur un modèle d'annotation en plus du modèle source. Comme tout modèle, le modèle d'annotation peut être décrit par un métamodèle. Il peut être défini par héritage des classes du modèle source. Dans ce cas, le modèle annoté remplace le modèle source dans la transformation de modèle. Il peut également être séparé du modèle source et avoir des relations d'association avec lui.

Dans notre cas, les syntaxes des modèles ALFA et des modèles d'analyse des aspects dysfonctionnels sont susceptibles d'évoluer. De ce fait, les annotations nécessaires sont également susceptibles de varier. Il est alors nécessaire de disposer d'une solution d'annotation à la fois modulaire et facile à maintenir. Nous choisissons alors d'exprimer un modèle d'annotation sur la base d'un métamodèle permettant d'exprimer des annotations de façon générique (voir la section 2.2.2 pour l'implémentation via Express).

Deux bénéfices majeurs sont obtenus par la définition du modèle d'annotation :

- les analystes de la sécurité pilotent la transformation de modèles grâce aux données qu'ils annotent ;
- la procédure de transformation de modèles ne change pas. Elle est adaptée aux différents types d'annotations.

2.2.2 Mise en œuvre de l'annotation

La table 6.5 montre la prise en compte de l'annotation, comme entrée de la transformation de modèle. Dans un cas plus général, la transformation aurait pris en entrée un ensemble de modèles d'annotation afin que chaque modèle d'annotation annote un unique modèle source.

```
1   ENTITY Transformation;
2         source : Modèle_source;
3         annotations : M_annot;
4   DERIVE
5   cible : Modèle_cible:=
6   transform(SELF.source,SELF.annotations);
7   END_ENTITY;
8
9   ENTITY Modèle_source SUBTYPE OF (graph);
10  END_ENTITY;
11  ENTITY Modèle_cible SUBTYPE OF (graph);
12  END_ENTITY;
13
14  ENTITY M_annot;
15  END_ENTITY;
```

Tab. 6.5 - Entité de la transformation de modèle (avec le modèle d'annotation)

Le modèle d'annotation lui même est inspiré de [Vara et al. 2009]. Un modèle d'annotation M_{annot} s'appuie sur un modèle de référence, qui est un graphe dirigé acyclique. Le modèle d'annotation contient plusieurs annotations sur des éléments de ce modèle (des nœuds ou des arcs dans notre cas). Une annotation consiste à relier un ensemble d'éléments du graphe annoté à plusieurs propriétés, définies sous la forme d'ensembles de valeurs appartenant à des types prédéfinis. La table 6.6 présente, à l'aide du langage Express, le métamodèle des modèles d'annotations que nous avons utilisé.

```
1   ENTITY M_annot;
2         référence : graph;
3         annotations : SET OF Annotation;
4   INVERSE
5   in_transfo:  M_T_ALFA_MBFHA FOR annotations;
6   WHERE
7   -- le modèle de référence est le modèle
8   source de la transformation
9   […]
10  END_ENTITY;
11
12  ENTITY Annotation;
13        éléments : SET OF élément;
14        propriétés : SET OF Propriété;
15  INVERSE
16  modèle_annotation : M_annot FOR annotations;
17  WHERE
18  -- tous les nœuds annotés font partie du
19  modèle référencé.
```

```
20      […]
21      END_ENTITY;
22
23      ENTITY Propriété;
24            valeurs_type : E_type;
25            valeurs : SET OF SELF.valeurs_type;
26      END_ENTITY;
```

Tab. 6.6 - Métamodèle des annotations

Les éléments nœuds et arcs du modèle source annoté sont définis par un type « élément » comme étant soit un nœud soit un arc, grâce au mot clé « *SELECT* » (voir la table 6.7).

```
1       TYPE élément = SELECT (node,arc);
2       END_TYPE;
```

Tab. 6.7 - Les éléments comme des nœuds ou des arcs

Dans la table 6.6, une contrainte est ajoutée à l'entité M_{annot} : le modèle sur lequel sont ajoutées les annotations est le modèle source de la transformation. Ici, l'attribut « référence » est redondant car un modèle d'annotation est lié à une transformation ayant un unique modèle source. Ce code est en revanche plus générique car il prend en compte le cas où la transformation disposerait de plusieurs modèles sources. Cette contrainte indique alors que chaque modèle source doit avoir son propre modèle d'annotation. La contrainte sur les annotations est plus évidente : tous les éléments annotés doivent être des éléments du modèle annoté.

Une instance du métamodèle de la table 6.6 est un modèle d'annotation décrivant par exemple une annotation sur plusieurs fonctions du modèle source. Le nom de cette annotation est « Modes de perte » qui affecte des valeurs de mode de panne à un ensemble de fonctions. Cette annotation peut signifier que les fonctions annotées peuvent subir des pertes totales et partielles, mais pas des fonctionnements erronés. Par exemple, la fonction régissant la visibilité extérieure à travers la vitre du cockpit est liée à une telle annotation car son fonctionnement erroné n'a aucun sens dans la réalité.

2.3 Traçabilité dans la transformation de modèle

La traçabilité est une problématique importante en Ingénierie Dirigée par les Modèles ([Czarnecki et al. 2003]). Pendant tout le processus de développement d'un système embarqué, il est nécessaire de faire de la gestion des modèles, depuis leur création jusqu'à leur validation et lors de chacune de leurs évolutions. La conception d'un système embarqué s'appuie notamment sur des exigences qu'il est nécessaire de tracer d'un métier à un autre, éventuellement à travers des modèles. L'exploitation des modèles d'analyse des aspects

dysfonctionnels que nous avons introduits est susceptible de provoquer des évolutions des modèles ALFA. Il est alors impératif de tracer les liens entre les éléments de ces deux familles de modèles.

2.3.1 Principes de la traçabilité

D'après [Czarnecki et al. 2003] et [Drivalos et al. 2008], il existe deux stratégies pour enregistrer et gérer les traces d'une transformation de type « *model to model* ».

1. La première stratégie est la plus intuitive. Elle consiste à modéliser la sauvegarde des traces directement dans le modèle de transformation, de façon intrusive dans les règles de transformation. Cette solution est difficilement maintenable, principalement dans le cas d'une évolution des métamodèles des modèles sources ou cibles, car elle nomme directement des éléments des modèles, leurs attributs ou leurs relations. En revanche, cette solution présente l'avantage d'être très précise car la définition des traces se base sur la connaissance des métamodèles.
2. La seconde stratégie est de sauvegarder les traces dans un modèle dédié et séparé du modèle de transformation. C'est la deuxième stratégie qui a été retenue pour des raisons de modularité et de maintenabilité. Il devient en effet possible de choisir les traces souhaitées tout en conservant la même transformation de modèle. Le modèle des traces peut être métamodélisé et donc s'adapter à différentes transformations de modèles et différents modèles sources et modèles cibles. [Drivalos et al. 2008] présente un tel métamodèle de traces (voir la section 2.3.2).

A l'instar des annotations, les traces sont des informations qui peuvent être pertinentes pour enrichir des règles de transformation. En particulier, même si les transformations de modèle suivent des procédures logiques bien établies, la majorité d'entre elles s'appuient, en cours d'exécution, sur une connaissance de l'ensemble des éléments déjà transformés ou sur la recherche d'éléments source à un élément cible considéré. Il s'agit dans ce cas de traçabilité active où les traces courantes de la transformation sont non seulement des résultats de la transformation mais aussi des données d'entrée (voir [Bondé et al. 2005] et [Vanhooff et al. 2007] pour plus de détail) : les traces générées en cours de transformation peuvent être lues par cette même transformation afin de simplifier des procédures.

2.3.2 Mise en œuvre de la traçabilité

La traçabilité active est mise en œuvre dans la génération des modèles de sécurité à partir des modèles ALFA. La table 6.8 présente l'évolution du code Express définissant les modèles de transformation en prenant en considération les traces. La fonction « transform » réalise la génération du modèle cible et écrit les traces. De plus, la fonction prend en compte les traces courantes dans les règles de transformation.

```
1    ENTITY M_T_ALFA_MBFHA;
2          source : M_ALFA;
3          annotations : M_annot;
4          traces : M_trace;
```

```
5       DERIVE
6       cible : M_MBFHA:= transform(SELF.source,
7       SELF.annotations, SELF.traces);
8       END_ENTITY;
9
10      ENTITY Modèle_source SUBTYPE OF (graph);
11      END_ENTITY;
12      ENTITY Modèle_cible SUBTYPE OF (graph);
13      END_ENTITY;
14
15      ENTITY M_annot;
16      END_ENTITY;
17
18      ENTITY M_trace;
19      END_ENTITY;
```

Tab. 6.8 - Entité de la transformation de modèle (avec le modèle des traces)

Le métamodèle définissant les modèles des traces s'inspire de ceux présentés dans [Vanhooff et al. 2007] et [Drivalos et al. 2008]. Il définit qu'un modèle de traces est composé de plusieurs traces reliant des éléments des modèles sources à des éléments des modèles cibles. La table 6.9 présente, à l'aide du langage Express, le métamodèle des traces que nous avons utilisé.

```
1       ENTITY M_trace;
2             traces : SET OF Trace;
3       INVERSE
4       in_transfo : M_T_ALFA_MBFHA for traces;
5       END_ENTITY;
6
7       ENTITY Trace;
8             sources : SET OF élément;
9             cibles : SET OF élément;
10      INVERSE
11      modèle_trace : M_trace for traces;
12      WHERE
13      -- tous les éléments sources font partie du
14      modèle source de la transformation
15      -- tous les éléments cibles font parti du
16      modèle cible de la transformation
17      […]
18      END_ENTITY;
```

Tab. 6.9 - Métamodèle des traces

Une instance de ce métamodèle est par exemple la trace de transformation de la fonction « Guider aérodynamiquement ». Le nœud qui représente cette fonction dans le modèles ALFA du contrôle de la trajectoire est sauvegardé en tant que nœud source. Ce nœud est transformé pour obtenir une fonction de même nom dans les modèles de sécurité. Le nœud transformé est sauvegardé en tant que nœud cible.

3 Mise en œuvre des principes de transformation de modèle

Dans cette section, nous mettons en œuvre les principes présentés dans la section 2 sur les deux transformations $T_{ALFA \to MBFHA}$ et $T_{MBFHA \to AltaRica}$.

3.1 Transformation des concepts

La transformation $T_{ALFA \to MBFHA}$ se base sur les métamodèles des représentations abstraites des modèles ALFA (voir la section 3.3.3 du chapitre 2 pour les détails) et des modèles d'analyse des aspects dysfonctionnels (voir le chapitre 4). La table 6.10 définit la transformation $T_{ALFA \to MBFHA}$ en suivant les principes présentés dans la section 2 : génération d'un modèle MB_{FHA} cible à partir d'un modèle M_{ALFA} source, d'un modèle d'annotation du modèle M_{ALFA} et d'un modèle des traces de transformation.

```
1   ENTITY M_ALFA_MBFHA;
2         source : M_ALFA;
3         annotations : M_annot;
4         traces : M_trace;
5   DERIVE
6        cible : MBFHA:=
7   transform(SELF.source, SELF.annotations,
8   SELF.traces);
9   END_ENTITY;
10
11  ENTITY M_ALFA SUBTYPE OF (graph);
12        activities : SET [0:?] OF Act_ALFA;
13        […]
14  END_ENTITY;
15
16  ENTITY M_MBFHA SUBTYPE OF (graph);
17        activities : SET [0:?] OF Act_MBFHA;
18        […]
19  END_ENTITY;
```

Tab. 6.10 - Transformation des concepts

3.1.1 Annotation des modèles ALFA

D'après le chapitre 4, les données annotées aux modèles ALFA sont des informations complémentaires et nécessaires pour construire les modèles de sécurité. Ces données sont listées exhaustivement ci-dessous.

- Le type de service : dans la section 2.3.4 du chapitre 5, nous avons mis en évidence que les lois de comportement des activités dépendent du sens des services. Nous définissons alors un type pour les services des activités. Conformément au chapitre 5, ce type permet de désigner soit un service symétrique, soit un service asymétrique. Cette annotation est réalisée sur les services, c'est-à-dire sur les arcs de type « Lien_sortant ».
- L'efficacité nominale des services des fonctions : il s'agit de définir la contribution des fonctions aux différents résultats, dans le mode de fonctionnement nominal. D'après le chapitre 5, les valeurs nominales d'efficacité sont restreintes par rapport au type de service. Une contrainte globale peut donc être écrite en Express pour modéliser cette restriction. Cette annotation est réalisée sur les services.
- Les modes de panne des fonctions : il s'agit de définir pour chaque fonction le sous-ensemble des modes de panne qui peuvent l'affecter. Ainsi, la combinaison des fonctions avec leurs modes de panne définit l'ensemble des pannes fonctionnelles potentielles.
- Les capacités de détection : cette annotation est réalisée sur les fonctions. Il s'agit d'une valeur booléenne, à vrai quand la fonction possède des moyens pour détecter les dégradations de ses données d'entrée.

Ces annotations sont réalisées par l'introduction de nouveaux types de valeurs. En Express, ces types sont définis dans la table 6.11 grâce au mot clé « *TYPE* ». Le type « E_type » fait la liste de tous les types afin de pouvoir les sélectionner.

```
1   TYPE E_type = ENUMERATION OF
2   (type_service,efficacité,puissance_env,mode_
3   panne,bool);
4   END_TYPE;
5
6   TYPE type_service = ENUMERATION OF
7   (service_symétrique, service_asymétrique);
8   END_TYPE;
9
10  TYPE efficacité = ENUMERATION OF (High, Low,
11  None, LNeg, HNeg);
12  END_TYPE;
13
14  TYPE puissance_env = ENUMERATION OF (Null,
15  Strong, Very_strong);
16  END_TYPE;
17
```

```
18      TYPE mode_panne = ENUMERATION OF
19      (Total_loss, Partial_loss_sym,
20      Partial_loss_asym, Erroneous_sym,
21      Erroneous_asym);
22      END_TYPE;
```

Tab. 6.11 - Les types

3.1.2 Processus d'annotation

La question qui se pose devant toutes ces annotations est : qui réalise l'annotation des modèles ALFA ? Les concepteurs des modèles ou les analystes de la sécurité.

Théoriquement, cette tâche revient aux analystes de la sécurité afin qu'ils puissent gérer les données de sécurité et piloter la transformation de modèle. Il est évident que l'annotation des modes de panne pertinents sur les fonctions doit être réalisée par un expert de la sécurité. En revanche, déterminer si un service est symétrique ou asymétrique semble davantage être du ressort des concepteurs de l'avion. De même, l'efficacité nominale des services n'est pas sans rappeler les objectifs de performance sur lesquels travaillent les concepteurs. Si aujourd'hui toutes ces annotations sont réalisées par les analystes de la sécurité souhaitant disposer de modèles de sécurité pour assister leurs analyses des risques, certaines d'entre elles pourraient découler à l'avenir d'autres données contenues dans des modèles ALFA enrichis. En particulier, des travaux futurs sont nécessaires pour étudier le lien entre les valeurs d'efficacité nominale des services des activités et les exigences de performance liées aux activités.

Exemple. La fonction permettant le guidage de l'avion à l'aide de la roulette de direction possède une exigence de performance exprimée en langage naturel par : « La roulette de direction seule doit permettre un guidage correct de l'avion en phase de roulage à basse vitesse » (exemple théorique). De cette exigence il découle naturellement que le service de cette fonction contribuant au contrôle de la trajectoire est « *High* ». La donnée, aujourd'hui annotée, découle donc d'une exigence textuelle.

3.1.3 Traçabilité des concepts

Il est important de noter que, dans notre cadre de transformation de modèle, les traces relient directement entre eux les nœuds de même type et les liens de même famille. Dans la section 2.1.2 nous avons en effet montré que la transformation globale est composée de multiples transformations locales à l'échelle de chaque nœud et chaque arc. Concrètement les traces sont simplifiées ; elles sauvegardent une source et son unique cible construite à partir de la transformation locale. L'ensemble des traces est alors composé de traces reliant chaque fonction, opération, donnée, condition environnementale et lien d'un modèle M_{ALFA} à la fonction, l'opération, la donnée, la condition environnementale ou le lien correspondant du modèle MB_{FHA} généré.

Le chapitre 7 donne un exemple de trace de la transformation d'un modèle ALFA sur le cas d'étude du décollage.

3.2 La concrétisation des modèles de sécurité en AltaRica

La transformation de modèle de la syntaxe abstraite MB_{FHA} vers des modèles concrets et exploitables en langage AltaRica avec le support graphique des outils OCAS ou RAMSES, notée $T_{MBFHA \to OCAS}$ consiste en la génération d'un code XML (transformation de type « *model to code* »), qui peut être interprétée par les outils OCAS et RAMSES. D'après le métamodèle associé à ce code XML des outils OCAS et RAMSES (voir la section 4.3.4 du chapitre 2), un modèle est défini par la classe « *ARCHITECTURE* », en relation avec des composants (les nœuds du modèle) et des liens. La transformation $T_{MBFHA \to OCAS}$ consiste alors à générer une architecture à partir d'un modèle MB_{FHA}. La table 6.12 présente, à l'aide du langage Express, la déclaration de cette transformation conformément aux principes présentés dans la section 2.

```
1   ENTITY M_MBFHA_OCAS;
2         source : M_MBFHA;
3         annotations : M_annot;
4         traces : M_trace;
5   DERIVE
6         cible : ARCHITECTURE:=
7   transform(SELF.source,
8   SELF.annotations, SELF.traces);
9   END_ENTITY;
```

Tab. 6.12 - Entité de la génération de code

Comme nous l'avons vu, la transformation $T_{MBFHA \to OCAS}$ s'appuie sur un modèle source, une annotation de ce modèle et une sauvegarde des traces de transformation. La transformation elle-même est réalisée grâce à la fonction « transform », qui génère une architecture à partir d'un modèle MB_{FHA}, des règles, des annotations et des traces.

Les sous-sections suivantes présentent quelques caractéristiques particulières de cette étape de transformation de modèle.

3.2.1 Génération des interfaces

La syntaxe des modèles OCAS et des modèles RAMSES fait apparaître la notion d'interface. La table 6.13 rappelle la définition d'un nœud AltaRica, selon la section 4.3.4 du chapitre 2. Les interfaces ont déjà été introduites dans la section 1.1.1 du chapitre 5. Chaque interface correspond à une variable d'entrée ou de sortie d'un nœud AltaRica.

```
1   ENTITY COMPONENT_MODEL;
2         interfaces : SET OF IO;
3         états : SET OF STATE;
4         événements : SET OF EVENT;
5         icônes : SET OF ICON;
6   END_ENTITY;
7
8   ENTITY COMPONENT;
9         nom : STRING;
10        type : COMPONENT_MODEL;
11  END_ENTITY;
```

Tab. 6.13 - Les nœuds dans les outils OCAS et RAMSES

Les interfaces mettent en évidence l'association d'un nœud et d'un lien. Les liens ne relient alors plus directement des nœuds entre eux, mais relient deux interfaces (entité « IO ») appartenant à des nœuds différents (voir la table 6.14).

```
1   ENTITY LINK SUBTYPE OF (arc);
2         init : IO;
3         final : IO;
4   DERIVE
5         SELF\arc.init : node :=
6   init_interface.in_node;
7         SELF\arc.final : node :=
8   final_interface.in_node;
9   WHERE
10  -- « init » appartient au nœud initial
11  -- « final » appartient au nœud final
12  END_ENTITY;
```

Tab. 6.14 - Entité détaillée des liens du métamodèle d'OCAS et de RAMSES

Les interfaces sont définies selon la table 6.16. L'affectation des variables de flux à leur domaine de valeur est réalisée par l'attribut « interface_type » qui désigne un type parmi l'ensemble des types possibles (ensemble nommé D dans la sémantique du langage AltaRica, voir la section 4.3.2 du chapitre 2). Cet ensemble est défini dans la table 6.15. Notons que les variables internes (ou états internes, « *STATE* ») des nœuds ont également une affectation d'un type.

```
1   TYPE E_type = ENUMERATION OF
2   (type_service,efficacité,puissance_env,mode_
3   panne,boolean);
4   END_TYPE;
```

Tab. 6.15 - L'ensemble des types

Les interfaces sont distinguées en deux sous-entités : les interfaces d'entrée et les interfaces de sortie, correspondant respectivement aux variables d'entrée et de sortie des nœuds dans la sémantique d'AltaRica. Si plusieurs liens peuvent être connectés à la même variable de sortie si ces liens propagent les mêmes valeurs, les variables d'entrée ne peuvent être reliées qu'à un unique lien. La valeur de la variable d'entrée est la valeur unique propagée par le lien.

```
1   ENTITY IO ABSTRACT SUPERTYPE;
2         nom : STRING;
3         interface_type : E_type;
4   INVERSE
5         in_node : COMPONENT FOR interfaces;
6   END_ENTITY;
7
8   ENTITY OUT_IO SUBTYPE OF (IO);
9   INVERSE
10        in_link : SET OF LINK FOR init;
11  END_ENTITY;
12
13  ENTITY IN_IO SUBTYPE OF (IO);
14  INVERSE
15        in_link : SET OF LINK FOR final;
16  WHERE
17  -- une interface d'entrée n'est utilisée que
18  par un seul lien
19  END_ENTITY;
```

Tab. 6.16 - Les interfaces du métamodèle d'OCAS et de RAMSES

L'étape de concrétisation des modèles de sécurité génère donc des interfaces pour les nœuds initiaux et finaux de chaque lien.

3.2.2 L'annotation du code AltaRica

Pour la transformation $T_{MBFHA \rightarrow OCAS}$, les données propres à la sécurité se trouvent déjà dans les modèles MB_{FHA}. D'après le chapitre 5, la seule connaissance des pannes fonctionnelles permet d'exprimer les transitions au sein de chaque nœud. De même, la connaissance des variables d'entrée dans les nœuds

permet d'exprimer en langage AltaRica les lois de comportement par rapport à des opérateurs pré-établis.

Exemple. Pour les nœuds de type « Donnée », que ces nœuds aient 3 ou 7 variables d'entrée, le calcul réalisé est toujours la somme tronquée des efficacités de toutes les variables d'entrée. Le code AltaRica correspondant varie en fonction du nombre d'entrée mais l'opération est connue. De même, l'agrégation des données nécessaires, des données utiles, des puissances environnementales et les lois de comportement des activités suivent des opérations logiques pré-établies aux chapitres 4 et 5.

Le modèle de sécurité est donc annoté de codes AltaRica décrivant ces opérateurs qui permettent de déterminer le code des assertions de chacun des nœuds par rapport à leurs variables d'entrée, leurs services etc. L'avantage de l'annotation est de pouvoir modifier les opérations mathématiques réalisées au niveau des assertions, afin d'affiner les lois de comportement, sans impacter la modélisation des dépendances fonctionnelles. Ces annotations représentent donc des choix de modélisation concrète en AltaRica par les analystes de la sécurité.

Au vu du nombre limité des primitives AltaRica, nous avons choisi de les annoter sous la forme de méthodes dont les paramètres sont par exemple un ensemble de variables d'entrée et dont le résultat est un code AltaRica. L'annotation est alors une collection de méthodes. La table 6.17 illustre une telle méthode pour la somme tronquée des variables d'entrée des nœuds de type « Donnée ». Comme nous l'avons vu dans le chapitre 5, le code AltaRica réalise une conversion des valeurs d'efficacité des variables d'entrée dans le domaine des entiers pour réaliser l'addition des variables. L'efficacité de la variable de sortie est ensuite déterminée par rapport au résultat de la somme des entiers.

```
1   FUNCTION résultat_donnée(i : SET OF IN_IO,
2   o: OUT_IO):STRING;
3   LOCAL
4         code : STRING;
5         Y : INTEGER;
6   END_LOCAL;
7
8   -- déclaration de l'assertion avec 'assert'
9   code:= 'assert';
10
11  REPEAT Y:=LOINDEX(i) TO HIINDEX(i);
12  -- conversion de chaque variable d'entrée
13  dans le domaine des entiers
14  code:= code + 'val_' + Y + ':=case{
15        i[Y]=High: 2,
16        i[Y]=Low: 1,
17        i[Y]=Lneg: -1,
18        i[Y]=Hneg: -2,
19        else 0};';
```

```
20      END_REPEAT;
21
22      -- calcul de la variable de sortie par
23      rapport à la somme des entiers
24      code:= code + o.nom + ':=case{' +
25             add(i)+ ' >= 2 : High,'+
26             add(i)+ ' =1 : Low,'+
27             add(i)+ ' =-1 : Lneg,'+
28             add(i)+ ' <=-2 : Hneg,'+
29             'else None};';
30
31      RETURN code ;
32      END_FUNCTION;
33
34
35      FUNCTION add(i : SET OF IN_IO):STRING;
36      LOCAL
37            code : STRING;
38            Y : INTEGER;
39      END_LOCAL;
40
41      code:= '';
42
43      REPEAT Y:=LOINDEX(i) TO HIINDEX(i);
44      -- addition de toutes les variables entières
45      correspondantes aux variables d'entrée
46      IF (Y == HIINDEX(i)) THEN
47            code:= code + 'val_'+ Y;
48      ELSE
49            code:= code + 'val_' + Y + '+';
50      END_IF;
51
52      RETURN code ;
53      END_FUNCTION;
```

Tab. 6.17 - Les fonctions Express générant le code AltaRica de la somme des entrées d'une donnée

Exemple. La table 6.18 donne le résultat retourné par la fonction « résultat_donnée » dans le cas d'une donnée à deux variables d'entrée, nommées « data_1 » et « data_2 ». Le résultat est une interface de sortie nommée « resultat ». Nous retrouvons le code AltaRica présenté dans la section 2.4.2 du chapitre 4.

```
1   assert
2
3   val_1:= case{data_1=High : 2,
4                data_1=Low : 1,
5                data_1=Lneg : -1,
6                data_1=Hneg : -2,
7                else 0};
8   val_2:= case{data_2=High : 2,
9                data_2=Low : 1,
10               data_2=Lneg : -1,
11               data_2=Hneg : -2,
12               else 0};
13  resultat:=case{val_1+val_2 >=2 : High,
14                 val_1+val_2 =1 : Low,
15                 val_1+val_2 =-1 : Lneg,
16                 val_1+val_2 <=-2 : Hneg,
17                 else None};
```

Tab. 6.18 - Assertion AltaRica générée pour une donnée à deux variables d'entrée

4 Bilan et perspectives sur la transformation de modèle

Le premier résultat obtenu de l'étude de la transformation de modèle est la réalisation d'un prototype, présenté dans la section 4.1. Puis, en section 4.2 nous présentons une conclusion sur nos travaux de transformation de modèle et l'apport de ces travaux dans le cadre des objectifs de cette thèse.

4.1 Prototypage

Un prototype de transformation de modèle a été réalisé tel que décrit dans les sections précédentes. La transformation $T_{ALFA \rightarrow MBFHA}$ a entièrement été réalisée en Express. La génération du code AltaRica respecte les spécifications présentées en section 3.2, mais a été réalisée dans l'environnement Eclipse, en langage Java, à l'aide de trois API :

- SDAI (voir la section 2.3.3 du chapitre 2) permettant de convertir le code Express des modèles abstraits de la sécurité dans le format XSD/XML ;
- JAXB (« Java Architecture for XML Binding »), une librairie Java, simplifiant la conversion des instances capitalisées dans un XML en objets Java manipulables et réciproquement ;
- et DOM (« Document Object Model »), permettant de parcourir des fichiers XML comme des arbres, pour effectuer de la recherche de données.

La figure 6.2 détaille les opérations du prototype. Premièrement, il s'agit de la transformation des modèles abstraits avec le langage Express. Puis, l'API SDAI est utilisée pour convertir le métamodèle de la syntaxe abstraite des modèles d'analyse des aspects dysfonctionnels et les instances associées dans le format XSD/XML. Dans la seconde transformation, les annotations sont codées en Java, à l'instar des opérations de transformation. JAXB et DOM permettent de manipuler les fichiers XML et XSD avec le langage Java. Le résultat final est un fichier XML pouvant être importé directement dans les outils d'édition graphique du langage AltaRica : OCAS et RAMSES.

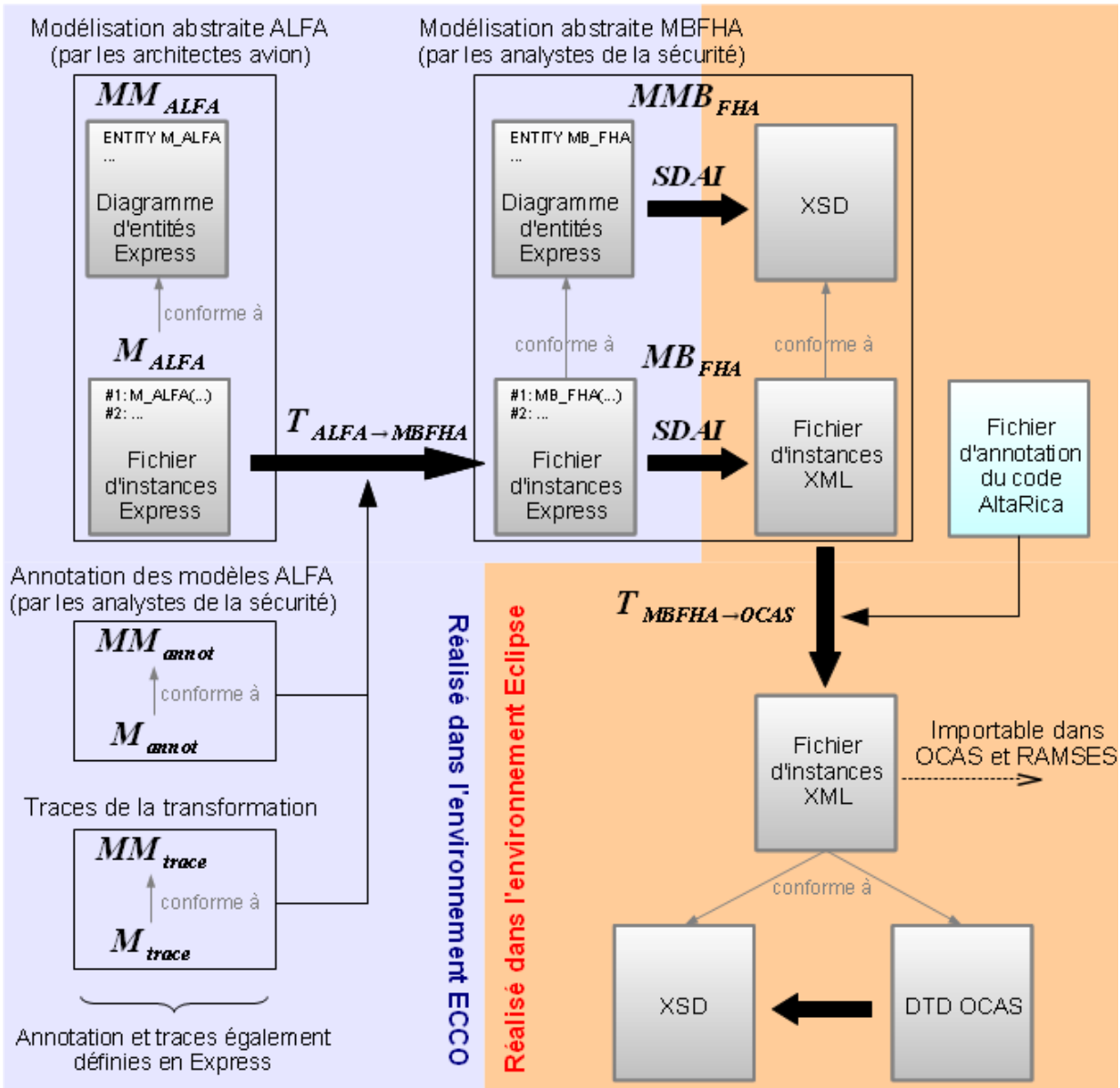


Fig. 6.2 - Prototype des deux transformations de modèle

Le code Express a été édité et interprété grâce à l'environnement de développement ECCO[4] ([ECCO Toolkit]), qui implémente l'API SDAI. Ce dernier permet la création des instances sur la base d'un métamodèle,

4 ECCO Toolkit est un outil de PDTec conforme au standard STEP. Il existe quelques outils concurrents, tous basés sur l'API SDAI, tels que : JSDAI (logiciel libre sous licence AGPL v3, disponible dans l'environnement Eclipse) et Platypus.

effectue les calculs procéduraux (notamment pour générer les modèles cibles dans les transformation de modèle) et réalise la vérification des contraintes exprimées.

Ce prototype a été testé sur plusieurs modèles dont le modèle sur le contrôle de la trajectoire au sol et les modèles réalisés dans le cadre de notre cas d'étude (voir le chapitre 7). Le modèle le plus imposant sur lequel le prototype a été validé est un modèle contenant 31 nœuds et 43 pannes fonctionnelles ou conditions environnementales. La validation du prototype s'est heurté à une difficulté pour définir les modèles abstraits des architectures opérationnelles et fonctionnelles. L'origine de cette difficulté est l'absence de maîtrise des modélisations concrètes et des mappings entre les modèles ALFA concrets et les modèles abstraits par les analystes de la sécurité.

Le prototypage a également fait apparaître une problématique sur les vues des modèles. Dans les environnements OCAS et RAMSES, l'organisation graphique des nœuds dans l'espace d'un modèle est définie par des coordonnées cartésiennes absentes dans les modèles de sécurité. Cette problématique rejoint les travaux réalisés sur la visualisation des graphes. Elle n'a pas fait l'objet de cette thèse.

4.2 Bilan

Dans ce chapitre, nous avons détaillé le contenu de notre approche visant à générer automatiquement des modèles de sécurité exploitables dans les outils d'édition graphique OCAS et RAMSES, basés sur le langage AltaRica, à partir de modèles ALFA. Une chaîne complète, composée de trois transformations permet de formaliser les liens entre les modèles concrets manipulés par les architectes avion et par les analystes de la sécurité. Toutefois, nous avons vu que le mapping des modèles EFFBD vers les modèles abstraits de la conception appartient au domaine de la conception. Ainsi, la génération des modèles de sécurité est tributaire des travaux d'abstraction réalisés par les architectes avion.

Notre approche a consisté à mettre en application un type particulier de transformation de modèle. Il ne s'agit plus de générer un modèle cible uniquement à partir d'un modèle source, mais de s'appuyer également sur un modèle d'annotation et un modèle de traces.

- Apport du modèle d'annotation : les analystes de la sécurité peuvent générer leur modèle à partir des modèles ALFA sans que les architectes avion n'aient besoin d'intervenir pour assister la démarche de transformation de modèle. En effet, toutes les connaissances nécessaires sont modélisées dans les modèles ALFA. Les analystes de la sécurité pilotent alors seuls les transformations $T_{ALFA \rightarrow MBFHA}$ et $T_{MBFHA \rightarrow OCAS}$ à l'aide des annotations. Notons cependant que certaines annotations actuelles, telles que les valeurs d'efficacité nominales des services, pourraient dériver d'attributs existants dans les modèles ALFA, tels que les exigences de performance (voir la section 3.1.2).
- Apport du modèle de traces : les modèles de la conception et ceux de la sécurité sont liés par des traces, générées pendant la transformation, qui facilitent la validation des modèles et l'échange de résultats d'analyse tels que les exigences de sécurité. Mais les traces sont également utiles pendant la procédure de transformation, puisqu'elles sauvegardent à chaque instant l'ensemble des éléments déjà

transformés. Elles permettent notamment à la transformation de rechercher un élément transformé.

Notre application de transformation de modèle renforce la confiance que les analystes de la sécurité peuvent avoir dans les modèles et leurs résultats d'exploitation, grâce à une traçabilité forte avec les modèles de la conception. De plus, la production des modèles se trouve accélérée et facilitée. Ce gain ne remet nullement en cause la maîtrise des modèles par les analystes de la sécurité ; au contraire, les annotations leur permettent de contrôler les lois de comportement des activités, les pannes fonctionnelles admissibles et d'autres données propres à la sécurité. Plusieurs travaux se sont déjà intéressés à la génération de modèles AltaRica à partir de diagrammes d'activités des outils CORE ou SysML ([David et al. 2010], [MDWorkbench]). Mais ce type de génération de modèle est indépendant de toute analyse particulière de la sécurité. Les modèles produits traduisent bien le flux d'activation des activités pour être conforme aux comportements des modèles EFFBD sources, mais ils n'intègrent aucune donnée spécifique propre aux analyses de sécurité, telles que les pannes fonctionnelles. De tels générateurs peuvent servir de base pour produire des squelettes de modèles AltaRica, mais ils doivent être complétés par les analystes de la sécurité. Dans notre approche de transformation de modèle, l'annotation se révèle être le moyen permettant d'éviter que les analystes de la sécurité (dont les connaissances en modélisation AltaRica sont limitées) soient contraints à modifier manuellement les modèles AltaRica produits. Nos travaux sur la transformation ont été également présentés dans [Maitrehenry et al. 2012 (2)].

4.3 Perspectives

4.3.1 Vers une ontologie du domaine des modèles de sécurité

[Grüber 1993] définie une **ontologie** comme la spécification d'une conceptualisation ; la conceptualisation étant une vue abstraite et simplifiée du monde que nous souhaitons représenter. Pragmatiquement, une ontologie est un ensemble structuré de connaissances permettant de définir le sens de concepts propres à un domaine de connaissance spécifique ([Guizzardi et al. 2007], [Tairas et al. 2009]). De telles connaissances sont en particulier les types des attributs des classes, c'est-à-dire leur domaine de valeurs. Il semble alors naturel que chaque domaine de modélisation (DSML) puisse être enrichi par des références explicites à une ontologie sur laquelle les définitions des modèles peuvent s'appuyer.

Dans le cadre de nos travaux, plusieurs types ont été mis en évidence pour spécifier quelques attributs des éléments des modèles MB_{FHA} : les valeurs d'efficacité, les modes de panne, les valeurs de puissance environnementales, les types de service etc. Ces types ont été formalisés dans la table 6.11 de la section 3.1.1.

L'ensemble de tous ces types peut être défini comme une ontologie ou faire partie d'une ontologie plus vaste, dont le périmètre serait par exemple l'analyse des risques voire l'ensemble du processus de sécurité. Nous identifions comme perspective le besoin de structurer davantage ces connaissances.

4.3.2 Vers une formalisation de la transformation inverse

Comme nous l'avons vu dans le chapitre 1, l'analyse des risques vise à exprimer les exigences de sécurité qui doivent être respectées par les architectures de l'avion. Nous avons également vu dans le chapitre 2 que le projet ALFA vise à capitaliser toutes les exigences contraignant la conception des architectures de l'avion. Le résultat de l'analyse des risques (et donc de l'exploitation des modèles de sécurité) est alors transmis aux concepteurs. Ce lien n'est pas formalisé dans le cadre de nos travaux. Il s'agit d'une transformation inverse à celle décrite dans ce chapitre, c'est-à-dire d'un transfert de données du domaine de la sécurité vers le domaine de la conception.

Chapitre 7

Déploiement sur un cas industriel

Sommaire

Résumé. Les trois derniers chapitres ont permis de répondre aux trois objectifs que nous nous étions fixés dans le chapitre 3. Dans le chapitre 5, nous avons identifié le besoin d'étudier davantage les bénéfices de l'exploitation de nos modèles de sécurité pour assister l'analyse des risques. Dans ce chapitre, nous présentons un cas d'étude industriel pour lequel nous avons réalisé une analyse des risques basée sur les modèles. Nous comparons également notre approche et ses résultats avec les analyses des risques déjà réalisées lors des précédents programmes avion. Nous rappelons toutefois que tous les exemples présentés dans ce mémoire, bien qu'inspirés de cas industriel réels, sont théoriques.

1 Présentation du cas d'étude industriel

Le cas industriel sur lequel nous avons déployé notre approche est le décollage d'un appareil de la classe des « short-range », bimoteur, tel que l'A320 d'Airbus. Le choix du décollage est motivé par le fait que cette phase fait partie des phases de vol les plus critiques avec la montée jusqu'à l'altitude de croisière et l'atterrissage.

Le scénario de vol choisi est un vol simple : l'avion s'est aligné normalement sur la piste de décollage. En cas de givre, les surfaces de l'avion sont dégivrées correctement. Le décollage commence alors à l'état nominal. Dans ce cas d'étude, nous nous intéressons simultanément au contrôle de la trajectoire au sol et à l'accélération/décélération de l'avion.

1.1 Présentation des phases de vol du décollage

Le décollage (TO, « *Take-Off* ») commence lorsque l'avion, aligné sur la piste de décollage, reçoit son autorisation pour décoller de la part des contrôleurs aériens. Les pilotes augmentent alors la poussée des moteurs pour accélérer l'avion tout en le maintenant bien aligné sur la piste. Deux vitesses sont caractéristiques de cette phase et leurs valeurs dépendent de plusieurs paramètres dont la famille d'avion considérée :

- V_1 est la vitesse à partir de laquelle l'avion ne dispose plus de la longueur de piste nécessaire pour freiner et s'arrêter sans sortir de la piste. Une fois cette vitesse atteinte, le décollage doit être poursuivi ;
- $V_r > V_1$ est une vitesse supérieure à V_1, à laquelle les roues de l'avion peuvent quitter le sol. Cette action est la « rotation ».

Dans le cadre du projet ALFA, le décollage se termine lorsque l'avion atteint sa vitesse V_r et effectue sa rotation. En fonctionnement nominal, le décollage est alors suivi de la phase de montée jusqu'à l'altitude de croisière. Cependant, en cas de pannes, il est possible de stopper le décollage (RTO, « *Rejected Take-Off* »), à condition que la vitesse V_1 ne soit pas atteinte. Nous identifions alors trois sous-phases de vol au décollage. Ces phases sont décrites par la figure 7.1 ; il s'agit d'un modèle réalisé à l'aide du formalisme EFFBD.

1. TO step 1 : il s'agit de la phase de décollage avant la vitesse V_1.
2. TO step 2 : il s'agit de la phase de décollage après la vitesse V_1.
3. RTO : il s'agit d'une phase qui peut être atteinte à n'importe quel instant durant la phase TO step 1 et qui consiste à arrêter l'avion. Il s'agit d'une action préventive réalisée par décision des pilotes s'ils jugent ne pas pouvoir réaliser le décollage de façon sûre.

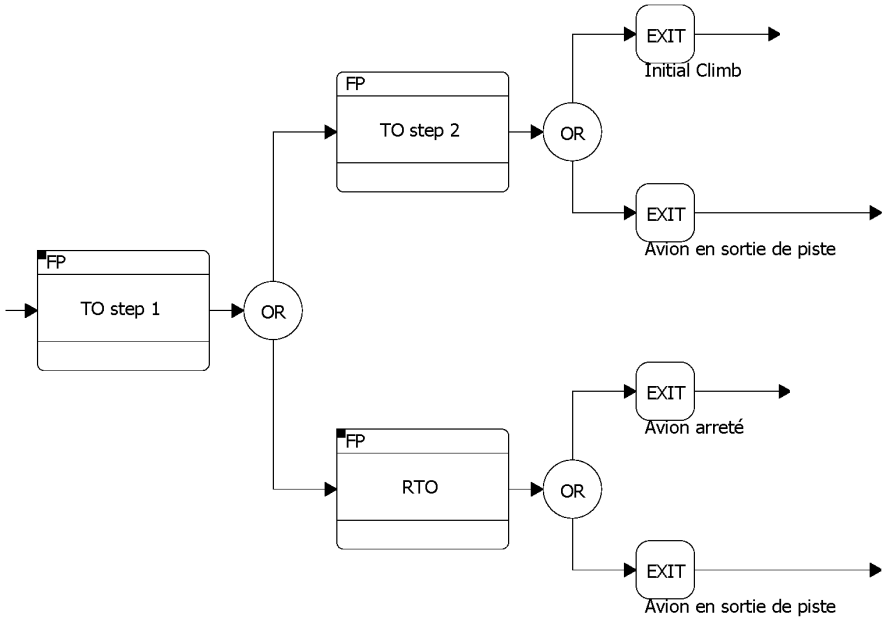


Fig. 7.1 - Modèle des phases du décollage (formalisme EFFBD)

1.2 La phase de décollage avant V_1

Dans le périmètre de notre étude, cette première phase du décollage consiste à accélérer l'avion jusqu'à la vitesse V_1 tout en maintenant sa trajectoire correctement alignée sur la piste. Nous utilisons alors la capacité des formalismes EFFBD et AltaRica à décrire des niveaux hiérarchiques pour représenter les deux principales tâches réalisées pendant cette phase de décollage (voir la figure 7.2). Les modèles de ces deux tâches sont naturellement composés d'activités, de données et de conditions environnementales.

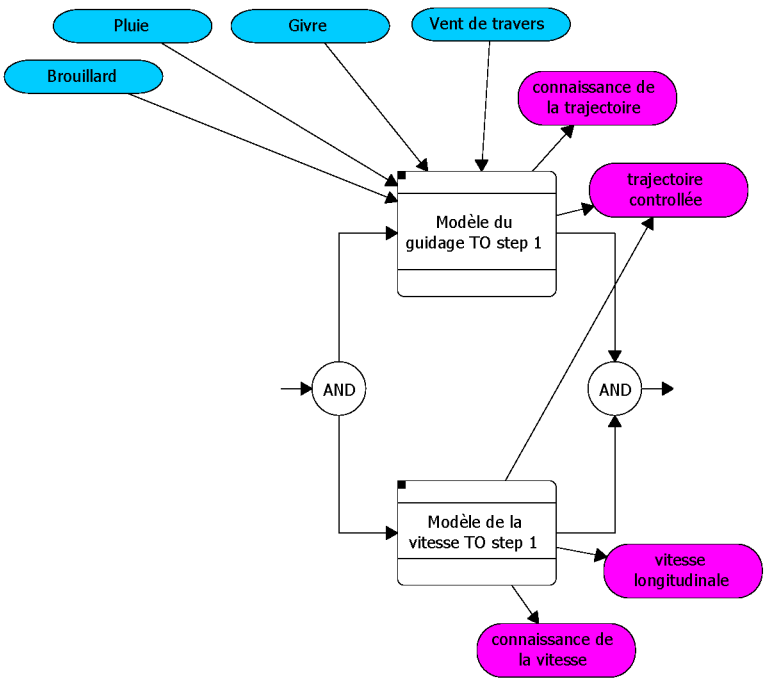


Fig. 7.2 - Les deux tâches réalisées pendant le TO step 1 (formalisme EFFBD)

L'utilisation de niveaux hiérarchiques ne modifie nullement la sémantique des modèles ALFA, ni celle de nos modèles de sécurité. En effet, il est toujours possible de produire un modèle « à plat », sans hiérarchie, qui est sémantiquement équivalent au modèle avec hiérarchie. La hiérarchie consiste simplement en la mise en place d'une vue permettant une compréhension plus facile des modèles pour les concepteurs et les analystes de la sécurité.

Un exemple de modèle A.O.F. du contrôle de la trajectoire au sol a été présenté par la figure 2.9 du chapitre 2. Cet exemple a été utilisé comme fil conducteur dans les différents chapitres de ce manuscrit. Le modèle sur l'accélération de l'avion a été réalisé de la même manière. La figure 7.3 en présente les principaux nœuds dans une représentation naturelle proche du formalisme EFFBD.

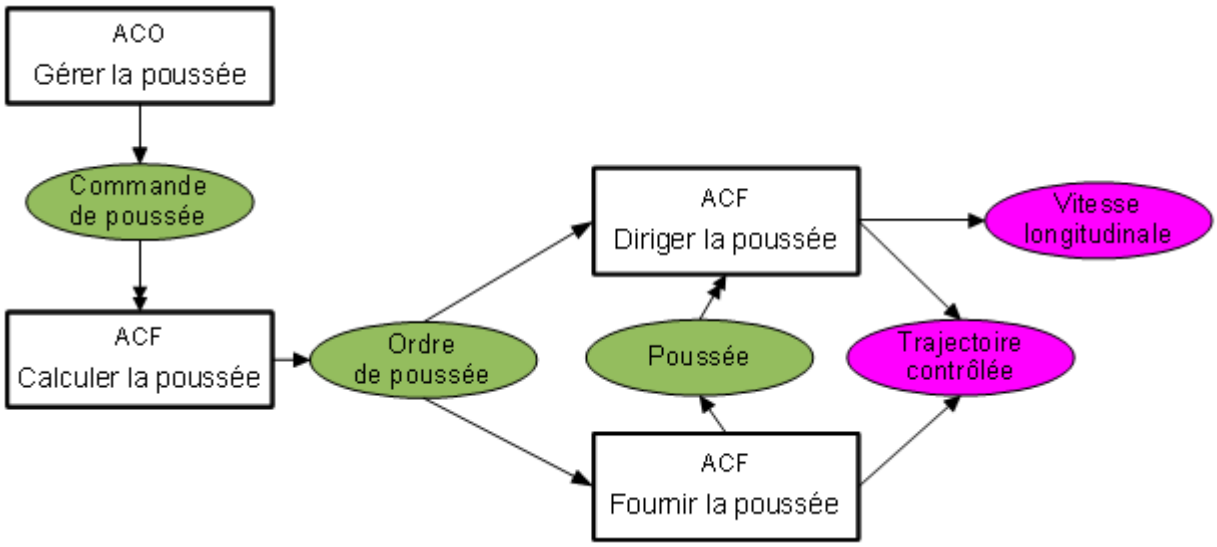


Fig. 7.3 - Modèle ALFA du contrôle de l'accélération (exemple théorique)

Comme pour le contrôle de la trajectoire, l'accélération est commandée par le pilote grâce à un opération. La commande est captée par l'avion qui calcule un ordre en vue de fournir et diriger la poussée motrice qui conduit à une maîtrise de la vitesse de l'avion. Nous pouvons remarquer que les deux fonctions « Fournir la poussée » et « Diriger la poussée » ont un service asymétrique en plus de leur service principal. En effet, ces deux fonctions sont supportées par les moteurs situés de part et d'autre de l'avion. La poussée a donc un impact sur la maîtrise de la trajectoire du fait d'un risque possible d'une poussée asymétrique.

Le modèle « à plat », complet, correspondant à l'association des deux tâches réalisées pendant la première phase du décollage est composé de 31 nœuds.

1.3 L'arrêt du décollage

Du fait d'un contrôle dégradé de la trajectoire de l'avion ou d'une accélération trop faible pendant la première phase du décollage, les pilotes peuvent décider d'interrompre le décollage. Dans cette phase aussi deux tâches principales sont identifiées : maintenir l'avion sur la piste et freiner jusqu'à l'arrêt de l'avion. La structure hiérarchique du modèle du RTO est donc identique à celle du modèle du décollage (figure 7.2).

La première tâche est identique à celle de la phase de décollage. Le modèle A.O.F. correspondant est donc le même. En revanche, la seconde tâche sur la décélération fait appel à de nouvelles fonctions dont principalement le freinage par les roues (voir la figure 7.4).

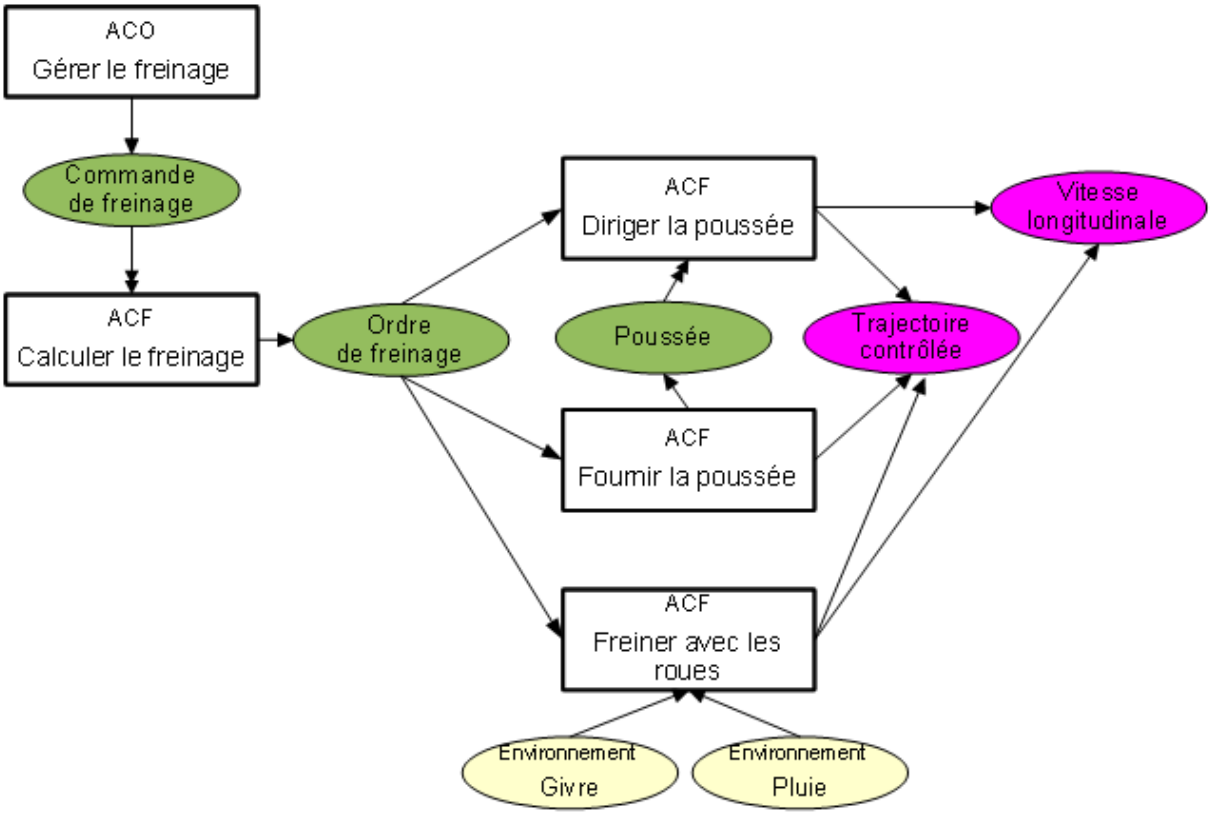


Fig. 7.4 - Modèle ALFA du contrôle du freinage (exemple théorique)

Le modèle « à plat » équivalent au modèle complet du RTO est composé de 33 nœuds.

2 Génération et analyse des modèles de sécurité

Les modèles à plat du décollage et de l'interruption du décollage peuvent être manipulées par la transformation de modèle présenté au chapitre 6 pour obtenir des modèles AltaRica graphique, compatibles avec les outils OCAS et RAMSES et conformes à la modélisation présentée dans les chapitres 4 et 5. Ces modèles AltaRica peuvent être exploité pour assister l'analyse des risques.

2.1 Génération des modèles

2.1.1 La génération automatique des modèles AltaRica

Les modèles A.O.F. peuvent être écrits comme des instances Express du métamodèle des modèles ALFA présentés au chapitre 2. La table 7.1 illustre quelques une de ces instances (l'opération, la fonction et la donnée situées à gauche de la figure 7.3) pour le contrôle de l'accélération lors du décollage.

```
1   -- Déclaration du modèle ALFA
2   #1= M_ALFA((#2,#3),(#4), *, (#5,#6));
3
4   -- Déclaration de l'opération et de la
5   fonction. Les attributs « running_time »
6   sont mis à 0.
7   #2= Op_ALFA('Gérer la poussée', 0);
8   #3= Fct_ALFA('Calculer la poussée', 0);
9
10  -- Déclaration de la donnée puis des liens
11  #4= Data_ALFA('Commande de poussée');
12  #5= Lien_out_ALFA(#2,#4);
13  #6= Lien_trig_ALFA(#4,#3);
```

Tab. 7.1 - Quelques instances du modèle ALFA sur l'accélération de l'avion (en Express)

La première étape de la transformation de modèle détermine les modèles conceptuels MB_{FHA} correspondant aux modèles ALFA. Elle s'appuie sur un modèle d'annotation (voir la table 7.2).

```
1   -- Déclaration du modèle d'annotation, qui
2    annote le modèle ALFA #1.
3   #10= M_annot(#1,(#11));
4
5   -- Déclaration d'une annotation sur la
6   fonction #3 qui annote les modes de pannes
7   admissibles sur la fonction.
8   #11= Annotation(#3, #12);
9   #12= Propriété(mode_panne,('Total_loss',
10  'Partial_loss_sym', 'Erroneous_sym'));
```

Tab. 7.2 - Exemple d'annotation du modèle ALFA (en Express)

Les instances générées du modèle de sécurité sont présentées dans la table 7.3.

```
1   -- Déclaration du modèle MBFHA
2   #100= M_MBFHA((#102,#103),(#104), *,
3   (#105,#106));
4
5   -- Déclaration de l'opération et de la
6   fonction. Les attributs de la fonctions
7   dépendent des annotations.
8   #102= Op_ALFA('Gérer la poussée');
```

```
9       #103= Fct_ALFA('Calculer la poussée',
10      ('Total_loss','Partial_loss_sym',
11      'Erroneous_sym'), .false.);
12
13      -- Déclaration de la donnée puis des liens.
14      Les attributs du lien sortant dépendent des
15      annotations.
16      #104= Data_ALFA('Commande de poussée');
17      #105= Lien_out_ALFA(#102,#104,
18      'service_principal', 'High', 'Hneg');
19      #106= Lien_trig_ALFA(#104,#103);
```

Tab. 7.3 - Instances générées du modèle de sécurité (en Express)

La seconde étape de la transformation de modèle déduit de la table précédente le modèle AltaRica correspondant. Les annotations par du code AltaRica, permettant l'expression des assertions dans les nœuds AltaRica, sont celles présentées dans le chapitre 5.

Le modèle AltaRica du contrôle de la trajectoire au sol a déjà été présenté au chapitre 5 sur la figure 5.7. La figure 7.5 illustre quant à elle le modèle AltaRica correspondant à l'accélération de l'avion (« trajectoire_1 » et « trajectoire_2 » sont agrégées en la donnée « trajectoire_contrôlée » dans le niveau hiérarchique supérieur).

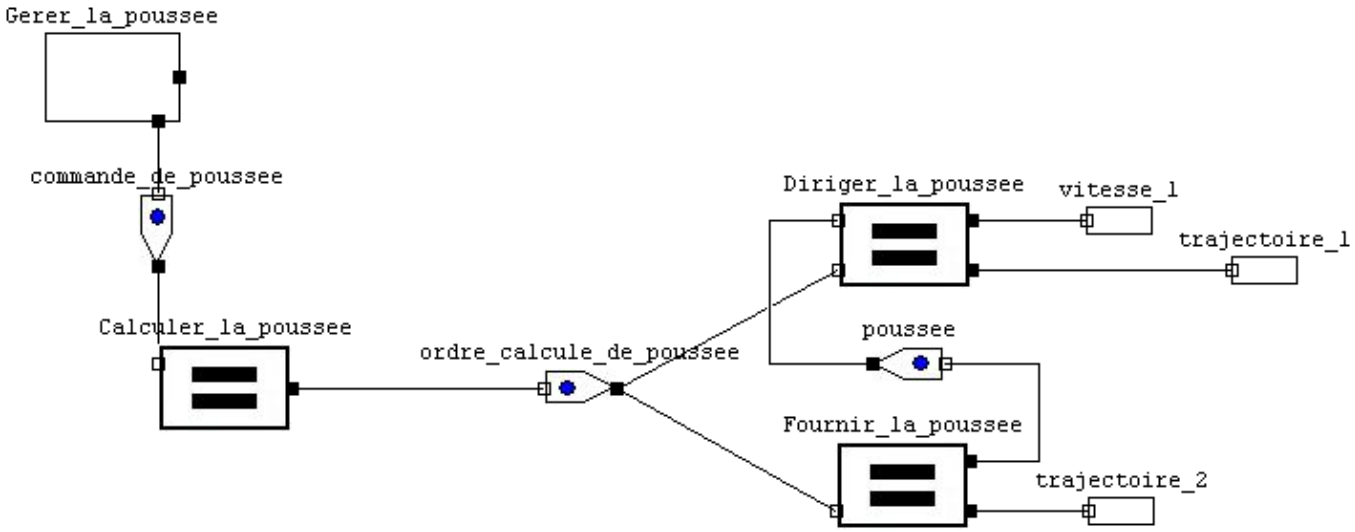


Fig. 7.5 - Modèle AltaRica du contrôle de l'accélération pendant le décollage.

Remarque. Les figures 5.7 et 7.5 montrent des modèles dont la vue a été modifiée manuellement pour être plus lisible. A la section 4.1 du chapitre 6, nous avons en effet évoqué que l'organisation graphique des nœuds est une problématique non traitée dans le cadre de cette thèse.

2.1.2 La complétion des modèles

Une fois la hiérarchie reconstituée, le modèle AltaRica du décollage prend la forme de la figure 7.6. A

gauche se trouve les deux tâches identifiées et dont les exemples de modèle sont ceux des figures 5.7 et 7.5. Au milieu de la figure, les quatre objectifs de vol sont mis en évidence (« connaissance de la trajectoire », « trajectoire_contrôlée » etc.) ; nous constatons qu'ils agrègent bien tous les services qui y contribuent.

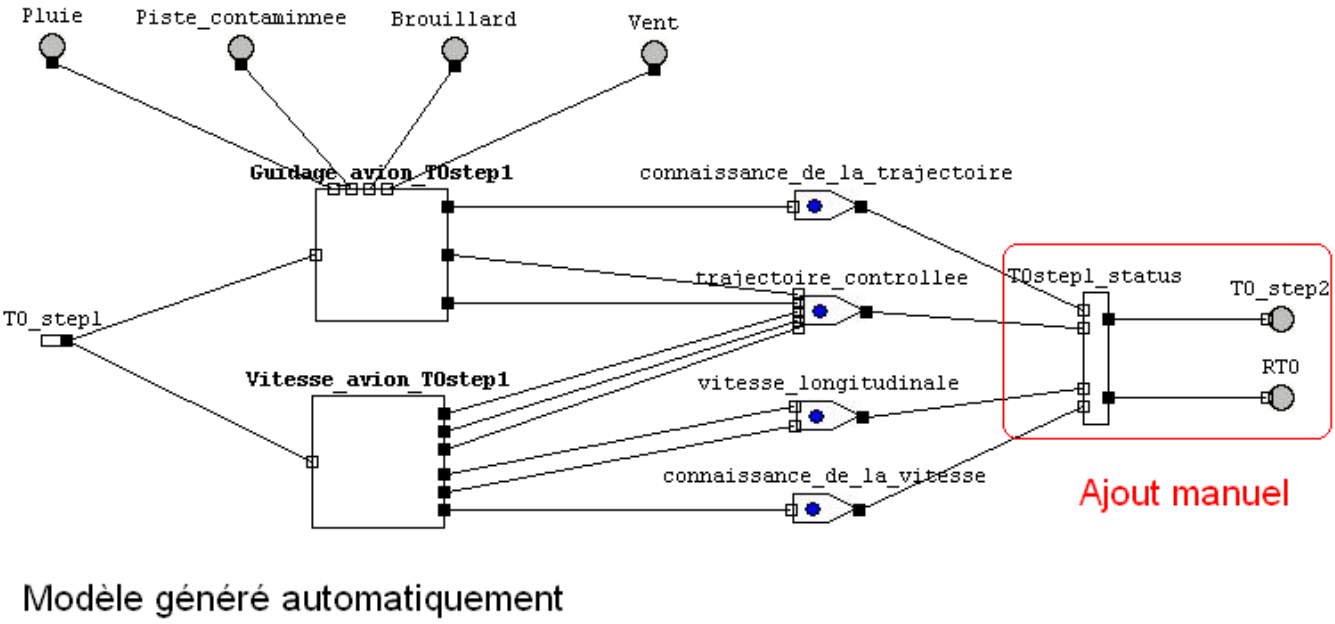


Fig. 7.6 - Modèle AltaRica du décollage avec ajout d'un observateur

Le modèle généré est incomplet pour réaliser des analyses de sécurité. Nous avons vu au chapitre 5 qu'il faut ajouter des observateurs représentant des situations redoutées. A droite de la figure, « TOstep1_status » joue le rôle d'observateur : il détermine, en fonction des objectifs de vol, si le décollage peut se poursuivre en toute sécurité ou si une interruption du décollage doit être réalisée. Comme nous l'avons vu aux chapitres 4 et 5, l'efficacité de chaque objectifs de vol est comparée à des seuils d'efficacité (voir la table 7.4) ; ces valeurs seuils sont « Low » pour « info_trajectoire » et « High » pour les trois autres objectifs de vol.

```
1     assert
2
3     TO_step2 = case{
4        ((info_trajectoire = High or
5         info_trajectoire = Low) and
6         controle_trajectoire = High and
7         info_vitesse = High and
8         controle_vitesse = High): true,
9         else false};
10
11    RTO = not(TO_step2);
```

Tab. 7.4 - Assertion AltaRica de l'observateur « TOstep1_status »

2.2 L'analyse des modèles pour assister l'analyse des risques

Puisque les activités représentées dans les modèles A.O.F. de la section 1 sont de niveau avion, l'analyse des risques qui en découle est une FHA avion. Nous souhaitons alors mettre en perspective l'exploitation de notre approche de modélisation avec les FHA avion existantes, pour les précédents programmes avion.

2.2.1 La recherche des conséquences de pannes fonctionnelles

Intéressons nous à la fonction « Fournir la poussée ». Il est évident que la perte totale de cette fonction pendant le décollage conduit à interrompre ce décollage. Mais cette fonction fait également partie du modèle concernant le contrôle de la décélération au sol grâce à la poussée inversée. Grâce à la simulation, nous pouvons montrer que la perte totale de cette fonction n'a pas d'impact dégradant sur le résultat de l'objectif de vol « vitesse longitudinale » du fait de la présence des freins de roues (voir la figure 7.7). Une analyse théorique de ce scénario de panne est proposée sur la partie supérieure de la table 7.5 : la criticité MAJ qui lui est associée prend en compte le fait que la panne n'a pas d'effets sévères sur l'arrêt de l'avion sur la piste.

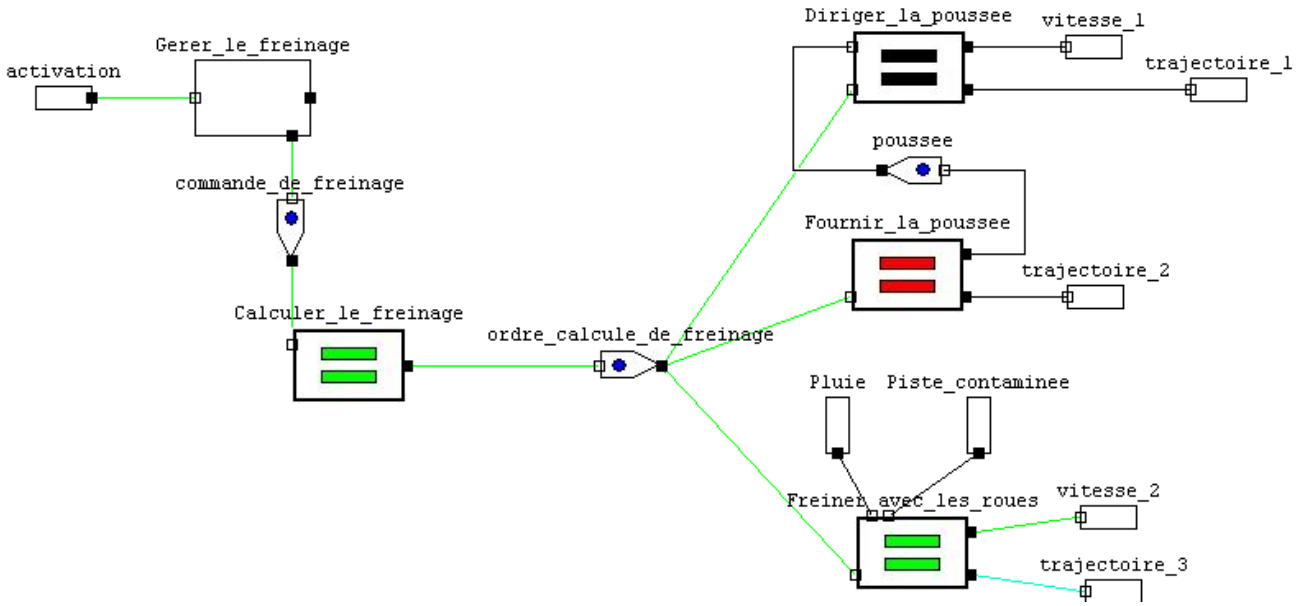


Fig. 7.7 - Simulation de la perte totale de la fonction « Fournir la poussée »

Mais, la simulation met également en évidence que si cette panne survient sous des conditions environnementales qui contaminent la piste, alors le freinage par les roues se dégrade et ne permet plus d'arrêter l'avion de façon satisfaisante (figure 7.8).

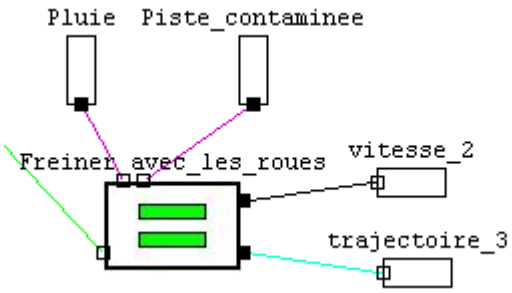


Fig. 7.8 - Simulation de l'effet de la contamination de la piste sur le freinage par les roues

Le risque est alors celui d'une sortie de piste ; les capacités de freinage restantes pourraient justifier que le scénario de panne n'est pas catastrophique, mais dangereux (HAZ), comme cela est illustré sur la partie inférieure de la table 7.5.

<table>
<tr><td colspan="5"><u>Aircraft/System Function :</u> Fournir de la <u>poussée</u></td></tr>
<tr><td><u>Failure :</u> A</td><td>Perte totale de production de la poussée</td><td>Classification</td><td>Justifications & Parameters</td><td>Requirements (FC)</td></tr>
<tr><td colspan="2"><u>Flight Phase :</u> décollage

<u>Repercussions :</u>
La perte totale de la poussée conduit à l'incapacité de l'avion à atteindre sa vitesse de rotation.

<u>Crew detection :</u>
• Alarmes associées à la perte d'accélération.
• Sensation physique associée à la perte d'accélération.

<u>Crew action :</u>
L'équipage cockpit réalise un RTO.</td><td>MAJ</td><td></td><td><u>FC :</u>
MAJ – Perte totale de la production de poussée pendant le décollage.</td></tr>
<tr><td><u>Failure :</u> B</td><td>Perte totale de production de la poussée sur piste contaminée</td><td>Classification</td><td>Justifications & Parameters</td><td>Requirements (FC)</td></tr>
<tr><td colspan="2"><u>Flight Phase :</u> décollage

<u>Repercussions :</u>
La perte totale de la poussée conduit à l'incapacité de l'avion à atteindre sa vitesse de rotation. La présence d'une piste contaminée (pluie, huile, givre etc.) rend plus difficile l’arrêt de l'avion sur la piste. Les répercussions sont donc considérées HAZ.

<u>Crew detection :</u>
• Alarmes associées à la perte d'accélération.
• Sensation physique associée à la perte d'accélération.

<u>Crew action :</u>
L'équipage cockpit réalise un RTO.</td><td>HAZ</td><td></td><td><u>FC :</u>
HAZ – Perte totale de la production de poussée pendant le décollage sur piste contaminée.</td></tr>
</table>

Tab. 7.5 - Perte totale de la production de la poussée (exemple théorique)

Cet exemple de simulation montre bien l'intérêt de la modélisation pour :

- aider les analystes à estimer la criticité de scénarios de pannes ;
- offrir un moyen visuel et interactif pour comprendre le scénario de panne ;
- faciliter les échanges avec les architectes avion et les concepteurs des systèmes aéronautiques.

2.2.2 La recherche de combinaisons de pannes fonctionnelles pertinentes

Le principe de recherche des causes d'une situation redoutée et le générateur de séquences présentés aux chapitres 4 et 5 permettent de rechercher des combinaisons de pannes particulièrement importantes à étudier. Notamment, dans la section 4.3 du chapitre 4, nous avons montré que les scénarios de pannes conduisant à une situation redoutée et empêchant le bon fonctionnement des moyens correctifs prévus dans cette situation sont des scénarios de pannes pertinents.

Nous effectuons cette recherche pour déterminer les pannes fonctionnelles qui, pendant le décollage, conduisent à devoir réaliser un RTO et dont le risque est de faire une sortie de piste malgré l'interruption du décollage. Ce critère de recherche, clairement exprimé et s'appuyant sur l'observateur illustré dans la section 2.1.2, a permis de révéler plusieurs combinaisons de pannes particulièrement pertinentes à analyser dont la perte totale combinée du guidage à la roulette de direction avec le guidage aérodynamique (voir la table 7.6).

Aircraft/System Function : Guider avec la roulette de direction et aérodynamiquement				
Failure : C	Perte totale combinée du guidage avec la roulette de direction et du guidage aérodynamique	Classification	Justifications & Parameters	Requirements (FC)
Flight Phase : Décollage, Atterrissage. Repercussions : Perte totale ou presque totale des systèmes nécessaires au contrôle de la direction de l'avion pendant le décollage ou l'atterrissage. Les capacités restantes sont insuffisantes pour empêcher une possible sortie de piste à haute vitesse. Crew detection : • Alarmes. • Sensation physique associée à la perte		**CAT**	Classification criteria : Une sortie de piste avec une vitesse supérieure à 60 nœuds est classée CAT. STUDY : Étudier la pertinence d'alarmes pour informer l'équipage cockpit de la perte totale de guidage de	**FC :** **CAT** – Perte totale du contrôle de la trajectoire pendant le décollage ou l'atterrissage.

de contrôle de la trajectoire. <u>Crew action :</u> L'équipage cockpit applique une décélération maximale afin de minimiser les répercussions (RTO)		l'avion.	

Tab. 7.6 - Perte totale du contrôle de la trajectoire (exemple théorique)

2.2.3 Bilan sur l'analyse des modèles de l'étude de cas

Dans ce chapitre, nous avons montré sur une étude de cas que les modèles A.O.F. peuvent servir de base à une approche de modélisation permettant d'assister les analystes de la sécurité lors de l'analyse des risques. Les analystes de la sécurité interviennent à trois niveaux :

1. lors de la transformation de modèle, en définissant les annotations sur les modèles A.O.F. ;
2. en ajoutant des observateurs décrivant les situations redoutées qu'ils souhaitent étudier ;
3. en utilisant les outils de simulation interactive et de génération de séquences pour déterminer les conséquences de pannes fonctionnelles ou rechercher des combinaisons de pannes pertinentes.

Les résultats obtenus avec notre approche de modélisation sont pertinents. En effet, nous les avons confrontés avec les analyses de risque effectuées selon la méthode traditionnelle. En particulier, sur la recherche de combinaisons de pannes pertinentes, nous constatons que les scénarios identifiés sur notre cas d'étude, sont déjà traités dans les analyses de risques courantes. Nos travaux de modélisation ont alors validé les choix des combinaisons de pannes étudiés en mettant en évidence un critère de recherche et donc une justification claire.

Chapitre 8

Extension des modèles à l'étude des Facteurs Humains

Sommaire

Résumé. Dans ce chapitre, nous présentons une extension possible des modèles de sécurité afin de prendre en considération des aspects Facteurs Humains. L'objectif est de renforcer l'analyse de la sécurité autour des interactions Homme/Machine. Ce chapitre introduit le domaine Facteurs Humains et ses liens avec la sûreté de fonctionnement. Puis, nous montrons plusieurs perspectives d'extension de nos modèles de sécurité : étude de la charge de travail de l'équipage, modélisation de conditions humaines dégradées et modélisation et analyse de la variabilité du système socio-technique.

1 Introduction : vers la prise en compte d'aspects Facteurs Humains

Nos travaux propose une évaluation de la sécurité des architectures fonctionnelles des systèmes socio-techniques. Or, d'après la section 3.1.3 du chapitre 2, un tel système socio-technique ne se résume pas seulement à sa composante technique, mais il comprend également une composante humaine (les acteurs du produit) et organisationnelle (les procédures de pilotage et la hiérarchisation de l'équipage par exemple). Nous avons vu au chapitre 1 que l'analyse des risques de l'avion est une activité réalisée sur différents scénarios de vol et évaluant entre autre les capacités de détection et de réaction de l'équipage à des pannes. Cette analyse particulière du système socio-technique considéré fait donc intervenir également des caractéristiques humaines et organisationnelles. Ainsi, une nouvelle perspective d'enrichissement de l'analyse des risques consiste à approfondir l'étude des composantes humaines et organisationnelles, insuffisamment traitées aujourd'hui à partir des connaissances des analystes de la sécurité.

1.1 Le domaine des Facteurs Humains

Le domaine d'étude des Facteurs Humains ([Villemeur, 1988], [Reason 1990], [Hollnagel 2012]) repose sur les sciences humaines et sociales. Les sciences humaines s'intéressent à la composante humaine des systèmes socio-techniques (voir la section 3.1.3 du chapitre 2). Les sciences sociales, quant à elle, traitent la composante organisationnelle. Le cadre théorique et méthodologique du domaine d'étude des Facteurs Humains est celui de la psychologie cognitive : un acteur humain est perçu comme un système cognitif, c'est-à-dire une entité capable d'acquérir, de stocker, de transmettre et d'utiliser une quantité limitée d'information. Les apports de cette science sont multiples, en particulier dans le cadre des interactions hommes-machines, de l'ergonomie, de l'amélioration des conditions de travail et de la maîtrise des erreurs humaines. Or, ces bénéfices fournissent indirectement plus de sécurité dans l'utilisation des systèmes embarqués. Les études des facteurs humains ont en effet de nombreux liens sous-jacents avec la sûreté de fonctionnement.

Contrairement à de nombreux domaines de l'ingénierie comme l'étude des performances ou de la sécurité, l'étude des facteurs humains ne s'appuie pas sur des sciences exactes (mathématiques) mais sur des raisonnements empiriques et rarement exhaustifs. La psychologie cognitive est l'étude des processus de traitement de l'information qui interviennent dans les comportements humains, à partir d'expériences et de modèles statistiques. L'objectif de telles études est d'évaluer statistiquement les probabilités d'occurrence de situations déjà observées (par exemple des accidents) et de définir des recommandations sur les plans humain et organisationnel afin de renforcer la sécurité.

1.2 Sûreté de Fonctionnement et Facteurs Humains

A l'instar de ce qui a été constaté au niveau de la composante technique, les interactions entre les acteurs des composantes humaines et organisationnelles sont très nombreuses et rendent difficiles les analyses à l'échelle de chaque acteur. Pour s'affranchir de cette difficulté, les sciences humaines et sociales s'intéressent

également à un niveau d'abstraction supérieur : l'étude des fonctions du système socio-technique. Dans le domaine des Facteurs Humains, une fonction désigne généralement n'importe quelle activité humaine ou matérielle. Sa définition correspond alors à la définition des activités de la section 3.1.1 du chapitre 2. Nous conservons le terme d'activité par la suite et maintenons la définition des fonctions telle qu'elle a été formulée au chapitre 2.

Comme pour la sûreté de fonctionnement, l'objectif des facteurs humains est d'étudier les interactions entre les activités et d'évaluer les activités grâce à des critères mesurables. Traditionnellement, les analyses des facteurs humains et de la sécurité des systèmes socio-techniques, sont réalisées séparément et des activités supplémentaires viennent éventuellement couvrir les interactions entre ces deux domaines d'étude. Grâce à la modélisation des couches opérationnelles et fonctionnelles, les domaines des facteurs humains et de la sécurité peuvent partager un référentiel commun ([Woods et al. 2005], [Hollnagel 2012]). Les modèles A.O.F. du projet ALFA peuvent être utilisés comme un tel référentiel. Dans cette perspective, D. Woods et E. Hollnagel proposent d'étudier les systèmes socio-techniques, et non plus seulement les acteurs du système, comme des systèmes cognitifs (ou « *Joint Cognitive Systems* »). De cette approche sont nées plusieurs méthodes, dont des analyses par des modèles, tels que FRAM présentés dans la section 3. Les bilans actuels de l'utilisation de telles méthodes montrent la nécessité d'une collaboration entre les spécialistes de la sûreté de fonctionnement et les spécialistes des sciences humaines et sociales ([Bieder 2006], [Belmonte 2009]).

2 Extension des modèles de sécurité avec des aspects Facteurs Humains

L'analyse des risques possède naturellement plusieurs liens avec les études des facteurs humains. Les facteurs humains apportent une assistance pour déterminer la criticité d'un scénario de panne, grâce à l'évaluation des moyens de détection des pannes par l'équipage et des capacités à appliquer des actions correctives. Et inversement les résultats de l'analyse des risques pilotent certaines activités orientées facteurs humains. Par exemple, il est nécessaire d'étudier le comportement des pilotes dans tous les cas de pannes menant à une condition de panne catastrophique (voir la section 3.3.1 du chapitre 1).

Nos modèles d'analyse des aspects dysfonctionnels sont construits sur la base d'un référentiel fonctionnel. Dans le cadre de nouvelles perspectives de nos travaux de recherche, nous pouvons nous interroger sur la capacité à étendre l'exploitation des modèles de sécurité en intégrant des connaissances du domaine des facteurs humains. Les sous-sections suivantes présentent quelques pistes de réflexion :

- modélisation des prises de décision ;
- évaluation de la charge de travail de l'équipage ;
- modélisation des conséquences du stress ou de la fatigue.

2.1 Modélisation des prises de décision et des modes de fonctionnement

Dans les modèles ALFA, les prises de décision par l'équipage ne sont pas directement prises en compte, mais elles sont intégrées au sein du scénario de vol (voir la section 3.3.5 du chapitre 2). Dans la couche fonctionnelle se trouve une problématique similaire avec la modélisation des modes de fonctionnement nominaux des activités. Eux aussi sont occultés car, en se plaçant dans un scénario de vol particulier, le modèle présuppose un mode de fonctionnement pour chaque fonction (voir la section 3.1 du chapitre 4).

Une perspective est d'intégrer à l'avenir les prises de décision, les modes de fonctionnement et les changements de mode de fonctionnement dans les modèles ALFA puis dans les modèles de sécurité afin de diminuer le nombre de modèle à développer pour chacun des scénarios de vol.

Une solution à explorer pourrait être celle mise en œuvre dans [Chaudemar 2012] où ces choix sont modélisés dans AltaRica par des événements, à l'instar des pannes fonctionnelles.

Exemple. A n'importe quel moment pendant le décollage, les pilotes peuvent choisir de faire un RTO. Le modèle intégrerait alors un événement nommé RTO, dont la transition permettrait de passer du modèle du décollage à celui du RTO.

Cette perspective se heurte toutefois à deux problématiques :

- du point de vue des analyses de sécurité, il faut s'assurer que l'effort supplémentaire de modélisation pour prendre en compte des prises de décision et des changements de mode contribue à renforcer l'analyse des risques en apportant des résultats supplémentaires pertinents ;
- la modélisation des prises de décision et des changements de modes de fonctionnement par des événements induit que les pannes fonctionnelles et les conditions environnementales ne sont plus les seuls événements du modèle. Afin de conserver la notion de coupe minimale, ces événements nominaux doivent être distingués des pannes. Les outils OCAS et RAMSES permettent une telle distinction grâce aux sous-familles d'événements ; le travail de modélisation n'est guère plus important. Le générateur de séquences permet de ne faire varier qu'un seul type d'événements (les pannes) et donc d'identifier des combinaisons de pannes.

2.2 Évaluation de la charge de travail de l'équipage

D'après la table 1.2 du chapitre 1 qui liste les critères qualitatifs et quantitatifs de classification des scénarios de panne dans l'analyse des risques, la charge de travail de l'équipage et plus précisément celle des pilotes fait partie des critères les plus importants. Les analystes de la sécurité sont amenés à estimer les potentielles augmentations de la charge de travail dans les cas de panne. Cette problématique rejoint celle traitée à la section 3.4.2 du chapitre 1 concernant la difficulté à estimer les conséquences des pannes sur les performances de l'avion. Prendre en considération la charge de travail dans l'analyse des risques revient à estimer les conséquences des pannes sur les performances des acteurs humains de l'avion.

Or, dans les modèles A.O.F., les opérations réalisées par les acteurs humains dans les différents scénarios de vol et scénarios de pannes sont connues et modélisées. Grâce à l'étude des liens d'activation, il est possible de déterminer les opérations réalisées successivement et en parallèle. Ainsi, une seconde perspective est d'intégrer une mesure de la charge de travail nécessaire pour réaliser chaque opération et d'en déduire la charge de travail globale par rapport à la séquence des opérations réalisées dans chaque scénario de panne.

Concrètement, en première approche, une valeur de charge de travail peut être associée à chaque opération. Il est possible de choisir une valeur entière ou des valeurs qualitatives comme pour l'efficacité. A l'instar de la notion d'efficacité, cette annotation est susceptible de découler d'autres données propres à la conception, telles que des exigences opérationnelles. Puis, la charge de travail globale peut être obtenue en première approche par l'addition des valeurs de charge de toutes les opérations réalisées simultanément par le même acteur. Chaque acteur possède ainsi sa valeur de charge globale. Si cette valeur globale dépasse un seuil défini par les spécialistes alors l'acteur peut-être jugé en surcharge, ce qui peut conduire potentiellement à des erreurs humaines.

Cette approche a le mérite de ne pas présumer de l'impact d'une surcharge ; elle vise plutôt à identifier les scénarios de pannes qui semblent conduire à des surcharges de travail. Ces scénarios de pannes sont alors susceptibles d'être plus critiques qu'initialement prévus.

Les modèles AltaRica que nous proposons doivent toutefois être soumis à une modification majeure : la modélisation explicite des états d'activation. En effet, savoir si une activité est exécutée au cours d'une phase de vol n'est pas suffisant puisqu'il faut déterminer l'ensemble des activités réalisées simultanément à chaque instant de la phase de vol. Il faut donc modéliser l'activation et la désactivation pour combiner les valeurs de charge de travail des opérations activées simultanément. Une proposition de modélisation de l'activation peut être trouvée dans [Chaudemar 2012] (voir aussi la section 2.3.2 du chapitre 5).

2.3 Modélisation de conditions humaines dégradées

Les sciences humaines étudient en particulier les conditions humaines dégradées, c'est-à-dire des états dans lesquelles les performances des acteurs sont anormales. Le stress et la fatigue sont des exemples courants de telles situations. Parfois, ces dégradations ont des conséquences positives sur les performances. Le stress par exemple peut accroître les réflexes.

Dans le cadre de la sûreté de fonctionnement, les cas pertinents sont ceux où des pannes conduisent à des risques plus sévères lorsque les performances de certains acteurs sont dégradées. Dans de tels cas, il est nécessaire de vérifier que les pannes elles même n'ont pas provoqué les dégradations dans le comportement des acteurs.

Exemple. Une panne moteur peut conduire le pilote a un état de stress dans lequel il risque de se focaliser davantage sur la vitesse de l'avion au dépend de la bonne maîtrise de sa trajectoire. Ceci est un exemple théorique de combinaison qu'il peut être pertinent d'analyser.

Comme pour la recherche des combinaisons de pannes, l'objectif de la modélisation est d'identifier automatiquement et selon des critères précis et pré-établis les combinaisons de pannes avec des comportements dégradés pouvant conduire à des situations redoutées.

La perspective associée à cette problématique est donc de modéliser les comportements dégradés à l'instar des pannes fonctionnelles. Concrètement, dans les modèles AltaRica, il s'agit de définir de nouveaux événements dont les transitions affectent en premier lieu les opérations pour en dégrader les services. Le choix des comportements dégradés à modéliser et l'impact des transitions sur les services ne peuvent être définis qu'en utilisant des connaissances propres aux facteurs humains. Cette perspective met à nouveau en évidence la nécessité d'une collaboration entre les analystes de la sécurité et les spécialistes des facteurs humains.

3 Modélisation FRAM : « *Functional Resonance Analysis Method* »

L'approche FRAM (« *Functional Resonance Analysis Method* »), développée par E. Hollnagel ([Hollnagel 2012]) permet d'analyser un système socio-technique à travers une identification des dépendances entre des activités réalisées par ce système. A ce titre, l'étude des dépendances fonctionnelles pour assister l'analyse des risques rentre dans le cadre de l'approche FRAM. Il existe donc un lien très fort entre cette approche et nos travaux de modélisation. La section 3.1 présente l'approche de modélisation FRAM. Puis, dans la section 3.2, nous introduisons la perspective de développer des modèles FRAM à l'aide du langage AltaRica.

3.1 Présentation de l'approche FRAM

L'approche FRAM se base sur une représentation des dépendances entre les activités du système socio-technique afin d'analyser la performance globale du système par rapport aux performances individuelles de chaque activité. Dans cette approche, la performance individuelle de chaque activité est considérée comme variable. La **variabilité** est définie comme une mesure des écarts potentiels de performances intrinsèques à chaque activité. Pour les systèmes techniques, la variabilité représente les imperfections de la spécification, de la conception et de la production. La variabilité humaine et organisationnelle, quant à elle, résulte des capacités naturelles de l'homme à s'adapter à diverses situations.

L'exploitation des modèles FRAM repose sur la notion de **résonance**. Il s'agit d'analyser la superposition, au travers des dépendances entre les activités, de la variabilité de chaque activité pour détecter des variabilités potentiellement critiques du point de vue de la sécurité. Cette théorie est empruntée à la physique ondulatoire dans laquelle des bruits (signaux non linéaires) sont additionnés et peuvent devenir prépondérants par rapport à un signal périodique attendu.

Dans les modèles FRAM, les activités sont représentées par les nœuds de la figure 8.1. Dans l'approche FRAM, ces nœuds s'appellent molécules.

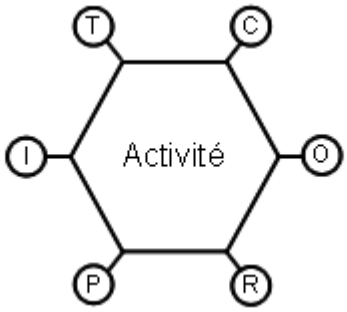


Fig. 8.1 - Nœud (molécule) d'un modèle FRAM

Chaque activité est caractérisée par six paramètres.

1. **I** (« *Input* ») : les entrées sont les données utilisées et transformées par l'activité pour produire ses services.
2. **O** (« *Output* ») : les sorties sont les services de l'activité.
3. **T** (« *Time* ») : il s'agit de toutes les contraintes temporelles de l'activité, notamment son temps d'exécution et sa fréquence d'activation.
4. **C** (« *Control* ») : le contrôle de l'activité est l'ensemble des moyens permettant de vérifier que l'activité se déroule comme prévue.
5. **P** (« *Preconditions* ») : les préconditions forment l'ensemble des conditions d'activation de l'activité.
6. **R** (« *Resources* ») : les ressources sont utilisées ou consommées par les activités pour leur exécution. Les ressources désignent principalement les liens vers la couche matérielle et les acteurs humains.

La figure 8.2 présente un modèle FRAM proche de l'exemple du contrôle de la trajectoire au sol. Nous y retrouvons l'opération « Gérer la trajectoire » qui transmet une donnée nécessaire (précondition) à la fonction « Calculer le guidage » (voir le modèle ALFA, figure 2.9, du chapitre 2). Les données utiles sont modélisées en connectant la sortie d'une molécule à l'entrée et non à la précondition. Sur ce modèle FRAM est illustré également le lien de contrôle car la visibilité extérieure obtenue par les pilotes leur permet de contrôler le bon fonctionnement des modifications de trajectoire. Une ressource est également représentée : il s'agit du réseau informatique d'échange de données de sein d'un avion (ADCN, « *Avionic Data Central Network* »).

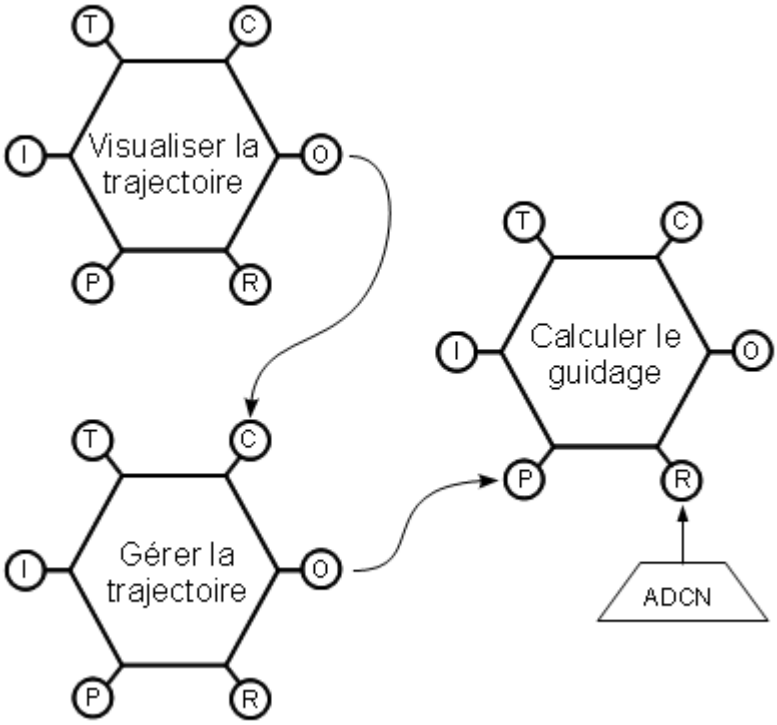


Fig. 8.2 - Exemple théorique d'un modèle FRAM

L'utilisation des modèles FRAM est récente et couvre principalement l'analyse d'accidents. Dans [Mawhin et al. 2011], l'approche FRAM est utilisée pour étudier les répartitions des tâches dans le cockpit d'un avion : une première répartition est analysée entre le pilote et le copilote et la seconde entre les différents écrans permettant aux pilotes de visualiser des données de l'avion. Le cas d'étude est celui de l'approche finale, lorsqu'il se produit tardivement un changement de piste d'atterrissage (par exemple sur demande de la tour de contrôle). Cet article est un exemple d'application de la méthode FRAM pour assister la conception des systèmes embarqués.

Dans [Martinie et al. 2012], FRAM est utilisé en complément de deux autres méthodes de modélisation pour représenter et étudier la variabilité du contrôle du trafic aérien (ATM pour « *Air traffic Management* ») sous différents points de vue. Chaque méthode de modélisation se focalise sur l'une des trois composantes d'un système socio-technique : l'organisation des activités, les activités humaines (opérations) et les activités techniques (fonctions). Dans ce cadre, FRAM est dédié à la modélisation de la composante organisationnelle du système étudié, permettant l'étude qualitative de la propagation de la variabilité entre les activités. Les deux autres méthodes de modélisation enrichissent cette étude grâce à une évaluation quantitative de la variabilité des opérations et des fonctions respectivement.

3.2 Expérimentation avec le langage AltaRica

L'approche de modélisation FRAM offre un moyen d'étude graphique des dépendances entre les activités. Cependant, une des principales limitations de cette approche est le manque d'un formalisme mathématique permettant d'exploiter les modèles à l'aide d'outils informatiques. Dans [Belmonte 2009], F. Belmonte

présente deux perspectives pour réaliser des évaluations quantitatives d'un modèle FRAM :

- traduire le modèle FRAM par un réseau de neurones afin d'évaluer le niveau de performance des activités ;
- calculer à partir du modèle FRAM des probabilités de franchissement de seuils critiques de performances d'activités à partir de la combinaison des variabilités.

La performance des activités dans les modèles FRAM peut être modélisée à l'aide de la notion d'efficacité que nous avons introduite à la section 2.3.3 du chapitre 4. Nous constatons également que les entrées, les sorties et les préconditions des molécules FRAM permettent de modéliser les activités et les échanges de données entre les activités. La représentation FRAM semble alors compatible avec les caractéristiques fondamentales du formalisme que nous avons proposé dans les chapitres 4 et 5. Nous identifions la perspective suivante : combiner nos travaux réalisés à l'aide du langage formel AltaRica avec l'approche FRAM ([Maitrehenry et al. 2011]). Les bénéfices attendus sont triples :

- fournir aux modèles FRAM un formalisme mathématique générique à l'aide du langage AltaRica ;
- formaliser les lois de comportement des activités des modèles MB_{FHA} à l'aide de la théorie de la résonance et des variabilités (les pannes fonctionnelles représentant la variabilité des systèmes techniques) ;
- modéliser la variabilité humaine dans les modèles MB_{FHA} pour prendre en compte des combinaisons de pannes fonctionnelles avec des écarts de performance humaine.

Suite à une première expérimentation avec le langage AltaRica, nous avons identifié que le formalisme FRAM ne permet pas de prendre en considération les conditions environnementales. A l'inverse, les moyens de contrôle, les ressources et des contraintes temporelles peuvent être ajoutées alors que ces trois caractéristiques ne sont pas étudiées actuellement dans les modèles de sécurité que nous proposons. Des travaux supplémentaires sont nécessaires pour utiliser l'approche FRAM dans la phase de conception pour les analyses préliminaires de sécurité. Toutefois, cette approche de modélisation peut présenter l'avantage de réunir autour d'une même représentation des analystes de la sécurité et des spécialistes des facteurs humains.

Conclusion

Bilan de nos travaux

Nos travaux ont tout d'abord permis d'identifier cinq axes d'amélioration sur l'analyse des risques.

1. Mieux identifier les fonctions à analyser : l'analyse des risques étant une analyse fonctionnelle, les fonctions du système étudié et leur périmètre doivent être explicités.
2. Mieux déterminer les effets des scénarios de pannes : le nombre important d'interactions entre les fonctions rend difficile l'analyse de la propagation des pannes.
3. Mieux prendre en compte des données supplémentaires comme la charge de travail de l'équipage, les conditions environnementales ou des aspects facteurs humains.
4. Mieux traiter les scénarios de pannes multiples : du fait des interactions entre les fonctions, certaines pannes peuvent se combiner pour conduire à des situations sévères du point de vue de la sécurité.
5. Renforcer la communication et la traçabilité entre l'analyse des risques et les documents de la conception.

Notre proposition s'appuie sur une nouvelle approche de modélisation ayant pour but l'étude de la propagation des pannes fonctionnelles à travers les dépendances fonctionnelles. Nous avons organisé nos travaux pour répondre aux trois objectifs majeurs.

Objectif 1 : formaliser les concepts d'une nouvelle approche de modélisation pour assister l'analyse des risques

L'étude des interactions entre les fonctions, appelées dépendances fonctionnelles, permet l'analyse de la propagation des pannes fonctionnelles. Or, des modèles représentant les dépendances fonctionnelles existent déjà : il s'agit des modèles d'architectures opérationnelles et fonctionnelles (ou modèles A.O.F.). Chez Airbus, de tels modèles sont développés à l'aide du formalisme EFFBD dans le cadre du projet ALFA. Les modèles A.O.F. décrivent une phase d'exploitation du système étudié (par exemple le décollage d'un avion dans le cadre d'un vol classique). Et les éléments composants ces modèles sont : les activités décomposées en opérations et fonctions, les données échangées entre les activités, les conditions environnementales pouvant dégrader la performance de certaines activités et des liens décrivant les interactions entre les éléments précédents.

Cependant, les modèles A.O.F. du projet ALFA ne permettent pas de décrire des pannes et d'étudier le

comportement dysfonctionnel du système. Nous avons identifié les insuffisances suivantes :

- pas de formalisation des pannes fonctionnelles en lien avec les fonctions ;
- description limitée de la propagation des pannes fonctionnelles du fait de la non-évaluation des échanges de données entre les activités ;
- limites de l'utilisation du formalisme EFFBD du fait de l'introduction de dépendances fonctionnelles parasites et du manque de formalisme pour décrire l'impact des conditions environnementales sur le système.

Pour remplir ce premier objectif, nous avons formalisé les concepts nécessaires pour l'analyse des risques en s'appuyant sur les concepts existant dans les modèles ALFA. Les modèles de sécurité que nous avons proposé sont novateurs dans le sens où ils s'inscrivent dans une démarche d'identification des risques et non de vérification du respect des exigences de sécurité. Notre approche de modélisation peut donc être déployée au cours des premières phases de développement d'un système alors que les architectures matérielles ne sont pas précisément connues.

La formalisation mathématique proposée permet de rechercher, à partir d'un modèle, les conséquences d'un scénario de pannes ou les causes d'une situation redoutée. En cela, l'introduction de la notion d'efficacité pour évaluer la qualité des services des fonctions a été fondamentale. Elle permet d'associer une valeur à toutes les variables du modèle de sécurité selon les états de pannes et des conditions environnementales. Les lois de comportement des fonctions, évaluant l'efficacité des services à partir des dépendances fonctionnelles entrantes, sont génériques, c'est-à-dire qu'elles sont indépendantes de la signification de chaque fonction. Cette abstraction permet de déployer notre approche pour étudier différents systèmes dans différents cadres d'utilisation.

Objectif 2 : implémenter et expérimenter les concepts, formalisés précédemment, à l'aide du langage de modélisation AltaRica

Pour répondre au premier objectif, les concepts des modèles de sécurité ont été définis. Cependant, pour que les modèles soient exploitables par les analystes de la sécurité, il faut implémenter ces concepts dans un langage de modélisation compréhensible par un ordinateur. Le langage de modélisation choisi est AltaRica, supporté par des outils tels que le simulateur pas à pas et le générateur de coupes minimales. Comme tout langage de modélisation, AltaRica impose des contraintes sur lesquelles les concepts doivent s'adapter.

Pour remplir cet objectif, nous avons tout d'abord montré que le langage AltaRica est suffisamment expressif pour modéliser les éléments de nos modèles de sécurité. En revanche, nous avons mis en avant le fait que les variables ne peuvent être évaluées que dans des ensembles bornés. Par conséquent, nous avons proposé une discrétisation de l'efficacité à l'aide de cinq valeurs qualitatives. Nous avons mené des expériences pour déterminer la « bonne » granularité, permettant de garantir une précision satisfaisante des résultats tout en conservant un faible effort de modélisation de la part des analystes de la sécurité.

Nous avons également mis en évidence que les couches graphiques OCAS et RAMSES permettent l'édition, la visualisation et l'exploitation facile des modèles par les analystes de la sécurité. Les vues proposées

participent à renforcer la communication avec les concepteurs.

Enfin, la recherche des conséquences de pannes et des causes de situations redoutées est possible directement à l'aide d'outils d'analyse existant (le simulateur pas à pas et le générateur de coupes minimales). Il est important de noter que ces outils ne modifient nullement les principes d'analyse envisagés. Comme nous l'avons souhaité, le développement et l'exploitation des modèles, bien que dédié à l'analyse des risques, vient en support du processus de sécurité, sans le perturber. La manière même de réaliser une analyse des risques n'est pas modifiée par l'utilisation ou non de modèles.

Objectif 3 : formaliser la transformation de modèle entre les modèles de sécurité et les modèles ALFA. Implémenter et expérimenter cette formalisation à travers le développement d'un démonstrateur.

Les modèles A.O.F. du projet ALFA et les modèles de sécurité que nous proposons partagent de nombreuses données : les activités du système dans le cadre d'utilisation étudié, toutes les dépendances fonctionnelles (échanges de données et séquencement) et les conditions environnementales pouvant dégrader des activités. Les modèles ALFA possèdent des données propres telles que les durées d'exécution des activités. De même, les modèles de sécurité se distinguent par la formalisation des pannes fonctionnelles et de leur propagation. Le troisième objectif consiste à formaliser le lien entre les modèles ALFA et les modèles de sécurité que nous proposons ; le but est de disposer d'une passerelle permettant de générer les modèles AltaRica directement à partir des modèles ALFA.

Face à l'hétérogénéité des modèles, cet objectif a tout d'abord fait apparaître la nécessité de manipuler les modèles comme des objets. Pour cela, en se basant sur les techniques issues de l'Ingénierie Dirigée par les Modèles, nous avons décrit les modèles ALFA (les modèles de sécurité respectivement) par un métamodèle des modèles ALFA (un métamodèle des modèles de sécurité respectivement). Ces métamodèles ont été écrits à l'aide du langage de modélisation de données Express.

Notre approche se caractérise par son orientation métier et non outil de modélisation. Cela signifie que nous avons mis en évidence des domaines spécifiques de modélisation correspondant à des activités précises des processus de la conception et de la sécurité (un premier DSML pour les modèles ALFA et un deuxième pour les modèles de sécurité). Nous avons notamment mis en évidence les concepts fondateurs des différentes activités de modélisation à travers la définition des syntaxes abstraites des DSML (voir figure 6.1, rappelée ci-dessous). Cette distinction orientée métier permet aux concepteurs et aux analystes de la sécurité de garder la maîtrise de leurs propres concepts.

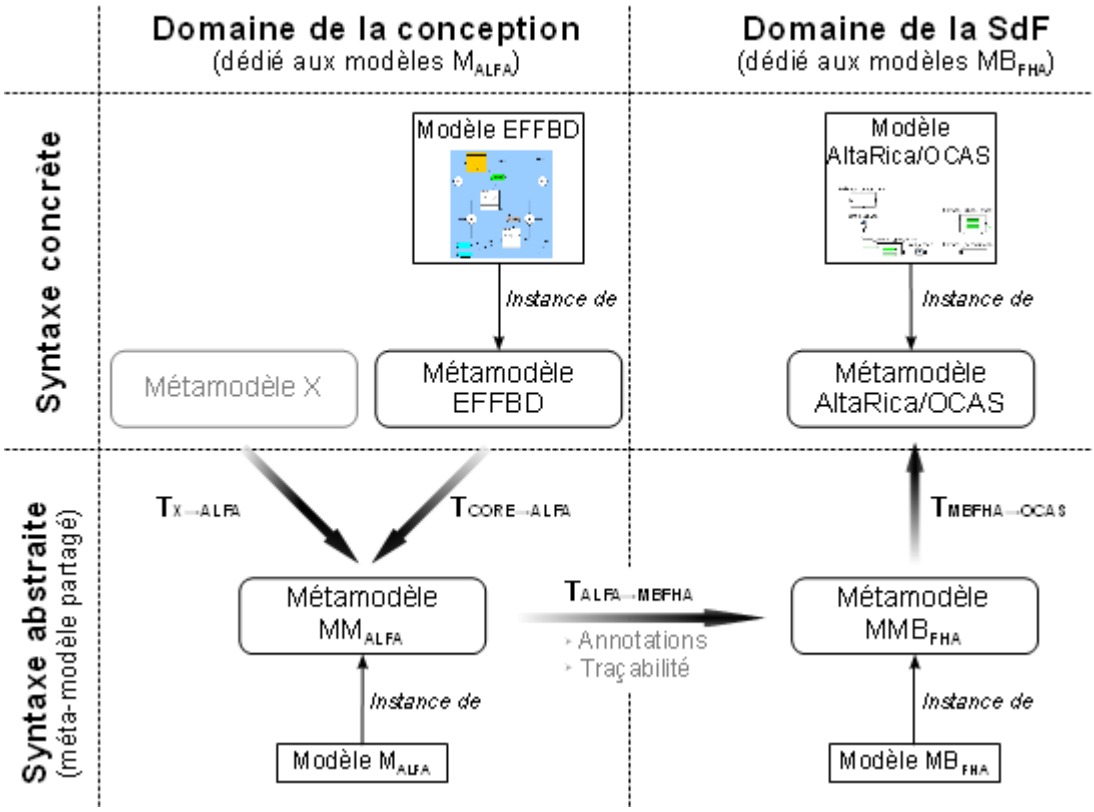


Fig. 6.1 - Approche de la transformation de modèle

La génération des modèles de sécurité à partir des modèles ALFA suit trois étapes de transformation :

- la première étape consiste à extraire les concepts des modèles ALFA à partir des modèles EFFBD correspondant ;
- la deuxième étape consiste à générer les concepts des modèles de sécurité correspondant aux modèles ALFA ;
- la troisième et dernière étape consiste à implémenter les concepts des modèles de sécurité dans le langage AltaRica.

Nous avons décrit une approche de transformation valable pour les différentes étapes : elle consiste à générer un modèle cible à partir d'un modèle source, d'un modèle d'annotations et d'un modèle de traces.

Les modèles d'annotations sont explicites, c'est-à-dire que les modèles annotés ne subissent aucune modification. Il est alors possible de définir plusieurs annotations sur un même modèle et de n'en sélectionner qu'une seule à chaque procédure de transformation. Par ce biais, les experts des modèles cibles pilotent la transformation grâce aux annotations, sans toucher aux modèles sources dont ils n'ont pas la maîtrise.

Les modèles de traces, quant à eux, sont nécessaires pour renforcer la confiance que les analystes de la sécurité peuvent avoir dans les modèles et leurs résultats d'exploitation. De plus, nous avons décrit une transformation dans laquelle les traces sont non seulement produites au cours de la transformation, mais servent également de données d'entrée au fur et à mesure de la transformation pour en simplifier les

procédures.

Nous avons appliqué cette approche générique de transformation aux deux dernières étapes de transformation identifiées sur la figure 6.1. Ces applications ont permis d'identifier que certaines annotations, telles que l'efficacité nominale des fonctions, ne devraient peut-être pas être maîtrisées par les analystes de la sécurité, mais pourraient découler, au contraire, d'autres données, telles que les exigences qualitatives de performance. Du point de vue du processus de sécurité, la génération automatique des modèles de sécurité renforce la confiance que les analystes de la sécurité peuvent avoir dans les modèles et leurs résultats d'exploitation. En cela, la traçabilité joue un rôle important.

Perspectives

Les trois objectifs de cette thèse ont été étudiés et remplis. Notre approche de modélisation et de transformation peut entrer dans une phase de renforcement par de futurs déploiements industriels. D'ores et déjà les modèles proposés ont montré leurs bénéfices pour assister l'analyse des risques : meilleure compréhension de la propagation des pannes, communication facilitée avec les concepteurs et gain en confiance et en temps de travail sur la réalisation de l'analyse des risques. Plusieurs perspectives à nos travaux ont été présentées tout au long de ce mémoire.

- Améliorer la précision dans le calcul de l'efficacité des résultats et des services des modèles de sécurité : en particulier, il s'agit de pallier les approximations résultant de l'agrégation de valeurs pour la combinaison des liens entrants dans les activités. Ces approximations, aujourd'hui nécessaires afin de pouvoir proposer des lois de comportement génériques, peuvent être sources d'erreurs de calcul dans certains modèles (les erreurs sont alors toujours pessimistes du point de vue de la sécurité).
- Enrichir la description des pannes : il s'agit notamment de modéliser le fonctionnement intempestif de certaines fonctions. Ce mode de panne a été écarté des modèles AltaRica actuels, car sa formalisation s'appuie sur des dépendances fonctionnelles absentes dans les modèles ALFA.
- Étudier les liens entre notre approche de modélisation et les modèles de sécurité développés pour la vérification du respect des exigences par les systèmes (approche MBSA, « *Model-Based Safety Assessment* »).
- Étudier les alternatives au langage AltaRica. Bien que ce langage permette de représenter les concepts des modèles de sécurité, il contraint les ensembles de valeurs pour l'évaluation de l'efficacité.
- Formaliser la transformation inverse permettant de transmettre des résultats issus de l'exploitation de nos modèles pour l'analyse des risques aux concepteurs des architectures opérationnelles et fonctionnelles.
- Décrire formellement une ontologie pour le domaine de modélisation de la sécurité : les modes de pannes, l'efficacité et d'autres types peuvent être davantage structurés, dans le but d'être directement réutilisables pour d'autres travaux de modélisation.

– Étendre les modèles de sécurité au domaine des Facteurs Humains afin de prendre en compte de nouvelles informations importantes pour la maîtrise de la sécurité, telles que la charge de travail, le stress ou encore la fatigue de l'équipage d'un avion.

Enfin, insistons sur le fait que même si les modèles proposés offrent un précieux support à l'analyse des risques, il n'en reste pas moins que l'expertise des analystes de la sécurité demeure essentielle et ne peut être substituée par des techniques de modélisation.

Bibliographie

[ATSB, 2010] : ATSB - Australian Transport Safety Bureau (2010). In-flight uncontained engine failure - overhead Batam Island, Indonesia - 4 November 2010, VH-OQA, Airbus A380-842.

[Villemeur, 1988] : Villemeur, A. (1988). Sûreté de fonctionnement des système industriels. Collection de la Direction des Etudes et de Recherches d'Electricité de France.

[Laprie et al., 1995] : Laprie J.-C., Arlat J., Blanquart J.-P., Costes A., Crouzet Y., Deswarte Y., Fabre J.-C., Guillermain H., Kaâniche M., Kanoun K., Mazet C., Powell D., Rabéjac C., Thévenod P. (1995). Guide de la Sûreté de Fonctionnement. Cépaduès - Editions.

[Mo, 1997] : Mo J. (1997). Contribution de l'informatique à la sûreté de fonctionnement d'un système socio-technique - Le cas de la maintenance en ligne des systèmes de contrôle de la navigation aérienne. Mémoire de thèse.

[Kehren, 2005] : Kehren, C. (2005). Motifs formels d'architectures de systèmes pour la sûreté de fonctionnement. Mémoire de thèse.

[CS-25 1309] : EASA (European Aviation Safety Agency) (2011). CS-25 Certification Specifications for Airworthiness of Large Aeroplanes, Amendment 11.

[EASA, 2011] : EASA - European Aviation Safety Agency (2011). Annual Safety Review.

[ARP4754A / ED-79A] : SAE aerospace & EUROCAE (2010). Guidelines for Development of Civil Aircraft and Systems.

[ARP4761 / ED-135] : SAE aerospace & EUROCAE (1996). Guidelines and Methods for Conducting the Safety Assessment Process on Civil Airborne Systems and Equipment.

[Robertson, 2006] : Robertson S., Robertson J. (2006). Mastering the Requirements Process. Addison-Wesley Professional.

[Wilkinson et al., 1998] : Wilkinson P.J., Kelly T.P. (1998). Functional Hazard Analysis for Highly Integrated Aerospace Systems. In *Proceedings of IEE Seminar on Certification of Ground/Air Systems*.

[Allenby et al., 2001] : Allenby K., Kelly T.P. (2001). Deriving Safety Requirements using Scenarios. In *5th IEEE International Symposium on Requirements Engineering*, pages 228-235. IEEE Computer Society Press.

[Flaus, 2008] : Flaus J.-M. (2008). A Model-Based Approach for Systematic Risk Analysis. In *Journal of Risk and Reliability*, pages 222: 79-93.

[Wassmuth et al., 2011] : Wassmuth M., Stilkerich S., Lübbers E. (2011). Distributed Safety Assessment for Airborne Systems: an industrial relevant approach for automated safety analysis and reporting. In *Proceedings of the International Workshop on Security and dependability for Resource Constrained Embedded Systems*.

[ALFA, 2012] : Vecchione E. (2012). ALFA Project Presentation. Airbus Technical Note.

[Reynolds, 2006] : Reynolds J. (2006). Validating DoD Architectures: The Promise of Systems Engineering. In *Proceedings of the CCRTS 2006: the state of the art and the state of the practice*.

[Jézéquel et al. 2012] : Jézéquel J.-M., Combemale B., Vojtisek D. (2012). Ingénierie Dirigée par les Modèles. Des concepts à la pratique. Ellipses Edition.

[Combemale 2008] : Combemale B. (2008). Ingénierie Dirigée par les Modèles (IDM). Etat de l'art. Institut de Recherche en Informatique de Toulouse.

[OMG MOF 2006] : Object Management Group, Inc (2006). Meta Object Facility (MOF) 2.0 Core Specification.

[OMG UML 2007] : Object Management Group, Inc. (2007). Unified Modeling Language (UML).

[Spinellis 2001] : Spinellis D. (2001). Notable design patterns for domain-specific languages. In *The Journal of Systems and Sofware*, pages 56: 91-99.

[Taha 2008] : Taha W. (2008). Domain-Specific Languages. In *International Conference on Computer Engineering Systems (ICCES)*, pages xxiii-xxviii.

[Winskel 1993] : Winskel G. (1993). The formal semantics of programming languages : an introduction. MIT Press.

[Czarnecki et al. 2003] : Czarnecki K., Helsen S. (2003). Classification of Model Transformation Approaches. In *OOPSLA'03 Workshop on Generative Techniques in the Context of Model-Driven Architecture*.

[Sendall et al. 2003] : Sendall S., Kozaczynski W. (2003). Model Transformation - the Heart and Soul of Model-Driven Software Development. In *IEEE Software*.

[OMG OCL 2010] : Object Management Group, Inc. (2010). Object Constraint Language (OCL) 2.2 Specification.

[Steinberg et al. 2008] : Steinberg D., Budinsky F., Paternostro M., Merks E. (2008). EMF: Eclipse Modeling Framework. Addison-Wesley Professional.

[Muller et al. 2005] : Muller P.-A., Fleurey F., Jézéquel J.-M. (2005). Weaving executability into Object-Oriented Meta-Languages. In *Proceedings of the 8th IEEE/ACM International Conference on Model Driven Engineering Languages and Systems (MoDELS)*, pages 264-278. Springer-Verlab.

[OMG QVT 2011] : Object Management Group, Inc. (2011). Meta Object Facility (MOF) 2.0 Query/View/Transformation Specification, version 1.1.

[Express 1994] : (1994). ISO 10303-11 : EXPRESS Language Reference Manual.

[Aït-Ameur et al. 2000] : Aït-Ameur Y., Pierra G., Sardet E. (2000). An object oriented approach to represent behavioral knowledge inhererogeous information systems. In *International Conference on Object-Oriented Information Systems (OOIS)*, pages 315-339.

[Dehainsala et al. 2005] : Dehainsala H., Jean S., Xuan Dung N., Pierra G. (2005). Ingénierie dirigée par les modèles en Express : un exemple d'application. In *Premières journées sur l'Ingénierie Dirigée par les Modèles*.

[MBSE] : INCOSE (2007). INCOSE SE Vision 2020.

[Duval et al. 2012] : Duval C., Fallet-Fidry G., Sibler A., Iung B. (2012). L'analyse intégrée des risques au profit des système socio-techniques en lien fort avec l'environnement. In *Actes du congrès Lambda Mu 18*.

[Estefan 2008] : Estefan J. A. (2008). Survey of Model-Based Systems Engineering (MBSE) Methodologies. In *Incose'08*.

[Feiler et al., 2006] : Feiler P. H., Lewiset B. A., Vestal S. (2002). The SAE Architecture Analysis & Design Language (AADL) a standard for engineering performance critical systems. In *IEEE Symposium on Computer Aided Control Systems Design*, pages 1206-1211.

[Dormoy, 2008] : Dormoy F. X. (2008). Scade 6 a model based solution for safety critical software development. In *Embedded Real-Time Systems Conference*.

[Halbwachs et al., 1991] : Halbwachs N., Caspi P., Raymond P., Pilaud D. (1991). The synchronous dataflow programming language lustre. In *Proceedings of the IEEE*, pages 79(9): 1305-1320.

[Matlab Simulink] : Mathworks. MATLAB SIMULINK - Simulation and Model Based Design.

http://www.mathworks.com.

[Baguet 2005] : Baguet A. (2005). Analyse Fonctionnelle.

[Seidner, 2009] : Seidner C. (2009). Vérification des EFFBDs: Model-checking en Ingenierie Système. Mémoire de thèse.

[Long, 1995] : Long J. (1995). Relationships b etween common graphical representations in Systems Engineering. In *Proceedings of the 5th International Symposium of the INCOSE*.

[McInnes et al., 2011] : McInnes A., Eames B., Grover R. (2011). Formalizing functional flow block diagrams using process algebra and metamodels. In *IEEE Transactions on System, Man and Cybernetics*, pages (vol. 41): 34-49.

[CORE] : http://www.vitechcorp.com/products/core.shtml, Vitech Corporation.

[OMG SysML, 2007] : OMG Document Number: formal/2007-09-01 (2007). OMG Systems Modeling Language (OMG SysML™), V1.0.

[Papyrus] : http://www.papyrusuml.org.

[TopCased] : http://www.topcased.org.

[SCADE Suite] : http://www.esterel-technologies.com/products/scade-suite/, ESTEREL Technologies.

[Thulasiraman et al., 1992] : Thulasiramal K., Swamy M. N. S. (1992). Graphs: Theory and Algorithms (5.7 Acyclic Directed Graphs). John Wiley and Son.

[Howard, 1971] : Howard R. A. (1971). Dynamic Probabilistic Systems, Volume 1: Markov Models.

[Chatelet, 2000] : Chatelet E. (2000). Sûreté de fonctionnement : méthodes et outils de base . Universitéde technologie de troyes edition.

[Ericson, 1999] : Ericson C. A. (1999). Fault Tree Analysis - A History. In *Proceedings of the 17th International System Safety conference*.

[Rauzy, 2001] : Rauzy A. (2001). Mathematical Foundation of Minimal Cutsets. In *IEEE Transaction on Reliability*, pages 50(4): 389-396.

[Thomas, 2002] : Thomas P. (2002). Contribution à l'approche booléenne de la sûreté de fonctionnement : l'atelierlogiciel Aralia Workshop. Mémoire de thèse.

[Hammersley et al., 1964] : Hammersley J. M., Handscomb D. C. (1964). Monte Carlo Methods. Chapman and Hall.

[Bryant, 1986] : Bryant R. E. (1986). Graph-based algorithms for boolean function manipulation. In *IEEE Transactions on Computers*, pages C-35(8): 677-691.

[Rauzy, 1993] : Rauzy A. (1993). New algorithms for fault trees analysis. In *Reliability Engineering and System Safety*, pages 40: 203–211.

[Dugan et al., 1998] : Dugan J. B., Manian R., Coppit D., Sullivan K. J. (1998). Combining various tech-niques for dynamic fault tree analysis of computer systems. In *Proceedings ofInternational Symposium on High Assurance System Engineering*.

[Bouissou et al., 2003] : Bouissou M., Bon J.-L. (2003). A new formalism that combines advantages of fault-trees and Markov models: Boolean logic driven Markov processes. In *Reliability Engineering and System Safety*.

[Cepin et al., 2002] : Cepin M., Mavko B. (2002). A dynamic fault tree. In *Reliability Engineering andSystem Safety*, pages 75: 83–91.

[MBSA Guideline, 2011] : Airbus Operation S.A.S. (2011). Model Based Safety Assessment Guideline.

[Joshi et al., 2006] : Joshi A., Whalen M., Heimdahl M. (2006). Model-Based Safety Analysis Final Report. In *NASA/CR-2006-213953*. NASA contractor report.

[Bouissou et al., 1991] : Bouissou M, Bouhadana H., Bannelier M., Villatte N. (1991). Knowledge modelling and reliability processing: presentation of the FIGARO language and associated tools. In *Proceedings of Safecomp'91*.

[Point et al., 1999] : Point G., Rauzy A. (1999). AltaRica: constraint automata as a description language. In *Journal Européen des Systèmes Automatisés*, pages 33:1033-1052.

[Point, 2000] : Point G. (2000). AltaRica: contribution à l'unification des méthodes formelles et de la sûreté de fonctionnement. Mémoire de thèse.

[Arnold et al., 2000] : Arnold A., Point G., Griffault A., Rauzy A. (2000). The AltaRica formalism for describing concurrent systems. In *Fundamenta Informaticae*, pages 40:109-124.

[Bozzano et al., 2003] : Bazzano M., Villafiorita A., Âkerlund O., Bieber P., Bougnol C., Böde E., Bretschneider M., Cavallo A., Castel C., Cifaldi M., Cimatti A., Griffault A., Kehren C., Lawrence B., Lüdtke A., Metge S., Papadopoulos C., Passarello R., Peikenkamp T., Persson P., Seguin C., Trotta L., Valacca L., Zacco G. (2003). ESACS: an integrated methodology for design and safety analysis of complex systems. In *Proceedings of ESREL*.

[Bozzano et al., 2006] : Akerlund O., Bieber P., Boede E., Bozzano M., Bretschneider M., Castel C., Cavallo A., Cifaldi M., Gauthier J., Griffault A., Lisagor O., Lüdtke A., Metge S., Papadopoulos C.,Peikenkamp T., Sagaspe L., Seguin C., Trivedi H., Valacca L. (2006). ISAAC, a framework forintegrated safety analysis of functionale, geometrical and human aspects. In *Proceedings of ERTS*.

[Bozzano et al., 2011] : Bozzano M., Cimatti A., Lisagor O., Mattarei C., Mover S., Reveri M., Tonetta S. (2011). Symbolic Model Checking and Safety Assessment of AltaRica Models. In *Proceedings of the 11th International Workshop on Automated Verification of Critical Systems (AVoCS 2011)*.

[Humbert, 2008] : 2008 (Humbert S.). Déclinaison d'exigences de sécurité du système vers le logiciel, assistée par des modèles formels. Mémoire de thèse.

[Sagaspe, 2008] : Sagaspe L. (2008). Allocation sûre dans les systèmes aéronautiques: Modélisation, Vérification et Génération. Mémoire de thèse.

[Bernard, 2009] : Bernard R. (2009). Analyses de sûreté de fonctionnement multi-systèmes. Mémoire de thèse.

[Adeline, 2011] : Adeline R. (2011). Méthodes pour la validation de modèles formels pour la sûreté de fonctionnement et extension aux problèmes multi-physiques. Mémoire de thèse.

[Chaudemar 2012] : Chaudemar J.-C. (2012). Etude des architectures de sécurité de systèmes autonomes : formalisation et évaluation en Event-B. Mémoire de thèse.

[Bozzano et al., 2007] : Bozzano M., Villafiorita A. (2007). The FSAP/NuSMV-SA Safety Analysis Platform. In *International Journal on Software Tools for Technology Transfer*, pages 9(1): 5-24.

[Vincent, 2003] : Vincent A. (2003). Conception et réalisation d'un vérificateur de modèles AltaRica. Mémoire de thèse.

[Arnold, 1994] : Arnold A. (1994). Finite Transition Systems. Prentice-hall edition.

[Hopcroft et al., 2006] : Hopcroft J., Ullman J., Motwani R. (2006). Introduction to Automata Theory, Languages and Computation (3rd Edition). Addison-Wesley.

[Rauzy, 2002] : Rauzy A. (2002). Modes automata and their compilation into fault trees. In *Reliability Engineering and System Safety*, pages 78: 1-12.

[Boulanger, 2012] : Boulanger J.-L. (2012). Outils de mise en oeuvre industrielle des techniques formelles. Hermes Science Publications.

[Bouissou et al., 2006] : Bouissou M., Seguin C. (2006). Comparaison des langages de modélisation AltaRica et Figaro. In *Proceedings of LM-15*.

[Papadopoulos et al., 1999] : Papadopoulos Y., McDermid J. A. (1999). Hierarchically Performed Hazard Origin and Propagation Studies. In *Proceedings of SAFECOMP'99, 18th International Conference on Computer*

Safety, Reliability and Security, pages 1698: 139-152. Springer Verlab.

[Papadopoulos et al., 2008] : Papadopoulos Y., Walker M., Wolforth I. (2008). A language for failure patterns and application in safety analysis. In *IEEE Conference on Dependable Computing Systems (DEPCOS08)*. IEEE Computer Society.

[SAE-AS5506/1] : International Society of Automotive Engineers (2006). SAE Architecture Analysis and Design Language (AADL) Annex Volume 1, Annex E: Error Model Annex.

[Feiler et al., 2007] : Feiler P., Rugina A. (2007). Dependability Modeling with the Architecture Analysis & Design Language (AADL). Software Engineering Institute.

[Rugina, 2007] : Rugina A.-E. (2007). Modélisation et évaluation de la sûreté de fonctionnement – De AADL vers les réseaux de Petri stochastiques. Mémoire de thèse.

[Joshi et al. 2007] : Joshi A., Vestal S., Binns P. (2007). Automatic Generation of Static Fault Trees. In *Workshop on Architecting Dependable Systems of The 37th Annual IEEE/IFIP Int. Conference on Dependable Systems and Networks*.

[Papadopoulos et al., 2001] : Papadopoulos C., Maruhn M. (2001). Model-based Synthesis of Fault Trees from Matlab-Simulink Models. In *Proceedings of DSN'01*.

[Abdulla et al., 2004] : Abdulla P. A., Deneux J., Stalmarck G., Agren H., Akerlund O. (2004). Designing safe, reliable systems using Scade. In *Sympothium on Leveraging Applications of Formal Methods (ISoLA)*.

[Joshi et al., 2005] : Joshi A., Heimdahl M. P. (2005). Model-Based Safety Analysis of Simulink Models Using SCADE Design Verifier. In *Proceedings of SAFECOMP, vol. 3688 of LNCS*, pages 122-135. Springer-Verlag.

[Bozzano, 2003] : Bozzano M., Villafiorita A. (2003). Improving System Reliability via Model Checking: The FSAP/NuSMV-SA Safety Analysis Platform. In *Proceeding of Computer Safety, Reliability and Security: 22th International Conference, SAFECOMP 2003*, pages 22: 49-62. Springer Varlab.

[Maitrehenry et al. 2011 (2)] : Maitrehenry S., Metge S. Bieber P., Ait-Ameur Y. (2011). Towards Model-Based Functional Hazard Assessment at Aircraft Level. In *Proceedings of the ESREL'11 conference*. Springer Verlab.

[Rasmussen 1983] : Rasmussen J. (1983). Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models. In *IEEE Transactions on Systems, Man, and Cybernetics*.

[Doguc et al. 2009] : Doguc O., Kardes O. (2009). Sensitivity Analysis Method for System Operational Effectiveness in Complex Systems. In *Proceedings of the 7th Annual Conference on Systems Engineering Research (CSER)*.

[Bernard et al., 2002] : Bernard R., Aubert J.-J., Bieber P., Merlini C., Metge S. (2002). Experiments in Model-Based Safety Analysis: Flight Controls. In *Dependable Control of Discrete Systems*, pages (vol. 1, part 1) 43-48.

[Castel et al., 2001] : Castel C., Seguin C. (2001). Modèles Formels pour l'Evaluation de la Sûreté de Fonctionnement des Architectures Logicielles d'Avionique Modulaire intégrée.

[Louis et al. 2012] : Louis V., Many F. (2012). New DGA activities driven by model based safety analysis: Initial military rotorcraft qualification and accident investigation. In *Model Based Safety Assessment Workshop (MBSA 2012)*.

[Belmonte 2009] : Belmonte F. (2009). Impact des postes centraux de supervision de trafic ferroviaire sur la sécurité. Mémoire de thèse.

[Sandberg et al. 2010] : Sandberg A., Chen D., Lönn H., Johansson R., Feng L., Törngren M., Torchiaro S., Tavakoli-Kolagari R., Abele A. (2010). Model-Based Safety Engineering of Interdependent Functions in Automotive Vehicles Using EAST-ADL2. In *Proceeding of Safecomp'10*, pages 6351: 332-346. Springer Verlab.

[Maitrehenry et al. 2012] : Maitrehenry S., Metge S., Bieber P. (2012). Modélisation de l'efficacité fonctionnelle pour l'identification de scénario de pannes. In *Actes du 18ème congrès de maitrise des risques et sûreté de*

fonctionnement (Lambda Mu 18).

[Chaudemar et al., 2009] : Chaudemar J.-C., Bensana E., Castel C., Seguin C. (2009). AltaRica and Event-B Models for Operational Safety Analysis: Unmanned Aerial Vehicle Case Study. In *Workshop on Integration of Model-based Formal Methods and Tools*.

[Belmonte et al. 2010] : Belmonte F., Blas A., Mejia L.-F., Thomas F. (2010). Evaluation des risques des systèmes ferroviaires, supporté par des langages et outils de modélisation. In *Proceeding of Lambda-Mu 17*.

[Whittle et al. 2005] : Whittle J., Gajanovic B. (2005). Model transformations should be more than just model generators. In *Briand L.C. Williams C. (eds) MoDELS*, pages 3713: 32-38. Springer-Verlab.

[Vara et al. 2009] : Vara J. M., Bollati V. A., Vela B., Marcos E. (2009). Leveraging Model Transformation by means of Annotation Models. In *MtATL*, pages 96-102. F. Jouault, editor.

[Drivalos et al. 2008] : Drivalos N., Paige R., Fernanders F., Kolovos D. (2008). Towards Rigorously defined Model-to-Model traceability. In *Traceability Workshop, European Conference in Model Driven Architecture (EC-MDA)*.

[Bondé et al. 2005] : Bondé L., Boulet P., Dekeyser J. (2006). Traceability and Interoperability at Different Levels of Abstraction in Model Transformations. In *Applications of Specification and Design Languages for SoCs*, pages 263-276. Springer Netherlands.

[Vanhooff et al. 2007] : Vanhooff B., Van Baelen S., Joosen W., Berbers Y. (2007). Traceability as Input for Model Transformations. In *Proceedings of the European Conference in Model Driven Architecture (EC-MDA), Traceability Workshop*.

[ECCO Toolkit] : PDTec. ECCO Toolkit, a new generation of EXPRESS-based Development Environments.

[David et al. 2010] : David P., Idasiak V., Kratz F. (2010). MéDISIS, l'intégration des analyses de SdF aux processus d'Ingénierie Système Basée sur les Modèles. In *Actes de la conférence LambdaMu 17*.

[MDWorkbench] : SODIUS. http://sodius.com/mdworkbench.

[Maitrehenry et al. 2012 (2)] : Maitrehenry S., Metge S., Bieber P., Ait-Ameur Y. (2012). An MDE-Based Synthesis of Aircraft Safety Models. In *Second International Conference on Model & Data Engineering (MEDI)*.

[Grüber 1993] : Grüber T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledges Sharing. In *Formal Ontology in Conceptual Analysis and Knowledge Representation*, pages . Kluwer Academic Publishers.

[Guizzardi et al. 2007] : Guizzardi G. (2007). On Ontology, Ontologies, Conceptualizations, Modeling Languages, and (Meta) Models. In *Frontiers in Artificial Intelligence and Applications, Databases and Information Systems IV*, pages . IOS Press.

[Tairas et al. 2009] : Tairas R., Mernik M., Gray J. (2009). Using Ontologies in the Domain Analysis of Domain-Specific Languages. In *Models in Software Engineering*, pages LNCS 5421: 332-342. Springer Verlab.

[Reason 1990] : Reason J. (1990). Human error. Cambridge University Press.

[Hollnagel 2012] : Hollnagel E. (2012). FRAM: the Functional Resonance Analysis Method. Modelling Complex Socio-Technical Systems. Ashgate Publishing Limited.

[Woods et al. 2005] : Woods D., Hollnagel E. (2005). Joint cognitive systems: Foundations of cognitive systems engineering. CRC Press.

[Bieder 2006] : Bieder C. (2006). Les facteurs humains dans la gestion des risques, évolution de la pensée et des outils. Hermes.

[Mawhin et al. 2011] : Mawhin B., Cabon P., Buratto F. (2011). Integrating aircrew resources variability in the design of future cockpits. In *Proceedings of the 16th International Symposium on Aviation Psychology*.

[Martinie et al. 2012] : Martinie C., Palanque P., Pasquini A., Ragosta M., Rigaud E., Silvagni S. (2012). Using

Complementary Models-Based Approaches for Representing and Analysing ATM Systems Variability. In *Proceedings of the 2nd International Conference on Application and Theory of Automation in Command and Control Systems (ATACCS '12)*, pages 146-157. IRIT Press.

[Maitrehenry et al. 2011] : Maitrehenry S., Mawhin B. (2011). Développement de modèles formels analysant les dépendances entre les fonctions avion et les aspects humains. MEMO Airbus Operation S.A.S..

Annexe A

Preuve de la composition des modèles de sécurité

Démonstration de l'existence de la fonction σ

La sémantique de nos modèles de sécurité (voir la section 2.2.3 du chapitre 4) fait apparaître une fonction $\sigma()$ permettant de calculer les valeurs des résultats d'un modèle par rapport à la structure du modèle et aux comportements des nœuds du modèle. Cette fonction correspond alors à une loi de comportement sur l'ensemble du modèle. Sa définition rappelle en effet celle des lois de comportement des activités, des données et des conditions environnementales. Nous constatons que tous les nœuds possèdent une sémantique de la forme suivante :

$$\forall n \in Noeud,\ \|n\| = (D, c(n), \sigma(n)),\ \text{avec :}$$

- $c(n)$ le comportement intrinsèque du nœud, évalué par un vecteur de x éléments de D ;
- $\sigma(n)\colon\ D^x \times dom(input(n)) \rightarrow dom(output(n))$ la loi de comportement du nœud, tel que :

$$val(output(n)) = \sigma(n)(c(n), val(input(n)))$$

Pour chacun des nœuds nous avons en effet la correspondance :

$$\forall n \in Noeud,\ \begin{cases} n \in Env_{MBFHA} \Rightarrow \sigma(n) = Loi_{env}(n)\ et\ c(n) \in C_{act} \\ n \in Act_{MBFHA} \Rightarrow \sigma(n) = Loi_{act}(n)\ et\ c(n) \in P_{env} \\ n \in Data_{MBFHA} \Rightarrow \sigma(n) = Loi_{donnée}(n)\ et\ c(n) \in \emptyset \end{cases}$$

Introduction des liens abstraits

Puisque les valeurs calculées par nos modèles de sécurité sont les résultats modélisés dans ces modèles,

nous introduisons une famille de liens abstraits connectés exclusivement à un nœud initial de type donnée (voir la figure A.1).

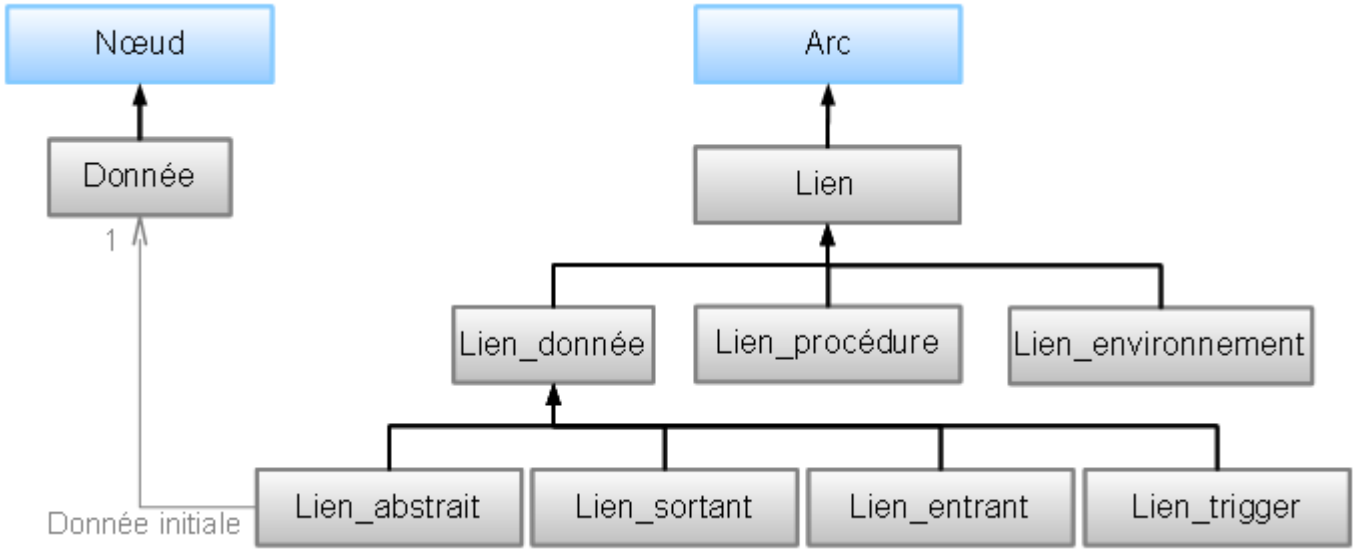


Fig. A.1 - Définition des liens abstraits

Pour chaque donnée du modèle nous construisons un lien abstrait sortant de cette donnée. D'après la loi de comportement des données, ce lien est évalué dans l'ensemble des valeurs d'efficacité et prend la valeur du résultat de la donnée auquel il est attaché.

La composition des nœuds des modèles de sécurité

La fonction $\sigma()$ du modèle MB_{FHA} est alors obtenue par composition de ses nœuds. Cette méthode de composition est présentée à la section 4.3.2 du chapitre 2 dans le cadre de la composition des nœuds du langage AltaRica. Nous montrons ci-dessous que la composition de deux nœuds donne un nouveau nœud.

Composition des nœuds des modèles MB_{FHA}. Soit deux nœuds, notés $n_i \in Noeud$, $\forall i \in [1,2]$ avec leur sémantique $\|n_i\| = (D, c(n_i), \sigma(n_i))$. Nous les considérons liés entre eux par le lien $l_{liaison} \in output(n_1) / l \in input(n_2)$.

Nous notons $input(n_i) = < i_1(n_i), \ldots, i_{card(input(n_i))}(n_i) >$ et $output(n_i) = < o_1(n_i), \ldots, o_{card(output(n_i))}(n_i) >$ les vecteurs des liens entrants et sortants des nœuds. Alors $\exists x \in [1, card(output(n_1))] / l_{liaison} = o_x(n_1)$ et $\exists y \in [1, card(input(n_2))] / l_{liaison} = i_y(n_2)$.

Nous notons les $card(output(n_i))$ fonctions composantes de $\sigma(n_i)$ de la façon suivante :

$$val(output(n_i)) = \sigma(n_i)(c(n_i), val(input(n_i))) =$$
$$< \sigma_1(n_i)(c(n_i), val(input(n_i))), \ldots, \sigma_{card(output(n_i))}(n_i)(c(n_i), val(input(n_i))) >$$

Alors la composition des deux nœuds $n_i \in Noeud$, $\forall i \in [1,2]$ est un nœud $n \in Noeud$ dont la sémantique est $\forall n \in Noeud$, $\|n\| = (D, c(n), \sigma(n))$, où :

- $c(n) = c(n_1) \times c(n_2)$ (vecteur des comportements) ;
- $\sigma(n)$: $D^x \times dom(input(n)) \rightarrow dom(output(n))$, avec :
  - $input(n) = input(n_1) \cup (input(n_2) - \{l_{liaison}\})$ (les entrées du nœud n sont les entrées des deux sous-nœuds sans le lien $l_{liaison}$ qui les relient),
  - $output(n) = (output(n_1) - \{l_{liaison}\}) \cup output(n_2)$, (les sorties du nœud n sont les sorties des deux sous-nœuds sans le lien $l_{liaison}$ qui les relient),
  - Le calcul des liens sortants :

$$\begin{aligned} val(output(n)) &= val(output(n_1) - \{l_{liaison}\}) \times val(output(n_2)) \\ &= \sigma_{/l_{liaison}}(n_1)(c(n_1), val(input(n_1))) \times \sigma(n_2)(c(n_2), val(input(n_2))) \\ &= \sigma_{/l_{liaison}}(n_1)(c(n_1), val(input(n_1))) \times \\ &\sigma(n_2)(c(n_2), val(<i_1(n_2), \dots, i_{(y-1)}(n_2), l_{liaison}, i_{(y+1)}(n_2), \dots, i_{(card(input(n_2)))}(n_2)>)) \\ &= \sigma_{/l_{liaison}}(n_1)(c(n_1), val(input(n_1))) \times \\ &\sigma(n_2)(c(n_2), val(<i_1(n_2), \dots, i_{(y-1)}(n_2), \sigma_{l_{liaison}}(n_1)(c(n_1), val(input(n_1))), i_{(y+1)}(n_2), \dots, i_{(card(input(n_2)))}(n_2)>)) \end{aligned}$$

avec :

$$\begin{aligned} \sigma_{/l_{liaison}}(n_1)(c(n_1), val(input(n_1))) = <&\sigma_1(n_1)(c(n_1), val(input(n_1))), \dots, \sigma_{x-1}(n_1)(c(n_1), val(input(n_1))), \\ &\sigma_{x+1}(n_1)(c(n_1), val(input(n_1))), \dots, \sigma_{card(output(n_i))}(n_i)(c(n_i), val(input(n_i)))> \end{aligned}$$

L'évaluation des liens sortants s'exprime exclusivement par rapport aux composantes du vecteur des liens d'entrée du nœud, $input(n) = input(n_1) \cup (input(n_2) - \{l_{liaison}\})$, et des deux composantes du vecteur des comportements $c(n) = c(n_1) \times c(n_2)$.

Nous en concluons donc qu'un modèle MB_{FHA} est un nœud particulier, composé d'autres nœuds. Il admet donc la sémantique des nœuds. Les propriétés liées à la structure de ces modèles de sécurité permettent d'en déduire la sémantique tel qu'elle est exprimée dans la section 2.2.3 du chapitre 4.

1. Graphe dirigé acyclique : Le calcul des liens sortants du nœud composé fait apparaître la nécessité de calculer la valeurs du lien $l_{liaison} \in output(n_1) / l \in input(n_2)$ au niveau du premier nœud avant de pouvoir réaliser le calcul des liens sortant du deuxième nœud. A l'échelle d'un modèle MB_{FHA}, cela signifie qu'il y a un ordre de calcul à suivre impérativement. Le calcul n'est possible que si le modèle est dirigé (pour l'ordre) acyclique (pour éviter les boucles de calcul entre les nœuds), comme c'est le cas par définition pour les modèles MB_{FHA}.
2. Graphe borné : Le caractère borné des modèles MB_{FHA} signifie que ces derniers n'ont concrètement aucun lien entrant ou sortant. La loi de comportement des modèles ne dépend donc plus que des

comportements des nœuds qui les composent. De plus, les uniques sorties calculées par nos modèles de sécurité sont donc les valeurs des liens abstraits que nous avons définis précédemment (voir la figure A.1). Formellement nous obtenons :

$\|MB_{FHA}\| = (D, C, \sigma)$ avec :

- $C = c(Act_{MBFHA}) \times c(Env_{MBFHA})$ est le vecteur des comportements de tous les nœuds du modèle (les données n'ayant pas de comportement intrinsèque),
- $\sigma : D^{card(Act_{MBFHA}) + card(Env_{MBFHA})} \times dom(input(MB_{FHA})) \rightarrow dom(output(MB_{FHA}))$, qui s'exprime plus précisément par $\sigma : (C_{act})^{card(Act_{MBFHA})} \times (C_{env})^{card(Env_{MBFHA})} \rightarrow E^{card(Data_{MBFHA})}$, car :
  - $input(MB_{FHA}) = \emptyset$,
  - $output(MB_{FHA}) = Lien\_abstrait$ dans lequel chaque lien est évalué par la notion d'efficacité.

Principe de parcours d'un modèle

Le calcul des résultats d'un modèle MB_{FHA} peut être interprété en parcourant les liens du modèle, des résultats aux nœuds intervenant de le calcul de ces résultats. Les points suivants montrent dans l'ordre le principe de calcul d'un résultat.

1. La valeur d'un résultat est calculée dans le cadre de la loi des données par rapport aux valeurs des services contribuant à ce résultat :

   $\forall d \in Donnée, r(d) = Res(val(input(d)))$.

2. Chacun de ces services est un « Lien_sortant » dont le nœud initial est une activité. La valeur de ce service est alors calculé à partir de la loi des activités :

   $\forall l \in input(d), \exists a \in Activité / l = output(a)$
   $val(l) = Loi_{act,l}(c(a), val(input(a)))$

   où $Loi_{act,l}()$ est la fonction partielle correspondante à $Loi_{act}()$ suivant la coordonnée l.

   L'évaluation du lien l est basée sur le comportement de l'activité a et la valeur des liens entrants dans l'activité.

3. Parmi les liens entrant dans l'activité, certains sont des liens de procédure (voir le point 4), certains sont des liens de donnée nécessaire ou utile (voir le point 5) et certains sont des liens d'environnement (voir le point 6.

4. Le nœud initial d'un lien de procédure est une activité (et plus précisément une opération). L'évaluation de ce lien dépend donc de la loi des activités et plus précisément de la fonction d'activation :

   $\forall o \in Opération, \ \forall l \in output(o) / l \in Lien\_procédure, \ val(l) = Activation(o)$

   avec $Activation(o) = (val_p(o) \wedge (val_n(o) \neq e_{nul}) \wedge (val_p(o) \leqslant p_{cond}))$.

Cette évaluation est alors fonction des liens entrants dans l'opération. Nous nous retrouvons alors au point 3 précédent.

5. Le nœud initial d'un lien entrant ou trigger est une donnée, régie par la loi des données :

   $\forall d \in Donnée,\ val(output(d)) = Loi_{donnée}(val(input(d)))$.

   Cette loi nécessite une évaluation des services entrant dans la donnée. Nous nous retrouvons alors dans le cas du point 2 précédent.

6. Le nœud initial d'un lien d'environnement est une condition environnementale. L'évaluation de ce lien dépend de la loi des conditions environnementales à partir d'une valeur caractéristique de la puissance de la condition environnementale :

   $\forall n_{env} \in Environnement,\ val(output(n_{env})) = Loi_{env}(Val_{env}(n_{env}))$.

En partant d'un résultat, il est donc possible de parcourir pas à pas les données, les activités et les conditions environnementales dont les valeurs des liens entrants ont un impact sur celle du résultat considéré. Comme les modèles MB_{FHA} sont continus, bornés et acycliques, le parcours aboutit à un sous-ensemble de nœuds n'ayant aucun lien en entrée. Finalement, la valeur du résultat ne dépend plus que des comportements des activités contributrices et des puissances des conditions environnementales concernées. D'où $R(Donnée) = \sigma(c(Activité), Val_{env}(Environnement))$.

Nous avons en outre montré que la fonction $\sigma()$ est construite par application des lois de comportement des conditions environnementales, des données et des activités.

Annexe B

Synchronisation d'activités dans les modèles de sécurité

Représentations multiples d'une même activité

Certains modèles peuvent requérir plusieurs modélisations différentes d'une même activité. Ce cas peut être rencontré pour diverses raisons.

– Durant la phase de vol modélisée, le contexte opérationnel évolue si bien que le comportement d'une activité n'est plus exactement le même du début à la fin de la phase de vol. Le changement peut se situer par exemple au niveau du choix de l'efficacité nominale.

Exemple. Pendant le décollage, la fonction « Guider aérodynamiquement » (voir la figure du chapitre 2) contribue au contrôle de la trajectoire. Toutefois, au début du décollage, cette fonction est faiblement efficace du fait des faibles forces aérodynamiques. Mais à la fin du décollage, la vitesse de l'avion a fortement augmenté, si bien que cette fonction a fortement gagné en efficacité.

– Si une activité est définie avec un périmètre très large, elle peut transmettre plusieurs services distincts à différentes étapes d'une séquence d'activités.

Exemple. Considérons par exemple une fonction nommée « Gérer la communication audio au sein de l'avion », qui permet entre autre aux pilotes de s'adresser à l'équipage cabine ou aux passagers. Or, juste avant le décollage, l'équipage cockpit informe simultanément l'équipage cabine et les passagers que le décollage va bientôt avoir lieu. Deux services, entre autre, sont alors identifiés pour la même fonction : mettre en communication les équipages cockpit et cabine et mettre en communication l'équipage cockpit et l'ensemble de la cabine. Les deux activités de communication peuvent être séparées dans le temps,

pendant lequel les moyens de communication sont inactifs. L'une des façons de modéliser ce comportement est de faire intervenir la fonction deux fois pour assister les deux opérations différentes.

Méthodes de modélisation

Dans le cadre de nos expérimentations, ce genre de cas ne nous est apparu que de façon très marginale. Nous avons identifié deux méthodes de modélisation pour pallier ces situations.

1. La première méthode est de réduire le périmètre d'étude afin de séparer les séquences d'activité faisant apparaître la même activité. Dans le cadre de nos applications, nous avons toujours pu appliquer cette méthode sans perte du point de vue des analyses. La charge de modélisation est simplifiée, mais davantage de modèles doivent être développés pour obtenir l'ensemble des analyses désirées sur le périmètre choisi. En pratique, nous constatons également que diminuer la taille des modèles en rend l'exploitation plus aisée. Nous conseillons donc vivement l'utilisation de cette méthode.
2. La seconde méthode est de modéliser les différentes utilisations d'une même activité comme s'il s'agissait d'activités distinctes. Puis, dans un second temps, il faut mettre en place des synchronisations entre les pannes de ces activités afin de synchroniser indirectement leurs états de fonctionnement. Nous simulons ainsi le fait que ces activités sont en réalité une seule activité avec un état de fonctionnement unique. D'après la section 4.3.2 du chapitre 2, nous avons montré que la synchronisation dans AltaRica se présente comme la réunion d'événements de différents nœuds du modèle. Dans nos modèles MB_{FHA}, les événements liés aux activités sont bien les pannes fonctionnelles. Le type de synchronisation choisi est la synchronisation forte puisque nous souhaitons interdire la possibilité que les activités synchronisées puissent changer d'état indépendamment des autres.

Annexe C

Modélisation du fonctionnement intempestif

Dans la section 2.3 du chapitre 5, nous avons présenté notre proposition de modélisation des activités des modèles de sécurité. Nous avons alors volontairement écarté le mode de panne intempestif parmi les modes de pannes fonctionnelles modélisés. Nous avons vu que les modèles proposés sont néanmoins riches en information et aptes à assister efficacement l'analyse des risques. Toutefois, il est évident que certaines combinaisons de pannes ne sont pas traitées par nos modèles de sécurité du fait de la non modélisation de l'activation intempestive potentielle. Nous identifions donc la prise en compte de ce mode de panne comme une première perspective future.

D'ores et déjà, quelques tests ont été menés sur la base des principes développés dans la section 3.2.4 du chapitre 4. Ils montrent en particulier les difficultés que nous pouvons rencontrer avec ce mode de panne et qui justifient de ne pas le prendre en compte dans les premières approches de modélisation.

Ajout du mode de panne intempestif dans les modèles

Déclaration du mode de fonctionnement intempestif

Considérons une extension de l'ensemble des états fonctionnels pour intégrer le mode de fonctionnement intempestif (voir la table C.1, construite sur la base de la table 5.3 du chapitre 5).

```
1  domain MBFHA_types_Failure_modes_2 =
2  {Erroneous_asym, Erroneous_sym, Nominal,
3  Partially_lost_asym, Partially_lost_sym,
4  Totally_lost, Inadvertant};
```

Tab. C.1 - Code AltaRica de l'ensemble des modes de fonctionnement avec le mode intempestif

L'activation intempestive est un nouvel événement. Sa transition ne peut avoir lieu que si l'état de fonctionnement courant est nominal et si la fonction n'est pas activée. La table C.2 présente alors le code AltaRica de cette transition.

```
1  node Fonction_intempestif
2    flow
3  //Variables d'entrée
4  //Variables d'agrégation
5  //Variable d'activation
6   Act:bool:private;
7    state
8  //Variable interne
9      status:MBFHA_types_Failure_modes_2;
10   event
11     total_loss,
12     partial_loss_sym,
13     partial_loss_asym,
14     error_sym,
15     error_asym,
16     inadvertant;
17   trans
18   ((status = Nominal) AND not(Act)) |-
19 inadvertant -> status := Inadvertant;
20 […]
21   assert
22 […]
23   init
24     status := Nominal;
25 edon
```

Tab. C.2 - Code AltaRica de la modélisation du fonctionnement intempestif

L'activation des fonctions intempestives

Enfin, les assertions qui calculent les services de la fonction doivent prendre en compte le nouvel état interne. Le changement se situe en premier lieu au niveau de l'activation. D'après les concepts (voir la section 3.2.4 du chapitre 4), l'assertion pour le calcul de l'activation est :

```
1   node Fonction_intempestif
2   […]
3     assert
4   //Calcul des valeurs agrégées
5   […]
6   //Calcul de l'activation
7   Act:=((val_N!=None) OR (status =
8   Inadvertant);
9   […]
10  edon
```

Tab. C.3 - Code AltaRica de l'activation d'une fonction avec le fonctionnement intempestif

Nous en déduisons donc que le seul moyen pour qu'une telle fonction puisse s'activer intempestivement est que la valeur agrégée de ses données nécessaires soit nulle, c'est à dire que l'une de ses données nécessaires au moins est manquante. Grâce au fonctionnement intempestif, la fonction s'exécute malgré tout et fournit ses services.

D'après l'un des principes de la loi de comportement concrète des activités, la valeur du service principal ne peut pas excéder la valeur agrégée des données nécessaires, soit la valeur « None » (voir la section 2.3.4 du chapitre 5). Alors, nous vérifions bien que le service principal est négatif ou nul selon les dégradations supplémentaires de la part des conditions environnementales ou des données utiles. Sans le fonctionnement intempestif, le service principal est nul ; nous en concluons donc que le fonctionnement intempestif est bien un cas potentiellement aggravant d'une situation globale d'un modèle de sécurité.

Toutefois, en pratique, pour visualiser un effet d'une activation intempestive d'une fonction du modèle, il faut au minimum trois pannes :

1. la panne fonctionnelle ayant conduit à la non-production d'une donnée nécessaire qui a entraîné la non-activation de la fonction considérée ;
2. l'activation intempestive de la fonction considérée ;
3. la panne ou la présence d'une condition environnementale qui dégrade les autres entrées de la fonction pour que ses services deviennent négatifs.

Logiquement, si nous nous limitons dans un premier temps à l'étude des pannes doubles, la modélisation des activations intempestives ne nous apporte aucun résultat.

Modélisation du déclenchement intempestif de fonctions externes au modèle

Pourtant, il y a des cas évidents de déclenchement fonctionnel intempestif sans autre panne qui sont pertinents et analysés au cours de l'analyse des risques.

Exemple. L'activation intempestive du freinage des roues pendant le décollage est un scénario de panne sérieux puisque ce freinage intempestif s'oppose clairement à l'objectif de vitesse que l'avion doit atteindre pour réussir son décollage. Il s'agit bien d'une panne simple, non modélisée par l'approche de la section précédente.

Si un tel scénario de panne n'est pas modélisé par la méthode de la section précédente, c'est pour une raison évidente : la fonction mise en cause n'est pas modélisée car elle n'est pas utilisée dans la phase de vol correspondant au modèle. En effet, les modèles A.O.F. n'intègrent pas les fonctions qui ne sont pas exploitées. Les concepteurs de l'avion conçoivent en effet leurs modèles en vue d'analyses fonctionnelles ; même si nous avons montré que les dépendances fonctionnelles sont intéressantes pour l'étude des pannes, certains dysfonctionnements ne peuvent pas être capturés à partir de tels modèles. C'est le cas pour le fonctionnement intempestif !

Proposition de modélisation des fonctions externes au modèle

Nous proposons alors de définir dans nos modèles de sécurité une nouvelle famille de fonctions non exploitées, mais dont l'activation intempestive s'oppose à certains résultats du modèle. Nous identifions alors trois états internes possibles à ces fonctions : l'état non activé, l'état activé et l'état activé asymétriquement (voir la table C.4). Ce troisième état interne est en effet pertinent pour certaines fonctions pour dégrader d'éventuels services asymétriques.

```
1    domain MBFHA_types_Failure_inadvertant =
2    {Not_activated, Partially_activated,
3    Totally_activated};
```

Tab. C.4 - Code AltaRica de la modélisation du fonctionnement intempestif

Exemple. Pour le freinage des roues pendant le décollage, l'état activé représente un freinage total au niveau de toutes les roues. L'impact est alors très négatif sur la vitesse. L'état activé asymétriquement quant à lui représente un freinage des roues situées du même coté de l'avion. La vitesse subit donc un impact négatif, mais du fait du freinage différentiel, le contrôle de la trajectoire subit également une dégradation.

La figure C.1 présente les icônes de ces fonctions. Les flèches noires signifient que la fonction n'est pas activée. Si les deux flèches sont rouges, alors la fonction est dans l'état activé. Une flèche rouge seulement

représente un état activé asymétrique.

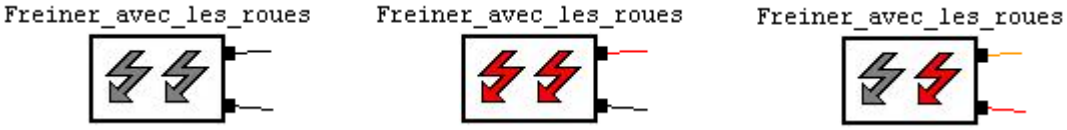


Fig. C.1 - Représentation graphique des fonctions intempestives

Le code AltaRica d'une telle fonction est présenté dans la table C.3.

```
1   node Fonction_intempestif
2     flow
3   //Service principal
4    S1:MBFHA_types_Effectiveness_5:out;
5   //Service asymétrique
6    S2:MBFHA_types_Effectiveness_5:out;
7     state
8   //Variable interne
9       status:MBFHA_types_Failure_inadvertant;
10    event
11      partial_inadvertant,
12      total_inadvertant;
13    trans
14   (status = Not_activated) |-
15  partial_inadvertant -> status :=
16  Partially_activated;
17   (status = Not_activated) |-
18  total_inadvertant -> status :=
19  Totally_activated;
20    assert
21  //Calcul du service principal
22   S1:= Case{
23      status = Partially_activated : LNeg,
24      status = Totally_activated : Hneg,
25      else None };
26  //Calcul du service asymétrique
27   S1:= Case{
28      status = Partially_activated : HNeg,
29      else None };
30    init
31      status := Not_activated;
32  edon
```

Tab. C.5 - Code AltaRica du comportement en fonctionnement intempestif

L'initialisation du nœud est l'état interne non-activé dans le code AltaRica précédent. Mais, comme nous l'avons vu dans la section 3 du chapitre 5, cet état interne peut varier selon la situation initiale choisie pour les analyses.

Exploitation du fonctionnement intempestif de fonctions externes au modèle

Notre proposition de modélisation de l'activation intempestive de fonctions externes aux modèles nominaux ne fait intervenir aucun lien entrant, mais uniquement des services négatifs ou nuls. Nous constatons alors que les dépendances fonctionnelles concernant ces fonctions se limitent exclusivement à leurs impacts directs sur des résultats. Il s'agit d'une première critique que nous pouvons faire sur notre proposition : elle ne nous permet pas de modéliser des séquences entières d'activités externes aux modèles et pouvant s'activer de façon non désirée, selon une réaction en chaîne à partir d'un unique déclenchement intempestif.

Néanmoins, notre proposition de fonction à activation intempestive peut impacter directement des résultats existants d'un modèle pour en diminuer la valeur d'efficacité d'après la loi de combinaison des services (voir la section 2.2.1 du chapitre 5). Dans le cadre de tests, nous avons enrichi les modèles d'étude du contrôle de la vitesse et de la trajectoire pendant le décollage que nous avons présentés au chapitre 7. Nous avons ajouté une fonction à activation intempestive, le freinage des roues, au sous-modèle permettant le contrôle de la vitesse. La figure C.2, ci-dessous, illustre cet ajout.

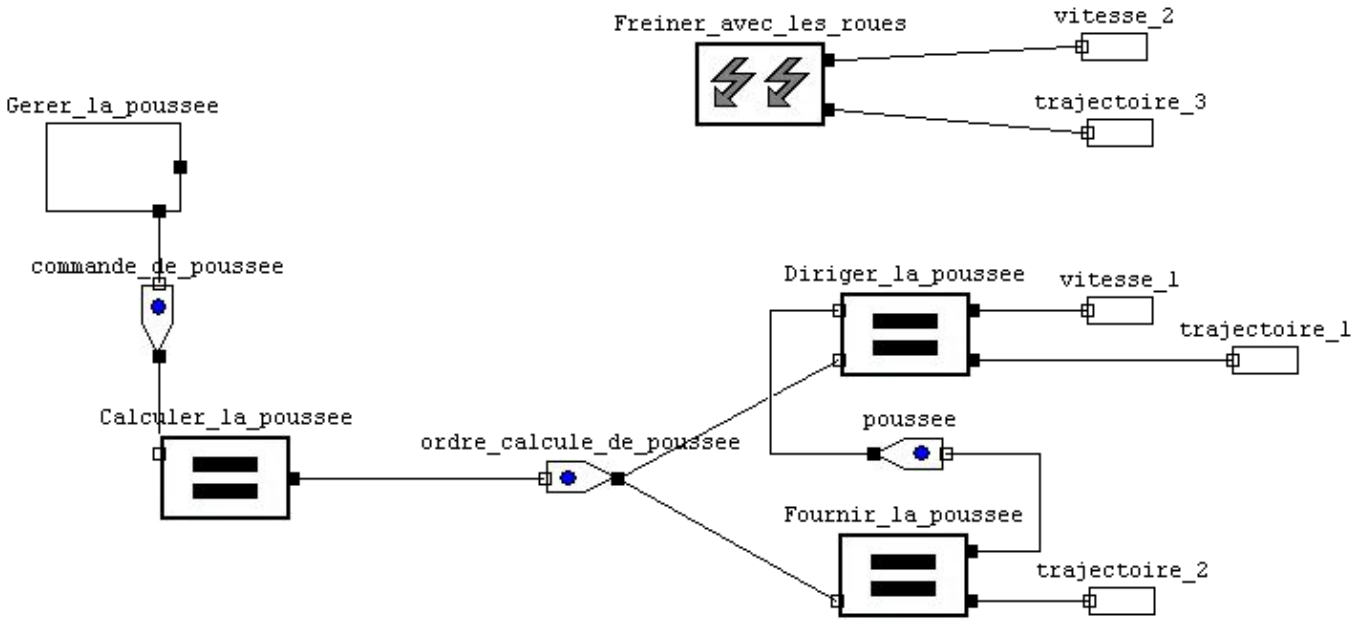


Fig. C.2 - Modèle AltaRica de la gestion de la poussée avec le freinage intempestif des roues

Le principe des analyses de ce modèle enrichi n'est pas modifié par rapport à ce que nous avons présenté en section 3 du chapitre 5. Le fonctionnement intempestif est alors traité au niveau des analyses comme les autres modes de panne : il peut être simulé et il peut être identifié dans des séquences.

Index des figures

Index des tables

Glossaire

A/C	Aircraft
ADCN	Avionic Data Central Network
ADD	Arbre De Défaillance
ALFA	Aircraft Level Functional Approach
ARP	Aerospace Recommended Practice
ASA/SSA	Aircraft/System Safety Assessment
ATM	Air Traffic Management
CAT	Catastrophic
DAG	Directed Acyclic Graph
DAL	Development Assurance Level
DD	Dependance Diagram
DSML	Domain Specific Modeling Language
EASA	European Aviation Safety Agency
EFFBD	Enhanced Functional Flow Block Diagram
ESFL	Ensure Safe Flight and Landing
FC	Failure Condition
FHA	Functional Hazard Assessment
FRAM	Functional Resonance Analysis Method
FT	Fault Tree
HAZ	Hazardous
HiP-HOPS	Hierarchically Performed Hazard Origin and Propagation Studies

LaBRI	Laboratoire Bordelais de Recherche en Informatique
MAJ	Major
MBFHA	Model-Based FHA
MBSA	Model-Based Safety Assessment
MIN	Minor
MISSA	More Integrated and cost efficient System Safety Assessment
MOF	Meta Object Facility
NSE	No Safety Effect
OCAS	Outil de Conception et d'Analyse Système
PASA/PSSA	Preliminary Aircraft/System Safety Assessment
SysML	System Modeling Language
UML	Unified Modeling Language
WG	Working Group

Modélisation des dépendances fonctionnelles pour l'analyse des risques de niveau avion

Résumé :
Nos travaux se situent au croisement de trois domaines : la sûreté de fonctionnement, l'analyse fonctionnelle et l'ingénierie des modèles. Dans l'objectif d'assister les analyses préliminaires des risques, nous avons proposé d'exploiter les modèles issus de l'analyse fonctionnelle de l'avion. Ces modèles décrivent les dépendances entre les fonctions qui doivent être réalisées durant une phase de vol. Pour exploiter ces modèles, nous avons introduit la notion d'efficacité qui mesure le degré de contribution d'une fonction à la réalisation nominale d'une phase de vol. Cette notion est utile pour les analyses de risques car elle permet de formaliser divers cas de dysfonctionnements des fonctions et pour évaluer le niveau de dégradation d'une phase de vol en cas de dysfonctionnement d'une ou plusieurs fonctions. Nous avons proposé d'annoter les modèles issus de l'analyse fonctionnelle avec des informations relatives à l'efficacité des fonctions et à leurs dysfonctionnements possibles. En suivant les principes de la transformation de modèles, nous avons étudié les moyens de produire le plus automatiquement possible des modèles utiles aux analyses de risques à partir des modèles annotés. Les modèles produits sont décrits avec le langage AltaRica, ils peuvent être analysés avec les outils associés à ce langage afin d'évaluer l'effet du dysfonctionnement de fonctions de l'avion ou de rechercher les combinaisons de dysfonctionnements les plus critiques. L'approche proposée a été appliquée pour analyser les risques associés aux fonctions utiles lors du décollage d'un avion.

Mots clés : Aéronautique, Système socio-technique, Analyse des risques, Analyse fonctionnelle, Fiabilité, Méthodes formelles, Ingénierie dirigée par les modèles.

Functional dependencies modelling for aircraft level risk analysis

Abstract :
Our work links three domains: safety, functional analysis and model based engineering. In order to assist preliminary risk analysis, we have proposed to exploit models developed for functional analysis. These models describe dependencies between functions that have to be performed during a flight phase. To exploit these models, we have introduced the notion of efficiency that measures the degree of contribution of a function to the nominal realisation of a flight phase. This notion is useful for risk analysis because it enables the formalisation of various cases of function failures and the evaluation of the level of degradation of a flight phase in case of function failures. We have proposed to annotate functional analysis models with information related with function efficiency and potential function failures. Following the principles of model transformation, we have studied the means to produce as automatically as possible models that could be used to support risk analysis starting from annotated models. Produced models are described with the AltaRica language, they can be analysed with the tools associated with this language in order to evaluate the effect of function failures or to search for the most critical combinations of failures. The approach was applied in order to analyse the risks associated with the functions used during an aircraft take-off.

Keywords : Aeronautics, Socio-technical systems, Risk analysis, Functional analysis, Reliability, Formal methods, Model based engineering.